

ARDUPILOT PARAMETER REFERENCE

COMPLETE CONFIGURATION GUIDE

GENERATED 2025 // MAVLINK HUD SYSTEMS

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY

ArduPilot Parameter Reference

Accelerometer Configuration (ACC) (ACC) ...
Acrobatic Mode Configuration (ACRO) (ACRO) ...
Actuator / Analog Sensor Configuration (ACT) (ACT) ...
ADS-B Configuration (ADSB) (ADSB) ...
Advanced Failsafe (AFS) (AFS) ...
Attitude Heading Reference System (AHRS) (AHRS) ...
Airspeed Limits (AIRSPEED) (AIRSPEED) ...
Automatic Identification System (AIS) (AIS) ...
Altitude Configuration (ALT) (ALT) ...
Altitude Error Checks (ALTERR) (ALTERR) ...
Angle Limits & Configuration (ANGLE) (ANGLE) ...
ARM Parameter Group (ARM) ...
Arming Checks (ARMING) (ARMING) ...
Autonomous Autorotation (AROT) (AROT) ...
Airspeed Sensor Configuration (ARSPD) (ARSPD) ...
Autorotation Speed Control (AS) (AS) ...
Attitude Control (ATC) (ATC) ...
Auto Mode Configuration (AUTO) (AUTO) ...
Autoland Configuration (AUTOLAND) (AUTOLAND) ...
AutoTune Configuration (AUTOTUNE) (AUTOTUNE) ...
ADSB Traffic Avoidance (AVD) (AVD) ...
AVOID Parameter Group (AVOID) ...
Axis Configuration (AXIS) (AXIS) ...
Logging Backend (BACKEND) (BACKEND) ...
Balance Bot Configuration (BAL) (BAL) ...
Barometer Configuration (BARO) (BARO) ...
BAROALT Parameter Group (BAROALT) ...
BAROALTERR Parameter Group (BAROALTERR) ...
BAROEXT Parameter Group (BAROEXT) ...
BAROFIELD Parameter Group (BAROFIELD) ...
BAROFLTR Parameter Group (BAROFLTR) ...
BAROGND Parameter Group (BAROGND) ...
BAROPROBE Parameter Group (BAROPROBE) ...
BAROSPEC Parameter Group (BAROSPEC) ...
Battery Monitor Configuration (BATT) (BATT) ...
BATTA Parameter Group (BATTA) ...
BATTB Parameter Group (BATTB) ...
BATTC Parameter Group (BATTC) ...
BATTD Parameter Group (BATTD) ...
BATTE Parameter Group (BATTE) ...
BATTF Parameter Group (BATTF) ...
BATTG Parameter Group (BATTG) ...
Beacon Positioning (BCN) (BCN) ...
GPS Blending Configuration (BLEND) (BLEND) ...
BLK Parameter Group (BLK) ...
Brake Mode Configuration (BRAKE) (BRAKE) ...
Board Configuration (BRD) (BRD) ...
Integrated Radio Configuration (BRD_RADIO) (BRD_RADIO) ...
Smart Battery Tags (BTAG) (BTAG) ...
Button Input Configuration (BTN) (BTN) ...
Buzzer Configuration (BUZZER) (BUZZER) ...
Camera Trigger Configuration (CAM) (CAM) ...
CAN Bus Configuration (CAN) (CAN) ...
Custom Controller (CC) (CC) ...
Custom Controller Parameters (CCNT) (CCNT) ...
Cellular Telemetry (CELL) (CELL) ...
Parachute Configuration (CHUTE) (CHUTE) ...
Circle Mode Configuration (CIRCLE) (CIRCLE) ...
Mission Command Status (CMD) (CMD) ...
Common Parameters (COMMON) (COMMON) ...
Compass Configuration (COMPASS) (COMPASS) ...
Crash Detection (CRASH) (CRASH) ...
Cruise Mode Configuration (CRUISE) (CRUISE) ...
Custom Rotation (CUST) (CUST) ...
Disarmed Logging Configuration (DARM) (DARM) ...
DDS / ROS 2 Interface (DDS) (DDS) ...
Data Enable Pin (DE) (DE) ...
Scheduler Debug (DEBUG) (DEBUG) ...
Deceleration Limits (DECEL) (DECEL) ...
Default Configuration (DEFLT) (DEFLT) ...
Delay Configuration (DELAY) (DELAY) ...
Developer Options (DEV) (DEV) ...
Remote ID / Drone ID (DID) (DID) ...
Wind Direction Sensor (DIR) (DIR) ...
Disable Axis Mask (DIS) (DIS) ...
Auto Disarm Configuration (DISARM) (DISARM) ...
Distance Limits (DIST) (DIST) ...
Distance Minimum (DISTANCE) (DISTANCE) ...
Docking Mode (DOCK) (DOCK) ...
DDS Domain Configuration (DOMAIN) (DOMAIN) ...
Driver Options (DRV) (DRV) ...
DSARM Parameter Group (DSARM) ...
DShot Configuration (DSHOT) (DSHOT) ...
Differential Spoiler Configuration (DSPOILER) (DSPOILER) ...
DSPOILR Parameter Group (DSPOILR) ...
External AHRS Configuration (EAHRS) (EAHRS) ...
Electronic Fuel Injection (EFI) (EFI) ...
Extended Kalman Filter 2 (EK2) (EK2) ...
Extended Kalman Filter 3 (EK3) (EK3) ...
IMU Enable Configuration (ENABLE) (ENABLE) ...
ESC Telemetry & Calibration (ESC) (ESC) ...
External Bus Configuration (EXT) (EXT) ...
Frequency Analysis (F) (F) ...
Fast Sampling Configuration (FAST) (FAST) ...
Fly-By-Wire B Mode (FBWB) (FBWB) ...
Camera Feedback Configuration (FEEDBAK) (FEEDBAK) ...
Geofence Configuration (FENCE) (FENCE) ...
In-Flight FFT Configuration (FFT) (FFT) ...
FlowHold Mode Configuration (FHLD) (FHLD) ...
Field Altitude Configuration (FIELD) (FIELD) ...
Log File Management (FILE) (FILE) ...
Sensor Filtering (FILT) (FILT) ...
Flap Configuration (FLAP) (FLAP) ...
Flash Memory & Bootloader (FLASH) (FLASH) ...
Flight Options (FLIGHT) (FLIGHT) ...
Optical Flow Configuration (FLOW) (FLOW) ...
Flight Mode Configuration (FLTMODE) (FLTMODE) ...
Filter Configuration (FLTR) (FLTR) ...
Format Configuration (FMT) (FMT) ...
Follow Mode Configuration (FOLL) (FOLL) ...
Parameter Format Version (FORMAT) (FORMAT) ...
Frame Configuration (FRAME) (FRAME) ...
Frequency Configuration (FREQ) (FREQ) ...
FrSky Telemetry Configuration (FRSKY) (FRSKY) ...
Failsafe Configuration (FS) (FS) ...
Fast Rate Threading (FSTRATE) (FSTRATE) ...
Forward/Vertical Decoupling (FV) (FV) ...
Forward Velocity Control (FW) (FW) ...

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Forward Thrust Configuration (FWD) (FWD) ...
Ground Control Station Interface (GCS) (GCS) ...
Generator Configuration (GEN) (GEN) ...
Glide Slope Configuration (GLIDE) (GLIDE) ...
Ground Effect Compensation (GND) (GND) ...
GPIO Configuration (GPIO) (GPIO) ...
GPS / GNSS Configuration (GPS) (GPS) ...
Gripper Configuration (GRIP) (GRIP) ...
Ground Steering Configuration (GROUND) (GROUND) ...
Guided Mode Configuration (GUID) (GUID) ...
GUIDED Parameter Group (GUIDED) ...
Gyroscope Configuration (GYR) (GYR) ...
GYRO Parameter Group (GYRO) ...
Traditional Helicopter Configuration (H) (H) ...
Hardpoint Configuration (HARDPOINT) (HARDPOINT) ...
Heading Correction Configuration (HDG) (HDG) ...
Heel Control Configuration (HEEL) (HEEL) ...
Height Override Configuration (HGT) (HGT) ...
Home Point Configuration (HOME) (HOME) ...
Internal Combustion Engine (ICE) (ICE) ...
Input Management (IM) (IM) ...
IMU General Configuration (IMU) (IMU) ...
Initial State Configuration (INITIAL) (INITIAL) ...
Data Injection Configuration (INJECT) (INJECT) ...
Inertial Navigation System (INS) (INS) ...
Interval Configuration (INTRVAL) (INTRVAL) ...
Jerk Limit Configuration (JERK) (JERK) ...
Joystick Configuration (JS) (JS) ...
KDE Direct CAN Configuration (KDE) (KDE) ...
Feedforward Mixing (KFF) (KFF) ...
Landing Configuration (LAND) (LAND) ...
Lead Filter Configuration (LEAD) (LEAD) ...
Leak Detector Configuration (LEAK) (LEAK) ...
LED Configuration (LED) (LED) ...
Left Button Configuration (LEFT) (LEFT) ...
Level Flight Safety (LEVEL) (LEVEL) ...
Landing Gear Configuration (LGR) (LGR) ...
Motor Load Testing (LOAD) (LOAD) ...
DataFlash Logging Configuration (LOG) (LOG) ...
Loiter Mode Configuration (LOIT) (LOIT) ...
Magnetic Anomaly Configuration (MAG) (MAG) ...
Manual Input Configuration (MAN) (MAN) ...
Manual Mode Configuration (MANUAL) (MANUAL) ...
MAVLink Stream Configuration (MAV) (MAV) ...
Maximum Control Limits (MAX) (MAX) ...
Moving Baseline Configuration (MB) (MB) ...
Minimum Threshold Configuration (MIN) (MIN) ...
Mission Sequencing Configuration (MIS) (MIS) ...
Mixer Configuration (MIXING) (MIXING) ...
Mount / Gimbal Configuration (MNT) (MNT) ...
Mode Selection Configuration (MODE) (MODE) ...
Motor Configuration & Thrust Logic (MOT) (MOT) ...
MOT10 Parameter Group (MOT10) ...
MOT11 Parameter Group (MOT11) ...
MOT12 Parameter Group (MOT12) ...
Message Display Configuration (MSG) (MSG) ...
MSP Protocol Configuration (MSP) (MSP) ...
L1 Navigation Controller (NAVL) (NAVL) ...
Network Configuration (NET) (NET) ...
NMEA Output Configuration (NMEA) (NMEA) ...
No-Go Zone Configuration (NO) (NO) ...
Notify Configuration (NTF) (NTF) ...
Battery Cell Count (NUM) (NUM) ...
Object Avoidance (OA) (OA) ...
Offset Calibration Checks (OFF) (OFF) ...
Follow Offset Type (OFS) (OFS) ...
OneShot ESC Configuration (ONESHOT) (ONESHOT) ...
On/Off Tracker Configuration (ONOFF) (ONOFF) ...
Global and Subsystem Options (OPTIONS) (OPTIONS) ...
Sensor Orientation Configuration (ORIENT) (ORIENT) ...
Local Frame Origin (ORIGIN) (ORIGIN) ...
On-Screen Display Configuration (OSD) (OSD) ...
Output Configuration (OUT) (OUT) ...
Input Override Configuration (OVERRIDE) (OVERRIDE) ...
Generic Custom Parameters (PARAM) (PARAM) ...
PosHold Mode Configuration (PHLD) (PHLD) ...
PID Deadzone Configuration (PID) (PID) ...
Pilot Input Configuration (PILOT) (PILOT) ...
Pitch Axis Configuration (PITCH) (PITCH) ...
Payload Place Configuration (PLDP) (PLDP) ...
Precision Landing Configuration (PLND) (PLND) ...
Position Controller Configuration (POS) (POS) ...
Sensor Probing Configuration (PROBE) (PROBE) ...
Proximity Sensor Configuration (PRX) (PRX) ...
Position Controller Configuration (PSC) (PSC) ...
Pitch Controller Configuration (PTCH) (PTCH) ...
PTCH2SRV Parameter Group (PTCH2SRV) ...
Pullup Configuration (PUP) (PUP) ...
PWM Configuration (PWM) (PWM) ...
QuadPlane Configuration (Q) (Q) ...
Quality Thresholds (QUAL) (QUAL) ...
QuickTune Configuration (QWIK) (QWIK) ...
Rally Point Configuration (RALLY) (RALLY) ...
Rate Loop Configuration (RATE) (RATE) ...
Raw Data Configuration (RAW) (RAW) ...
RC Input Configuration (RC) (RC) ...
RCIN Parameter Group (RCIN) ...
RC Channel Mapping (RCMAP) (RCMAP) ...
Relay Configuration (RELAY) (RELAY) ...
Roll Controller Configuration (RLL) (RLL) ...
Roll-to-Servo Configuration (RLL2SRV) (RLL2SRV) ...
Rangefinder Configuration (RNGFND) (RNGFND) ...
RNGFNDA Parameter Group (RNGFNDA) ...
ROLL Parameter Group (ROLL) ...
RPM Sensor Configuration (RPM) (RPM) ...
RSSI Configuration (RSSI) (RSSI) ...
Reset Configuration (RST) (RST) ...
Real-Time Clock Configuration (RTC) (RTC) ...
Return to Launch Configuration (RTL) (RTL) ...
Rudder Configuration (RUDD) (RUDD) ...
RUDDER Parameter Group (RUDDER) ...
Safety Configuration (SAFE) (SAFE) ...
Sailboat Configuration (SAIL) (SAIL) ...
Configuration Persistence (SAVE) (SAVE) ...
OSD Sidebar Configuration (SB) (SB) ...
SBAS Parameter Group (SBAS) ...
SBP Parameter Group (SBP) ...
Control Scaling Configuration (SCALING) (SCALING) ...
Scan Mode Configuration (SCAN) (SCAN) ...
Real-Time Scheduler (SCHED) (SCHED) ...
Lua Scripting Configuration (SCR) (SCR) ...
Serial Port Configuration (SERIAL) (SERIAL) ...
Servo Output Configuration (SERVO) (SERVO) ...
SFML Joystick Configuration (SF) (SF) ...
System Identification Configuration (SID) (SID) ...
SITL Simulation Configuration (SIM) (SIM) ...

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Simple Mode Configuration (SIMPLE) (SIMPLE) ...
Autonomous Soaring (SOAR) (SOAR) ...
SONAR Parameter Group (SONAR) ...
Speed Scaling Configuration (SPD) (SPD) ...
Specific Gravity Configuration (SPEC) (SPEC) ...
Navigation Speed Configuration (SPEED) (SPEED) ...
Crop Sprayer Configuration (SPRAY) (SPRAY) ...
Smart RTL Configuration (SRTL) (SRTL) ...
Stabilization Options (STAB) (STAB) ...
Startup Origin Configuration (START) (START) ...
Startup Delay Configuration (STARTUP) (STARTUP) ...
Vehicle Statistics Configuration (STAT) (STAT) ...
Steering Controller Configuration (STEER2SRV) (STEER2SRV) ...
Stick Mixing Configuration (STICK) (STICK) ...
Motion Detection Threshold (STILL) (STILL) ...
Stopping Distance Configuration (STOP) (STOP) ...
Steering Gain Configuration (STR) (STR) ...
Super Simple Mode (SUPER) (SUPER) ...
Surface Configuration (SURFACE) (SURFACE) ...
Surface Tracking Configuration (SURFTRAK) (SURFTRAK) ...
Switch Method Configuration (SW) (SW) ...
MAVLink System Identification (SYSID) (SYSID) ...
Target Configuration (TARG) (TARG) ...
Temperature Calibration (TCAL) (TCAL) ...
Total Energy Control System (TECS) (TECS) ...
Temperature Sensor Configuration (TEMP) (TEMP) ...
Terrain Following Configuration (TERRAIN) (TERRAIN) ...
Throttle Configuration (THR) (THR) ...
THROTTLE Parameter Group (THROTTLE) ...
Throw Mode Configuration (THROW) (THROW) ...
TIMEOUT Parameter Group (TIMEOUT) ...
Takeoff Configuration (TKOFF) (TKOFF) ...
Toy Mode Configuration (TMODE) (TMODE) ...
Neutral Trim Configuration (TRIM) (TRIM) ...
Torqeedo Motor Interface (TRQD) (TRQD) ...
TRUE Parameter Group (TRUE) ...
In-Flight Tuning Configuration (TUNE) (TUNE) ...
TURBO Parameter Group (TURBO) ...
Turn Dynamics Configuration (TURN) (TURN) ...
UART Physical Port Configuration (UART) (UART) ...
UDP Parameter Group (UDP) ...
USE Parameter Group (USE) ...
V Parameter Group (V) ...
VEC Parameter Group (VEC) ...
Velocity Estimator Configuration (VEL) (VEL) ...
Vibration Analysis Configuration (VIB) (VIB) ...
VICON Parameter Group (VICON) ...
Visual Odometry Configuration (VISO) (VISO) ...
Video Transmitter Configuration (VTX) (VTX) ...
W Parameter Group (W) ...
Wheel Encoder Configuration (WENC) (WENC) ...
Winch Configuration (WINCH) (WINCH) ...
Wind Estimation Configuration (WIND) (WIND) ...
Wind Speed Thresholds (WNDSPD) (WNDSPD) ...
Wind Vane Configuration (WNDVN) (WNDVN) ...
Waypoint Configuration (WP) (WP) ...
Waypoint Navigation Configuration (WPNAV) (WPNAV) ...
Wheel Rate Control Configuration (WRC) (WRC) ...
Weather Vane Configuration (WVANE) (WVANE) ...
Crosstrack Configuration (XTRACK) (XTRACK) ...
Horizontal Controller Scaling (XY) (XY) ...
Yaw Controller Configuration (YAW) (YAW) ...
Yaw-to-Servo Configuration (YAW2SRV) (YAW2SRV) ...
ZigZag Mode Configuration (ZIGZ) (ZIGZ) ...

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY

Overview

The **ACC** parameter group contains configuration settings for the **Accelerometers** within the Inertial Measurement Units (IMUs). These parameters manage the low-level identification, alignment, and calibration offsets of the physical sensors.

In ArduPilot, accelerometers are the primary source for the "Gravity Vector," which establishes the vehicle's "Down" reference (Level Horizon). They are also integrated to estimate velocity and position (INS).

Key Concepts

1. Sensor Identification (`ACC_ID`)

Every accelerometer detected on the hardware bus (SPI, I2C, DroneCAN) is assigned a unique ID. This ID allows the autopilot to track the health and calibration status of a specific physical sensor even if the board is rebooted or if the sensor order changes.

2. Body Frame Alignment (`ACC_BODYFIX`)

The autopilot needs to know which sensor is "perfectly aligned" with the flight controller board. This sensor becomes the reference for the "Body Frame." If the board itself is mounted at an angle (see `AHRS_ORIENTATION`), the `ACC_BODYFIX` sensor rotates with the board, while the mathematical solution is rotated to match the vehicle frame.

3. Temperature Calibration (`ACC_CALTEMP`)

MEMS sensors drift with temperature. During factory or user calibration, the temperature is recorded (`ACC_CALTEMP`). This reference allows the `TCAL` (Temperature Calibration) system to apply compensation curves effectively.

Parameter Breakdown

- `ACC_ID`: Unique hardware identifier for the sensor.
- `ACC_BODYFIX`: Selects the primary "Body Fixed" accelerometer (usually IMU 1 or 2).
- `ACC_CALTEMP`: The temperature (°C) measured during the 6-point accelerometer calibration.
- `ACC_TRIM`: (SITL Only) Used for simulating sensor installation errors.

Integration Guide

Calibration

These parameters are typically set automatically during the **Accelerometer Calibration** process:

1. **Level Horizon:** Sets the AHRS trim.
2. **6-Point Calibration:** Rotates the vehicle through all 6 axes to determine offsets and scaling factors. This populates `ACC_CALTEMP` and the internal `INS_ACCn_OFFS` / `INS_ACCn_SCAL` parameters (often visible in the `INS` group, not `ACC`).

Troubleshooting

- **"Inconsistent Accelerometers":** If you see this pre-arm failure, it means the vector length or alignment of the secondary accelerometers disagrees with the primary (`ACC_BODYFIX`). Recalibrate.
- **Drifting Horizon:** If the horizon drifts during flight, check for high vibrations (Vibe Logging) or poor temperature calibration.

Developer Notes

- **Library:** `libraries/AP_InertialSensor`
- **Key Class:** `AP_InertialSensor`
- **Legacy Note:** In newer ArduPilot versions, many raw sensor parameters are grouped under `INS_` (Inertial Navigation System). The `ACC_` group often persists for legacy compatibility or specific `AP_Periph` configurations.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





ACC_BODYFIX (Body-fixed accelerometer)

DEFAULT: 2	RANGE: 1 3

Body-fixed accelerometer (ACC_BODYFIX)

Description

Specifies which IMU's accelerometer is considered "body-fixed" for the purpose of calculating attitude trims.

Tuning & Behavior

- **Default Value:** 2 (IMU 2)
- **Values:** 1:IMU 1, 2:IMU 2, 3:IMU 3

ACC_CALTEMP (Calibration temperature for accelerometer)

DEFAULT: -300	UNITS: degC

Calibration temperature for accelerometer (ACC_CALTEMP)

Description

Temperature recorded during accelerometer calibration.

Tuning & Behavior

- **Default Value:** -300 degC

ACC_ID (Accelerometer ID)

DEFAULT: 0

Accelerometer ID (ACC_ID)

Description

Accelerometer sensor ID.

Tuning & Behavior

- **Default Value:** 0
- **ReadOnly:** True

ACC_TRIM (Accelerometer Trim (SITL))

DEFAULT: 0

Accelerometer Trim (ACC_TRIM)

Description

Simulated accelerometer trim used in the SITL environment to test trim estimation logic.

Tuning & Behavior

- **Default Value:** 0

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





ACROBATIC MODE CONFIGURATION (ACRO)

Overview

The **ACRO** parameter group governs the behavior of the **Acro** flight mode (also known as Rate Mode). In this mode, the pilot directly controls the **angular velocity** (rotation rate) of the vehicle. When the sticks are centered, the vehicle holds its current attitude (angle), not level.

- **For Copters:** This is the primary mode for FPV racing, freestyle, and precise cinematic movements. It feels "locked in" and requires constant pilot input to hover.
- **For Planes:** This mode acts like a "Fly-By-Wire Rate" mode. The control surfaces move to produce a commanded rotation rate. If you let go of the stick, the plane stops rotating and holds that bank/pitch angle (stabilized), counteracting turbulence.

Key Concepts

1. Rate Command (`ACRO_RP_RATE` / `ACRO_ROLL_RATE`)

This defines the responsiveness of the vehicle.

- **Definition:** The maximum rotation rate (in degrees per second) achieved at full stick deflection.
- **Impact:** Higher values make the vehicle snap and roll faster (snappier). Lower values make it smoother and more docile (cinematic).
- **Copter:** `ACRO_RP_RATE` controls both Roll and Pitch rates. `ACRO_Y_RATE` controls Yaw.
- **Plane:** `ACRO_ROLL_RATE` and `ACRO_PITCH_RATE` are often independent.

2. Expo (`ACRO_EXPO`)

Exponential (Expo) softens the stick response around the center without reducing the maximum rate at the endpoints.

- **Positive Expo:** Makes fine adjustments near the center stick easier while still allowing fast flips at full stick.
- **Linear (0):** Direct mapping of stick position to rate command.

3. Acro Trainer (`ACRO_TRAINER`)

A safety feature for learning Acro.

- **Function:** It prevents the pilot from flipping the vehicle completely over. If the vehicle reaches a certain bank angle limit (e.g., 45 degrees), the autopilot overrides the rate command to stop further rotation.
- **Levels:**
  - 0: Disabled (Full freedom).
  - 1: Leveling (Automatically returns to level when sticks are released).
  - 2: Limited (Stops at angle limits, holds attitude when released).

4. Heading Lock (`ACRO_LOCKING`)

Determines whether the autopilot aggressively fights to hold a specific heading/attitude when the sticks are centered.

- **Locked:** The integrator builds up to correct any uncommanded drift (wind, CG shift).
- **Unlocked/Drift:** The vehicle acts more like a raw mechanical machine, allowing slight drift which can feel more "natural" for airplanes in smooth turns.

Parameter Breakdown

Copter Specific

- `ACRO_RP_RATE`: Maximum Roll/Pitch rate (e.g., 360 deg/s).
- `ACRO_Y_RATE`: Maximum Yaw rate.
- `ACRO_RP_EXPO`: Exponential curve for Roll/Pitch.
- `ACRO_THR_MID`: Throttle stick position that corresponds to "Hover Throttle" (only used for Expo calculation on throttle).

Plane Specific

- `ACRO_ROLL_RATE`: Maximum Roll rate (deg/s).
- `ACRO_PITCH_RATE`: Maximum Pitch rate (deg/s).
- `ACRO_LOCKING`: Enable/Disable strict attitude locking (1=Locked, 0=Rate Only).

Integration Guide

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Tuning for FPV Racing (Copter)

1. **Set Rates:** Start with `ACRO_RP_RATE` at 360 deg/s. Pros often use 600-900 deg/s.
2. **Add Expo:** Increase `ACRO_RP_EXPO` (e.g., 0.1 to 0.3) to regain precision for small movements.
3. **Disable Trainer:** Ensure `ACRO_TRAINER` is 0 for full flips and rolls.

Tuning for Smooth Flight (Plane)

1. **Check Mechanical Limits:** Ensure `ACRO_ROLL_RATE` is achievable by your airframe. Commanding 360 deg/s on a slow trainer will just saturate the servos (integral windup).
2. **Locking:** Use `ACRO_LOCKING = 1` for a "fly on rails" feel. Use `0` if you want the nose to drop naturally in turns.

Developer Notes

- **Code Locations:**
  - **Copter:** `ArduCopter/mode_acro.cpp` - Implements the `ModeAcro` class.
  - **Plane:** `ArduPlane/mode_acro.cpp` - Implements the fixed-wing rate controller logic.
- **Logic:** The pilot input is converted to a `target_rate` vector. This is fed into the PID controller (`ATC_RAT_RLL`, etc.) which compares `target_rate` vs `gyro_rate` to drive the motors/servos.

ACRO_BAL_PITCH (Acro Balance Pitch)

```
DEFAULT: 1.0
```

Acro Balance Pitch (ACRO_BAL_PITCH)

Description

Determines the speed of pitch leveling in ACRO mode (when trainer mode is active).

Tuning & Behavior

- **Default Value:** 1.0

ACRO_BAL_ROLL (Acro Balance Roll)

```
DEFAULT: 1.0
```

Acro Balance Roll (ACRO_BAL_ROLL)

Description

Determines the speed of roll leveling in ACRO mode (when trainer mode is active).

Tuning & Behavior

- **Default Value:** 1.0

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





ACRO_EXPO (Acro Mode Exponential)

DEFAULT: 0.3	RANGE: 0 1.0

Acro Mode Exponential (ACRO_EXPO)

Description

`ACRO_EXPO` makes the drone feel less "twitchy" when the control sticks are near the center.

In **ACRO** mode, the drone rotates based on stick position. Without Expo (0), the response is linear—small movements result in immediate rotation. This can make steady flight difficult. Setting an Expo value (like **0.3**) flattens the curve near the center, providing fine control for precision flying, while still allowing for full-speed flips and rolls when the stick is pushed to the edge.

Tuning & Behavior

- **0:** Linear response (Very sensitive).
- **0.3 (Default):** Standard aerobatic feel.
- **0.5+:** Very "soft" center stick, very aggressive at the edges.
- **Recommendation:** Start at **0.3** and increase if you find yourself accidentally over-correcting during smooth flight.

ACRO_LOCKING (Acro Mode Attitude Locking)

DEFAULT: 0	RANGE: 0 1

Acro Mode Attitude Locking (ACRO_LOCKING)

Description

`ACRO_LOCKING` (also known as "Stick Locking") helps the drone stay exactly where you point it.

ACRO_OPTIONS (Acro mode options)

DEFAULT: 0

Acro mode options (ACRO_OPTIONS)

Description

Bitmask for configuring specific Acro mode behaviors like Air-mode.

Tuning & Behavior

- **Default Value:** 0
- **Bitmask:**
  - 0: Air-mode (Enables mixer at all times)
  - 1: Rate Loop Only (Disables angle stabilization completely)

ACRO_PITCH_RATE (ACRO mode pitch rate)

DEFAULT: 180	RANGE: 10 500	UNITS: deg/s

ACRO mode pitch rate (ACRO_PITCH_RATE)

Description

Maximum angular velocity for pitch in Plane ACRO mode.

Tuning & Behavior

- **Default Value:** 180 deg/s
- **Range:** 10 to 500 deg/s

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





ACRO_ROLL_RATE (ACRO mode roll rate)

DEFAULT: 180	RANGE: 10 500	UNITS: deg/s

ACRO mode roll rate (ACRO_ROLL_RATE)

Description

Maximum angular velocity for roll in Plane ACRO mode.

Tuning & Behavior

- **Default Value:** 180 deg/s
- **Range:** 10 to 500 deg/s

ACRO_RP_EXPO (ACRO roll and pitch expo)

DEFAULT: 0.3	RANGE: -0.5 1.0

ACRO roll and pitch expo (ACRO_RP_EXPO)

Description

Exponential curve applied to roll and pitch inputs in ACRO mode, providing fine control near the center while maintaining high rates at the stick extremes.

Tuning & Behavior

- **Default Value:** 0.3
- **Range:** -0.5 to 1.0
- **Higher values make the center of the stick less sensitive.**

ACRO_RP_P (Acro Mode Roll/Pitch Gain)

DEFAULT: 4.5	RANGE: 1.0 10.0

Acro Mode Roll/Pitch Gain (ACRO_RP_P)

Description

`ACRO_RP_P` defines how fast the drone flips and rolls when in **Acro** mode.

A higher value increases the rotation rate for a given stick deflection. This is the primary parameter for adjusting the "Feel" of an FPV racing or freestyle drone.

Tuning & Behavior

- **Default:** 4.5.
- **Usage:** For smooth cinematic flight in Acro, use a lower value (**2.0 to 3.0**). For aggressive freestyle, use a higher value (**6.0+**).
- **Note:** This is functionally similar to "Rates" in Betaflight.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





ACRO_RP_RATE (ACRO roll and pitch rate)

DEFAULT: 360	RANGE: 1 360	UNITS: deg/s

ACRO roll and pitch rate (ACRO_RP_RATE)

Description

Maximum angular velocity achievable in roll and pitch when in ACRO mode.

Tuning & Behavior

- **Default Value:** 360 deg/s
- **Range:** 1 to 360 deg/s

ACRO_RP_RATE_TC (ACRO roll and pitch rate time constant)

DEFAULT: 0.1	RANGE: 0 1	UNITS: s

ACRO roll and pitch rate time constant (ACRO_RP_RATE_TC)

Description

Smoothing factor for roll and pitch rate commands in ACRO mode.

Tuning & Behavior

- **Default Value:** 0.1 s
- **Range:** 0 to 1 s
- **Values:** 0.5:Very Soft, 0.2:Soft, 0.15:Medium, 0.1:Crisp, 0.05:Very Crisp

ACRO_THR_MID (Acro Thr Mid)

DEFAULT: 0	RANGE: 0 1

Acro Thr Mid (ACRO_THR_MID)

Description

Defines the mid-throttle position specific to ACRO mode.

Tuning & Behavior

- **Default Value:** 0
- **Range:** 0 1

ACRO_TRAINER (ACRO trainer)

DEFAULT: 2	RANGE: 0 2

ACRO trainer (ACRO_TRAINER)

Description

Assistance mode for ACRO flight, allowing for automatic leveling or angle limiting.

Tuning & Behavior

- **Default Value:** 2 (Limited)
- **Values:** 0:Disabled, 1:Leveling, 2:Limited

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





ACRO_TURN_RATE (ACRO mode turn rate)

DEFAULT: 0	RANGE: 10 500	UNITS: deg/s

ACRO mode turn rate (ACRO_TURN_RATE)

Description

Maximum rate of heading change in ACRO mode.

Tuning & Behavior

- **Default Value:** 0 (Plane), 180 (Rover)
- **Range:** 10 to 500 deg/s

ACRO_YAW_P (Acro Mode Yaw Gain)

DEFAULT: 4.5	RANGE: 1.0 10.0

Acro Mode Yaw Gain (ACRO_YAW_P)

Description

`ACRO_YAW_P` defines the "Sensitivity" of the yaw stick while flying in **Acro** mode.

In Acro mode, your sticks do not control the drone's angle; they control its *rate of rotation*. This parameter sets the multiplier for that command. A higher value means the drone will spin faster for the same amount of stick movement.

Tuning & Behavior

- **Default:** 4.5.
- **Effect of Increasing:** The drone feels "faster" and more responsive on the yaw axis.
- **Recommendation:** If you find the drone rotates too fast to control precisely during stunts, decrease this value to **3.0 or 2.0**.

ACRO_YAW_RATE (ACRO mode yaw rate)

DEFAULT: 0	RANGE: 0 500	UNITS: deg/s

ACRO mode yaw rate (ACRO_YAW_RATE)

Description

Maximum angular velocity for yaw in Plane ACRO mode.

Tuning & Behavior

- **Default Value:** 0 (Rudder directly controlled by stick)
- **Range:** 0 to 500 deg/s
- **Requires YAW_RATE_ENABLE = 1 to take effect.**

ACRO_Y_EXPO (ACRO yaw expo)

DEFAULT: 0	RANGE: -0.5 1.0

ACRO yaw expo (ACRO_Y_EXPO)

Description

Exponential curve applied to yaw inputs in ACRO mode.

Tuning & Behavior

- **Default Value:** 0
- **Range:** -0.5 to 1.0

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





ACRO_Y_RATE (ACRO yaw rate)

DEFAULT: 200	RANGE: 1 360	UNITS: deg/s

ACRO yaw rate (ACRO_Y_RATE)

Description

Maximum angular velocity achievable in yaw when in ACRO mode.

Tuning & Behavior

- **Default Value:** 200 deg/s
- **Range:** 1 to 360 deg/s

ACRO_Y_RATE_TC (ACRO yaw rate time constant)

DEFAULT: 0.1	RANGE: 0 1	UNITS: s

ACRO yaw rate time constant (ACRO_Y_RATE_TC)

Description

Smoothing factor for yaw rate commands in ACRO mode.

Tuning & Behavior

- **Default Value:** 0.1 s
- **Range:** 0 to 1 s
- **Values:** 0.5:Very Soft, 0.2:Soft, 0.15:Medium, 0.1:Crisp, 0.05:Very Crisp

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





ACTUATOR / ANALOG SENSOR CONFIGURATION (ACT)

Overview

The **ACT** parameter group is primarily associated with the **AD7091R5** Analog-to-Digital Converter (ADC) driver, commonly used on **DroneCAN Peripheral** nodes (such as the Zubax GNSS or specialized power modules) for high-precision battery monitoring.

While the prefix "ACT" suggests "Actuator," in the current ArduPilot ecosystem, these parameters are almost exclusively used to calibrate the analog sensing of Voltage and Current on distributed power systems.

Key Concepts

1. Analog Scaling (`ACT_VOLT_MULT` / `ACT_AMP_PERVLT`)

To convert the raw ADC voltage (0-3.3V or 0-5V) into meaningful real-world units (Battery Volts and Amps), linear scaling factors are applied.

- **Multiplier:** Converts sensor logic voltage to battery voltage.
- **Amps per Volt:** Converts the current sensor's output voltage to Amperes.

2. Zero-Point Calibration (`ACT_AMP_OFFSET`)

Current sensors (especially Hall Effect sensors) often output a non-zero voltage even when 0 Amps are flowing. This "Offset" must be subtracted to prevent the autopilot from integrating phantom current consumption (mAh) while the vehicle is sitting idle.

3. Peripheral Telemetry (`ACT_TELEM_RATE`)

Controls how frequently the peripheral node broadcasts its status (or actuator feedback) over the CAN bus to the central flight controller.

Parameter Breakdown

- `ACT_AMP_OFFSET`: Voltage offset of the current sensor at 0 Amps.
- `ACT_AMP_PERVLT`: Sensitivity of the current sensor (Amps per sensed Volt).
- `ACT_VOLT_MULT`: Voltage divider multiplier.
- `ACT_CURR_MAX`: Maximum expected current (sanity check/scaling limit).
- `ACT_TELEM_RATE`: Frequency of status updates (Hz).

Integration Guide

Calibrating Power Modules

1. **Measure:** Use a multimeter to measure the actual battery voltage.
2. **Adjust:** Tweak `ACT_VOLT_MULT` until the GCS reported voltage matches the multimeter.
3. **Zero Current:** Disconnect the load (if possible) or ensure motors are off. Adjust `ACT_AMP_OFFSET` until the current reads ~0.0A.
4. **Load Test:** Apply a known load (or use a wattmeter) to calibrate `ACT_AMP_PERVLT`.

Developer Notes

- **Library:** `libraries/AP_BattMonitor` (specifically `AP_BattMonitor_AD7091R5`) and `Tools/AP_Periph`.
- **Context:** These parameters usually exist on the **Peripheral Node** itself, not the main Autopilot. You access them via the "DroneCAN/UAVCAN Parameters" menu in your GCS, not the main parameter list.
- **Naming:** The `ACT_` prefix is a specific choice for this driver in the `AP_Periph` firmware build to avoid conflicts with the main `BATT_` group.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





ACT_AMP_OFFSET (AMP offset)

DEFAULT: 0	UNITS: V

AMP offset (ACT_AMP_OFFSET)

Description

Voltage offset at zero current on current sensor.

Tuning & Behavior

- **Default Value:** 0 V

ACT_AMP_PERVLT (Amps per volt)

DEFAULT: 0	UNITS: A/V

Amps per volt (ACT_AMP_PERVLT)

Description

Number of amps that a 1V reading on the current sensor corresponds to.

Tuning & Behavior

- **Default Value:** 0 A/V

ACT_CURR_MAX (Actuator Current Max (Invalid/Legacy))

DEFAULT: 45	RANGE: 20 80	UNITS: deg

Actuator Current Max (ACT_CURR_MAX)

Description

This parameter is not found as ACT_CURR_MAX in the main codebase. It likely refers to Q_TILT_MAX (Tiltrotor maximum VTOL angle).

ACT_CURR_PIN (Battery Current sensing pin)

DEFAULT: 0

Battery Current sensing pin (ACT_CURR_PIN)

Description

Sets the analog input pin that should be used for Current monitoring on AD7091R5.

Tuning & Behavior

- **Default Value:** 0
- **Values:** -1:Disabled

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





ACT_CURR_PIN1 (Actuator Current Pin 1 (Invalid/Legacy))

DEFAULT: -1

Actuator Current Pin 1 (ACT_CURR_PIN1)

Description

This parameter is not found as ACT_CURR_PIN1 in the main codebase. It likely refers to BTN_PIN1 (First button Pin).

ACT_TELEM_RATE (Actuator Telemetry Rate (Invalid/Legacy))

DEFAULT: 0	UNITS: Hz

Actuator Telemetry Rate (ACT_TELEM_RATE)

Description

This parameter is not found as ACT_TELEM_RATE in the main codebase.

ACT_VLT_OFFSET (Volage offset)

DEFAULT: 0	UNITS: V

Volage offset (ACT_VLT_OFFSET)

Description

Voltage offset on voltage pin. This allows for an offset due to a diode. This voltage is subtracted before the scaling is applied.

Tuning & Behavior

- **Default Value:** 0 V

ACT_VOLT_MULT (Voltage Multiplier)

DEFAULT: 0

Voltage Multiplier (ACT_VOLT_MULT)

Description

Used to convert the voltage of the voltage sensing pin (ACT_VOLT_PIN) to the actual battery's voltage (pin_voltage * VOLT_MULT).

Tuning & Behavior

- **Default Value:** 0

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





ACT_VOLT_PIN (Battery Voltage sensing pin on the AD7091R5 Ic)

DEFAULT: 0

Battery Voltage sensing pin on the AD7091R5 Ic (ACT_VOLT_PIN)

Description

Sets the analog input pin that should be used for voltage monitoring on AD7091R5.

Tuning & Behavior

- **Default Value:** 0
- **Values:** -1:Disabled

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY

Overview

The **ADSB** parameter group configures the **Automatic Dependent Surveillance–Broadcast** (ADS-B) subsystem. This system allows the autopilot to:

1. **Receive (ADS-B In):** Detect nearby manned aircraft equipped with ADS-B transmitters. This data is fed into the **AVOID** library for potential collision avoidance actions.
2. **Transmit (ADS-B Out):** Broadcast the drone's position, altitude, and velocity to air traffic control and other aircraft (requires a capable transceiver like the uAvionix Ping2020 or CubeOrange+ ADSB carrier board).

Key Concepts

1. Hardware Selection (`ADSB_TYPE`)

Tells ArduPilot which driver to use for the connected hardware.

- **1 (uAvionix):** For Ping/PingRX modules communicating via the uAvionix serial protocol.
- **2 (MAVLink):** For devices that send standard ADSB_VEHICLE MAVLink messages (e.g., OpenOpener, some remote IDs).
- **3 (Sagetech):** For Sagetech transponders.

2. Vehicle Identification (`ADSB_ICAO_ID` / `ADSB_SQUAWK`)

For "ADS-B Out" capable hardware, these parameters define the aircraft's identity.

- **ICAO ID:** The unique 24-bit airframe address assigned by your aviation authority (FAA, EASA, etc.). **Do not make this up.**
- **Squawk:** The 4-digit octal transponder code (usually 1200 for VFR in the US, 7000 in Europe).
- **Callsign:** Often set via a separate parameter or string (`ADSB_CALLSIGN` or `FLIGHT_ID`).

3. Emitter Category (`ADSB_EMIT_TYPE`)

Defines the "Wake Turbulence Category" or type of vehicle broadcasted in the ADS-B message.

- **14:** UAV (Unmanned Aerial Vehicle). This is the correct setting for most drones.

4. RF Configuration (`ADSB_RF_SELECT`)

Configures the radio receiver sensitivity (UAT 978MHz vs 1090ES).

- **RX_ENABLED:** Enables receiving.
- **UAT_RX:** Enable 978MHz reception (US general aviation below 18k ft).
- **1090_RX:** Enable 1090MHz reception (Commercial airliners, international).

Parameter Breakdown

- `ADSB_ENABLE`: Master switch for the library.
- `ADSB_TYPE`: Driver selection.
- `ADSB_BAUDRATE`: Serial port speed (usually 57600).
- `ADSB_LIST_MAX`: Maximum number of tracked targets held in RAM.
- `ADSB_LIST_RADIUS`: Detection radius (meters) for tracking targets. Targets outside this are ignored to save memory.
- `ADSB_LOG`: Log detected targets to the DataFlash log (ADSB message).

Integration Guide

ADS-B In Setup (PingRX)

1. Connect PingRX to a Telem port.
2. Set `SERIALx_PROTOCOL` to 2 (MAVLink 2) or specific ADSB protocol if required by driver logic (rare, usually auto-detected or mapped to `ADSB_TYPE`).
3. Set `ADSB_TYPE = 1` (uAvionix).
4. Reboot.
5. Verify detections in GCS "Traffic" map.

ADS-B Out Setup

1. Requires GPS lock and valid time.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





2. Set `ADSB_ICAO_ID` to your assigned hex code.
3. Set `ADSB_EMIT_TYPE` to 14.
4. **Legal Warning:** Transmitting invalid ADS-B data is a federal offense in many jurisdictions. Ensure your ICAO ID is registered.

Developer Notes

- **Library:** `libraries/AP_ADSB`
- **Singleton:** `AP_ADSB::get_singleton()`
- **Interaction:** Feeds data into `AP_Avoidance` for "Simple Avoidance" or "ADSB Avoidance" behaviors (climb/descend/loiter).

ADSB_BAUDRATE (ADSB serial baudrate)

DEFAULT: 57600

ADSB serial baudrate (ADSB_BAUDRATE)

Description

This parameter sets the communication speed (baud rate) for the serial port on the DroneCAN peripheral that is connected to an ADS-B receiver (like the uAvionix PingRX).

The Mathematics

Standard Serial Baud Rates (57600, 115200, etc.).

The Engineer's View

Defined in `Tools/AP_Periph/Parameters.cpp`. Configures the UART attached to the ADS-B driver stack on the peripheral node.

Tuning & Behavior

- **Default Value:** 57600
- **Match Hardware:** Must match the fixed baud rate of your ADS-B receiver.

ADSB_EMIT_TYPE (Emitter type)

DEFAULT: 14

Emitter type (ADSB_EMIT_TYPE)

Description

ADSB classification for the vehicle type.

Tuning & Behavior

- **Default Value:** 14 (UAV)
- **Values:** 0:NoInfo, 1:Light, 2:Small, 3:Large, 4:HighVortexlarge, 5:Heavy, 6:HighlyManuv, 7:Rotocraft, 8:RESERVED, 9:Glider, 10:LightAir, 11:Parachute, 12:UltraLight, 13:RESERVED, 14:UAV, 15:Space, 16:RESERVED, 17:EmergencySurface, 18:ServiceSurface, 19:PointObstacle

ADSB_ICAO_ID (ICAO_ID vehicle identification number)

DEFAULT: 0	RANGE: -1 16777215

ICAO_ID vehicle identification number (ADSB_ICAO_ID)

Description

Unique identifier for this aircraft in the ADS-B system.

Tuning & Behavior

- **Default Value:** 0 (Randomly generated)
- **Range:** -1 to 16777215
- **-1 means static information is not sent (pre-programmed transceiver).**

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





ADSB_ICAO_SPECL (ICAO_ID of special vehicle)

DEFAULT: 0

ICAO_ID of special vehicle (ADSB_ICAO_SPECL)

Description

Enables tracking of a specific vehicle by its ICAO ID, bypassing distance and altitude filters.

Tuning & Behavior

- **Default Value:** 0 (Disabled)

ADSB_LEN_WIDTH (Aircraft length and width)

DEFAULT: 1

Aircraft length and width (ADSB_LEN_WIDTH)

Description

Physical dimensions of the aircraft for ADS-B reporting.

Tuning & Behavior

- **Default Value:** 1 (L15W23)
- **Values:** 0:NO_DATA, 1:L15W23, 2:L25W28P5, 3:L25W34, 4:L35W33, 5:L35W38, 6:L45W39P5, 7:L45W45, 8:L55W45, 9:L55W52, 10:L65W59P5, 11:L65W67, 12:L75W72P5, 13:L75W80, 14:L85W80, 15:L85W90

ADSB_LIST_ALT (ADSB vehicle list altitude filter)

DEFAULT: 0	RANGE: 0 32767	UNITS: m

ADSB vehicle list altitude filter (ADSB_LIST_ALT)

Description

Altitude filter for ADSB-detected vehicles.

Tuning & Behavior

- **Default Value:** 0 m
- **Range:** 0 32767
- **A value of 0 disables the filter.**

ADSB_LIST_MAX (ADSB vehicle list size)

DEFAULT: 25	RANGE: 1 100

ADSB vehicle list size (ADSB_LIST_MAX)

Description

Maximum number of vehicles to track in the ADSB list.

Tuning & Behavior

- **Default Value:** 25
- **Range:** 1 100

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





ADSB_LIST_RADIUS (ADSB vehicle list radius filter)

DEFAULT: 2000	RANGE: 0 100000	UNITS: m

ADSB vehicle list radius filter (ADSB_LIST_RADIUS)

Description

Distance filter for ADSB-detected vehicles.

Tuning & Behavior

- **Default Value:** 2000 m (Copter/Rover), 10000 m (Plane)
- **Range:** 0 100000
- **A value of 0 disables the filter.**

ADSB_LOG (ADS-B logging)

DEFAULT: 1

ADS-B logging (ADSB_LOG)

Description

Controls logging of ADS-B data.

Tuning & Behavior

- **Default Value:** 1 (Special ID Only)
- **Values:** 0:no logging, 1:log only special ID, 2:log all

ADSB_OFFSET_LAT (GPS antenna lateral offset)

DEFAULT: 4

GPS antenna lateral offset (ADSB_OFFSET_LAT)

Description

Physical location offset from the center of the GPS antenna on the aircraft.

Tuning & Behavior

- **Default Value:** 4 (Center)
- **Values:** 0:NoData, 1:Left2m, 2:Left4m, 3:Left6m, 4:Center, 5:Right2m, 6:Right4m, 7:Right6m

ADSB_OFFSET_LON (GPS antenna longitudinal offset)

DEFAULT: 1

GPS antenna longitudinal offset (ADSB_OFFSET_LON)

Description

Physical location offset along the longitudinal axis of the GPS antenna.

Tuning & Behavior

- **Default Value:** 1 (AppliedBySensor)
- **Values:** 0:NO_DATA, 1:AppliedBySensor

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





ADSB_OPTIONS (ADS-B Options)

DEFAULT: 0

ADS-B Options (ADSB_OPTIONS)

Description

Configuration options for ADS-B behavior and failsafes.

Tuning & Behavior

- **Default Value:** 0
- **Bitmask:** 0:Ping200X Send GPS, 1:Squawk 7400 on RC failsafe, 2:Squawk 7400 on GCS failsafe, 3:Sagetech MXS use External Config

ADSB_PORT (ADSB Serial Port (Invalid/Legacy))

DEFAULT: 0

ADSB Serial Port (ADSB_PORT)

Description

This parameter is not found in the current ArduPilot ADS-B library. Hardware serial mapping is handled via SERIALx_PROTOCOL.

ADSB_RF_CAPABLE (RF capabilities)

DEFAULT: 0

RF capabilities (ADSB_RF_CAPABLE)

Description

Description of the transceiver's hardware RF capabilities.

Tuning & Behavior

- **Default Value:** 0
- **Bitmask:** 0:UAT_in, 1:1090ES_in, 2:UAT_out, 3:1090ES_out

ADSB_RF_SELECT (Transceiver RF selection)

DEFAULT: 1

Transceiver RF selection (ADSB_RF_SELECT)

Description

Selection of Rx/Tx modes for the ADSB transceiver.

Tuning & Behavior

- **Default Value:** 1 (Rx Enabled)
- **Bitmask:** 0:Rx, 1:Tx

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





ADSB_SQUAWK (Squawk code)

DEFAULT: 1200	RANGE: 0 7777	UNITS: octal

Squawk code (ADSB_SQUAWK)

Description

Pre-programmed default VFR squawk code.

Tuning & Behavior

- **Default Value:** 1200
- **Range:** 0 to 7777 (Octal)

ADSB_TYPE (ADSB Type)

DEFAULT: 0

ADSB Type (ADSB_TYPE)

Description

Type of ADS-B hardware.

Tuning & Behavior

- **Default Value:** 0 (Disabled)
- **Values:** 0:Disabled, 1:uAvionix-MAVLink, 2:Sagetech, 3:uAvionix-UCP, 4:Sagetech MX Series

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **AFS** parameter group configures the **Advanced Failsafe** library. This subsystem is distinct from the standard `FS_` (Failsafe) parameters found in Plane/Copter. It is designed to meet strict **Flight Termination System (FTS)** requirements for regulatory compliance or competitions (e.g., UAV Challenge Outback Rescue).

Unlike standard failsafes which try to save the vehicle (RTL, Land), **AFS is often designed to "Terminate" the flight** to ensure safety of people on the ground if the vehicle goes rogue.

Key Concepts

1. Flight Termination (`AFS_TERMINATE`)

The core function of AFS. If a critical failure condition is met (RC Loss, GCS Loss, Geofence Breach), the system can be configured to:

- **Kill Motors:** Immediately stop propulsion.
- **Move Servos:** Force control surfaces to a "Crash" or "Spin" position (e.g., full rudder, full up elevator) to limit the impact area.
- **Hardware Pin:** Drive a GPIO pin (`AFS_TERM_PIN`) high/low to trigger an external parachute deployment module or airbag.

2. Dual-Link Monitoring

AFS can monitor both the RC link and the GCS (MAVLink) link independently (`AFS_DUAL_LOSS`). It ensures that you always have at least one valid command link.

3. Heartbeat Hardware (`AFS_HB_PIN`)

Can output a "heartbeat" signal to an external watchdog timer. If the autopilot freezes, the heartbeat stops, and the external hardware terminates the flight.

Parameter Breakdown

- `AFS_ENABLE`: Master switch.
- `AFS_TERM_ACTION`: 0=Disable, 1=Terminate immediately on failure.
- `AFS_MAN_PIN`: Input pin for a manual "Kill Switch" from an external receiver.
- `AFS_TERM_PIN`: Output pin to trigger external termination hardware.
- `AFS_RC_FAIL_TIME`: Timeout (seconds) for RC loss before termination.
- `AFS_AMSL_LIMIT`: Hard altitude ceiling (AMSL) for termination.

Integration Guide

Typical Setup for FTS

1. **Enable:** Set `AFS_ENABLE = 1`.
2. **Define Output:** Set `AFS_TERM_PIN` to the Servo/GPIO connected to your parachute release or ignition kill.
3. **Define Logic:** Configure `AFS_RC_FAIL_TIME` and `AFS_GCS_TIMEOUT` appropriately.
4. **Testing: REMOVE PROPELLERS.** Use the `AFS.TERMINATE` MAVLink command or simulate a link loss to verify the pin switches state.

Developer Notes

- **Library:** `libraries/AP_AdvancedFailsafe`
- **Vehicle:** Primarily used in **ArduPlane**.
- **Warning:** Misconfiguring this WILL crash your plane. This is a "destruct" system, not a "save me" system.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





AFS_AMSL_ERR_GPS (Error margin for GPS based AMSL limit)

DEFAULT: -1	UNITS: m

Error margin for GPS based AMSL limit (AFS_AMSL_ERR_GPS)

Description

Safety margin for GPS-based altitude limiting when the barometer fails.

Tuning & Behavior

- **Default Value:** -1 m
- **-1 means barometer failure leads to immediate termination.**

AFS_AMSL_LIMIT (AMSL limit)

DEFAULT: 0	UNITS: m

AMSL limit (AFS_AMSL_LIMIT)

Description

Maximum altitude (Above Mean Sea Level) allowed before forced termination.

Tuning & Behavior

- **Default Value:** 0 m (Disabled)
- **Units are in meters.**

AFS_DUAL_LOSS (Enable dual loss terminate)

DEFAULT: 1

Enable dual loss terminate (AFS_DUAL_LOSS)

Description

Triggers termination if both GPS and GCS link are lost simultaneously.

Tuning & Behavior

- **Default Value:** 1 (Enabled)
- **Values:** 0:Disabled, 1:Enabled

AFS_ENABLE (Enable Advanced Failsafe)

DEFAULT: 0

Enable Advanced Failsafe (AFS_ENABLE)

Description

Enables the advanced failsafe system.

Tuning & Behavior

- **Default Value:** 0 (Disabled)
- **Values:** 0:Disabled, 1:Enabled

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





AFS_GCS_TIMEOUT (GCS timeout)

DEFAULT: 10	UNITS: s

GCS timeout (AFS_GCS_TIMEOUT)

Description

Duration of GCS link loss before triggering failsafe.

Tuning & Behavior

- **Default Value:** 10 s

AFS_GEOFENCE (Enable geofence Advanced Failsafe)

DEFAULT: 1

Enable geofence Advanced Failsafe (AFS_GEOFENCE)

Description

Enables geofence-triggered failsafes within the AFS system.

Tuning & Behavior

- **Default Value:** 1 (Enabled)
- **Requires AFS_ENABLE = 1 to function.**

AFS_HB_PIN (Heartbeat Pin)

DEFAULT: -1

Heartbeat Pin (AFS_HB_PIN)

Description

Digital output pin used for monitoring autopilot health via a 10Hz heartbeat.

Tuning & Behavior

- **Default Value:** -1 (Disabled)
- **Values:** -1:Disabled, 49:BB Blue GP0 pin 4, 50:AUXOUT1, 51:AUXOUT2, 52:AUXOUT3, 53:AUXOUT4, 54:AUXOUT5, 55:AUXOUT6, 57:BB Blue GP0 pin 3, 113:BB Blue GP0 pin 6, 116:BB Blue GP0 pin 5

AFS_MAN_PIN (Manual Pin)

DEFAULT: -1

Manual Pin (AFS_MAN_PIN)

Description

Digital output pin indicating Manual mode.

Tuning & Behavior

- **Default Value:** -1 (Disabled)

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





AFS_MAX_COM_LOSS (Maximum number of comms loss events)

DEFAULT: 0

Maximum number of comms loss events (AFS_MAX_COM_LOSS)

Description

Limit on how many times communication can be lost before the aircraft refuses to resume its mission.

Tuning & Behavior

- **Default Value:** 0 (Unlimited)

AFS_MAX_GPS_LOSS (Maximum number of GPS loss events)

DEFAULT: 0

Maximum number of GPS loss events (AFS_MAX_GPS_LOSS)

Description

Limit on how many times GPS signal can be lost before the aircraft refuses to resume its mission.

Tuning & Behavior

- **Default Value:** 0 (Unlimited)

AFS_MAX_RANGE (Max allowed range)

DEFAULT: 0	UNITS: km

Max allowed range (AFS_MAX_RANGE)

Description

Distance limit from the arming point, triggering termination if exceeded.

Tuning & Behavior

- **Default Value:** 0 km (Disabled)
- **Units are in kilometers.**

AFS_OPTIONS (AFS options)

DEFAULT: 0

AFS options (AFS_OPTIONS)

Description

Bitmask for configuring specific Advanced Failsafe behaviors.

Tuning & Behavior

- **Default Value:** 0
- **Bitmask:**
  - 0: Continue the mission even after comms are recovered (does not go to the mission item at the time comms were lost)
  - 1: Enable AFS for all autonomous modes (not just AUTO)
    Internal logic check: bit 0 skips returning to the exact WP where comms were lost; bit 1 extends protection beyond just the AUTO mission mode.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





AFS_QNH_PRESSURE (QNH pressure)

DEFAULT: 0	UNITS: hPa

QNH pressure (AFS_QNH_PRESSURE)

Description

Reference sea-level pressure for barometric altitude limit calculation.

Tuning & Behavior

- **Default Value:** 0 hPa (Disabled)
- **Must be set correctly for AMSL limiting to be accurate.**

AFS_RC (Enable RC Advanced Failsafe)

DEFAULT: 1

Enable RC Advanced Failsafe (AFS_RC)

Description

Enables RC-link loss detection and associated actions within the AFS system.

Tuning & Behavior

- **Default Value:** 1 (Enabled)
- **Requires AFS_ENABLE = 1 to function.**

AFS_RC_FAIL_TIME (RC failure time)

DEFAULT: 0	UNITS: s

RC failure time (AFS_RC_FAIL_TIME)

Description

Duration of RC link loss in manual mode before forced termination.

Tuning & Behavior

- **Default Value:** 0 s (Disabled)
- **Recommended for OBC rules: 1.5 seconds.**

AFS_RC_MAN_ONLY (Enable RC Termination only in manual control modes)

DEFAULT: 1

Enable RC Termination only in manual control modes (AFS_RC_MAN_ONLY)

Description

Restricts RC-loss termination to manual flight modes only.

Tuning & Behavior

- **Default Value:** 1 (Manual Only)
- **Values:** 0:All Modes, 1:Manual Modes Only

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





AFS_TERMINATE (Force Terminate)

DEFAULT: 0

Force Terminate (AFS_TERMINATE)

Description

Software switch to immediately trigger vehicle termination.

Tuning & Behavior

- **Default Value:** 0 (Normal)
- **Setting this to 1 in flight will terminate the vehicle.**

AFS_TERM_ACTION (Terminate action)

DEFAULT: 0

Terminate action (AFS_TERM_ACTION)

Description

Specific vehicle behavior to execute when termination is triggered.

Tuning & Behavior

- **Default Value:** 0
- **Common values include crashing or immediate landing.**

AFS_TERM_PIN (Terminate Pin)

DEFAULT: -1

Terminate Pin (AFS_TERM_PIN)

Description

Digital output pin triggered high when the flight is terminated.

Tuning & Behavior

- **Default Value:** -1 (Disabled)
- **Values:** -1:Disabled, 49:BB Blue GP0 pin 4, 50:AUXOUT1, 51:AUXOUT2, 52:AUXOUT3, 53:AUXOUT4, 54:AUXOUT5, 55:AUXOUT6, 57:BB Blue GP0 pin 3, 113:BB Blue GP0 pin 6, 116:BB Blue GP0 pin 5

AFS_WP_COMMS (Comms Waypoint)

DEFAULT: 0

Comms Waypoint (AFS_WP_COMMS)

Description

Waypoint designated for navigation when communication link is lost.

Tuning & Behavior

- **Default Value:** 0 (None)

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





AFS_WP_GPS_LOSS (GPS Loss Waypoint)

DEFAULT: 0

GPS Loss Waypoint (AFS_WP_GPS_LOSS)

Description

Waypoint designated for navigation when GPS signal is lost.

Tuning & Behavior

- **Default Value:** 0 (None)

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





ATTITUDE HEADING REFERENCE SYSTEM (AHRS)

Overview

The **AHRS** (Attitude Heading Reference System) parameters control the fundamental orientation logic of the autopilot. This library is responsible for determining "Which way is Up" and "Which way is North."

While the modern **Extended Kalman Filter (EKF)** does the heavy lifting for state estimation, the AHRS library manages:

1. **Board Orientation:** Telling the software how the flight controller is mounted relative to the vehicle frame.
2. **Estimator Selection:** Choosing between EKF2, EKF3, or the legacy DCM (Direction Cosine Matrix) filter.
3. **Fallback Logic:** Providing a backup attitude solution if the EKF fails or diverges.

Key Concepts

1. Board Orientation (`AHRS_ORIENTATION`)

This is the most critical parameter in setup. If this is wrong, the vehicle will flip over instantly on takeoff.

- It defines the rotation of the flight controller board relative to the vehicle's forward direction.
- **Example:** If the USB port points to the right wing, you likely need `YAW_90`.

2. Estimator Type (`AHRS_EKF_TYPE`)

ArduPilot supports multiple estimator backends.

- **0:** Disabled (DCM only). Not recommended for modern GPS flight.
- **2:** EKF2 (Legacy, 22-state). Stable, widely used in older versions.
- **3:** EKF3 (Modern, 24-state). Supports newer features like Moving Baseline Yaw, Optical Flow, and Visual Odometry. **Default for 4.1+.**

3. Custom Rotations

If standard 90-degree increments are insufficient (e.g., mounting on a 45-degree arm), the `AHRS_CUSTOM_ROLL`, `PITCH`, and `YAW` parameters (or `CUSTOM_ORIENTATION` group) allow for precise alignment.

Parameter Breakdown

- `AHRS_ORIENTATION`: Board rotation enum (0=None, 2=Yaw90, 8=Roll180, etc.).
- `AHRS_EKF_TYPE`: Active state estimator selector.
- `AHRS_GPS_USE`: Determines if the legacy DCM filter is allowed to use GPS for correction.
- `AHRS_TRIM_X` / `Y`: (Radians) Internal trim states for level horizon. Usually set via "Calibrate Level".

Integration Guide

Setting Orientation

1. **Check Arrow:** Look for the arrow on your flight controller case.
2. **Mount:** Secure the board.
3. **Configure:** If the arrow points Forward, `AHRS_ORIENTATION = 0`. If Right, `AHRS_ORIENTATION = 2` (Yaw 90).
4. **Verify:** Open the HUD in your GCS. Tilt the vehicle nose DOWN. The HUD horizon should move UP. Tilt Right, HUD moves Left. **If it moves incorrectly, your orientation is wrong.**

Developer Notes

- **Library:** `libraries/AP_AHRS`
- **Classes:** `AP_AHRS_DCM`, `AP_AHRS_NavEKF`.
- **Redundancy:** The AHRS library often runs the DCM filter in the background as a "sanity check" for the EKF. If the EKF "blows up" (diverges), the system may failover to the DCM (Msg: "EKF Fail, switching to DCM").

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





AHRS_COMP_BETA (AHRS Velocity Complementary Filter Beta Coefficient)

DEFAULT: 0.1	RANGE: 0.001 0.5

AHRS Velocity Complementary Filter Beta Coefficient (AHRS_COMP_BETA)

Description

Controls the weighting between GPS data and air data for ground velocity estimation.

Tuning & Behavior

- **Default Value:** 0.1
- **Range:** 0.001 to 0.5
- **A larger beta value increases reliance on air data (airspeed/heading).**
- **A smaller beta value increases reliance on GPS data.**

AHRS_CUSTOM_ROLL (Board orientation roll offset)

DEFAULT: 0	RANGE: -180 180	UNITS: deg

Board orientation roll offset (AHRS_CUSTOM_ROLL)

Description

Fine-tuning for the autopilot mounting angle in the roll axis.

Tuning & Behavior

- **Default Value:** 0 deg
- **Range:** -180 to 180 deg
- **Only active if AHRS_ORIENTATION is set to 100 (Custom).**

AHRS_CUSTOM_YAW (Board orientation yaw offset)

DEFAULT: 0	RANGE: -180 180	UNITS: deg

Board orientation yaw offset (AHRS_CUSTOM_YAW)

Description

Fine-tuning for the autopilot mounting angle in the yaw axis.

Tuning & Behavior

- **Default Value:** 0 deg
- **Range:** -180 to 180 deg
- **Only active if AHRS_ORIENTATION is set to 100 (Custom).**

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





AHRS_EKF_TYPE (Use NavEKF Kalman filter version)

DEFAULT: 3

Use NavEKF Kalman filter version (AHRS_EKF_TYPE)

Description

Selects the primary Extended Kalman Filter (EKF) version for flight estimation.

Tuning & Behavior

- **Default Value:** 3 (EKF3)
- **Values:** 0:Disabled, 2:Enable EKF2, 3:Enable EKF3, 11:ExternalAHRS
- **EKF3 is the modern standard for ArduPilot.**

AHRS_GPS_GAIN (AHRS GPS gain)

DEFAULT: 1.0	RANGE: 0.0 1.0

AHRS GPS gain (AHRS_GPS_GAIN)

Description

Weighting of GPS data for correcting the attitude estimate.

Tuning & Behavior

- **Default Value:** 1.0
- **Range:** 0.0 to 1.0
- **Critical for Plane safety during coordinated turns.**

AHRS_GPS_MINSATS (AHRS GPS Minimum satellites)

DEFAULT: 6	RANGE: 0 10

AHRS GPS Minimum satellites (AHRS_GPS_MINSATS)

Description

Threshold for GPS satellite count required for velocity-based attitude correction.

Tuning & Behavior

- **Default Value:** 6
- **Range:** 0 to 10

AHRS_GPS_USE (AHRS use GPS for DCM navigation)

DEFAULT: 1	RANGE: 0 2

AHRS use GPS for DCM navigation (AHRS_GPS_USE)

Description

Switch between GPS-aided navigation and pure dead-reckoning.

Tuning & Behavior

- **Default Value:** 1 (Use GPS for DCM position)
- **Values:** 0:Disabled, 1:Use GPS for DCM position, 2:Use GPS for DCM position and height

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





AHRS_OPTIONS (Optional AHRS behaviour)

DEFAULT: 0

Optional AHRS behaviour (AHRS_OPTIONS)

Description

Bitmask for advanced AHRS configuration, primarily controlling fallback logic.

Tuning & Behavior

- **Default Value:** 0
- **Bitmask:**
  - 0: DisableDCMFallbackFW
  - 1: DisableDCMFallbackVTOL
  - 2: DontDisableAirspeedUsingEKF

AHRS_ORIENTATION (Board Orientation)

DEFAULT: 0	RANGE: 0 43

Board Orientation (AHRS_ORIENTATION)

Description

Software rotation for the flight controller mounting orientation.

Tuning & Behavior

- **Default Value:** 0 (None)
- **Common Values:** 0:None, 1:Yaw45, 2:Yaw90, 4:Yaw180, 8:Roll180, 100:Custom
- **Requires reboot and re-leveling after change.**

AHRS_RP_P (AHRS RP_P)

DEFAULT: 0.2	RANGE: 0.1 0.4

AHRS RP_P (AHRS_RP_P)

Description

Proportional gain for roll and pitch attitude correction using accelerometer data.

Tuning & Behavior

- **Default Value:** 0.2
- **Range:** 0.1 to 0.4
- **Higher values correct tilt errors faster but are more sensitive to vibration.**

AHRS_WIND_MAX (Maximum wind)

DEFAULT: 0.0	RANGE: 0 127	UNITS: m/s

Maximum wind (AHRS_WIND_MAX)

Description

Safety limit for the discrepancy between groundspeed and airspeed, providing redundancy against airspeed sensor failure.

Tuning & Behavior

- **Default Value:** 0.0 m/s (Disabled)
- **Range:** 0 to 127 m/s

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





AHRS_YAW_P (Yaw P)

DEFAULT: 0.2	RANGE: 0.1 0.4

Yaw P (AHRS_YAW_P)

Description

Proportional gain for heading correction using compass or GPS data.

Tuning & Behavior

- **Default Value:** 0.2
- **Range:** 0.1 to 0.4
- **Higher values track heading sources more aggressively.**

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **AIRSPEED** parameter group (often aliased or related to `ARSPD_FBW_` parameters in the full parameter list) defines the **Flight Envelope** for ArduPlane's autonomous and assisted modes.

These limits are critical for the TECS (Total Energy Control System) controller. They tell the autopilot the safe range of speeds within which the aircraft can fly.

- **Too Slow:** Risk of Stall.
- **Too Fast:** Risk of Structural Failure (Vne) or Flutter.

Key Concepts

1. Minimum Airspeed (`AIRSPEED_MIN`)

This is the "Floor" speed. In auto-throttle modes (Auto, Cruise, FBWB), the autopilot will apply maximum throttle and pitch down if necessary to prevent the airspeed from dropping below this value.

- **Setting:** Should be set slightly above the stall speed of the airframe (e.g., 1.2x Stall Speed).

2. Maximum Airspeed (`AIRSPEED_MAX`)

This is the "Ceiling" speed. The autopilot will reduce throttle or pitch up to prevent exceeding this value.

- **Setting:** Should be set below the structural limit of the airframe.

3. Cruise Speed (`TRIM_ARSPD_CM` or `AIRSPEED_CRUISE`)

(Related parameter) The target speed the autopilot tries to maintain during normal flight. It must lie between MIN and MAX.

Parameter Breakdown

- `AIRSPEED_MIN`: Minimum allowed airspeed (m/s).
- `AIRSPEED_MAX`: Maximum allowed airspeed (m/s).

Integration Guide

Initial Tuning

1. **Stall Test (Manual):** Fly high in manual mode. Reduce throttle and pull back elevator until the nose drops (stall). Note the speed.
2. **Set MIN:** Set `AIRSPEED_MIN` to Stall Speed + 2-3 m/s.
3. **High Speed Test:** Fly fast. Note the speed where the airframe feels stable.
4. **Set MAX:** Set `AIRSPEED_MAX` to this safe top speed.

Developer Notes

- **Vehicle: ArduPlane** Only.
- **Interaction:** These limits directly constrain the `AP_TECS` speed demand.
- **Aliases:** In some firmware versions, these appear as `ARSPD_FBW_MIN` and `ARSPD_FBW_MAX`.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





AIRSPEED_MAX (Maximum Airspeed)

DEFAULT: 22	RANGE: 5 100	UNITS: m/s

Maximum Airspeed (AIRSPEED_MAX)

Description

Defines the upper bound for airspeed during autonomous flight. The flight controller will prioritize staying below this airspeed by pitching the nose up if necessary.

Tuning & Behavior

- **Default Value:** 22 m/s (Approx. 43 knots)
- **Range:** 5 to 100 m/s
- **Should be set below the Vne (Never Exceed Speed) of the airframe.**

AIRSPEED_MIN (Minimum Airspeed)

DEFAULT: 9	RANGE: 5 100	UNITS: m/s

Minimum Airspeed (AIRSPEED_MIN)

Description

Defines the lower bound for airspeed during autonomous flight. The flight controller will prioritize maintaining this airspeed by pitching the nose down if necessary.

Tuning & Behavior

- **Default Value:** 9 m/s (Approx. 17.5 knots)
- **Range:** 5 to 100 m/s
- **Must be set above the stall speed of the aircraft with a safe margin.**

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **AIS** parameter group configures the **Marine AIS** subsystem. This is the nautical equivalent of ADS-B. It allows ArduRover (Boat) to detect nearby commercial vessels equipped with AIS transmitters.

Key Concepts

1. Collision Avoidance

AIS data is fed into the **AVOID** library. If a vessel is detected on a collision course, the autopilot can warn the operator or autonomously steer away (if avoidance is configured).

2. Hardware

Typically requires a serial AIS receiver connected to a UART port.

Parameter Breakdown

- `AIS_TYPE`: 0=Disabled, 1=NMEA (Standard Serial AIS).
- `AIS_TIME_OUT`: Time (seconds) without a signal before a vessel is considered lost/gone.
- `AIS_LIST_MAX`: Maximum number of vessels to track in memory.

Integration Guide

1. **Hardware:** Connect NMEA 0183 AIS Receiver to a Serial Port.
2. **Serial Config:** Set `SERIALx_PROTOCOL = 40` (AIS). Baud rate usually 38400.
3. **Enable:** Set `AIS_TYPE = 1`.

Developer Notes

- **Library:** `libraries/AP_AIS`
- **Vehicle:** **ArduRover** (Boats).

AIS_LIST_MAX (AIS vessel list size)

DEFAULT: 25	RANGE: 1 100

AIS vessel list size (AIS_LIST_MAX)

Description

Maximum number of vessels to track in the AIS list.

Tuning & Behavior

- **Default Value:** 25
- **Range:** 1 to 100

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





AIS_LOGGING (AIS logging options)

DEFAULT: 6

AIS logging options (AIS_LOGGING)

Description

Controls which AIS messages are recorded in the data logs.

Tuning & Behavior

- **Default Value:** 6 (AIS_OPTIONS_LOG_UNSUPPORTED_RAW | AIS_OPTIONS_LOG_DECODED)
- **Bitmask:**
  - 0: Log all AIVDM messages
  - 1: Log only unsupported AIVDM messages
  - 2: Log decoded messages

AIS_TIME_OUT (AIS vessel time out)

DEFAULT: 600	RANGE: 1 2000	UNITS: s

AIS vessel time out (AIS_TIME_OUT)

Description

Duration after which a vessel is removed from the tracking list if no updates are received.

Tuning & Behavior

- **Default Value:** 600 s (10 minutes)
- **Range:** 1 to 2000 s

AIS_TYPE (AIS receiver type)

DEFAULT: 0

AIS receiver type (AIS_TYPE)

Description

Enables and selects the type of Automatic Identification System (AIS) receiver.

Tuning & Behavior

- **Default Value:** 0 (None)
- **Values:** 0:None, 1:NMEA AIVDM message

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **ALT** parameter group contains miscellaneous altitude-related settings.

Note: This group is a bit of a "catch-all" in some documentation sets.

- `ALT_OFFSET` is a core **ArduPlane** parameter for adjusting the mission altitude frame.
- `ALT_MIN` is typically associated with the **Proximity** library (`PRX_ALT_MIN`), defining the floor for obstacle avoidance.

Key Concepts

1. Altitude Offset (`ALT_OFFSET`)

(ArduPlane Only)
This parameter shifts the autopilot's internal reference for "Home Altitude" without actually moving the Home point.

- **Usage:** Useful for landing at a location that is at a different physical elevation than the takeoff point, or for adjusting altitude during repeated auto-landings in a circuit.
- **Math:** `Target_Alt = Mission_Alt + ALT_OFFSET`.

2. Proximity Floor (`ALT_MIN`)

(From `AP_Proximity`)
Defines the minimum altitude (AGL) required for the obstacle avoidance system to engage.

- **Purpose:** Prevents the ground itself from being detected as an "obstacle" during takeoff and landing, which could cause the vehicle to panic or refuse to land.

Parameter Breakdown

- `ALT_OFFSET`: Altitude offset in meters. Positive adds height to targets.
- `ALT_MIN`: Minimum altitude for proximity sensors to be active.

Developer Notes

- **Context:** `ALT_OFFSET` is defined in `ArduPlane/Parameters.cpp`. `ALT_MIN` is defined in `libraries/AP_Proximity`.

ALT_MIN (Proximity lowest altitude)

DEFAULT: 1.0	RANGE: 0 10	UNITS: m

Proximity lowest altitude (ALT_MIN)

Description

Defines the minimum altitude above ground (AGL) required for proximity sensors and avoidance logic to become active. This prevents false triggers from the ground during takeoff and landing.

Tuning & Behavior

- **Default Value:** 1.0 m
- **Range:** 0 to 10 m

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





ALT_OFFSET (Altitude offset)

DEFAULT: 0	UNITS: m

Altitude offset (ALT_OFFSET)

Description

A manual or GCS-provided offset added to the calculated barometric altitude. It is typically used to align the vehicle's altitude with a known ground reference.

Tuning & Behavior

- **Default Value:** 0 m
- **It is automatically reset to 0 when the barometer is calibrated on reboot.**

ALT_SLOPE_MIN (Glide slope minimum)

DEFAULT: 15	UNITS: m

Glide slope minimum (ALT_SLOPE_MIN)

Description

Defines the minimum vertical distance required between waypoints to trigger a linear glide slope (gradual climb/descent). For smaller changes, the aircraft may attempt an immediate altitude change.

Tuning & Behavior

- **Default Value:** 15 m
- **A value of 0 disables glide slope calculations, leading to immediate altitude setpoint changes.**

ALT_TYPE (Follow Altitude Type)

DEFAULT: 0

Follow Altitude Type (ALT_TYPE)

Description

Specifies the altitude reference frame used when in Follow mode.

Tuning & Behavior

- **Default Value:** 0 (AMSL)
- **Values:** 0:AMSL (Above Mean Sea Level), 1:Relative (Above Home)

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **ALTERR** parameter group (often appearing as `BARO_ALTERR_` in the full parameter list) configures the pre-flight safety checks related to altitude consistency.

Before allowing the vehicle to arm, the autopilot compares the altitude reported by the Barometer against the altitude reported by the GPS. Large discrepancies indicate a potential sensor failure, blockage (pitot/static port), or extreme weather changes, which could lead to dangerous behavior if the EKF (State Estimator) cannot reconcile the data.

Key Concepts

1. Altitude Discrepancy Check (`ALTERR_MAX`)

This parameter sets the threshold for the difference between the two altitude sources.

- **Logic:** `if abs(BaroAlt - GPSAlt) > ALTERR_MAX: PRE-ARM FAILURE`
- **Common Error:** "GPS alt error [X]m (see BARO_ALTERR_MAX)"

2. Sensor Health

This check helps identify:

- **Drifting Barometers:** Due to temperature changes or sunlight hitting the sensor.
- **Bad GPS Vertical Accuracy:** Poor satellite geometry.
- **Pressurized Compartments:** If the flight controller is sealed in a box that pressurizes as it heats up.

Parameter Breakdown

- `ALTERR_MAX` (or `BARO_ALTERR_MAX`): Maximum allowable error in meters (or centimeters, check units). Default is typically 2000 (20 meters).

Integration Guide

Troubleshooting Pre-Arm Failures

1. **Check GPS:** Ensure you have a 3D Lock and low HDOP/VDOP. GPS altitude is notoriously noisy indoors.
2. **Calibrate:** Perform a barometer calibration (reboot) if the vehicle has moved significant elevation since power-on.
3. **Adjust Threshold:** If you are operating in a challenging environment (e.g., indoors with GPS repeaters), you may need to increase `ALTERR_MAX` to bypass the check, but this increases the risk of EKF instability.

Developer Notes

- **Library:** `libraries/AP_Baro`
- **Implementation:** The check is performed within the `AP_Baro::update()` loop or the `AP_Arming` library.

`ALTERR_MAX` (Altitude error maximum)

DEFAULT: 2000	RANGE: 0 5000	UNITS: m

Altitude error maximum (ALTERR_MAX)

Description

Safety check during arming that compares barometric altitude with GPS altitude. If the discrepancy exceeds this value, arming will be blocked.

Tuning & Behavior

- **Default Value:** 2000 m
- **Range:** 0 to 5000 m
- **A value of 0 disables this arming check.**
- **Commonly used to detect hardware mismatches (e.g., MS5607 vs MS5611).**

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





ANGLE LIMITS & CONFIGURATION (ANGLE)

Overview

The **ANGLE** parameter group is a mixed category containing angle-related limits for different vehicle types.

Note: The parameters here serve completely different functions depending on whether you are flying a Multicopter or sailing a Boat.

Key Concepts

1. Copter Bank Limit (`ANGLE_MAX`)

This is the **Global Lean Angle Limit** for ArduCopter in all stabilized modes (Loiter, AltHold, Auto, RTL, PosHold, Drift).

- **Function:** It clamps the maximum pitch and roll angle the autopilot will ever command.
- **Safety:** Prevents the copter from banking so steeply that it loses altitude (insufficient vertical thrust component).
- **Performance:** Higher angles = Faster top speed in GPS modes (Speed = sqrt(tan(Angle) * g * Radius)).

2. Sailboat Configuration (`ANGLE_IDEAL`, `ANGLE_MIN`)

(ArduRover - Sailboats Only)
These parameters configure the automated sail winch control.

- `ANGLE_IDEAL`: The optimal angle of attack for the mainsail to generate thrust.
- `ANGLE_MIN`: The tightest angle the sail can physically be pulled in (close-hauled limit).

Parameter Breakdown

Copter

- `ANGLE_MAX`: Maximum lean angle in centi-degrees (e.g., 4500 = 45 degrees).

Rover (Sailboat)

- `ANGLE_IDEAL`: Ideal sail angle (deg).
- `ANGLE_MIN`: Minimum sail angle (deg).

Integration Guide

Tuning Copter Speed

1. **Default:** 45 degrees (`ANGLE_MAX = 4500`). Good for most camera drones.
2. **Fast Flight:** Increase to 55-60 degrees for high-speed autonomous missions (ensure your T/W ratio is > 2:1).
3. **Filming:** Decrease to 25-30 degrees for very smooth, slow movements.

Tuning Sailboats

1. **Bench Test:** Verify winch movement limits.
2. **Set Limits:** Ensure `ANGLE_MIN` prevents the boom from hitting the shrouds or superstructure.

Developer Notes

- **Copter:** `ANGLE_MAX` is a `GSCALAR` in `ArduCopter/Parameters.cpp`.
- **Rover:** `ANGLE_IDEAL` is defined in `Rover/sailboat.cpp`.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





ANGLE_IDEAL (Sailboat Ideal Angle)

DEFAULT: 25	RANGE: 0 90	UNITS: deg

Sailboat Ideal Angle (ANGLE_IDEAL)

Description

Optimal sail angle relative to the apparent wind for efficient sailing in Rover sailboat configurations.

Tuning & Behavior

- **Default Value:** 25 deg
- **Range:** 0 to 90 deg

ANGLE_MAX (maximum angle error for tune abort)

DEFAULT: 10	UNITS: deg

maximum angle error for tune abort (ANGLE_MAX)

Description

Safety threshold for Quicktune. If the aircraft's attitude deviates more than this from the target during tuning, the process will immediately abort to prevent potential crashes from unstable PID gains.

Tuning & Behavior

- **Default Value:** 10 deg
- **If you get "Quicktune: attitude error ABORTING" frequently, ensure the aircraft is well-balanced or slightly increase this value.**

ANGLE_MIN (Sailboat Minimum Angle)

DEFAULT: 0	RANGE: 0 90	UNITS: deg

Sailboat Minimum Angle (ANGLE_MIN)

Description

Defines the minimum allowable angle for the sail in Rover sailboat configurations.

Tuning & Behavior

- **Default Value:** 0 deg
- **Range:** 0 to 90 deg

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

Key Concepts

ARM_SCR (Arm Screen)

DEFAULT: 0	RANGE: 0 4	UNITS: Index

Arm Screen (ARM_SCR)

Description

This parameter automates the On-Screen Display (OSD) management by automatically switching to a specific layout when the vehicle arms.

Pilots often prefer a "Pre-flight" OSD screen (showing GPS health and battery cell voltages) while on the ground, but want a "Flight" OSD screen (showing artificial horizon, altitude, and current draw) as soon as they are ready to fly. Setting `ARM_SCR` eliminates the need to manually toggle screens using a transmitter switch.

The Mathematics

The parameter uses a 1-based index for user convenience, which the internal code converts to a 0-indexed array element:

$$\text{InternalScreenID} = ARM_SCR - 1$$

If $ARM_SCR = 0$, the feature is disabled and no automatic switching occurs.

The Engineer's View

In `AP_OSD::update()` (`libraries/AP_OSD/AP_OSD.cpp`), the OSD system monitors the armed state of the vehicle.

1. It checks for a "Rising Edge" transition where `was_armed` is false but the current state is armed.
2. It verifies that `arm_scr` is within the valid range (typically 1 to 4).
3. It checks if the target screen layout is actually enabled in the parameters.
4. If all checks pass, it updates the `current_screen` variable, forcing the display to refresh with the new layout.

Tuning & Behavior

- **0:** Disabled (No automatic switching).
- **1 - 4:** Switches to the corresponding OSD screen number on arming.
- **Recommendation:** Set OSD Screen 1 as your "Summary/Pre-flight" screen and OSD Screen 2 as your "Combat/Flight" screen. Then set `ARM_SCR` to **2**.
- **Related:** Use `OSD_DSARM_SCR` to automatically switch back to a summary screen upon landing and disarming.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **ARMING** parameter group configures the **Pre-Arm Safety Checks**. ArduPilot performs a comprehensive system health check before allowing the motors to spin (Arming). This prevents taking off with bad sensors, no GPS lock, or unsafe configurations.

Key Concepts

1. The Arming Check Bitmask (`ARMING_CHECK`)

This is the master configuration for what to check.

- **1 (All):** Performs all standard checks (Barometer, Compass, GPS, Battery, RC, Board Voltage, Safety Switch, Logging). **Recommended.**
- **0 (None):** Disables all checks. **Dangerous.**
- **Specific Bits:** You can disable individual checks (e.g., disable GPS check for indoor flight) by summing the bitmask values.

2. Rudder Arming (`ARMING_RUDDER`)

Configures whether holding the rudder stick (Yaw) to the bottom-right corner arms the vehicle.

- **1 (ArmOrDisarm):** Standard for Copters.
- **2 (ArmOnly):** Cannot disarm via stick (requires GCS or Switch). Safer for flight.
- **0 (Disabled):** Must use GCS or Switch to arm.

3. Required Accuracy

Parameters like `ARMING_ACCTHRESH` (Accelerometer error) and `ARMING_MAGTHRESH` (Compass error) set the strictness of the sensor health checks.

Parameter Breakdown

- `ARMING_CHECK` : Bitmask of enabled checks.
- `ARMING_RUDDER` : Enable/Disable stick arming.
- `ARMING_ACCTHRESH` : Max acccelometer inconsistency.
- `ARMING_MAGTHRESH` : Max magnetic field inconsistency.

Integration Guide

Troubleshooting "Pre-Arm: Check ..."

1. **Read the Message:** The HUD will tell you exactly what failed (e.g., "Pre-Arm: Compass not calibrated").
2. **Fix the Root Cause:** Calibrate the sensor or move away from magnetic interference.
3. **Last Resort:** Disable the specific check in `ARMING_CHECK` only if you understand the risk.

Developer Notes

- **Library:** `AP_Arming` (Vehicle specific implementations in `ArduCopter/AP_Arming.cpp`, etc.).
- **Safety:** These checks are the last line of defense against configuration errors.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





ARMING_ACCTHRESH (Accelerometer error threshold)

DEFAULT: 0.75	RANGE: 0.25 3.0	UNITS: m/s^2

Accelerometer error threshold (ARMING_ACCTHRESH)

Description

Defines the maximum allowable difference between multiple accelerometers. If the readings diverge by more than this value, the aircraft will refuse to arm, indicating a potential sensor failure or calibration mismatch.

Tuning & Behavior

- **Default Value:** 0.75 m/s^2
- **Range:** 0.25 to 3.0 m/s^2
- **A lower value makes the pre-arm check more strict.**
- **A higher value allows for more sensor noise/inconsistency but increases risk.**

ARMING_BBOX_SPD (Blackbox speed)

DEFAULT: 5	RANGE: 1 20	UNITS: m/s

Blackbox speed (ARMING_BBOX_SPD)

Description

GPS speed threshold for "Blackbox" mode, which automatically arms the flight controller to trigger data logging when a certain speed is reached.

Tuning & Behavior

- **Default Value:** 5 m/s
- **Range:** 1 to 20 m/s
- **Primarily used for high-speed logging on non-controlled platforms (e.g., as a dedicated logger).**
- **WARNING: Do not use with propellers attached if the FC is not the primary controller.**

ARMING_CHECK (Arm Checks to Perform (bitmask))

DEFAULT: 1

Arm Checks to Perform (ARMING_CHECK)

Description

Bitmask selecting which internal subsystems must pass their health checks before the vehicle is allowed to arm.

Tuning & Behavior

- **Default Value:** 1 (All checks enabled)
- **Bitmask Examples:**
  - 1: All
  - 2: Barometer
  - 4: Compass
  - 8: GPS lock
  - 16: INS (IMU)
  - 32: Parameters
  - 64: RC Channels
- **Setting to 0 disables all safety checks (NOT RECOMMENDED for flight).**

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





ARMING_CRSDP_IGN (Disable CrashDump Arming check)

DEFAULT: 0	RANGE: 0 1

Disable CrashDump Arming check (ARMING_CRSDP_IGN)

Description

A safety "acknowledgment" required if the flight controller previously suffered a hard crash resulting in a crash dump. Arming is blocked until the user acknowledges the crash by setting this to 1.

Tuning & Behavior

- **Default Value:** 0
- **Values:** 0:Check Active, 1:Acknowledge/Ignore
- **Setting this to 1 allows arming even if crash data is present.**

ARMING_MAGTHRESH (Compass magnetic field strength error threshold)

DEFAULT: 100	RANGE: 0 500	UNITS: mGauss

Compass magnetic field strength error threshold (ARMING_MAGTHRESH)

Description

Maximum allowable deviation of the measured magnetic field from the expected Earth magnetic model (based on GPS location).

Tuning & Behavior

- **Default Value:** 100 mGauss
- **Range:** 0 to 500 mGauss
- **Ensures the compass is properly calibrated and not affected by local interference.**

ARMING_MIS_ITEMS (Required mission items)

DEFAULT: 0

Required mission items (ARMING_MIS_ITEMS)

Description

Forces the pilot to include specific commands in their mission (e.g., Takeoff or Land) before the aircraft will arm.

Tuning & Behavior

- **Default Value:** 0
- **Bitmask:**
  - 0: Land
  - 1: VTOL Land
  - 2: DO_LAND_START
  - 3: Takeoff
  - 4: VTOL Takeoff
  - 5: Rallypoint
  - 6: RTL

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





ARMING_NEED_LOC (Require vehicle location)

DEFAULT: 0	RANGE: 0 1

Require vehicle location (ARMING_NEED_LOC)

Description

Strict requirement for a valid 3D GPS fix (or other absolute position) before arming is permitted.

Tuning & Behavior

- **Default Value:** 0 (Not required for most vehicles; Copter often defaults to 1 for safety)
- **Setting to 1 ensures the vehicle has a known "Home" position for failsafe RTL.**

ARMING_OPTIONS (Arming options)

DEFAULT: 0

Arming options (ARMING_OPTIONS)

Description

Bitmask for miscellaneous arming-related behaviors.

Tuning & Behavior

- **Default Value:** 0
- **Bitmask:**
  - 0: Disable prearm display
  - 1: Do not send status text on state change
  - 2: Skip IMU consistency checks when ICE motor running

ARMING_REQUIRE (Require Arming Motors)

DEFAULT: 1	RANGE: 0 2

Require Arming Motors (ARMING_REQUIRE)

Description

Determines if an explicit arming command (from GCS or RC sticks) is required to enable motor output.

Tuning & Behavior

- **Default Value:** 1
- **Values:**
  - 0: Disabled (Arm immediately on boot)
  - 1: Enabled (Minimum PWM when disarmed)
  - 2: Enabled (0 PWM when disarmed)
- **A value of 1 or 2 is strongly recommended for safety.**

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





ARMING_RUDDER (Arming with Rudder enable/disable)

DEFAULT: 1	RANGE: 0 2

Arming with Rudder enable/disable (ARMING_RUDDER)

Description

Enables or disables the ability to arm and disarm the vehicle using the rudder stick on the RC transmitter.

Tuning & Behavior

- **Default Value:** 1 (Arming Only for Plane, Arm/Disarm for Copter)
- **Values:** 0:Disabled, 1:ArmingOnly, 2:ArmOrDisarm
- **Rudder arming only works when the throttle is at its minimum position.**

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY

Overview

The **AROT** parameter group configures the **Autonomous Autorotation** capability, primarily for **Traditional Helicopters**.

If the engine fails (low RPM), this system automatically takes control to manage the collective pitch and cyclic to maintain rotor energy and guide the helicopter to a safe touchdown, performing a flare maneuver at the bottom.

Key Concepts

1. Entry Criteria

The system monitors the main rotor RPM (via `RPM` sensor). If it drops below a threshold while in flight, Autorotation is triggered.

2. Glide Phase

The autopilot pitches for optimal airspeed (`AROT_FW_V_P` gain) to maximize glide distance or rotor energy preservation.

3. Flare and Touchdown

Near the ground, the system pitches up (Flare) to convert forward airspeed into rotor RPM, slowing the descent for a soft landing.

Parameter Breakdown

- `AROT_ENABLE`: Master switch.
- `AROT_HS_P`: Head Speed (RPM) P gain (controls Collective).
- `AROT_FW_V_P`: Forward Velocity P gain (controls Pitch).
- `AROT_RAMP`: Time to ramp collective back in if power is restored (Bail-out).

Developer Notes

- **Library:** `libraries/AC_Autorotation`
- **Vehicle: ArduCopter** (Heli frame).

AROT_COL_FILT_E (Entry Phase Collective Filter)

DEFAULT: 0.7	RANGE: 0.2 0.5	UNITS: Hz

Entry Phase Collective Filter (AROT_COL_FILT_E)

Description

Low-pass filter frequency applied to collective commands during the initial entry into autorotation.

Tuning & Behavior

- **Default Value:** 0.7 Hz
- **Range:** 0.2 to 0.5 Hz (Note: Default in code is 0.7, range documentation says 0.2-0.5)
- **Controls the reactivity of the collective during the critical transition to autorotation.**

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





AROT_COL_FILT_G (Glide Phase Collective Filter)

DEFAULT: 0.1	RANGE: 0.03 0.15	UNITS: Hz

Glide Phase Collective Filter (AROT_COL_FILT_G)

Description

Low-pass filter frequency applied to collective commands during the steady-state glide phase of autorotation.

Tuning & Behavior

- **Default Value:** 0.1 Hz
- **Range:** 0.03 to 0.15 Hz
- **A lower value results in smoother collective changes during the glide.**

AROT_ENABLE (Enable autonomous autorotation)

DEFAULT: 0	RANGE: 0 1

Enable autonomous autorotation (AROT_ENABLE)

Description

Enables the autonomous autorotation feature for helicopters, allowing the flight controller to manage head speed and glide profile during an engine failure or manual entry.

Tuning & Behavior

- **Default Value:** 0 (Disabled)
- **Values:** 0:Disabled, 1:Enabled
- **Requires careful tuning of head speed and collective parameters for safe operation.**

AROT_FWD_D (Autorotation Forward Velocity D Gain (Alias))

DEFAULT: 0

Autorotation Forward Velocity D Gain (AROT_FWD_D)

Note: This parameter is likely a legacy alias or reserved slot.

AROT_FWD_FF (Autorotation Forward Velocity FeedForward (Alias))

DEFAULT: 0.15

Autorotation Forward Velocity FeedForward (AROT_FWD_FF)

Note: This parameter is an alias for AROT_FW_V_FF.

AROT_FWD_FLTD (Autorotation Forward Velocity Derivative Filter (Alias))

DEFAULT: 0

Autorotation Forward Velocity Derivative Filter (AROT_FWD_FLTD)

Note: This parameter is likely a legacy alias or reserved slot.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





AROT_FWD_FLTE (Autorotation Forward Velocity Error Filter (Alias))

DEFAULT: 0

Autorotation Forward Velocity Error Filter (AROT_FWD_FLTE)

Note: This parameter is likely a legacy alias or reserved slot.

AROT_FWD_I (Autorotation Forward Velocity I Gain (Alias))

DEFAULT: 0

Autorotation Forward Velocity I Gain (AROT_FWD_I)

Note: This parameter is likely a legacy alias or reserved slot. The current autorotation controller primarily uses P and FeedForward terms for forward speed.

AROT_FWD_IMAX (Autorotation Forward Velocity I-Max (Alias))

DEFAULT: 0

Autorotation Forward Velocity I-Max (AROT_FWD_IMAX)

Note: This parameter is likely a legacy alias or reserved slot.

AROT_FWD_P (Autorotation Forward Velocity P Gain (Alias))

DEFAULT: 0.9	RANGE: 0.1 6.0

Autorotation Forward Velocity P Gain (AROT_FWD_P)

Note: This parameter is an alias for AROT_FW_V_P.

AROT_FW_V_FF (Velocity (horizontal) feed forward)

DEFAULT: 0.15	RANGE: 0 1

Velocity (horizontal) feed forward (AROT_FW_V_FF)

Description

Feed-forward gain for the horizontal velocity controller.

Tuning & Behavior

- **Default Value:** 0.15
- **Range:** 0 to 1

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





AROT_FW_V_P (Velocity (horizontal) P gain)

DEFAULT: 0.9	RANGE: 0.1 6.0

Velocity (horizontal) P gain (AROT_FW_V_P)

Description

Proportional gain for the horizontal velocity controller during autorotation glide.

Tuning & Behavior

- **Default Value:** 0.9
- **Range:** 0.1 to 6.0
- **Controls how aggressively the aircraft maintains its target forward speed during descent.**

AROT_HS_P (P gain for head speed controller)

DEFAULT: 0.7	RANGE: 0.3 1

P gain for head speed controller (AROT_HS_P)

Description

Proportional gain for the rotor RPM (head speed) controller during autorotation.

Tuning & Behavior

- **Default Value:** 0.7
- **Range:** 0.3 to 1
- **A higher value makes the controller more aggressive in maintaining target head speed.**

AROT_HS_SENSOR (Main Rotor RPM Sensor)

DEFAULT: 0	RANGE: 0 1

Main Rotor RPM Sensor (AROT_HS_SENSOR)

Description

Selects which RPM sensor instance provides the head speed feedback for the autorotation controller.

Tuning & Behavior

- **Default Value:** 0 (RPM1)
- **Range:** 0 to 1

AROT_HS_SET_PT (Target Head Speed)

DEFAULT: 1500	RANGE: 1000 2800	UNITS: RPM

Target Head Speed (AROT_HS_SET_PT)

Description

Defines the desired rotor RPM to be maintained during the autorotation glide.

Tuning & Behavior

- **Default Value:** 1500 RPM
- **Range:** 1000 to 2800 RPM
- **Setting this too low may result in insufficient lift for flare; too high may cause excessive descent rates.**

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





AROT_RAMP (Autorotation Bailout Ramp Time)

DEFAULT: 2	UNITS: s

Autorotation Bailout Ramp Time (AROT_RAMP)

Description

Defines the duration of the motor power ramp-up when the pilot or autopilot "bails out" of an autorotation and returns to powered flight.

Tuning & Behavior

- **Default Value:** 2 s
- **Ensures a smooth but timely restoration of power without overstressing the drive system.**

AROT_XY_ACC_MAX (Forward Acceleration Limit)

DEFAULT: 60	RANGE: 30 60	UNITS: cm/s/s

Forward Acceleration Limit (AROT_XY_ACC_MAX)

Description

Limits the maximum horizontal acceleration (and thus the pitch angle) the aircraft can use to maintain forward speed during autorotation glide.

Tuning & Behavior

- **Default Value:** 60 cm/s/s
- **Range:** 30 to 60 cm/s/s

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **ARSPD** parameter group configures the hardware interface and signal processing for **Airspeed Sensors** (Pitot Tubes). ArduPilot supports multiple redundant airspeed sensors (`ARSPD`, `ARSPD2`, etc.).

Accurate airspeed data is vital for:

1. **Energy Management:** Efficient climbing and cruising (TECS).
2. **Stall Prevention:** Flying at optimal angle of attack.
3. **Wind Estimation:** Calculating wind speed and direction by comparing Airspeed vector vs. Ground Speed (GPS) vector.

Key Concepts

1. Sensor Selection (`ARSPD_TYPE`)

Tells the driver which sensor hardware is connected.

- **1 (Analog):** Older MPXV7002DP sensors connected to an ADC pin.
- **2 (I2C - MS4525):** The standard digital sensor for Pixhawk (MS4525DO).
- **3 (I2C - MS5525):** Higher resolution digital sensor.
- **6 (I2C - SDP3x):** Sensirion SDP3x (very high sensitivity, requires 0 calibration).
- **8 (DroneCAN):** UAVCAN/DroneCAN sensors.

2. Calibration (`ARSPD_RATIO` / `ARSPD_AUTOCAL`)

- **Ratio:** Adjusts the scaling between pressure (Pa) and speed (m/s) to account for pitot tube placement errors (position error).
- **Autocal:** If enabled, the autopilot automatically adjusts the Ratio during flight by comparing GPS ground speed and Airspeed during loiter turns.

3. Tube Order (`ARSPD_TUBE_ORDR`)

Digital differential pressure sensors are directional. If your airspeed reads negative when blowing into the pitot tube, you can flip this parameter instead of physically swapping the silicone tubes.

Parameter Breakdown

- `ARSPD_USE`: Enable/Disable using this sensor for flight control.
- `ARSPD_TYPE`: Driver type.
- `ARSPD_BUS`: I2C bus number (0, 1, 2).
- `ARSPD_PIN`: (Analog only) ADC Pin number.
- `ARSPD_RATIO`: Calibration factor (Default ~2.0 for MS4525).
- `ARSPD_AUTOCAL`: Enable in-flight learning.

Integration Guide

Pre-Flight Check

1. **Cover:** Ensure pitot tube is uncovered.
2. **Calibrate:** Perform "Preflight Calibration" in GCS to zero the pressure offset.
3. **Blow Test:** Gently blow into the pitot tube. Speed should rise positive. If negative, toggle `ARSPD_TUBE_ORDR`.

Tuning

1. **Enable Autocal:** Set `ARSPD_AUTOCAL = 1`.
2. **Fly:** Loiter in circles for 5 minutes.
3. **Verify:** Check if `ARSPD_RATIO` has converged to a stable value.
4. **Lock:** Set `ARSPD_AUTOCAL = 0` once tuned to prevent bad learning in turbulent conditions.

Developer Notes

- **Library:** `libraries/AP_Airspeed`
- **Redundancy:** The `AP_Airspeed` library manages sensor switching and health monitoring (stuck pressure detection).
- **Primary:** `ARSPD_PRIMARY` selects which sensor instance (0, 1, 2) is fed to the EKF.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





ARSPD2_AUTOCAL (Automatic airspeed ratio calibration)

DEFAULT: 0	RANGE: null

Automatic airspeed ratio calibration

Note: This parameter functions identically to ARSPD_AUTOCAL.

ARSPD2_BUS (Airspeed I2C bus)

DEFAULT: 1	RANGE: null

Airspeed I2C bus

Note: This parameter functions identically to ARSPD_BUS.

ARSPD2_DEVID (Airspeed ID)

DEFAULT: 0	RANGE: null

Airspeed ID

Note: This parameter functions identically to ARSPD_DEVID.

ARSPD2_OFFSET (Airspeed offset)

DEFAULT: 0	RANGE: null

Airspeed offset

Note: This parameter functions identically to ARSPD_OFFSET.

ARSPD2_PIN (Airspeed pin)

DEFAULT: 0	RANGE: null

Airspeed pin

Note: This parameter functions identically to ARSPD_PIN.

ARSPD2_PSI_RANGE (The PSI range of the device)

DEFAULT: PSI_RANGE_DEFAULT	RANGE: null

The PSI range of the device

Note: This parameter functions identically to ARSPD_PSI_RANGE.

ARSPD2_RATIO (Airspeed ratio)

DEFAULT: 2	RANGE: null

Airspeed ratio

Note: This parameter functions identically to ARSPD_RATIO.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





ARSPD2_SKIP_CAL (Skip airspeed offset calibration on startup)

DEFAULT: 0	RANGE: null

Skip airspeed offset calibration on startup

Note: This parameter functions identically to ARSPD_SKIP_CAL.

ARSPD2_TUBE_ORDR (Control pitot tube order)

DEFAULT: 2	RANGE: null

Control pitot tube order

Note: This parameter functions identically to ARSPD_TUBE_ORDR.

ARSPD2_TYPE (Airspeed type)

DEFAULT: 0	RANGE: null

Airspeed type

Note: This parameter functions identically to ARSPD_TYPE.

ARSPD2_USE (Airspeed use)

DEFAULT: 0	RANGE: null

Airspeed use

Note: This parameter functions identically to ARSPD_USE.

ARSPD3_AUTOCAL (Automatic airspeed ratio calibration)

DEFAULT: 0	RANGE: null

Automatic airspeed ratio calibration

Note: This parameter functions identically to ARSPD_AUTOCAL.

ARSPD3_BUS (Airspeed I2C bus)

DEFAULT: 1	RANGE: null

Airspeed I2C bus

Note: This parameter functions identically to ARSPD_BUS.

ARSPD3_DEVID (Airspeed ID)

DEFAULT: 0	RANGE: null

Airspeed ID

Note: This parameter functions identically to ARSPD_DEVID.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





ARSPD3_OFFSET (Airspeed offset)

DEFAULT: 0	RANGE: null

Airspeed offset

Note: This parameter functions identically to ARSPD_OFFSET.

ARSPD3_PIN (Airspeed pin)

DEFAULT: 0	RANGE: null

Airspeed pin

Note: This parameter functions identically to ARSPD_PIN.

ARSPD3_PSI_RANGE (The PSI range of the device)

DEFAULT: PSI_RANGE_DEFAULT	RANGE: null

The PSI range of the device

Note: This parameter functions identically to ARSPD_PSI_RANGE.

ARSPD3_RATIO (Airspeed ratio)

DEFAULT: 2	RANGE: null

Airspeed ratio

Note: This parameter functions identically to ARSPD_RATIO.

ARSPD3_SKIP_CAL (Skip airspeed offset calibration on startup)

DEFAULT: 0	RANGE: null

Skip airspeed offset calibration on startup

Note: This parameter functions identically to ARSPD_SKIP_CAL.

ARSPD3_TUBE_ORDR (Control pitot tube order)

DEFAULT: 2	RANGE: null

Control pitot tube order

Note: This parameter functions identically to ARSPD_TUBE_ORDR.

ARSPD3_TYPE (Airspeed type)

DEFAULT: 0	RANGE: null

Airspeed type

Note: This parameter functions identically to ARSPD_TYPE.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





ARSPD3_USE (Airspeed use)

DEFAULT: 0	RANGE: null

Airspeed use

Note: This parameter functions identically to ARSPD_USE.

ARSPD4_AUTOCAL (Automatic airspeed ratio calibration)

DEFAULT: 0	RANGE: null

Automatic airspeed ratio calibration

Note: This parameter functions identically to ARSPD_AUTOCAL.

ARSPD4_BUS (Airspeed I2C bus)

DEFAULT: 1	RANGE: null

Airspeed I2C bus

Note: This parameter functions identically to ARSPD_BUS.

ARSPD4_DEVID (Airspeed ID)

DEFAULT: 0	RANGE: null

Airspeed ID

Note: This parameter functions identically to ARSPD_DEVID.

ARSPD4_OFFSET (Airspeed offset)

DEFAULT: 0	RANGE: null

Airspeed offset

Note: This parameter functions identically to ARSPD_OFFSET.

ARSPD4_PIN (Airspeed pin)

DEFAULT: 0	RANGE: null

Airspeed pin

Note: This parameter functions identically to ARSPD_PIN.

ARSPD4_PSI_RANGE (The PSI range of the device)

DEFAULT: PSI_RANGE_DEFAULT	RANGE: null

The PSI range of the device

Note: This parameter functions identically to ARSPD_PSI_RANGE.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





ARSPD4_RATIO (Airspeed ratio)

DEFAULT: 2	RANGE: null

Airspeed ratio

Note: This parameter functions identically to ARSPD_RATIO.

ARSPD4_SKIP_CAL (Skip airspeed offset calibration on startup)

DEFAULT: 0	RANGE: null

Skip airspeed offset calibration on startup

Note: This parameter functions identically to ARSPD_SKIP_CAL.

ARSPD4_TUBE_ORDR (Control pitot tube order)

DEFAULT: 2	RANGE: null

Control pitot tube order

Note: This parameter functions identically to ARSPD_TUBE_ORDR.

ARSPD4_TYPE (Airspeed type)

DEFAULT: 0	RANGE: null

Airspeed type

Note: This parameter functions identically to ARSPD_TYPE.

ARSPD4_USE (Airspeed use)

DEFAULT: 0	RANGE: null

Airspeed use

Note: This parameter functions identically to ARSPD_USE.

ARSPD5_AUTOCAL (Automatic airspeed ratio calibration)

DEFAULT: 0	RANGE: null

Automatic airspeed ratio calibration

Note: This parameter functions identically to ARSPD_AUTOCAL.

ARSPD5_BUS (Airspeed I2C bus)

DEFAULT: 1	RANGE: null

Airspeed I2C bus

Note: This parameter functions identically to ARSPD_BUS.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





ARSPD5_DEVID (Airspeed ID)

DEFAULT: 0	RANGE: null

Airspeed ID

Note: This parameter functions identically to ARSPD_DEVID.

ARSPD5_OFFSET (Airspeed offset)

DEFAULT: 0	RANGE: null

Airspeed offset

Note: This parameter functions identically to ARSPD_OFFSET.

ARSPD5_PIN (Airspeed pin)

DEFAULT: 0	RANGE: null

Airspeed pin

Note: This parameter functions identically to ARSPD_PIN.

ARSPD5_PSI_RANGE (The PSI range of the device)

DEFAULT: PSI_RANGE_DEFAULT	RANGE: null

The PSI range of the device

Note: This parameter functions identically to ARSPD_PSI_RANGE.

ARSPD5_RATIO (Airspeed ratio)

DEFAULT: 2	RANGE: null

Airspeed ratio

Note: This parameter functions identically to ARSPD_RATIO.

ARSPD5_SKIP_CAL (Skip airspeed offset calibration on startup)

DEFAULT: 0	RANGE: null

Skip airspeed offset calibration on startup

Note: This parameter functions identically to ARSPD_SKIP_CAL.

ARSPD5_TUBE_ORDR (Control pitot tube order)

DEFAULT: 2	RANGE: null

Control pitot tube order

Note: This parameter functions identically to ARSPD_TUBE_ORDR.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





ARSPD5_TYPE (Airspeed type)

DEFAULT: 0	RANGE: null

Airspeed type

Note: This parameter functions identically to ARSPD_TYPE.

ARSPD5_USE (Airspeed use)

DEFAULT: 0	RANGE: null

Airspeed use

Note: This parameter functions identically to ARSPD_USE.

ARSPD6_AUTOCAL (Automatic airspeed ratio calibration)

DEFAULT: 0	RANGE: null

Automatic airspeed ratio calibration

Note: This parameter functions identically to ARSPD_AUTOCAL.

ARSPD6_BUS (Airspeed I2C bus)

DEFAULT: 1	RANGE: null

Airspeed I2C bus

Note: This parameter functions identically to ARSPD_BUS.

ARSPD6_DEVID (Airspeed ID)

DEFAULT: 0	RANGE: null

Airspeed ID

Note: This parameter functions identically to ARSPD_DEVID.

ARSPD6_OFFSET (Airspeed offset)

DEFAULT: 0	RANGE: null

Airspeed offset

Note: This parameter functions identically to ARSPD_OFFSET.

ARSPD6_PIN (Airspeed pin)

DEFAULT: 0	RANGE: null

Airspeed pin

Note: This parameter functions identically to ARSPD_PIN.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





ARSPD6_PSI_RANGE (The PSI range of the device)

DEFAULT: PSI_RANGE_DEFAULT	RANGE: null

The PSI range of the device

Note: This parameter functions identically to ARSPD_PSI_RANGE.

ARSPD6_RATIO (Airspeed ratio)

DEFAULT: 2	RANGE: null

Airspeed ratio

Note: This parameter functions identically to ARSPD_RATIO.

ARSPD6_SKIP_CAL (Skip airspeed offset calibration on startup)

DEFAULT: 0	RANGE: null

Skip airspeed offset calibration on startup

Note: This parameter functions identically to ARSPD_SKIP_CAL.

ARSPD6_TUBE_ORDR (Control pitot tube order)

DEFAULT: 2	RANGE: null

Control pitot tube order

Note: This parameter functions identically to ARSPD_TUBE_ORDR.

ARSPD6_TYPE (Airspeed type)

DEFAULT: 0	RANGE: null

Airspeed type

Note: This parameter functions identically to ARSPD_TYPE.

ARSPD6_USE (Airspeed use)

DEFAULT: 0	RANGE: null

Airspeed use

Note: This parameter functions identically to ARSPD_USE.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





ARSPD_AUTOCAL (Automatic airspeed ratio calibration)

DEFAULT: 0	RANGE: 0 1

Automatic airspeed ratio calibration (ARSPD_AUTOCAL)

Description

A feature that dynamically tunes the ARSPD_RATIO during flight by comparing groundspeed and estimated wind to measured airspeed.

Tuning & Behavior

- **Default Value:** 0 (Disabled)
- **Should be disabled once the ratio has converged to a stable value.**

ARSPD_BUS (Airspeed I2C bus)

DEFAULT: 1	RANGE: 0 3

Airspeed I2C bus (ARSPD_BUS)

Description

Selects the I2C physical bus on which the digital airspeed sensor is located.

Tuning & Behavior

- **Default Value:** 1
- **May not match the labels on the flight controller board.**

ARSPD_DEVID (Airspeed ID)

DEFAULT: 0

Airspeed ID (ARSPD_DEVID)

Description

Automatically detected sensor hardware ID.

Tuning & Behavior

- **Default Value:** 0
- **ReadOnly:** True

ARSPD_ENABLE (Airspeed Enable)

DEFAULT: 0	RANGE: 0 1

Airspeed Enable (ARSPD_ENABLE)

Description

Global master switch to enable the airspeed sensor library.

Tuning & Behavior

- **Default Value:** 0 (Disabled)
- **Values:** 0:Disable, 1:Enable
- **Requires reboot to take effect.**

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





ARSPD_OFFSET (Airspeed offset)

DEFAULT: 0

Airspeed offset (ARSPD_OFFSET)

Description

Stored calibration value for the pressure sensor's zero-point.

Tuning & Behavior

- **Default Value:** 0
- **Usually set automatically during the "Preflight Calibration" process with the pitot tube covered.**

ARSPD_OFF_PCNT (Maximum offset cal speed error)

DEFAULT: 0	RANGE: 0.0 10.0	UNITS: %

Maximum offset cal speed error (ARSPD_OFF_PCNT)

Description

Safety threshold to detect if an airspeed calibration was performed incorrectly (e.g., pitot tube not covered).

Tuning & Behavior

- **Default Value:** 0 (Disabled)
- **Range:** 0.0 to 10.0 %

ARSPD_OPTIONS (Airspeed options bitmask)

DEFAULT: 11

Airspeed options (ARSPD_OPTIONS)

Description

Advanced configuration bitmask for airspeed sensor health monitoring and EKF consistency checks.

Tuning & Behavior

- **Default Value:** 11 (SpeedMismatchDisable | AllowSpeedMismatchRecovery | UseEkf3Consistency)
- **Bitmask:**
  - 0: SpeedMismatchDisable (Mark unhealthy on large groundspeed mismatch)
  - 1: AllowSpeedMismatchRecovery (Mark healthy if mismatch resolves)
  - 2: DisableVoltageCorrection
  - 3: UseEkf3Consistency (Require airspeed to match EKF estimated wind)
  - 4: Report cal offset to GCS

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





ARSPD_PIN (Airspeed pin)

DEFAULT: 0

Airspeed pin (ARSPD_PIN)

Description

Specifies the analog input pin used for analog airspeed sensors.

Tuning & Behavior

- **Default Value:** 0
- **Common Values:** 2:Pixhawk/Pixracer, 13:Pixhawk2_PM2, 14:CubeOrange
- **Only relevant if ARSPD_TYPE is set to 2 (Analog).**

ARSPD_PRIMARY (Primary airspeed sensor)

DEFAULT: 0	RANGE: 0 1

Primary airspeed sensor (ARSPD_PRIMARY)

Description

Selects the primary sensor for flight control in dual-airspeed configurations.

Tuning & Behavior

- **Default Value:** 0 (First Sensor)
- **Values:** 0:FirstSensor, 1:2ndSensor

ARSPD_PSI_RANGE (PSI range of the device)

DEFAULT: 1.0	UNITS: PSI

PSI range of the device (ARSPD_PSI_RANGE)

Description

The pressure range of the airspeed sensor hardware.

Tuning & Behavior

- **Default Value:** 1.0 PSI
- **Do not change this unless required by the specific sensor datasheet (e.g., DLVR sensors).**

ARSPD_RATIO (Airspeed ratio)

DEFAULT: 2.0	RANGE: 1.5 3.0

Airspeed ratio (ARSPD_RATIO)

Description

Scaling factor used to convert measured differential pressure into airspeed. It accounts for pitot tube design and placement.

Tuning & Behavior

- **Default Value:** 2.0
- **Range:** 1.5 to 3.0
- **Can be automatically tuned using ARSPD_AUTOCAL.**

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





ARSPD_SKIP_CAL (Skip airspeed offset calibration on startup)

DEFAULT: 0	RANGE: 0 1

Skip airspeed offset calibration on startup (ARSPD_SKIP_CAL)

Description

Saves time during boot by skipping the zero-airspeed pressure calibration.

Tuning & Behavior

- **Default Value:** 0 (Always calibrate)
- **Only recommended if your sensor has very low drift and you cannot easily cover the pitot tube during boot.**
- **Requires a prior successful manual calibration.**

ARSPD_TUBE_ORDR (Control pitot tube order)

DEFAULT: 2	RANGE: 0 2

Control pitot tube order (ARSPD_TUBE_ORDR)

Description

Defines which pressure port (stagnation vs static) is connected to which sensor input.

Tuning & Behavior

- **Default Value:** 2 (Auto Detect)
- **Values:** 0:Normal, 1:Swapped, 2:Auto Detect
- **Set to 0 or 1 if the sensor consistently reports negative airspeed on the ground.**

ARSPD_TYPE (Airspeed type)

DEFAULT: 0

Airspeed type (ARSPD_TYPE)

Description

Enables and selects the hardware type for the primary airspeed sensor.

Tuning & Behavior

- **Default Value:** 0 (None)
- **Common Values:** 1:I2C-MS4525D0, 2:Analog, 3:I2C-MS5525, 6:I2C-SDP3X, 8:DroneCAN, 100:SITL
- **Requires reboot to initialize the driver.**

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY

ARSPD_USE (Airspeed use)

DEFAULT: 0	RANGE: 0 2

Airspeed use (ARSPD_USE)

Description

Determines if the measured airspeed is used by the flight controller for autonomous speed and altitude management (TECS).

Tuning & Behavior

- **Default Value:** 0 (DoNotUse - Logging only)
- **Values:** 0:DoNotUse, 1:Use, 2:UseWhenZeroThrottle
- **Set to 1 only after verifying the sensor is healthy and properly calibrated.**

ARSPD_WIND_GATE (Re-enable Consistency Check Gate Size)

DEFAULT: 5.0	RANGE: 0.0 10.0

Re-enable Consistency Check Gate Size (ARSPD_WIND_GATE)

Description

Sensitivity of the logic that re-enables a previously failed airspeed sensor once its readings return to a statistically consistent range.

Tuning & Behavior

- **Default Value:** 5.0
- **Range:** 0.0 to 10.0
- **Higher values make re-enabling easier but carry higher risk of using bad data.**

ARSPD_WIND_MAX (Maximum airspeed and ground speed difference)

DEFAULT: 0	RANGE: 0 127	UNITS: m/s

Maximum airspeed and ground speed difference (ARSPD_WIND_MAX)

Description

Sanity check threshold comparing reported airspeed to groundspeed. If the difference is physically impossible (given known wind limits), the sensor is flagged as failed.

Tuning & Behavior

- **Default Value:** 0 (Disabled)
- **Range:** 0 to 127 m/s
- **Works in conjunction with ARSPD_OPTIONS bit 0.**

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





ARSPD_WIND_WARN (Airspeed and GPS speed difference warning)

DEFAULT: 0	RANGE: 0 127	UNITS: m/s

Airspeed and GPS speed difference warning (ARSPD_WIND_WARN)

Description

Threshold for issuing a non-fatal warning about suspected airspeed sensor inaccuracies compared to GPS groundspeed.

Tuning & Behavior

- **Default Value:** 0 (Disabled)
- **Range:** 0 to 127 m/s

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **AS** parameter group currently contains parameters related to the **Acceleration Limits** used by the Autonomous Autorotation controller (`AC_Autorotation`).

Key Parameters

- `AS_ACC_MAX`: Maximum forward acceleration allowed during the autorotation glide/maneuvers.

See Also

- AROT - Main Autorotation configuration.

AS_ACC_MAX (Forward Acceleration Limit)

DEFAULT: 60	RANGE: 30 60	UNITS: cm/s/s

Forward Acceleration Limit (AS_ACC_MAX)

Description

Limits the maximum horizontal acceleration the aircraft can use to maintain forward speed during autorotation.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **ATC** parameter group contains the core **PID (Proportional-Integral-Derivative)** tuning parameters for **ArduCopter** and **QuadPlane** (in VTOL modes).

This library is the "muscle" of the flight controller. It is responsible for taking the pilot's desire (e.g., "Lean 30 degrees right") and converting it into precise motor commands to achieve and maintain that attitude against wind and disturbances.

Key Concepts: The Control Loop

The attitude control logic works in a cascaded (nested) structure:

1. Angle Controller (Outer Loop)

- **Input:** Desired Angle (from Pilot stick or Navigation).
- **Action:** Calculates how fast the vehicle needs to rotate to reach that angle.
- **Output:** Target Rate (deg/s).
- **Parameters:** `ATC_ANG_RLL_P`, `ATC_ANG_PIT_P`, `ATC_ANG_YAW_P`.
- **Logic:** `Target_Rate = Error_Angle * P_Gain`.

2. Rate Controller (Inner Loop)

- **Input:** Target Rate (from Angle Controller) vs. Actual Rate (from Gyro).
- **Action:** Calculates the necessary motor power adjustments.
- **Output:** Motor Mixer Command (-1 to +1).
- **Parameters:** `ATC_RAT_RLL_P/I/D/FF`.
- **Logic:**
  - **FF (FeedForward):** Immediate output proportional to the target. Primary driver for sharp handling.
  - **P (Proportional):** Immediate correction based on error.
  - **I (Integral):** Long-term correction for imbalances/wind.
  - **D (Derivative):** Dampening to prevent overshoot.

3. Acceleration Limits

- `ATC_ACCEL_R_MAX` / `P_MAX` / `Y_MAX`: Limits how quickly the Angle Controller can request a change in rate. This limits the "jerkiness" or physical aggression of the drone.

Parameter Breakdown

Rate Loops (The most critical tuning)

- `ATC_RAT_RLL_P`, `I`, `D`, `FF`: Roll axis tuning.
- `ATC_RAT_PIT_P`, `I`, `D`, `FF`: Pitch axis tuning.
- `ATC_RAT_YAW_P`, `I`, `D`, `FF`: Yaw axis tuning.

Angle Loops (Feel & Stability)

- `ATC_ANG_RLL_P`: Roll angle stiffness (typically 4.5 - 10.0).
- `ATC_ANG_PIT_P`: Pitch angle stiffness.

Filters

- `ATC_RAT_RLL_FLTT`: Target filter frequency (smooths inputs).
- `ATC_RAT_RLL_FLTE`: Error filter frequency (smooths D-term noise).

Integration Guide

Basic Tuning Workflow

1. **AutoTune:** The best way to tune `ATC` parameters is to use the **AutoTune** flight mode. It automatically tests the vehicle's response and calculates optimal P, I, D, and Angle P values.
2. **Manual Adjustment:**

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





- **Oscillation (Fast Wobble):** Reduce P and D terms.
- **Sluggishness:** Increase P and FF.
- **Drift/Angle Hold Failure:** Increase I.
- **Overshoot (Bounce back):** Increase D or Reduce P.

Developer Notes

- **Library:** `libraries/AC_AttitudeControl`
- **Vehicle: ArduCopter**, **ArduPlane** (QuadPlane), **ArduSub**.
- **Evolution:** Modern ArduPilot relies heavily on **FeedForward (FF)** for responsiveness, whereas older versions relied more on P.

ATC_ACCEL_MAX (Speed Control Acceleration Max)

DEFAULT: 1.0	RANGE: 0 10	UNITS: m/s^2

Speed Control Acceleration Max (ATC_ACCEL_MAX)

Description

`ATC_ACCEL_MAX` is the "Launch Control."

It limits how fast the rover tries to speed up.

- **Low Value (e.g. 0.5):** Slow, smooth starts. No wheelspin.
- **High Value (e.g. 5.0):** Drag race mode.

Tuning & Behavior

- **Default Value:** 1.0 m/s^2.
- **Recommendation:** Set slightly lower than the physical limit of your traction to prevent wheelspin.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





ATC_ACCEL_P_MAX (Pitch Acceleration Max)

DEFAULT: 0.0	RANGE: 0 180000	UNITS: cdeg/s/s

Pitch Acceleration Max (ATC_ACCEL_P_MAX)

Description

`ATC_ACCEL_P_MAX` defines the maximum "torque authority" allowed around the pitch axis (tilting forward or backward). It limits the intensity of the drone's longitudinal rotation changes.

Think of it as the drone's "pitch throttle." If you command a sudden dive, `ATC_ACCEL_P_MAX` ensures the drone doesn't flip into that dive so violently that it stalls its props or causes a brown-out. It provides the "softness" or "sharpness" of the start and stop of a pitch maneuver.

- **Low Value:** Pitch changes are smooth and gradual. The drone will tilt forward or backward like a heavy ship.
- **High Value:** Pitch changes are sharp and aggressive. The drone snaps into forward flight instantly.
- **Disabled (0):** No limit is applied; the drone will rotate as fast as the motors can physically push the frame.

The Mathematics

This parameter limits the rate of change of the pitch angular velocity (ω_pitch):

$$\left| \frac{\omega_pitch(t) - \omega_pitch(t - dt)}{dt} \right| \leq \text{ACCEL_P_MAX}$$

Where:

- ACCEL_P_MAX is in centidegrees/sec².
- **Example:** A value of 36000 means the drone can increase its pitch rotation by 360 degrees per second, every second.

On frames like "Dead Cats," where the mass is distributed longitudinally, `ACCEL_P_MAX` often needs to be lower than `ACCEL_R_MAX` because the airframe has more inertia along the pitch axis.

The Engineer's View

This parameter is the `_accel_pitch_max` member in `AC_AttitudeControl`.

It is used in the **Input Shaper** for all pitch-related attitude commands:

```
// AC_AttitudeControl.cpp
_ang_vel_target.y = input_shaping_angle(error, _input_tc, get_accel_pitch_max_radss(), ...);
```

By constraining the input before it reaches the PID loop, ArduPilot ensures that the inner Rate Loop never receives a "step" command that it can't physically follow. This prevents P-gain "overshoot" and motor saturation during high-authority pitch changes.

Tuning & Behavior

- **Default Value:** 0.0 (Standard)
- **Range:** 0 - 180,000
- **Effect of Increasing:** Snappier pitch response. Better for high-speed tracking and stopping.
- **Effect of Decreasing:** Smoother cinematic movement. Reduces the risk of "pitch bobbing" during aggressive flight.

Use Case Recommendations

- **Cinematography: Set to 36000.** Provides a very professional, weighted feel to forward flight starts and stops.
- **Long-Range Flying: Set to 20000.** High pitch accelerations are inefficient and increase drag; a lower limit keeps the drone in its "sweet spot" for cruise efficiency.
- **Racing / Acro: Keep at 0.0.** Racers need the absolute maximum pitch authority to perform "power loops" and clear sharp turns.

Troubleshooting

- **Scenario:** Drone nose "dips" or "bobs" when you release the stick after flying forward.
  - **Diagnosis:** `ATC_ACCEL_P_MAX` is likely set too high, allowing the drone to stop rotating faster than the PID loop can stabilize.
  - **Fix:** Reduce `ATC_ACCEL_P_MAX` to 40000 and re-test.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





ATC_ACCEL_R_MAX (Roll Acceleration Max)

DEFAULT: 0.0	RANGE: 0 180000	UNITS: cdeg/s/s

Roll Acceleration Max (ATC_ACCEL_R_MAX)

Description

`ATC_ACCEL_R_MAX` defines the "maximum torque" the flight controller is allowed to request around the roll axis. While `ATC_RAT_RLL_P` determines how much authority the drone *has*, `ATC_ACCEL_R_MAX` determines how much authority it is *allowed to use* to start or stop a rotation.

Think of it as the "traction control" for your drone's rotation. If this value is too high, the drone will try to snap into a roll so fast that the motors saturated or the battery voltage drops. If it is too low, the drone will feel "heavy" and slow to start moving, even if your PID gains are high.

- **Low Value:** Flight feels very smooth, stable, and "heavy." Rotations start and stop with a noticeable ramp-up.
- **High Value:** Rotations feel instant and sharp.
- **Disabled (0):** The software does not limit angular acceleration (limited only by physical motor power).

The Mathematics

This parameter limits the rate of change of the target angular velocity (ω_target):

$$\left| \frac{\omega_target(t) - \omega_target(t - dt)}{dt} \right| \leq \text{ACCEL_R_MAX}$$

Where:

- ACCEL_R_MAX is in centidegrees/sec².
- **Example:** A value of 72000 means the drone can increase its rotation speed by 720 degrees per second, every second.

The Engineer's View

This parameter is the `_accel_roll_max` member in `AC_AttitudeControl`.

It is used in the **Input Shaper** logic inside `AC_AttitudeControl::input_shaping_angle()` and `input_shaping_ang_vel()`.
These functions ensure that the setpoints sent to the PID loops are physically achievable by the drone, preventing the PID loops from "winding up" against impossible targets.

```
// AC_AttitudeControl.cpp
_ang_vel_target.x = input_shaping_angle(error, _input_tc, get_accel_roll_max_radss(), ...);
```

By keeping the target acceleration within this limit, ArduPilot ensures that the drone's attitude remains predictable and that the "Square Root Controller" (which handles large errors) doesn't over-command the motors.

Tuning & Behavior

- **Default Value:** 0.0 (Standard)
- **Range:** 0 - 180,000 (0 to 1800 deg/s^2)
- **Effect of Increasing:** Snappier, more immediate rotations. The drone feels "lighter."
- **Effect of Decreasing:** Smoother, more cinematic rotations. Greatly reduces current spikes during aggressive flight.

Use Case Recommendations

- **Cinematography / Aerial Inspection: Set to 36000 - 54000.** Limits the "jerkiness" of the drone, making it easier for the camera operator and reducing mechanical stress.
- **Large Professional Rig (>25 inch props): Set to 20000.** Large props have enormous inertia; trying to accelerate them faster than 200 deg/s^2 usually results in wasted energy and heat.
- **High Performance / Racing: Keep at 0.0 or set >110000.** Racers need the absolute maximum torque available to clear obstacles and make 180-degree turns.

Troubleshooting

- **Scenario:** Drone feels "laggy" to start a roll, even though the PID gains are high.
  - **Diagnosis:** `ATC_ACCEL_R_MAX` is set too low, bottlenecking the command.
  - **Fix:** Increase `ATC_ACCEL_R_MAX` by 10000.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY

ATC_ACCEL_Y_MAX (Yaw Acceleration Max)

DEFAULT: 0.0	RANGE: 0 72000	UNITS: cdeg/s/s

Yaw Acceleration Max (ATC_ACCEL_Y_MAX)

Description

`ATC_ACCEL_Y_MAX` defines the maximum "spin authority" allowed around the vertical axis. It limits how fast the drone's heading can accelerate or decelerate.

Since multirotors yaw by changing torque, they have significantly less "grip" on the yaw axis than they do on roll or pitch. If you try to snap into a 360-degree spin too fast, the drone may lose its vertical lift (since motors are busy generating torque instead of lift). `ATC_ACCEL_Y_MAX` prevents this by forcing heading changes to be gradual and within the physical limits of the motors.

- **Low Value:** Yaw turns are smooth and cinematic. The tail "glides" into and out of turns.
- **High Value:** Yaw turns are sharp and robotic. The drone stops spinning instantly.
- **Disabled (0):** The software does not limit heading acceleration.

The Mathematics

This parameter limits the rate of change of the yaw angular velocity (ω_yaw):

$$\left| \frac{\omega_yaw(t) - \omega_yaw(t - dt)}{dt} \right| \leq \text{ACCEL_Y_MAX}$$

Where:

- ACCEL_Y_MAX is in centidegrees/sec².
- **Example:** A value of 9000 means the drone can increase its spin rate by 90 degrees per second, every second.

Safety Limit: Yaw acceleration is typically set much lower than Roll/Pitch (e.g. 9,000 vs 50,000) because high yaw torque demands can "starve" the other axes, leading to a loss of control.

The Engineer's View

This parameter is the `_accel_yaw_max` member in `AC_AttitudeControl`.

It is used in the **Input Shaper** for heading commands:

```
// AC_AttitudeControl.cpp
_ang_vel_target.z = input_shaping_angle(error, _input_tc, get_accel_yaw_max_radss(), ...);
```

By keeping the heading command within this acceleration limit, ArduPilot ensures that the drone's vertical lift remains consistent during turns, preventing "altitude dip" that occurs when motors on one diagonal are spun up to their limit to provide yaw torque.

Tuning & Behavior

- **Default Value:** 0.0 (Standard)
- **Range:** 0 - 72,000
- **Effect of Increasing:** Snappier heading response. Better for high-speed tracking and sharp "Yaw Spins."
- **Effect of Decreasing:** Smoother, more elegant yaw pans. Essential for professional videography.

Use Case Recommendations

- **Professional Cinematography: Set to 9000 - 18000.** Creates perfectly smooth pans that don't look like they were made by a robot.
- **FPV Freestyle: Keep at 0.0 or set >54000.** Required for maneuvers that rely on instant heading changes to change the drone's direction of momentum.
- **Mapping / Survey: Set to 18000.** Ensures the drone turns at the end of each grid line smoothly without overshooting or causing gimbal vibrations.

Troubleshooting

- **Scenario:** Drone loses 1-2 meters of altitude every time you perform a fast 360-degree spin.
  - **Diagnosis:** Yaw torque is starving the vertical lift (Saturation). `ATC_ACCEL_Y_MAX` is too high.
  - **Fix:** Reduce `ATC_ACCEL_Y_MAX` to 12000.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY

ATC_ANGLE_BOOST (Angle Boost)

DEFAULT: 1	RANGE: 0 1

Angle Boost (ATC_ANGLE_BOOST)

Description

`ATC_ANGLE_BOOST` is the "Auto-Altitude" feature of ArduPilot.

When a drone tilts to move forward, some of its propeller force is redirected sideways. Because less force is now pointing straight down, the drone will naturally lose altitude and "sink" unless the pilot increases the throttle. `ATC_ANGLE_BOOST` automates this. It "boosts" the throttle command based on how steep the lean is, ensuring the drone maintains a consistent hover height whether it's sitting still or zooming forward.

- **Enabled (1):** The drone stays at the same height automatically during turns and forward dashes.
- **Disabled (0):** The drone will sink whenever it tilts. The pilot must manually manage altitude with the throttle stick.

The Mathematics

The boost factor is based on the trigonometry of the drone's tilt angle (θ):

$$ \text{Throttle}_{boosted} = \frac{\text{Throttle}_{base}}{\cos(\theta)} $$

Where:

- θ is the combined roll and pitch angle.
- $\cos(\theta)$ represents the vertical component of the total thrust vector.

Example: If a drone tilts by 60 degrees, its vertical lift is cut in half ($\cos(60^\circ) = 0.5$). To stay level, it must **double** its total motor power. `ATC_ANGLE_BOOST` handles this math instantly.

The Engineer's View

This parameter is the `_angle_boost_enabled` member in `AC_AttitudeControl`.

It is applied in `AC_AttitudeControl_Multi::get_throttle_boosted()`:

```
// AC_AttitudeControl_Multi.cpp
float cos_tilt = _ahrs.cos_pitch() * _ahrs.cos_roll();
float boost_factor = 1.0f / constrain_float(cos_tilt_target, 0.1f, 1.0f);
float throttle_out = throttle_in * inverted_factor * boost_factor;
```

Crucially, the code includes a **Tilt-Limit Safeguard**: As the drone approaches 90 degrees (vertical), the math would command "infinite" power. ArduPilot caps the boost factor to prevent motor saturation and ensures the drone maintains enough control authority to level itself out.

Tuning & Behavior

- **Default Value:** 1 (Enabled)
- **Range:** 0 (Off) or 1 (On)
- **Effect of Enabling:** Much easier to fly in manual modes. Essential for AltHold and Loiter to work correctly.
- **Effect of Disabling:** "Old school" flight feel. The drone will descend during any movement.

Use Case Recommendations

- **Standard Multirotor: Always Keep Enabled (1).** There is almost no reason to turn this off for modern multirotor flight.
- **Acrobatic Training: Disable (0) occasionally.** If you want to learn how full-sized helicopters fly (where you must manually coordinate throttle and cyclic), turning this off provides a realistic challenge.
- **Cinematography: Always Keep Enabled (1).** Ensures perfectly level tracking shots.

Troubleshooting

- **Scenario:** Drone "climbs" slightly every time I fly forward at speed.
  - **Diagnosis:** This is rare, but usually means your motor thrust is non-linear or you have extreme aerodynamic lift from the drone's body.
  - **Fix:** Check `MOT_THST_EXPO`. If the problem persists, you can't "tune" angle boost (it's binary), so look for mechanical or sensor issues.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





ATC_ANG_LIM_TC (Angle Limit Time Constant)

DEFAULT: 1.0	RANGE: 0.5 10.0	UNITS: s

Angle Limit Time Constant (ATC_ANG_LIM_TC)

Description

`ATC_ANG_LIM_TC` is a specialized safety parameter that handles "low-power" situations.

Multirotors need vertical thrust to stay in the air. When they tilt (lean), some of that thrust is redirected sideways for movement. If you are carrying a heavy load or have a weak battery, the motors might reach 100% power just to stay level. If you then try to tilt by 30 degrees, the drone will lose lift and fall.

ArduPilot automatically calculates a "Safe Tilt Angle" based on how much throttle you have left. `ATC_ANG_LIM_TC` determines how fast that limit is applied. It ensures the drone prioritizes "not falling out of the sky" over "moving sideways."

- **Low Value:** The drone will aggressively and instantly limit your tilt if it thinks it is about to lose altitude.
- **High Value:** The drone will allow temporary "overshoots" of the tilt limit, relying on momentum before slowly enforcing the new safety limit.
- **Default (1.0):** A slow, 1-second decay that provides a smooth transition when reaching power limits.

The Mathematics

The controller calculates the maximum tilt (θ_max) that can be sustained at the current throttle setting. This parameter smooths that limit:

$$\theta_lim_smoothed = \theta_lim_smoothed_prev + \frac{dt}{dt + \tau} \cdot (\theta_max_calculated - \theta_lim_smoothed_prev)$$

Where:

- τ is `ATC_ANG_LIM_TC`.
- $\theta_max_calculated$ is the physical limit based on motor saturation.

Implication: If you punch the throttle to climb, your available tilt angle drops. This parameter determines how quickly that drop is enforced on your stick inputs.

The Engineer's View

This parameter is the `_angle_limit_tc` member in `AC_AttitudeControl`.

It is used in `AC_AttitudeControl_Multi::update_althold_lean_angle_max()`:

```
// AC_AttitudeControl_Multi.cpp
float althold_lean_angle_max = acosf(constrain_float(throttle_in / (AC_ATTITUDE_CONTROL_ANGLE_LIMIT_THROTTLE_MAX * thr_max), 0.0f, 1
_althold_lean_angle_max = _althold_lean_angle_max + (_dt / (_dt + _angle_limit_tc)) * (althold_lean_angle_max - _althold_lean_angle_
```

This is a critical part of the **Altitude Hold prioritization** logic. By smoothing the limit, the drone prevents jerky "twitching" of the attitude when flying at the edge of its performance envelope.

Tuning & Behavior

- **Default Value:** 1.0s
- **Range:** 0.5 - 10.0s
- **Effect of Increasing:** The drone will be more "brave"—it will allow you to tilt aggressively even if it means losing a little altitude temporarily.
- **Effect of Decreasing:** The drone is very "cautious"—it will flatten itself out immediately if it feels the motors working too hard.

Use Case Recommendations

- **Heavy Cargo Drones: Decrease to 0.5.** You want the safety limits to kick in instantly to prevent any vertical drop.
- **Underpowered / Battery-Saving builds: Decrease to 0.5.** Helps keep the drone in a safe operating range.
- **Racing / Acro-Loiter: Increase to 2.0.** Allows the pilot to "force" a tilt for a second to clear an obstacle, accepting a temporary loss in height.

Troubleshooting

- **Scenario:** During a fast forward dash in AltHold, the drone suddenly "nods" its nose up and slows down, even though my stick is still forward.
  - **Diagnosis:** The drone hit its power limit and is enforcing the angle limit.
  - **Fix:** If you have motor headroom, increase `ATC_ANG_LIM_TC` to 2.0 to make the correction less jarring.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





ATC_ANG_PIT_P (Pitch Axis Angle P Gain)

DEFAULT: 4.5	RANGE: 3.0 12.0

Pitch Axis Angle P Gain (ATC_ANG_PIT_P)

Description

`ATC_ANG_PIT_P` controls the "stiffness" of the drone's pitch attitude (tilting forward or backward). It is the outer control loop responsible for ensuring the drone reaches and holds the requested pitch angle.

If you push the stick forward to fly, `ATC_ANG_PIT_P` determines how quickly the drone tilts into that forward movement. A high value makes the drone snap into the tilt; a low value makes it tilt gracefully.

- **Low Value:** Pitch response feels "soft." The drone may take a noticeable amount of time to start moving or stop.
- **High Value:** Pitch response feels immediate and sharp.
- **Too High:** The drone will oscillate (bob its nose up and down) during hover or when sticks are released.

The Mathematics

This is a Proportional (k_P) controller that converts an **Angle Error** into a **Target Angular Velocity** (ω):

$$\omega_pitch_target = k_P \cdot (\theta_desired - \theta_actual)$$

Where:

- ω_pitch_target is the requested rotation speed (deg/s) for the Pitch axis.
- $\theta_desired$ is the angle requested by the sticks or autopilot.
- θ_actual is the current estimated pitch from the AHRS.
- k_P is `ATC_ANG_PIT_P`.
- **Dimensionality:** $1/s$ (Hertz).

Note on Symmetry: On most symmetric multirotors (X-quads), this value should be identical to `ATC_ANG_RLL_P`. On rectangular frames ("Dead Cat" or Long-Range frames), it may need to be slightly different to account for the different weight distribution.

The Engineer's View

This parameter is an instance of the `AC_P` class, member `_p_angle_pitch`, inside `AC_AttitudeControl`.

It is executed in `AC_AttitudeControl::update_ang_vel_target_from_att_error()`:

```
// AC_AttitudeControl.cpp
const float angleP_pitch = _p_angle_pitch.kP() * _angle_P_scale.y;
rate_target_ang_vel.y = angleP_pitch * attitude_error_rot_vec_rad.y;
```

The output `rate_target_ang_vel.y` is the input setpoint for the inner **Pitch Rate PID** loop (`ATC_RAT_PIT_*`).

Tuning & Behavior

- **Default Value:** 4.5 (Standard)
- **Range:** 3.0 - 12.0
- **Effect of Increasing:** Faster nose-up/nose-down response. Better adherence to the flight path in missions.
- **Effect of Decreasing:** Smoother, less robotic pitching. Reduces landing gear stress during hard stops.

Use Case Recommendations

- **Cinematography: Decrease to 3.5 - 4.0.** Creates very smooth "tilt-up" or "tilt-down" shots.
- **Racing / FPV: Increase to 6.0 - 10.0.** Allows for the instant pitch changes required for high-speed maneuvering and gate navigation.
- **Long-Range Mapping: Keep Default (4.5).** Reliable, stable tracking.

Troubleshooting

- **Scenario:** Drone nose "bobs" up and down specifically when stopping after forward flight.
  - **Diagnosis:** `ATC_ANG_PIT_P` is too high (Angle loop overshoot).
  - **Fix:** Reduce `ATC_ANG_PIT_P` by 0.5.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





ATC_ANG_RLL_P (Roll Axis Angle P Gain)

DEFAULT: 4.5	RANGE: 3.0 12.0

Roll Axis Angle P Gain (ATC_ANG_RLL_P)

Description

`ATC_ANG_RLL_P` is the primary "stiffness" setting for the drone's horizontal attitude. It controls the **Outer Loop** of the flight controller, which is responsible for maintaining the requested roll angle.

This parameter defines how aggressively the drone should rotate to correct a mistake in its tilt. If the pilot (or autopilot) wants a 20-degree lean, but the drone is currently level, `ATC_ANG_RLL_P` determines how fast the drone should start rolling to reach that 20-degree target.

- **Low Value:** The drone feels "soft" or "soupy." It may take a long time to reach the desired tilt, making it feel non-responsive to stick inputs.
- **High Value:** The drone feels "locked-in" and very responsive. Stick inputs feel immediate.
- **Too High:** The drone will oscillate (jitter) in the roll axis, specifically when the pilot releases the sticks or during a hover.

The Mathematics

This is a Proportional (k_P) controller that generates a **Target Angular Velocity** (ω):

$$\omega_roll_target = k_P \cdot (\theta_desired - \theta_actual)$$

Where:

- ω_roll_target is the requested rotation speed (deg/s) sent to the inner Rate Loop.
- $\theta_desired$ is the angle requested by the pilot's sticks or the position controller.
- θ_actual is the current roll estimate from the AHRS/IMU.
- k_P is `ATC_ANG_RLL_P`.
- **Dimensionality:** Since it converts Angle (deg) to Angular Velocity (deg/s), the unit of k_P is $1/s$ (Hertz).

Example: If the drone is 10 degrees off-target and `ATC_ANG_RLL_P` is 4.5, it will demand a rotation rate of 45 deg/s to fix the error.

The Engineer's View

This parameter is an instance of the `AC_P` class, instantiated as `_p_angle_roll` within the `AC_AttitudeControl` library.

It is executed inside `AC_AttitudeControl::update_ang_vel_target_from_att_error()`:

```
// AC_AttitudeControl.cpp
const float angleP_roll = _p_angle_roll.kP() * _angle_P_scale.x;
rate_target_ang_vel.x = angleP_roll * attitude_error_rot_vec_rad.x;
```

Note that the result is often passed through a **Square Root Controller** logic if the error is large, which helps prevent overshooting during massive corrections.

Tuning & Behavior

- **Default Value:** 4.5 (Standard Multirotor)
- **Range:** 3.0 - 12.0
- **Effect of Increasing:** Snappier response to stick inputs. Tighter attitude hold in wind.
- **Effect of Decreasing:** Smoother, more organic flight feel. Reduces the "robotic" look of automated flight.

Use Case Recommendations

- **Cinematography: Decrease to 3.5 - 4.0.**
  - *Why:* Creates smoother transitions when starting and stopping rolls, making the camera footage less jarring.
- **FPV Freestyle / Racing: Increase to 6.0 - 9.0.**
  - *Why:* Provides the "instant" response needed for proximity flying and rapid gates.
- **Autonomous Survey / Mapping: Keep Default (4.5).**
  - *Why:* Provides reliable tracking of the mission grid without exciting mechanical resonances.

Troubleshooting

- **Scenario:** Drone hovers perfectly, but when you let go of the roll stick after a move, it "bounces" or wobbles 2-3 times before settling.
  - **Diagnosis:** `ATC_ANG_RLL_P` is too high (Angle loop oscillation).
  - **Fix:** Reduce `ATC_ANG_RLL_P` by 0.5.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





ATC_ANG_YAW_P (Yaw Axis Angle P Gain)

DEFAULT: 4.5	RANGE: 3.0 12.0

Yaw Axis Angle P Gain (ATC_ANG_YAW_P)

Description

`ATC_ANG_YAW_P` controls the drone's "heading stiffness." It is responsible for making sure the nose of the drone stays pointed where you want it.

If the wind pushes the tail of the drone, or if you command a 90-degree turn, `ATC_ANG_YAW_P` determines how hard the drone should start spinning to align with that new heading.

- **Low Value:** The drone feels "sloppy" in its heading. It may drift away from its heading when accelerating or in wind. The nose feels "vague" to the pilot.
- **High Value:** Heading hold is very precise. The drone snaps to new headings instantly.
- **Too High:** The drone will oscillate its tail (wagging) or make sharp, sudden yaw corrections that cause the whole frame to twitch.

The Mathematics

This Proportional (k_P) gain converts heading error into a **Rotation Rate**:

$$\omega_yaw_target = k_P \cdot (\psi_desired - \psi_actual)$$

Where:

- ω_yaw_target is the requested rotation speed around the vertical axis (deg/s).
- $\psi_desired$ is the target heading (0-360 deg).
- ψ_actual is the current heading (from Compass/EKF).
- k_P is `ATC_ANG_YAW_P`.
- **Dimensionality:** $1/s$ (Hertz).

Yaw Constraint: Heading error is always calculated using "Shortest Path" logic (the drone will never spin more than 180 degrees to fix its heading).

The Engineer's View

This parameter is an instance of the `AC_P` class, member `_p_angle_yaw`, inside `AC_AttitudeControl`.

It is executed in `AC_AttitudeControl::update_ang_vel_target_from_att_error()`:

```
// AC_AttitudeControl.cpp
const float angleP_yaw = _p_angle_yaw.kP() * _angle_P_scale.z;
rate_target_ang_vel.z = angleP_yaw * attitude_error_rot_vec_rad.z;
```

The output `rate_target_ang_vel.z` becomes the input setpoint for the inner **Yaw Rate PID** loop (`ATC_RAT_YAW_*`).

Tuning & Behavior

- **Default Value:** 4.5 (Standard)
- **Range:** 3.0 - 12.0
- **Effect of Increasing:** Tighter heading hold. Better rejection of motor-induced yaw torque (yaw-dip).
- **Effect of Decreasing:** Smoother yaw turns. More "drifty" heading feel.

Use Case Recommendations

- **Precision Photography / Survey: Keep Default (4.5).** Heading accuracy is important for georeferencing, but too much gain can cause vibration in the gimbal.
- **Yaw-Heavy Freestyle: Increase to 6.0 - 8.0.** Provides the crisp yaw stops needed for maneuvers like "Yaw Spins" or "Orbit" mode.
- **Large Professional Drones: Keep Default or decrease to 3.5.** Large frames have high yaw-inertia (due to long arms); pushing them too hard with P-gain leads to "frame flex" and low-frequency oscillations.

Troubleshooting

- **Scenario:** Drone hovers well, but its tail "wags" back and forth once every few seconds.
  - **Diagnosis:** `ATC_ANG_YAW_P` is too high (Angle loop resonance).
  - **Fix:** Reduce `ATC_ANG_YAW_P` by 0.5.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





ATC_BAL_LIM_TC (Pitch control limit time constant)

DEFAULT: 0.5	RANGE: 0.0 5.0	UNITS: s

Pitch control limit time constant (ATC_BAL_LIM_TC)

Description

`ATC_BAL_LIM_TC` is a safety parameter for **Balance Bots** (Segway-style rovers).

A balancing robot stays upright by moving its wheels to stay under its center of mass. If the motors reach 100% power (saturation), the robot can no longer accelerate fast enough to catch itself if it tilts further. This parameter defines how quickly the autopilot should "pull back" the maximum allowed tilt angle when it detects that the motors are working too hard.

- **Low Value (e.g. 0.1):** The drone aggressively limits its lean angle the moment it feels underpowered. High safety, but might feel "weak."
- **High Value (e.g. 2.0):** The drone allows for longer "pushes" near the limit, relying on momentum.

The Mathematics

The pitch limit (θ_lim) is dynamically adjusted based on throttle output:

$$ \dot{\theta}*{lim} = \frac{1}{\text{ATC_BAL_LIM_TC}} \cdot (\theta*{max} - \theta_{lim}) $$

When throttle exceeds `ATC_BAL_LIM_THR`, the limit is reduced. When throttle is low, the limit is "relaxed" back to its user-defined maximum.

Tuning & Behavior

- **Default Value:** 0.5 seconds.
- **Recommendation:** If your balance bot frequently "falls over" forward during hard acceleration, decrease this value to **0.2** to force it to stay more upright.

ATC_BAL_LIM_THR (Pitch control limit throttle threshold)

DEFAULT: 0.6	RANGE: 0.0 1.0

Pitch control limit throttle threshold (ATC_BAL_LIM_THR)

Description

`ATC_BAL_LIM_THR` sets the "Power Safety Line" for balancing robots.

If the motors are using more than 60% (0.6) of their power just to maintain speed, they have very little "headroom" left to fix a balance error. This parameter tells the autopilot at what point it should start worrying and limit how far the robot is allowed to tilt.

Tuning & Behavior

- **Default Value:** 0.6 (60% throttle).
- **Recommendation:** If you have high-torque motors, you can increase this to **0.8**. If your motors are weak, decrease to **0.4**.

ATC_BAL_PIT_FF (Balance Bot Pitch Feed-Forward)

DEFAULT: 0	RANGE: 0 1

Balance Bot Pitch Feed-Forward (ATC_BAL_PIT_FF)

Description

`ATC_BAL_PIT_FF` helps a Balance Bot (Segway-style) stay upright.

It adds throttle based on the current pitch angle. If the bot leans forward, it needs to accelerate forward to catch itself. This term provides the immediate response.

Tuning & Behavior

- **Default Value:** 0.
- **Recommendation:** Fundamental tuning parameter for Balance Bots. Increase until the bot resists falling over.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





ATC_BRAKE (Speed Control Brake Enable)

DEFAULT: 1	RANGE: 0 1

Speed Control Brake Enable (ATC_BRAKE)

Description

`ATC_BRAKE` allows the autopilot to put the car in "Reverse" to slow down.

- **0: Disabled.** The rover will coast to a stop (drag only).
- **1: Enabled.** The rover actively reverses the motor to brake.

Tuning & Behavior

- **Default Value:** 1.
- **Boats:** Be careful. Reversing the prop quickly can cause cavitation or unscrew the prop nut.

ATC_DECEL_MAX (Speed Control Deceleration Max)

DEFAULT: 0	RANGE: 0 10	UNITS: m/s^2

Speed Control Deceleration Max (ATC_DECEL_MAX)

Description

`ATC_DECEL_MAX` is the "ABS" setting.

It limits how hard the rover brakes.

- **0 (Default):** Use the same value as `ATC_ACCEL_MAX`.
- **Value:** Override braking limit.

Tuning & Behavior

- **Default Value:** 0.
- **Recommendation:** Rovers usually brake faster than they accelerate. Set this higher than `ATC_ACCEL_MAX` (e.g. 2.0).

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





ATC_HOVR_ROL_TRM (Hover Roll Trim)

DEFAULT: 0	RANGE: 0 1000	UNITS: cdeg

Hover Roll Trim (ATC_HOVR_ROL_TRM)

Description

`ATC_HOVR_ROL_TRM` is a helicopter-specific parameter. Traditional helicopters produce a side-force from their tail rotor while hovering. To stay in one spot, the helicopter must lean slightly (Roll) in the opposite direction of the tail thrust.

This parameter allows you to pre-set that lean angle so the autopilot doesn't have to "discover" it via the I-term every time you take off.

- **Units:** Centi-degrees (100 = 1 degree).
- **Effect:** Reduces horizontal drift immediately after takeoff in manual modes and improves position-hold performance in autonomous modes.

The Mathematics

The roll trim ϕ_trim is added directly to the Euler roll target ϕ_target:

$$\phi_final = \phi_target + \phi_trim$$

In inverted flight, the sign is automatically flipped by the `inverted_factor`:

$$\text{inverted_factor} = \text{clamp}(2.0 \cdot \cos(\text{roll}), -1.0, 1.0)$$

$$\phi_applied = \phi_trim \cdot \text{inverted_factor}$$

The Engineer's View

This parameter is the `_hover_roll_trim` member in `AC_AttitudeControl_Heli`.
It is retrieved via `get_roll_trim_rad()` and added to the attitude target in the `input_euler_angle_roll_pitch_euler_rate_yaw` and `input_euler_angle_roll_pitch_yaw` functions.

Tuning & Behavior

- **Default Value:** 0 (or board specific)
- **Range:** 0 to 1000 cdeg (0 to 10 degrees).
- **Typical Value:** Most helicopters require 3 to 5 degrees (**300 to 500 cdeg**) of roll trim to hover stationary.
- **Tuning Procedure:**
  1. Hover in **Stabilize** or **Acro** mode.
  2. Observe which way the heli drifts horizontally.
  3. Adjust `ATC_HOVR_ROL_TRM` until the heli stays stationary with the roll stick centered.
  4. Verify that in **Loiter**, the heli doesn't "lean" excessively or drift when first engaged.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





ATC_INPUT_TC (Attitude Control Input Time Constant)

DEFAULT: 0.15	RANGE: 0 1.0	UNITS: s

Attitude Control Input Time Constant (ATC_INPUT_TC)

Description

`ATC_INPUT_TC` is the "smoothness filter" for the pilot's fingers. It determines how fast the flight controller attempts to reach the attitude requested by your sticks.

Even if you move your stick instantly from 0 to 100%, the drone won't instantly command 100%. `ATC_INPUT_TC` defines the time it takes for the software target to catch up to your actual stick position. It removes the "jerkiness" from manual flight, making the drone feel like it has weight and momentum.

- **Low Value (e.g., 0.05):** The drone feels very "twitchy" and responsive. It reacts to every tiny movement of your hand.
- **High Value (e.g., 0.5):** The drone feels very "soft" or "delayed." It will take a long time to start and stop its movements.
- **Default (0.15):** A "Medium" setting that feels natural for most multirotor pilots.

The Mathematics

This parameter acts as the time constant (τ) for a first-order low-pass filter on the requested target:

$$\theta_target(t) = \theta_target(t - dt) + \frac{dt}{\tau + dt} \cdot (\theta_pilot - \theta_target(t - dt))$$

Where:

- θ_pilot is the raw angle requested by your stick position.
- θ_target is the smoothed value sent to the flight control loops.
- τ is `ATC_INPUT_TC`.

Physical Meaning: A time constant of 0.15s means that if you hold the stick at a fixed position, the drone will reach 63% of that target angle in 150 milliseconds.

The Engineer's View

This parameter is the `_input_tc` member in `AC_AttitudeControl`.

It is used in the **Input Shaper** for all manual modes (Loiter, AltHold, Stabilize):

```
// AC_AttitudeControl.cpp
_euler_rate_target.x = input_shaping_angle(wrap_PI(euler_roll_angle - _euler_angle_target.x), _input_tc, euler_accel.x, _euler_rate
```

By filtering the *input* rather than the output, ArduPilot ensures that the PID loops are always following a perfectly smooth, continuous path, which significantly reduces motor noise and frame stress compared to raw stick commands.

Tuning & Behavior

- **Default Value:** 0.15s
- **Range:** 0.0 - 1.0s
- **Effect of Increasing:** Smoother, more professional cinematic look. Reduces battery sag during aggressive stick movements.
- **Effect of Decreasing:** Tighter stick feel. Essential for racing or flying in tight spaces where you need immediate reaction.

Use Case Recommendations

- **Racing / FPV Freestyle: Decrease to 0.05 - 0.10.** You want the drone to follow your brain with zero delay.
- **Cinematography: Increase to 0.20 - 0.50.** This makes the drone feel "heavy," like a full-sized helicopter. It prevents small finger shakes from showing up on the camera.
- **Beginner Pilots: Set to 0.20.** Makes the drone easier to control by slowing down its reactions to "panic" stick movements.

Troubleshooting

- **Scenario:** The drone feels "delayed" or "disconnected." I move the stick, but I have to wait for the drone to react.
  - **Diagnosis:** `ATC_INPUT_TC` is set too high.
  - **Fix:** Reduce `ATC_INPUT_TC` to 0.10.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY

ATC_LAND_P_MULT (Landed Pitch Gain Multiplier)

DEFAULT: 1.0	RANGE: 0.25 1.0

Landed Pitch Gain Multiplier (ATC_LAND_P_MULT)

Description

`ATC_LAND_P_MULT` manages the stability of your drone's nose and tail during the critical landing phase.

When your drone's landing gear hits the ground, the air resistance is replaced by solid mechanical contact. If your pitch gains are high, the drone might feel the impact of the ground and try to "stabilize" against it by pitching up or down. Because the ground is rigid, this creates a high-speed vibration loop that can cause the drone to "dance" or flip forward/backward on its gear.

This parameter allows you to automatically scale down the pitch authority the moment the landing detector is triggered.

- **Set to 1.0 (Default):** No reduction. The drone maintains its full flight authority on the ground.
- **Set to 0.5:** The drone's pitch gains are halved on touchdown. This makes the drone feel "heavy" and settled, preventing it from bobbing its head or flipping over.

The Mathematics

The effective pitch gains are scaled by this multiplier (M) during the landing transition:

$$ \text{Gain}_{effective} = \text{Gain}_{flight} \cdot M $$

Where M is `ATC_LAND_P_MULT`.

Dynamic Transition: ArduPilot ramps the gain down over a short period of time (defined by the attitude shaper) to avoid a sudden "step" in motor power that could itself trigger a bounce.

The Engineer's View

This parameter is the `_land_pitch_mult` member in `AC_AttitudeControl`.

It is applied in `AC_AttitudeControl::landed_gain_reduction()`:

```
// AC_AttitudeControl.cpp
Vector3f scale_mult = VECTORF_111 * (1.0 - _landed_gain_ratio) + Vector3f(_land_roll_mult, _land_pitch_mult, _land_yaw_mult) * _lan
set_PD_scale_mult(scale_mult);
set_angle_P_scale_mult(scale_mult);
```

This ensures that any nose-heavy or tail-heavy imbalances (which the I-term usually fights) don't result in a violent motor surge when the drone is physically constrained by the landing pad.

Tuning & Behavior

- **Default Value:** 1.0 (No reduction)
- **Range:** 0.25 - 1.0
- **Effect of Decreasing:** Drone becomes much more stable on the ground. Prevents "nose-dives" during auto-landing.
- **Effect of Increasing:** Maintains full authority. Useful for landing on inclines (like a boat deck) where you must fight the tilt until the moment of disarm.

Use Case Recommendations

- **Cinematography (Gimbal Safety): Decrease to 0.5.** Ensures the drone stays perfectly flat and calm on the ground, protecting the gimbal from accidental "nose-bobs."
- **Speed / Freestyle Quads: Keep at Default.** These frames are so light that they rarely suffer from ground resonance.
- **Large Hybrid/Fixed-Wing VTOL: Decrease to 0.4.** High-inertia vehicles with large wings can catch the wind on the ground; reducing the gain prevents the motors from fighting that wind in a way that would over-stress the airframe.

Troubleshooting

- **Scenario:** On auto-landing, the drone touches down and then immediately "pitches forward" and strikes its front propellers on the ground.
  - **Diagnosis:** Longitudinal Ground Resonance or I-term windup.
  - **Fix:** Reduce `ATC_LAND_P_MULT` to 0.5.
  - **Also check:** `ATC_THR_MIX_MIN`.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





ATC_LAND_R_MULT (Landed Roll Gain Multiplier)

DEFAULT: 1.0	RANGE: 0.25 1.0

Landed Roll Gain Multiplier (ATC_LAND_R_MULT)

Description

`ATC_LAND_R_MULT` is a "de-tuning" parameter for the landing phase.

Your drone's PID gains are tuned for flight (where the drone is surrounded by air). When the drone touches the ground, the physics change instantly. The rigid ground acts like a massive spring. If your roll gains are high, the drone might feel a tiny vibration from the ground and try to "stabilize" against it. This creates a feedback loop known as **Ground Resonance**, where the drone starts shaking violently or "hopping" from side to side on its landing gear.

This parameter allows you to automatically lower your roll authority the moment the landing gear touches the earth.

- **Set to 1.0 (Default):** No reduction. The drone stays at full authority. Use this if your drone is very heavy and rigid.
- **Set to 0.5:** The drone's roll gains are cut in half on the ground. Makes for much "quieter," safer landings.

The Mathematics

The effective roll gains (P, I, D) are scaled by this multiplier (M) when the landing detector is active:

$$ \text{Gain}_{effective} = \text{Gain}_{flight} \cdot M $$

Where M is `ATC_LAND_R_MULT`.

Interpolation: ArduPilot doesn't just "snap" the gain down. It uses a transition logic (tied to `ATC_INPUT_TC`) to smoothly ramp the gains down as the drone settles onto its gear.

The Engineer's View

This parameter is the `_land_roll_mult` member in `AC_AttitudeControl`.

It is applied in `AC_AttitudeControl::landed_gain_reduction()`:

```
// AC_AttitudeControl.cpp
Vector3f scale_mult = VECTORF_111 * (1.0 - _landed_gain_ratio) + Vector3f(_land_roll_mult, _land_pitch_mult, _land_yaw_mult) * _lanc
set_PD_scale_mult(scale_mult);
set_angle_P_scale_mult(scale_mult);
```

This logic ensures that both the **Outer Angle Loop** and the **Inner Rate Loop** are softened, preventing any software-driven oscillations from reaching the motors while the drone is physically constrained by the ground.

Tuning & Behavior

- **Default Value:** 1.0 (Standard)
- **Range:** 0.25 - 1.0
- **Effect of Decreasing:** Drone becomes much "lazier" and more stable on the ground. Prevents tip-overs during auto-landings.
- **Effect of Increasing:** Maximum authority on touchdown. Useful for "hard" landings on uneven or rocky terrain where you must fight to stay upright.

Use Case Recommendations

- **Lightweight Racers / Carbon Frames: Decrease to 0.5 - 0.7.** These frames are very stiff and prone to ground resonance. Lowering the gain makes the drone "stick" to the ground.
- **Drones with Long, Springy Legs: Decrease to 0.4.** Springy legs amplify vibrations; reducing the gain is essential to keep the drone from "pogo-sticking."
- **Heavy Industrial Drones: Keep at 1.0.** High inertia usually dampens ground vibrations naturally.

Troubleshooting

- **Scenario:** On every landing, the drone touches down and then starts a rapid "shiver" or vibration that won't stop until it disarms.
  - **Diagnosis:** Ground Resonance. The PID loops are too fast for the physical contact with the earth.
  - **Fix:** Reduce `ATC_LAND_R_MULT` and `ATC_LAND_P_MULT` to 0.6.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY

ATC_LAND_Y_MULT (Landed Yaw Gain Multiplier)

DEFAULT: 1.0	RANGE: 0.25 1.0

Landed Yaw Gain Multiplier (ATC_LAND_Y_MULT)

Description

`ATC_LAND_Y_MULT` handles the drone's heading stability during the landing phase.

When a multirotor touches the ground, it often experiences "Yaw Torque" from friction between the legs and the earth. If your yaw gains are high, the drone will try to fight this friction to keep its nose straight. This can lead to a violent "spinning" or "twisting" vibration as the drone tries to yaw against the immovable ground.

This parameter allows the flight controller to automatically lower its yaw authority the moment the landing gear makes contact.

- **Set to 1.0 (Default):** No reduction. The drone fights to maintain its heading even on the ground.
- **Set to 0.5:** The drone's yaw authority is cut in half on the ground. This allows the drone to "relax" its heading slightly to match the ground friction, preventing mechanical stress on the frame and landing gear.

The Mathematics

The effective yaw gains are scaled by this multiplier (M) during the landing sequence:

$$ \text{Gain}_{effective} = \text{Gain}_{flight} \cdot M $$

Where M is `ATC_LAND_Y_MULT`.

Safety Note: Unlike Roll and Pitch, where a loss of control means a flip, a loss of Yaw control on the ground usually just means the nose rotates by a few degrees. Therefore, `ATC_LAND_Y_MULT` can often be set lower than the other multipliers without significant risk.

The Engineer's View

This parameter is the `_land_yaw_mult` member in `AC_AttitudeControl`.

It is applied in `AC_AttitudeControl::landed_gain_reduction()`:

```
// AC_AttitudeControl.cpp
Vector3f scale_mult = VECTORF_111 * (1.0 - _landed_gain_ratio) + Vector3f(_land_roll_mult, _land_pitch_mult, _land_yaw_mult) * _lan
set_PD_scale_mult(scale_mult);
set_angle_P_scale_mult(scale_mult);
```

This logic ensures that any motor torque imbalances don't cause the drone to "pivot" or "spin out" violently when one leg has more friction with the ground than the others.

Tuning & Behavior

- **Default Value:** 1.0 (Standard)
- **Range:** 0.25 - 1.0
- **Effect of Decreasing:** Drone becomes much "lazier" in its heading on the ground. Prevents tail-induced vibrations during touchdown.
- **Effect of Increasing:** Maintains full heading hold. Useful for landing on moving or slippery surfaces where the drone must maintain its heading relative to the vehicle below it.

Use Case Recommendations

- **Standard Multirotor: Set to 0.5 - 0.7.** This is generally safer and reduces the "growling" sound often heard when multirotors land on concrete.
- **Landing on Moving Boats: Keep at 1.0.** You want the drone to fight the friction and stay aligned with the boat's direction until it is fully disarmed.
- **Tricopters (Servo Yaw): Set to 0.4.** Servos can easily strip their gears if they try to yaw against ground friction; reducing the gain is critical for mechanical protection.

Troubleshooting

- **Scenario:** On landing, the drone "twists" its legs or frame violently just before the motors stop.
  - **Diagnosis:** Yaw ground resonance or motor torque spikes.
  - **Fix:** Reduce `ATC_LAND_Y_MULT` to 0.5.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





ATC_PIRO_COMP (Pirouette Compensation Enable)

DEFAULT: 0	RANGE: 0 1

Pirouette Compensation Enable (ATC_PIRO_COMP)

Description

`ATC_PIRO_COMP` is a specialized feature for **Traditional Helicopters**. In physics, a helicopter spinning rapidly around its vertical axis (a pirouette) experiences intense cross-coupling between its roll and pitch axes.

Without compensation, if a helicopter is tilted forward and starts to spin, the tilt will appear to "wobble" or "cone" as the helicopter rotates, potentially leading to a crash. `ATC_PIRO_COMP` enables advanced math that keeps the tilt direction fixed relative to the earth, even while the airframe is spinning at extreme rates.

- **Enabled (1):** The helicopter stays perfectly tilted in the desired direction regardless of its rotation speed. Essential for advanced 3D aerobatics.
- **Disabled (0):** Standard flight mode. The pilot must manually correct for coning during spins.

Note: For Multirotors (Quads, Octos), this compensation is handled inherently by the Body-Frame controller architecture and this specific parameter is usually not present or needed.

The Mathematics

The compensation logic involves rotating the roll and pitch targets into the moving frame of the helicopter:

$$ \begin{bmatrix} \text{Roll}_{comp} \\ \text{Pitch}_{comp} \end{bmatrix} = \mathbf{R}_z(\psi) \cdot \begin{bmatrix} \text{Roll}_{target} \\ \text{Pitch}_{target} \end{bmatrix} $$

Where $\mathbf{R}_z(\psi)$ is the rotation matrix around the yaw axis (ψ). This ensures that the cyclic commands (the "tilt" instructions) stay aligned with the earth's horizon rather than spinning with the heli's nose.

The Engineer's View

This parameter is the `_piro_comp_enabled` member in `AC_AttitudeControl_Heli`.

It affects how the attitude error is calculated in `AC_AttitudeControl_Heli::update_ang_vel_target_from_att_error()`:

```
// AC_AttitudeControl_Heli.cpp
if (_piro_comp_enabled) {
    // Perform frame rotation for cyclic commands
    // ...
}
```

This is particularly important for helicopters equipped with a **Mechanical Flybar**, as the physical flybar has its own gyroscopic inertia that must be accounted for by the software.

Tuning & Behavior

- **Default Value:** 0 (Disabled)
- **Range:** 0 (Off) or 1 (On)
- **Effect of Enabling:** Much more stable heading-independent flight. The heli feels more like a multirotor during fast spins.
- **Effect of Disabling:** "Classical" helicopter flight feel.

Use Case Recommendations

- **3D Aerobatics (Smack flight): Always Enable (1).** Instant heading-locked attitude is required for maneuvers like "Piro-flips."
- **Scale / Precision Helicopter: Enable (1).** Makes the helicopter much more predictable in wind during turns.
- **Legacy Flybar Helicopters: Keep at 0.** These rely on the physical stability of the mechanical flybar; software piro-comp can sometimes fight the mechanical stabilization.

Troubleshooting

- **Scenario:** During a fast yaw spin, the helicopter starts to wobble in a circular motion (coning), eventually tilting further and further.
  - **Diagnosis:** Pirouette coupling. The controller isn't keeping up with the rotation.
  - **Fix:** Set `ATC_PIRO_COMP` to 1.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





ATC_RATE_FF_ENAB (Rate Feedforward Enable)

DEFAULT: 1	RANGE: 0 1

Rate Feedforward Enable (ATC_RATE_FF_ENAB)

Description

`ATC_RATE_FF_ENAB` is one of the most important architectural switches in ArduPilot. It determines how the drone's "brain" talks to its "muscles."

- **Enabled (1):** The drone uses **Body-Frame Feedforward**. This means that when you move the stick, the command is instantly translated into a target rotation rate that the drone starts to follow immediately. This is the modern standard for all multirotors.
- **Disabled (0):** The drone uses **Euler-Angle Lead**. This is a legacy method used in older flight controllers where the drone waits to see an error in its angle before it starts moving the motors.

WARNING: For almost all modern drones, this parameter **MUST remain Enabled (1)**. Disabling it will make the drone feel incredibly sluggish, like it's flying in a thick liquid, and it will be unable to handle rapid maneuvers or high winds.

The Mathematics

When enabled, the controller bypasses the slow error-accumulation of the outer loop for stick inputs. The target rate (ω_target) is derived directly from the pilot's stick position and the shaper:

$$\omega_target = \text{Shaper}(\text{Stick_Pos}, \dots)$$

When disabled, the target rate is purely a function of angle error:

$$ \omega_{target} = k_P \cdot (\text{Angle}*{target} - |text{Angle}*{actual}) $$

The Engineer's View

This parameter is the `_rate_bf_ff_enabled` member in `AC_AttitudeControl`.

It is a core conditional in every input function (e.g., `input_euler_angle_roll_pitch_yaw`):

```
// AC_AttitudeControl.cpp
if (_rate_bf_ff_enabled) {
    // Modern: Use input shaper and feedforward
    _euler_rate_target.x = input_shaping_angle(...);
    // ...
} else {
    // Legacy: Euler target is simple angle
    _euler_angle_target.x = euler_roll_angle;
    _euler_rate_target.zero();
}
```

By enabling feedforward, ArduPilot allows the **Rate PID** loops to operate with zero-lag on pilot commands, while still using the **Angle Loop** to correct for any deviations from that command. This "Parallel Path" architecture is what gives ArduPilot its legendary stability.

Tuning & Behavior

- **Default Value:** 1 (Enabled)
- **Range:** 0 (Legacy) or 1 (Modern)
- **Effect of Enabling:** Tighter, more direct stick feel. Much better wind rejection. Required for AutoTune to work correctly.
- **Effect of Disabling:** "Loose," "Elastic" feel. Drone may overshoot targets significantly.

Use Case Recommendations

- **Standard Drone: Keep Enabled (1).**
- **Legacy Porting:** If you are porting very old code that doesn't support body-frame FF, you might set this to 0, but this is highly discouraged for modern flight.
- **Debugging:** Only disable (0) if you are trying to isolate a problem with the outer angle loop and want to remove all lead-authority for testing.

Troubleshooting

- **Scenario:** I moved my sticks, but it takes nearly a full second for the drone to start leaning.
  - **Diagnosis:** `ATC_RATE_FF_ENAB` has accidentally been set to 0.
  - **Fix:** Set to 1.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





ATC_RATE_P_MAX (Pitch Rate Max)

DEFAULT: 0.0	RANGE: 0 1080	UNITS: deg/s

Pitch Rate Max (ATC_RATE_P_MAX)

Description

`ATC_RATE_P_MAX` defines the "pitch speed limit" for your drone. It determines the maximum degrees per second the drone is allowed to tilt forward or backward.

This parameter is essential for keeping the drone within its physical flight envelope. If you pitch too fast, you risk motor saturation (where motors on one side hit 100% and cannot lift the drone anymore) or sensor loss. `ATC_RATE_P_MAX` acts as a safety governor, ensuring that even if you slam the stick forward, the drone will only accelerate its rotation up to this specific speed.

- **Low Value (e.g. 60 deg/s):** The drone pitches very slowly. Ideal for heavy-lift commercial drones and passenger vehicles.
- **High Value (e.g. 360 deg/s):** The drone can perform one full forward flip every second. Essential for high-performance flight.
- **Disabled (0):** The software does not limit pitch rotation speed.

The Mathematics

This parameter caps the target angular velocity (ω_pitch) calculated by the flight controller:

$$\omega_pitch_target = \text{constrain}(\omega_pitch_calculated, -\text{RATE_P_MAX}, \text{RATE_P_MAX})$$

Where:

- RATE_P_MAX is in degrees per second.
- $\omega_pitch_calculated$ is the desired rotation rate from stick input or the angle loop.

Combined Limits: ArduPilot enforces this limit in a way that respects the drone's total thrust. If you are at the limit of both Roll and Pitch, the controller will prioritize whichever axis has the larger error while keeping the total vector within safe bounds.

The Engineer's View

This parameter is the `_ang_vel_pitch_max` member in `AC_AttitudeControl`.

It is applied in `AC_AttitudeControl::ang_vel_limit()`:

```
// AC_AttitudeControl.cpp
if (!is_zero(ang_vel_pitch_max)) {
    euler_rad.y = thrust_vector_ang_vel.y * ang_vel_pitch_max / thrust_vector_length;
}
```

This limit is particularly important during **High Speed Braking**. When the drone stops moving forward, it must pitch up aggressively. If `RATE_P_MAX` is too low, the drone may take too long to pitch up and could potentially overshoot its stopping point or strike an obstacle.

Tuning & Behavior

- **Default Value:** 0.0 (Standard)
- **Range:** 0 - 1080 deg/s
- **Effect of Increasing:** Allows for sharper, more immediate forward/backward maneuvers. Necessary for high-speed automated missions.
- **Effect of Decreasing:** Smoother pitch transitions. Reduces landing gear stress during hard stops.

Use Case Recommendations

- **Large Professional Octocopters: Set to 60 - 90 deg/s.** These large vehicles have massive inertia; attempting to pitch them faster than 90 deg/s is inefficient and can cause structural fatigue.
- **Standard GPS Multirotor: Set to 180 deg/s.** Provides a good balance of agility and safety.
- **Freestyle / Racing Quads: Keep at 0.0.** Performance is the priority; the pilot is responsible for managing the rotation speed.

Troubleshooting

- **Scenario:** During an Auto mission, the drone seems to "glide" through waypoints instead of turning sharply.
  - **Diagnosis:** `ATC_RATE_P_MAX` (and potentially `ATC_RATE_R_MAX`) is set too low, limiting the drone's ability to change its attitude quickly.
  - **Fix:** Increase `ATC_RATE_P_MAX` to 180.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





ATC_RATE_R_MAX (Roll Rate Max)

DEFAULT: 0.0	RANGE: 0 1080	UNITS: deg/s

Roll Rate Max (ATC_RATE_R_MAX)

Description

`ATC_RATE_R_MAX` is the "top speed" for your drone's rotation. It defines the maximum number of degrees per second the drone is allowed to roll.

If you are flying in a GPS mode (like Loiter) or a stabilized mode, the flight controller will never request a rotation faster than this limit, even if you push your stick to the limit. It ensures the drone remains controllable and doesn't rotate so fast that the EKF (position estimator) loses its orientation.

- **Low Value (e.g. 90 deg/s):** The drone rotates very slowly and safely. Ideal for beginners and heavy cargo drones.
- **High Value (e.g. 360 deg/s):** The drone rolls very quickly (one full circle per second). Essential for FPV and aggressive maneuvers.
- **Disabled (0):** The software does not limit rotation speed. The drone will roll as fast as its motors can physically push it.

The Mathematics

This parameter acts as a hard "cap" on the requested angular velocity (ω):

$$\omega_roll_target = \text{constrain}(\omega_roll_calculated, -\text{RATE_R_MAX}, \text{RATE_R_MAX})$$

Where:

- RATE_R_MAX is in degrees/second.
- $\omega_roll_calculated$ is the output of the pilot's stick scaling or the outer angle loop.

Note: This limit is applied *before* the rate PID loop, ensuring the PID controller doesn't "wind up" trying to reach an impossible rotation speed.

The Engineer's View

This parameter is the `_ang_vel_roll_max` member in `AC_AttitudeControl`.

It is enforced in `AC_AttitudeControl::ang_vel_limit()`:

```
// AC_AttitudeControl.cpp
if (!is_zero(ang_vel_roll_max)) {
    euler_rad.x = thrust_vector_ang_vel.x * ang_vel_roll_max / thrust_vector_length;
}
```

ArduPilot uses a **Spherical Limit** for Roll and Pitch combined. If both axes are at their limits, the controller doesn't just clip them; it scales them both down to ensure the "total" rotation speed remains within the drone's physical capabilities.

Tuning & Behavior

- **Default Value:** 0.0 (Standard)
- **Range:** 0 - 1080 deg/s
- **Effect of Increasing:** Allows for faster flips and sharper turns.
- **Effect of Decreasing:** Limits how fast the drone can change its tilt. Makes the drone safer for novice pilots.

Use Case Recommendations

- **Cinematography / Inspection: Set to 60 - 90 deg/s.** Limits any accidental "jerks" of the stick, ensuring the horizon always moves in a smooth, cinematic way.
- **Autonomous Search & Rescue: Set to 180 deg/s.** Provides a good balance between scanning speed and stabilization accuracy.
- **Standard Multirotor: Keep at 0.0 or 360.** Standard 360 deg/s is common for most hobbyist flight.

Troubleshooting

- **Scenario:** I am trying to do a flip, but the drone rolls extremely slowly, like it's fighting me.
  - **Diagnosis:** `ATC_RATE_R_MAX` is set too low.
  - **Fix:** Increase `ATC_RATE_R_MAX` to 360 or 0.
- **Scenario:** During high-speed turns, the EKF loses its position and the drone enters "Land" mode.
  - **Diagnosis:** The drone is rotating faster than the sensors can accurately track.
  - **Fix:** Reduce `ATC_RATE_R_MAX` to 180.
  - **Also check:** `ATC_RATE_P_MAX`.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





ATC_RATE_Y_MAX (Yaw Rate Max)

DEFAULT: 0.0	RANGE: 0 1080	UNITS: deg/s

Yaw Rate Max (ATC_RATE_Y_MAX)

Description

`ATC_RATE_Y_MAX` is the "rotation speed limiter" for your drone's heading. It determines the maximum speed (in degrees per second) that the drone is allowed to spin around its center.

Yaw is generally the weakest axis for a multirotor because it relies on torque reaction. If a drone spins too fast, it creates a massive amount of centrifugal force on the motor arms and can easily cause "vertical wash," where the drone drops out of the sky because the motors are busy spinning the drone instead of lifting it. `ATC_RATE_Y_MAX` ensures the drone stays in a safe speed range, even if the pilot pushes the yaw stick all the way to the side.

- **Low Value (e.g. 45 deg/s):** Very slow, stately rotation. Perfect for smooth cinematic pans.
- **High Value (e.g. 180 deg/s):** Fast rotation (one full circle every 2 seconds). Good for general flight and rapid target re-acquisition.
- **Disabled (0):** The drone will spin as fast as its motor torque allows.

The Mathematics

This parameter acts as a hard ceiling on the requested heading rotation rate (ω_yaw):

$$\omega_yaw_target = \text{constrain}(\omega_yaw_calculated, -\text{RATE_Y_MAX}, \text{RATE_Y_MAX})$$

Where:

- `RATE_Y_MAX` is in degrees per second.
- $\omega_yaw_calculated$ is derived from the yaw stick or the heading controller.

Torque Prioritization: If the drone is at its `RATE_Y_MAX` limit, ArduPilot will often reduce the yaw torque slightly if more Roll or Pitch authority is needed to keep the drone level.

The Engineer's View

This parameter is the `_ang_vel_yaw_max` member in `AC_AttitudeControl`.

It is enforced in `AC_AttitudeControl::ang_vel_limit()`:

```
// AC_AttitudeControl.cpp
if (!is_zero(ang_vel_yaw_max)) {
    euler_rad.z = constrain_float(euler_rad.z, -ang_vel_yaw_max, ang_vel_yaw_max);
}
```

Unlike Roll and Pitch, which are often scaled together, Yaw is limited independently. This prevents high yaw rates from "stealing" the power needed for roll and pitch stability.

Tuning & Behavior

- **Default Value:** 0.0 (Standard)
- **Range:** 0 - 1080 deg/s
- **Effect of Increasing:** Allows for faster head-turns and "Yaw Spins." Necessary for tracking fast-moving objects.
- **Effect of Decreasing:** Smoother pans and less vertical altitude loss during aggressive spinning.

Use Case Recommendations

- **Professional Cinematography: Set to 30 - 45 deg/s.** This prevents accidental "jerky" heading moves from ruining a shot, ensuring every rotation is a smooth, high-quality pan.
- **Standard GPS Flying: Set to 90 - 120 deg/s.** Provides a responsive tail without risking motor saturation.
- **FPV Freestyle: Keep at 0.0 or set >360.** Pilots want the nose to follow their fingers with no artificial speed limits.

Troubleshooting

- **Scenario:** Every time I spin the drone 360 degrees, it drops significantly in altitude.
  - **Diagnosis:** Yaw torque is starving the lift motors. The spin is too fast for the available motor power.
  - **Fix:** Reduce `ATC_RATE_Y_MAX` to 60 deg/s.
- **Scenario:** My drone takes a very long time to turn its nose to the next waypoint in an Auto mission.
  - **Diagnosis:** `ATC_RATE_Y_MAX` is set too low.
  - **Fix:** Increase `ATC_RATE_Y_MAX` to 120.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





ATC_RAT_PIT_D (Pitch Axis Rate D Gain)

DEFAULT: 0.0036	RANGE: 0.0 0.05

Pitch Axis Rate D Gain (ATC_RAT_PIT_D)

Description

`ATC_RAT_PIT_D` is the vertical "damper" for your drone's nose. In physics, the **Derivative (D)** term acts like a brake that only activates when the drone is moving too fast toward its target.

If you command the drone to level out after a fast forward dash, it has a lot of pitch momentum. Without D-gain, the nose would swing back up, shoot past the level point, and then "bob" (oscillate) several times before settling. The D-gain "senses" this rapid swing and adds pre-emptive resistance, ensuring the drone stops perfectly level on the first try.

- **Low Value:** The drone will "bob its head" like a toy when you stop pitching. It feels loose and bouncy in the air.
- **High Value:** The drone feels extremely stable and "robotic." Flips stop exactly where you want them.
- **Too High:** The drone will emit a high-pitched "growl" or motor shriek. This is dangerous as it can overheat the motors and ESCs within seconds.

The Mathematics

The D-term is proportional to how fast the pitch error is changing:

$$ \text{Out}*D = k*_D \|cdot \|*frac{d(\|*omega{target} - \omega_{actual})}{dt} $$

Where:

- k_D is `ATC_RAT_PIT_D`.
- The signal is passed through the `ATC_RAT_PIT_FLTD` filter to remove noise.

Intuition: The D-term opposes the *speed* of the rotation. If you are rotating too fast toward the target, the D-term pushes *against* the movement to slow it down before you overshoot.

The Engineer's View

This parameter is the `_kd` member of the `_pid_rate_pitch` object.

It is executed in `AC_PID::update_all()`:

```
// AC_PID.cpp
if (is_positive(dt)) {
    float derivative = (error - _last_error) / dt;
    _derivative = filtered_derivative;
}
return ... + _derivative * _kd;
```

In multirotors, the Pitch axis often needs slightly different D-gain than the Roll axis because the weight of the drone (battery and camera) is usually distributed along the longitudinal axis, creating a different moment of inertia.

Tuning & Behavior

- **Default Value:** 0.0036 (Highly sensitive)
- **Range:** 0.0 - 0.05
- **Effect of Increasing:** Crisper stops after forward flight. Less vertical altitude loss during hard braking.
- **Effect of Decreasing:** Smoother, "softer" flight feel. Safer for older motors or low-quality propellers.

Use Case Recommendations

- **Large Professional Multirotors: Keep very low or set to 0.0.** Large props cannot physically react fast enough for D-gain to be effective; it usually just causes wasted heat.
- **FPV Racing / Cinema Whoops: Increase to 0.006 - 0.008.** High performance depends on the drone stopping its pitch exactly on command to maintain the correct forward speed.
- **Long-Range Mapping: Keep at Default.** Provides a stable platform without motor stress.

Troubleshooting

- **Scenario:** Drone nose "bobs" up and down 2-3 times every time you release the pitch stick.
  - **Diagnosis:** `ATC_RAT_PIT_D` is too low (Underdamped).
  - **Fix:** Increase `ATC_RAT_PIT_D` by 0.0005.
- **Scenario:** Motors sound like they are "grinding" only when you are hovering in wind.
  - **Diagnosis:** `ATC_RAT_PIT_D` is too high, amplifying turbulence noise.
  - **Fix:** Reduce `ATC_RAT_PIT_D` immediately.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





ATC_RAT_PIT_D_FF (Pitch Axis Rate Derivative Feed-Forward)

DEFAULT: 0.0	RANGE: 0 0.02

Pitch Axis Rate Derivative Feed-Forward (ATC_RAT_PIT_D_FF)

Description

`ATC_RAT_PIT_D_FF` is an advanced predictive control term for your drone's forward and backward rotation. While standard Feed-Forward (`ATC_RAT_PIT_FF`) looks at your desired pitch speed, the **Derivative Feed-Forward (D_FF)** looks at how fast you are *moving the stick*.

If you suddenly slam the pitch stick forward to dodge an obstacle, the D_FF term instantly commands a surge of power to the rear motors to tilt the nose down. It provides the "kick" needed to overcome the longitudinal inertia of the drone's frame and battery. This results in a drone that follows your commands with zero perceived lag.

- **Low Value (0.0):** Standard behavior. The drone relies on error-correction and base feed-forward.
- **High Value:** Creates a sharp "snap" at the beginning of pitch moves.
- **Too High:** Can cause "kickback" or jerky, non-linear movements in the pitch axis.

The Mathematics

The D_FF term is proportional to the rate of change of the pitch setpoint (ω_pit_target):

$$ \text{Out}*{D_FF} = \frac{d(\omega*{pit_target})}{dt} \cdot k_{D_FF} $$

Where:

- k_D_FF is `ATC_RAT_PIT_D_FF`.
- $\frac{d(\omega_pit_target)}{dt}$ is the rate of change of the requested pitch speed.

Vibration Immunity: Because this term is derived from the **Setpoint** (pilot stick) rather than the **Gyro** (sensor), it does not amplify motor vibration. You can use D_FF to get a "stiff" response without the risk of motor heating associated with standard D-gains.

The Engineer's View

This parameter maps to `_kdff` in the `_pid_rate_pitch` object.

It is executed in `AC_PID::update_all()`:

```
// AC_PID.cpp
if (is_positive(dt)) {
    float target_derivative = (target - _last_target) / dt;
    _pid_info.D_FF = target_derivative * _kdff;
}
```

In ArduPilot, D_FF is especially useful for "Dead Cat" or long-range frames where the Pitch axis has significantly more mass to move than the Roll axis. It helps "lead" the heavier longitudinal axis into maneuvers.

Tuning & Behavior

- **Default Value:** 0.0 (Standard)
- **Range:** 0.0 - 0.02
- **Effect of Increasing:** Snappier longitudinal transitions. Reduced altitude loss during aggressive pitch maneuvers.
- **Effect of Decreasing:** Smoother, more organic pitch behavior.

Use Case Recommendations

- **Speed Runs / Drag Racing: Increase to 0.01.** Essential for performing the instant, aggressive tilts needed to maximize acceleration from a hover.
- **Cinematography: Keep at 0.0.** High D_FF gains can make the horizon "snap" in a way that looks artificial on video.
- **Standard Multirotor: Keep at 0.0.** Standard tuning is safer and easier to manage for most pilots.

Troubleshooting

- **Scenario:** Drone nose "bobs" or twitches whenever I move the stick, even slightly.
  - **Diagnosis:** `ATC_RAT_PIT_D_FF` is likely too high, reacting to tiny finger twitches.
  - **Fix:** Reduce `ATC_RAT_PIT_D_FF` to 0.002 or set back to 0.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY

ATC_RAT_PIT_FF (Pitch Axis Rate Feed-Forward Gain)

DEFAULT: 0.0	RANGE: 0.0 0.5

Pitch Axis Rate Feed-Forward Gain (ATC_RAT_PIT_FF)

Description

`ATC_RAT_PIT_FF` is the "accelerator pedal" for your drone's pitch axis. While the P, I, and D terms wait for the drone to *miss* its target rotation speed before they correct it, the Feed-Forward (FF) term acts pre-emptively.

When you push the stick forward, the FF term instantly calculates how much power to send to the rear motors to tilt the drone forward *at the same moment* your finger moves. This eliminates the "mushy" or "laggy" feeling often associated with flight controllers and makes the drone feel like it is mechanically locked to your hands.

- **Low Value (0.0):** The drone relies entirely on error-correction. There will be a slight delay as the drone "spools up" its rotation authority.
- **High Value:** The drone feels extremely responsive. It starts and stops rotating the exact millisecond you move the sticks.
- **Too High:** The drone will "over-snap" when you start a movement, potentially causing the nose to dip or rise too far before leveling out at the correct speed.

The Mathematics

The FF term is a simple multiplier of the target rotation rate:

$$ \text{Out}*{FF} = \omega*{target} \cdot k_{FF} $$

Where:

- ω_target is the desired pitch rate (deg/s).
- k_FF is `ATC_RAT_PIT_FF`.

The total output is the sum of PID and FF contributions.

The Engineer's View

This parameter is the `_kff` member of the `_pid_rate_pitch` object.

It is applied in `AC_PID::update_all()`:

```
// AC_PID.cpp
_pid_info.FF = target * _kff;
// ...
return P + I + D + _pid_info.FF;
```

Feed-Forward is particularly important in autonomous flight (Auto missions) because it allows the drone to follow its planned path with much higher precision, as the controller doesn't need a large "error" to generate high motor power.

Tuning & Behavior

- **Default Value:** 0.0 (Standard Multirotor)
- **Range:** 0.0 - 0.5
- **Effect of Increasing:** Snappier, more immediate pitch response. Better for high-speed tracking and sharp turns.
- **Effect of Decreasing:** Smoother, softer pitch behavior.

Use Case Recommendations

- **Cinematography: Set to 0.05.** Just enough to remove the initial "laziness" of the drone without making the footage look robotic or twitchy.
- **FPV Racing: Increase to 0.2 - 0.3.** Essential for clearing gates and hitting gaps at high speed where any control lag is fatal.
- **Large Professional Drones: Keep at 0.0.** Heavy vehicles should be moved with deliberate, error-checked power to avoid over-stressing the airframe.

Troubleshooting

- **Scenario:** When you push the stick forward, the drone "jumps" forward too hard for a split second, then settles.
  - **Diagnosis:** `ATC_RAT_PIT_FF` is too high.
  - **Fix:** Reduce `ATC_RAT_PIT_FF` by 0.05.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





ATC_RAT_PIT_FLTD (Pitch Axis Rate D-Term Filter)

DEFAULT: 0.0	RANGE: 0 100	UNITS: Hz

Pitch Axis Rate D-Term Filter (ATC_RAT_PIT_FLTD)

Description

`ATC_RAT_PIT_FLTD` is a specialized safety filter for your drone's pitch damping. In a PID loop, the **Derivative (D)** term is extremely sensitive to noise because it looks at the *rate of change* between gyro samples. High-frequency vibrations from the motors look like massive pitch rotation rates to the D-term.

Without this filter, the D-gain (`ATC_RAT_PIT_D`) would amplify that motor noise, causing the motors to "chirp," get hot, and potentially burn out. This filter specifically cleans up the signal before it gets multiplied by the D-gain, stripping out high-frequency "jitter" while leaving the actual movements of the drone intact.

- **Higher Frequency (e.g., 40Hz):** Effective damping, prevents "nose-bobbing" at the end of maneuvers, but increases motor heat.
- **Lower Frequency (e.g., 10Hz):** Safe for older or noisy frames, but the pitch damping will feel "laggy" and may not stop the drone from bouncing after a movement.
- **Disabled (0):** Defaults to half of the main error filter or a safe internal value.

The Mathematics

The D-term calculates the derivative of the pitch error and then passes it through a low-pass filter:

$$D_filtered = D_prev + \alpha \cdot \left(\frac{E - E_prev}{dt} - D_prev \right)$$

Where the filter coefficient α is derived from `ATC_RAT_PIT_FLTD` (f_c):

$$\alpha = \frac{dt}{dt + \frac{1}{2\pi f_c}}$$

Engineer's Tip: In a stable tune, the D-filter frequency is almost always lower than the Error filter frequency (`ATC_RAT_PIT_FLTE`) because the derivative process is naturally "noisier."

The Engineer's View

This parameter maps to `_filt_D_hz` in the `_pid_rate_pitch` object.

It is applied in `AC_PID::update_all()`:

```
// AC_PID.cpp
if (is_positive(dt)) {
    float derivative = (error - _last_error) / dt;
    _derivative = _last_derivative + (dt / (_filt_D_hz + dt)) * (derivative - _last_derivative);
}
return ... + _derivative * _kd;
```

Modern ArduPilot versions run these calculations at the full loop rate, allowing for high-precision noise rejection that is specifically tuned for the resonant frequencies of the longitudinal frame.

Tuning & Behavior

- **Default Value:** 0.0 (Internal auto-selection)
- **Range:** 0.0 - 100.0 Hz
- **Effect of Increasing:** Crisper pitch stops, less "bounce" when you let go of the stick.
- **Effect of Decreasing:** Quieter motor operation, longer motor life, and less risk of "prop wash" jitters.

Use Case Recommendations

- **Clean Cinematic Drone: Set to 15.0 - 20.0 Hz.** Provides excellent damping for the horizon without introducing high-frequency noise.
- **FPV Racing (Rigid Frame): Set to 30.0 - 40.0 Hz.** Necessary for the "instant stop" handling required for racing.
- **Large Professional Rig (>18 inch props): Set to 5.0 - 10.0 Hz.** Large props generate significant low-frequency noise; keeping the D-filter low prevents heating.

Troubleshooting

- **Scenario:** Motors are getting warm and making a rhythmic "growling" sound specifically during vertical descent.
  - **Diagnosis:** D-term noise from the pitch axis.
  - **Fix:** Reduce `ATC_RAT_PIT_FLTD` by 5.0 Hz.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





ATC_RAT_PIT_FLTE (Pitch Axis Rate Error Filter)

DEFAULT: 0.0	RANGE: 0 100	UNITS: Hz

Pitch Axis Rate Error Filter (ATC_RAT_PIT_FLTE)

Description

`ATC_RAT_PIT_FLTE` is the primary "noise cleaner" for your pitch controller.

The drone's gyroscope is extremely sensitive and picks up every mechanical vibration from the motors and frame. If the flight controller tried to react to every tiny vibration, the motors would work overtime, wasting battery and potentially burning out. This filter "washes out" that high-frequency noise, ensuring the PID loop only reacts to the drone's actual movements.

- **Higher Frequency (e.g., 40Hz):** Faster control response. The drone feels more "locked-in," but motor noise and heat will increase.
- **Lower Frequency (e.g., 10Hz):** Smoother flight and quieter motors, but the drone will feel slightly "loose" or "mushy" in its pitch.
- **Disabled (0):** The software uses the main gyro sample rate. This is risky for all but the cleanest, most rigid racing frames.

The Mathematics

This is a single-pole low-pass filter applied to the pitch error signal (E):

$$E_filtered = E_prev + \alpha \cdot (E_raw - E_prev)$$

Where α is derived from `ATC_RAT_PIT_FLTE` (f_c):

$$\alpha = \frac{dt}{dt + \frac{1}{2\pi f_c}}$$

Trade-off: A lower f_c introduces "Phase Lag." If the lag becomes too great, the drone will develop a low-frequency vertical "nodding" or wobble because its pitch corrections are arriving too late.

The Engineer's View

This parameter maps to `_filt_E_hz` in the `_pid_rate_pitch` object.

It is applied in `AC_PID::update_all()`:

```
// AC_PID.cpp
if (_reset_filter) {
    _reset_filter = false;
    _error = target - measurement;
} else {
    _error += ((target - measurement) - _error) * get_filt_E_alpha(dt);
}
```

By filtering the error *before* the P, I, and D calculations, ArduPilot ensures that the entire rate loop is based on a smooth, consistent measurement of the vehicle's state.

Tuning & Behavior

- **Default Value:** 0.0 (Internal auto-selection)
- **Range:** 0.0 - 100.0 Hz
- **Effect of Increasing:** Snappier pitch response. Tighter tracking during rapid forward flight.
- **Effect of Decreasing:** Quieter motors, less "jello" in the camera gimbal. Essential for long-range frames with flexible motor arms.

Use Case Recommendations

- **Standard GPS Photography Drone: Set to 20.0 Hz.** Provides a very smooth and quiet vertical hover.
- **FPV Freestyle Quad: Set to 30.0 - 40.0 Hz.** High performance requires a high filter cutoff for instant response.
- **Heavy Lift Multirotors: Set to 10.0 Hz.** Large props generate significant low-frequency noise; a lower filter keeps the system stable.

Troubleshooting

- **Scenario:** Drone's nose "vibrates" or makes a growling sound during fast forward flight.
  - **Diagnosis:** Frame resonance passing through the PID loop.
  - **Fix:** Reduce `ATC_RAT_PIT_FLTE` by 5.0 Hz.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





ATC_RAT_PIT_FLTT (Pitch Axis Rate Target Filter)

DEFAULT: 0.0	RANGE: 0 100	UNITS: Hz

Pitch Axis Rate Target Filter (ATC_RAT_PIT_FLTT)

Description

`ATC_RAT_PIT_FLTT` smooths the commands sent to the drone's pitch rate controller. While other filters (like `FLTE`) clean up noise from the **Sensors**, this filter cleans up "jumps" from the **Software**.

When a pilot moves the pitch stick or an autopilot changes the flight path, the target rotation rate might change in a sudden "step." These sharp edges can cause the drone to "kick" its nose up or down violently. This filter rounds off those corners, ensuring the drone accelerates into its pitch rotation gracefully rather than with a mechanical shock.

- **Low Frequency (e.g., 10Hz):** Very smooth, "organic" flight feel. Pitch commands are softened.
- **High Frequency (e.g., 40Hz):** Instant response. The drone follows every tiny pitch command perfectly.
- **Disabled (0):** The target rate is unfiltered. Maximum responsiveness, but potential for mechanical "ringing."

The Mathematics

This is a single-pole low-pass filter applied to the target rate (ω_target):

$$\omega_target_filtered = \omega_target_prev + \alpha \cdot (\omega_target_raw - \omega_target_prev)$$

Where α is derived from `ATC_RAT_PIT_FLTT` (f_c):

$$\alpha = \frac{dt}{dt + \frac{1}{2\pi f_c}}$$

Effect: This acts as a "softener" for the pitch stick. A 20Hz filter frequency corresponds to a time constant of about 8 milliseconds—enough to save your drone from resonant frame vibrations during hard stops.

The Engineer's View

This parameter maps to `_filt_T_hz` in the `_pid_rate_pitch` object.

It is applied at the very beginning of the PID loop in `AC_PID::update_all()`:

```
// AC_PID.cpp
_target += (target - _target) * get_filt_T_alpha(dt);
```

By filtering the target *at the source*, ArduPilot ensures that the Feed-Forward (`RAT_PIT_FF`) and P-terms don't produce a massive, discontinuous spike when the command changes.

Tuning & Behavior

- **Default Value:** 0.0 (Disabled / Loop rate)
- **Range:** 0.0 - 100.0 Hz
- **Effect of Increasing:** Snappier, more "direct" stick feel in the pitch axis.
- **Effect of Decreasing:** Smoother cinematic forward/backward transitions. Essential for vehicles with long arms that can flex under sudden torque.

Use Case Recommendations

- **Standard Build: Keep at 0.0.** ArduPilot's internal path planning handles most command smoothing automatically.
- **High-End Cinematography: Set to 15.0 Hz.** Adds a professional "weighted" feel to the pitch axis, making forward moves look more like a heavy camera crane.
- **Racing / FPV: Keep at 0.0.** Minimum latency is priority for high-speed maneuvering.

Troubleshooting

- **Scenario:** Drone makes a "clunk" sound specifically when you suddenly release the pitch stick after a forward dash.
  - **Diagnosis:** Discontinuous target rate is causing a motor current spike (mechanical ringing).
  - **Fix:** Set `ATC_RAT_PIT_FLTT` to 20.0 Hz.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





ATC_RAT_PIT_I (Pitch Axis Rate I Gain)

DEFAULT: 0.135	RANGE: 0.01 2.0

Pitch Axis Rate I Gain (ATC_RAT_PIT_I)

Description

`ATC_RAT_PIT_I` is the "automatic trim" for your drone's pitch axis. Its job is to remember and correct for persistent imbalances in the drone's weight distribution or aerodynamic drag.

If your drone is **nose-heavy** (e.g., you mounted a large camera on the front), the drone will naturally want to tilt forward. Without an I-term, you would have to constantly pull back on the stick to keep it level. The I-term "notices" that the drone keeps falling forward and slowly increases the power to the front motors until the drone stays level on its own.

- **Low Value:** The drone will drift forward or backward in wind. It won't hold its pitch angle accurately during long movements.
- **High Value:** The drone holds its pitch angle very strictly.
- **Too High:** The drone will develop a slow "pitch wobble" as the integrator over-corrects and "hunts" for the perfect trim point.

The Mathematics

The I-term accumulates the pitch rate error over time:

$$ \text{Out} I += \int_0^t (\omega_{target} - \omega_{actual}) \cdot k_I \cdot dt $$

Where:

- k_I is `ATC_RAT_PIT_I`.
- The result is constrained by `ATC_RAT_PIT_IMAX`.

Longitudinal Stability: Because most drones have their battery mounted on top/bottom and payloads on the front/back, the Pitch I-term usually has to work much harder than the Roll I-term.

The Engineer's View

This parameter is the `_ki` member of the `_pid_rate_pitch` object.

It is executed in `AC_PID::update_all()`:

```
// AC_PID.cpp
_integrator += (error * _ki * dt);
_integrator = constrain_float(_integrator, -_kimax, _kimax);
```

ArduPilot's **Leakiness** logic ensures that the integrator doesn't stay wound up forever. When the drone is on the ground, the integrator is slowly decayed to zero to prevent a "jumpy" take-off.

Tuning & Behavior

- **Default Value:** 0.135 (Matches P-gain)
- **Range:** 0.01 - 2.0
- **Effect of Increasing:** Better tracking of forward velocity, less pitch drift in wind.
- **Effect of Decreasing:** Drone feels more "fluid," but may lose its pitch target over long distances.

Use Case Recommendations

- **Nose-Mounted Gimbal (Cinematic): Increase to 0.2.** Heavier cameras on the nose require more I-term authority to maintain a stable, level horizon.
- **Speed Runs: Increase to 0.25.** High aerodynamic drag at 60mph+ creates a massive torque trying to flip the drone; a strong I-term is required to fight this.
- **Small Whoops / Micros: Keep Default.** Lightweight frames are usually well-balanced.

Troubleshooting

- **Scenario:** When you stop flying forward, the drone levels out, but then its nose slowly sinks by 2-3 degrees over the next few seconds.
  - **Diagnosis:** `ATC_RAT_PIT_I` is too low. It isn't building up enough authority to fight the weight of the drone.
  - **Fix:** Increase `ATC_RAT_PIT_I` by 0.02.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





ATC_RAT_PIT_ILMI (Pitch Axis Rate I-Leak Minimum)

DEFAULT: 0.02	RANGE: 0 1.0

Pitch Axis Rate I-Leak Minimum (ATC_RAT_PIT_ILMI)

Description

`ATC_RAT_PIT_ILMI` is a "memory management" feature for the pitch axis of **Traditional Helicopters**.

Helicopter pitch control is complex because the rotor disc acts like a giant gyroscope. When you command the heli to tilt forward, the I-term builds up authority to achieve that tilt. On the ground or during slow takeoff, the I-term can build up "false" corrections.

"Leakiness" allows this software memory to slowly fade away over time. `ATC_RAT_PIT_ILMI` determines the speed of this fade. It ensures the helicopter always has a "fresh" sense of level and doesn't get stuck leaning forward or backward because of a correction that happened several seconds ago.

- **Higher Value:** The pitch memory clears faster. The heli feels more reactive and "natural."
- **Lower Value:** The pitch memory persists longer. Better for maintaining exact pitch angles during long, fast forward flight.

Note: This is an `AC_HELI_PID` specific parameter and is not used on standard Quads or Octocopters.

The Mathematics

The pitch integrator (Int) is decayed every time step (dt):

$$ \text{Int}_{new} = \text{Int}_{prev} \cdot (1.0 - \text{LeakRate} \cdot dt) $$

Where the Leak Rate is set by `ATC_RAT_PIT_ILMI`.

Physical Meaning: This acts as a "forgetting factor." It prevents the integrator from "stalling" at a fixed value, which is critical for the rotating physics of a mechanical rotor head.

The Engineer's View

This parameter is the `_leak_min` member in the `AC_HELI_PID` class.

Longitudinal (pitch) leakiness is especially important for helicopters because they are often physically nose-heavy or tail-heavy. The leak ensures that the software-learned "trim" doesn't become dangerous if the airframe's balance shifts (e.g., during high-speed forward flight).

Tuning & Behavior

- **Default Value:** 0.02 (2% decay per second)
- **Range:** 0.0 - 1.0
- **Effect of Increasing:** The helicopter becomes more forgiving during "punchy" pitch moves. Less risk of the nose "jumping" after you center the stick.
- **Effect of Decreasing:** Tighter pitch authority. Better for heavy helis that need constant correction to stay level.

Use Case Recommendations

- **3D Smack Flight: Increase to 0.05.** Prevents "longitudinal memory" from ruining rapid vertical/pitch transitions.
- **Heavy Cargo Heli: Decrease to 0.01.** Ensures the heli stays perfectly level even with an off-center payload.
- **Scale Heli: Keep at 0.02.**

Troubleshooting

- **Scenario:** After stopping forward flight, the helicopter's nose "bobs" up and then stays slightly elevated for several seconds.
  - **Diagnosis:** Stale I-term memory (Windup).
  - **Fix:** Increase `ATC_RAT_PIT_ILMI` to 0.04.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





ATC_RAT_PIT_IMAX (Pitch Axis Rate I-Term Maximum)

DEFAULT: 0.5	RANGE: 0 1.0

Pitch Axis Rate I-Term Maximum (ATC_RAT_PIT_IMAX)

Description

`ATC_RAT_PIT_IMAX` is a vital safety fuse specifically for the drone's longitudinal balance. It limits the maximum power the **Integral (I) term** is allowed to use to correct for pitch errors.

Because most drones carry their payloads (cameras) and batteries along the pitch axis, they are frequently slightly nose-heavy or tail-heavy. The I-term learns to compensate for this. However, if the drone is physically stuck (e.g., in a tree) or if a front motor has failed, the I-term will "wind up" to 100% power trying to level the drone. If the drone suddenly breaks free, this massive, stored-up correction will cause it to flip over instantly. `ATC_RAT_PIT_IMAX` prevents this by capping that "learned authority."

- **Low Value:** Limits the drone's ability to fly if it is nose-heavy or tail-heavy.
- **High Value:** Allows for great balance correction, but increase the risk of a flip-on-recovery.
- **Default (0.5):** Allows the integrator to use up to 50% of motor range to fix pitch errors.

The Mathematics

The pitch integrator ($extInt$) is accumulated but clamped by this value:

$$ ext{Int}*{new} = ext{constrain}(ext{Int}*{prev} + ext{error}
k_I
dt, - ext{IMAX}, ext{IMAX}) $$

Where:

- $extIMAX$ is `ATC_RAT_PIT_IMAX`.
- The output is a 0.0 to 1.0 value used in the motor mixer.

The Engineer's View

This parameter is the `_kimax` member of the `_pid_rate_pitch` object.

It is applied in `AC_PID::update_all()`:

```
// AC_PID.cpp
_integrator = constrain_float(_integrator + (error * _ki * dt), -_kimax, _kimax);
```

ArduPilot's **Saturation logic** also stops the integrator from increasing if the drone's throttle is already at zero or full, preventing it from fighting the pilot's primary vertical control.

Tuning & Behavior

- **Default Value:** 0.5 (Standard)
- **Range:** 0.0 - 1.0
- **Effect of Increasing:** Better ability to stabilize nose-heavy drones (like those with large DSLR gimbals on the front).
- **Effect of Decreasing:** Safer for racing and testing; drone is less likely to flip after a minor collision.

Use Case Recommendations

- **Delivery Drones (Variable Payloads): Increase to 0.7 - 0.8.** If the weight moves or is dropped, the drone needs high authority to find its new level point quickly.
- **Cinematography (Gimbal Protrusion): Keep at 0.5 or 0.6.** Ensures the drone stays perfectly level despite the long lever arm of a heavy camera.
- **Standard FPV / Racing: Decrease to 0.4.** Minimizes the "bobbing" after a crash or gate-strike.

Troubleshooting

- **Scenario:** Drone flies level at hover, but as soon as you fly forward, it "sags" its nose and you have to constantly pull back.
  - **Diagnosis:** `ATC_RAT_PIT_IMAX` is being hit. The drone is nose-heavy, but the integrator is capped.
  - **Fix:** Increase `ATC_RAT_PIT_IMAX` to 0.7 or move the battery further back.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





ATC_RAT_PIT_NEF (Pitch Axis Rate Error Notch Filter Index)

DEFAULT: 0	RANGE: 1 8

Pitch Axis Rate Error Notch Filter Index (ATC_RAT_PIT_NEF)

Description

`ATC_RAT_PIT_NEF` is a high-precision vibration filter for your drone's pitch axis.

Longitudinal vibrations (along the drone's spine) can be very different from lateral (roll) vibrations, especially if the drone has long, thin arms or a heavy battery mounted on a "pendulum" style mount. This parameter allows you to assign a specific **Harmonic Notch Filter** to the pitch axis to specifically "ignore" those mechanical ringing frequencies.

By surgically removing the noise generated by the propellers at their specific RPM, this filter allows the pitch PID loop to be much more aggressive, resulting in a drone that stops its pitch moves with robotic precision and has zero "growling" sounds during fast flight.

- **Set to 0:** No notch filter is applied to the pitch rate error.
- **Set to 1-8:** Uses the corresponding Harmonic Notch filter (configured via `INS_HNTCH_*`).

The Mathematics

The notch filter (H_notch) is applied to the pitch error signal (E_pit):

$$ \vec{E}*{pit_clean} = H*{notch}(\omega_{pit_target} - \omega_{pit_actual}) $$

The filter is dynamic, following the motor RPM:

$$f_center = \text{Base_Freq} \cdot \frac{\text{Current_RPM}}{\text{Ref_RPM}}$$

This RPM-scaling ensures the "blind spot" of the filter always matches the noise generated by the props, regardless of throttle position.

The Engineer's View

This parameter maps to `_notch_E_filter` in the `_pid_rate_pitch` object.

It is applied in `AC_PID::update_all()`:

```
// AC_PID.cpp
if (_notch_E_filter > 0) {
    // Passes the error through the INS library's harmonic notch pointer
    error = _notch_E_filter_ptr->apply(error);
}
```

Since the pitch axis often has a different moment of inertia than the roll axis, having an independent notch filter index allows for axis-specific tuning on non-symmetric (rectangular or "Dead Cat") frames.

Tuning & Behavior

- **Default Value:** 0 (Disabled)
- **Range:** 1 - 8 (Filter Index)
- **Effect of Enabling:** Cleaner pitch stability. Allows for higher `ATC_RAT_PIT_P` and `D` gains without causing "shaking" or noise.
- **Effect of Disabling:** Increased risk of longitudinal resonance. Pitch behavior may feel "sloppy" in high wind.

Use Case Recommendations

- **"Dead Cat" / Long Range Frames: Always Set to 1.** These frames have asymmetric arms that vibrate at complex frequencies; a notch filter is mandatory for high-quality flight.
- **Cinematography Drones: Set to 1.** Eliminates micro-vibrations in the pitch axis that can show up as "jello" or horizon-tilt in high-resolution video.
- **Racing Drones: Set to 1.** Provides the cleanest possible signal for the D-term damping to work effectively.

Troubleshooting

- **Scenario:** Drone nose "bobs" or oscillates rapidly when flying into a headwind.
  - **Diagnosis:** Wind-induced resonance is being amplified by the pitch controller.
  - **Fix:** Configure an RPM-based notch and set `ATC_RAT_PIT_NEF` to its index.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY

ATC_RAT_PIT_NTF (Pitch Axis Rate Target Notch Filter Index)

DEFAULT: 0	RANGE: 1 8

Pitch Axis Rate Target Notch Filter Index (ATC_RAT_PIT_NTF)

Description

`ATC_RAT_PIT_NTF` is a high-precision command filter for the drone's pitch axis.

In many industrial or cinematic drones, the pitch axis has a long lever arm (the "spine" of the drone). This structure can be prone to specific vibration frequencies. If your automated flight controller (the high-level software) commands pitch changes at a frequency that matches the frame's resonance, the drone will vibrate.

This parameter allows you to apply a **Harmonic Notch Filter** specifically to the command signal. It "scoops out" the problematic frequencies from your mission paths or GPS corrections before they are sent to the motors. This results in a drone that follows complex paths with much greater longitudinal stability and zero "growling" or ringing in the pitch axis.

- **Set to 0:** No notch filter is applied to the pitch target rate.
- **Set to 1-8:** Uses the corresponding Harmonic Notch filter (configured via `INS_HNTCH_*`).

The Mathematics

The notch filter (H_notch) processes the target pitch rate (ω_pit_target) before the PID loop sees it:

$$\omega_pit_clean = H_notch(\omega_pit_raw)$$

The filter moves its rejection frequency dynamically to match the noise:

$$f_center \propto \text{Motor_RPM}$$

The Engineer's View

This parameter maps to `_notch_T_filter` in the `_pid_rate_pitch` object.

It is applied in `AC_PID::update_all()`:

```
// AC_PID.cpp
if (_notch_T_filter > 0) {
    // Passes the target through the dynamic notch filter
    target = _notch_T_filter_ptr->apply(target);
}
```

This is particularly useful when using **Terrain Following** or **Object Avoidance**, as those systems can sometimes generate high-frequency "jitter" in the pitch setpoint as they react to noisy environmental data.

Tuning & Behavior

- **Default Value:** 0 (Disabled)
- **Range:** 1 - 8 (Filter Index)
- **Effect of Enabling:** Quieter flight during autonomous navigation. Reduces longitudinal "bobbing" caused by software setpoint jitter.
- **Effect of Disabling:** Maximum pitch command responsiveness.

Use Case Recommendations

- **Agricultural Survey (Terrain Following): Set to 1.** Helps smooth the drone's pitch corrections as it follows undulating hills at high speeds.
- **Lidar-Based Obstacle Avoidance: Set to 1.** Prevents "software chatter" from the Lidar sensor from reaching the motors.
- **Standard Multirotor: Keep at 0.** Standard low-pass filters are usually sufficient for command smoothing.

Troubleshooting

- **Scenario:** Drone nose "jitters" specifically when obstacle avoidance is enabled, but is smooth in manual flight.
  - **Diagnosis:** High-level setpoint noise.
  - **Fix:** Assign a notch filter to the pitch target via `ATC_RAT_PIT_NTF`.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





ATC_RAT_PIT_P (Pitch Axis Rate P Gain)

DEFAULT: 0.135	RANGE: 0.01 0.5

Pitch Axis Rate P Gain (ATC_RAT_PIT_P)

Description

`ATC_RAT_PIT_P` is the core stability parameter for the drone's forward and backward movement. It sits in the **Innermost Loop** of the pitch controller, managing the motor RPM differences between the front and rear of the drone to achieve a specific rotation rate.

- **Low Value:** The drone feels "heavy" and slow to pitch. It may struggle to stop after a forward dash and will be easily pushed around by wind.
- **High Value:** The drone feels very responsive and sharp.
- **Too High:** The drone will oscillate (jitter) rapidly in the pitch axis (nose-up/nose-down). This is often audible as a "growl" or rapid motor pulsing.

The Mathematics

This is the Proportional (k_P) term of the pitch rate controller:

$$ \text{Out}*P = k*_P |*cdot (|omega*{target} - \omega_{actual}) $$

Where:

- ω_target is the desired pitch rate (deg/s) from the outer Angle loop.
- ω_actual is the actual pitch rate measured by the Gyroscope.
- k_P is `ATC_RAT_PIT_P`.

The Engineer's View

This parameter is the `_kp` member of the `_pid_rate_pitch` object (class `AC_PID`) inside `AC_AttitudeControl_Multi`.

It is executed in `AC_AttitudeControl_Multi::rate_controller_run_dt()`:

```
// AC_AttitudeControl_Multi.cpp
_motors.set_pitch(get_rate_pitch_pid().update_all(ang_vel_body.y, gyro.y, dt, ...) + ...);
```

On most symmetric frames, `ATC_RAT_PIT_P` and `ATC_RAT_PIT_P` should be identical. However, on "Dead Cat" frames or those with long fore-aft arms, these values will often diverge during AutoTune to account for the difference in rotating mass.

Tuning & Behavior

- **Default Value:** 0.135 (Standard Quad)
- **Range:** 0.01 - 0.50
- **Effect of Increasing:** Better tracking of forward speed commands. Drone stays level more accurately when punching the throttle.
- **Effect of Decreasing:** Smoother, "lazier" pitch handling.

Use Case Recommendations

- **Long-Range / "Dead Cat" Frames: Rely on AutoTune.** These frames often require a significantly higher Pitch P-gain than Roll P-gain because the rear motors are further from the center than the front motors.
- **Racing Drone: Decrease to 0.08 - 0.1.** High-torque motors need lower gains to avoid over-correcting.
- **Heavy Cargo Drone: Increase to 0.18 - 0.22.** Needed to overcome the high longitudinal inertia of the airframe.

Troubleshooting

- **Scenario:** Drone nose "twitches" or bobs up and down rapidly when flying forward at high speed.
  - **Diagnosis:** `ATC_RAT_PIT_P` is too high (Aerodynamic resonance).
  - **Fix:** Reduce `ATC_RAT_PIT_P` by 0.01.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY

ATC_RAT_PIT_PDMX (Pitch Axis Rate PD Maximum)

DEFAULT: 0	RANGE: 0 1.0

Pitch Axis Rate PD Maximum (ATC_RAT_PIT_PDMX)

Description

`ATC_RAT_PIT_PDMX` is a safety clamp for your drone's longitudinal rotation authority.

In a pitch controller, the **P** and **D** terms are responsible for reacting to fast disturbances. If you hit an object or experience a massive headwind gust, the PD math might command an instantaneous motor power jump that exceeds the frame's mechanical limits.

This parameter sets a ceiling on the combined power of P and D. It ensures that the drone's *immediate* reaction to a pitch error is kept within a safe range, preventing extreme torque spikes that could damage ESCs or frame components.

- **Low Value:** Limits the drone's "recovery snap" during extreme maneuvers.
- **High Value:** Allows the drone to use its full potential motor torque for immediate pitch corrections.
- **Disabled (0):** The PD sum is unlimited (limited only by the physical capabilities of the battery and motors).

The Mathematics

The P and D terms are summed and then constrained by this value:

$$\text{PD_}output = \text{constrain}(P + D, -\text{PDMX}, \text{PDMX})$$

Where:

- $P = k_P \cdot \omega_pitch_error$
- $D = k_D \cdot \frac{d}{dt}\omega_pitch_error$
- PDMX is `ATC_RAT_PIT_PDMX`.

Note: The Integral (I) and Feed-Forward (FF) terms are added *after* this clamp. This is critical because it means the drone can still achieve full tilt authority via the I-term, but it won't react with a violent "kick" to high-frequency noise.

The Engineer's View

This parameter maps to `_kpdmax` in the `_pid_rate_pitch` object.

It is applied in `AC_PID::update_all()`:

```
// AC_PID.cpp
if (is_positive(_kpdmax)) {
    float pd_sum = P + D;
    pd_sum = constrain_float(pd_sum, -_kpdmax, _kpdmax);
}
```

By capping the PD sum, ArduPilot allows you to use very high D-gains (for tight damping) without the risk of those high gains creating destructive motor pulses if a sensor glitch occurs.

Tuning & Behavior

- **Default Value:** 0 (Unlimited)
- **Range:** 0 - 1.0 (0 to 100% motor authority)
- **Effect of Increasing:** Better longitudinal recovery. The drone fights harder to stay level in turbulent air.
- **Effect of Decreasing:** Smoother, safer pitch behavior. Limits the mechanical "mechanical shock" of the drone's reactions.

Use Case Recommendations

- **Large Professional Rig (>10kg): Set to 0.5.** Protects the expensive long-arm structure from extreme torsional shocks during abrupt stops.
- **Cinematography Drone: Set to 0.4.** Ensures that any software glitches or sensor spikes produce a limited "kick," keeping the video footage cleaner.
- **FPV Racing Quad: Keep at 0.** You want every available ounce of motor torque for recovery during high-speed gate navigation.

Troubleshooting

- **Scenario:** Drone is slow to level its nose after a fast dive, even with high P-gain.
  - **Diagnosis:** `ATC_RAT_PIT_PDMX` is set too low.
  - **Fix:** Increase `ATC_RAT_PIT_PDMX` or set it to 0.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





ATC_RAT_PIT_SMAX (Pitch Axis Rate Slew Rate Limit)

DEFAULT: 0.0	RANGE: 0 200

Pitch Axis Rate Slew Rate Limit (ATC_RAT_PIT_SMAX)

Description

`ATC_RAT_PIT_SMAX` is a protection mechanism for your drone's longitudinal stability. A **Slew Rate** is a limit on how fast a command signal can change over time.

In the pitch controller, if the PID loop calculates that it needs to jump from 20% motor power to 80% motor power instantly (due to a violent gust or a sensor spike), the ESCs and motors will experience a massive, potentially destructive surge in current. This parameter rounds off those spikes, ensuring that the motor command always increases or decreases at a rate your hardware can safely handle.

- **Low Value:** Pitch response feels extremely smooth and "damped." Motors will sound very quiet.
- **High Value:** Allows the pitch motors to reach their target power as fast as they physically can.
- **Too High:** Risk of "motor chirping" or ESC failure due to extreme current transients during fast pitch changes.
- **Disabled (0):** The software does not limit the rate of change of the pitch motor command.

The Mathematics

The slew limit acts as a "speed limit" for the change in the pitch motor command (O_pit):

$$|O_new - O_prev| \leq SMAX \cdot dt$$

If the calculated change in output exceeds $SMAX$, ArduPilot automatically reduces the P and D gains of the pitch rate loop for that specific cycle to keep the command within the limit:

$$\text{Gain Scale Factor} = \frac{SMAX}{\text{Calculated Change}}$$

The Engineer's View

This parameter maps to `_slew_rate_max` in the `_pid_rate_pitch` object.

It is executed in `AC_PID::update_all()`:

```
// AC_PID.cpp
if (is_positive(_slew_rate_max)) {
    _pid_info.slew_rate = (P + D - _last_out_pd) / dt;
    if (fabsf(_pid_info.slew_rate) > _slew_rate_max) {
        // Dynamically lower authority for this loop cycle to prevent mechanical shock
        float scale = _slew_rate_max / fabsf(_pid_info.slew_rate);
        // ... apply scaling ...
    }
}
```

This **Dynamic Gain Reduction** allows the drone to remain stable while protecting the hardware from high-frequency resonant "shocks" that could otherwise break motor mounts or frames.

Tuning & Behavior

- **Default Value:** 0.0 (Disabled)
- **Range:** 0 - 200
- **Effect of Increasing:** Sharper, more immediate pitch authority. Essential for high-speed tracking.
- **Effect of Decreasing:** Smoother motor response, longer motor/ESC life, and quieter overall flight.

Use Case Recommendations

- **Large Prop Professional Drones (>18 inch): Set to 50 - 80.** High rotating inertia makes instant speed changes impossible; the slew limit prevents wasted energy and heat.
- **Industrial VTOL: Set to 40.** Reliability and mechanical longevity are prioritized over razor-sharp freestyle handling.
- **FPV Freestyle: Keep at 0.0.** Pilots generally prefer the absolute minimum latency in their pitch response.

Troubleshooting

- **Scenario:** Drone makes a high-pitched "scratching" sound during vertical punches or sharp stops.
  - **Diagnosis:** High-frequency pitch transients are exceeding the frame's rigidity or the ESC's update capability.
  - **Fix:** Set `ATC_RAT_PIT_SMAX` to 100 and re-test.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





ATC_RAT_RLL_D (Roll Axis Rate D Gain)

DEFAULT: 0.0036	RANGE: 0.0 0.05

Roll Axis Rate D Gain (ATC_RAT_RLL_D)

Description

`ATC_RAT_RLL_D` is the vertical "shock absorber" for your drone's rotation. In physics, the **Derivative (D)** term looks at how fast the error is changing.

If you command a sharp 180-degree roll and then stop the stick, the drone has a massive amount of rotating momentum. Without D-gain, the P-gain would try to stop the drone, but it would shoot past the target angle and then "bounce back." The D-gain "notices" that the drone is rotating very fast toward its target and applies the brakes early to ensure a perfectly smooth, non-bouncing stop.

- **Low Value:** The drone will "bounce" or "wag" its wings when you stop a roll. It feels loose and imprecise during fast maneuvers.
- **High Value:** The drone feels extremely rigid and "on rails." Stops are instantaneous.
- **Too High:** The drone will emit a high-pitched "shriek" from the motors. This is caused by the D-term amplifying tiny sensor vibrations (noise) and trying to correct them hundreds of times per second. This can burn out your motors and ESCs in seconds.

The Mathematics

The D-term is proportional to the rate of change of the rate error:

$$ \text{Out}*D = k*{D} *cdot *frac{d(*omega{target} - \omega_{actual})}{dt} $$

Where:

- k_D is `ATC_RAT_RLL_D`.
- The derivative is heavily filtered by `ATC_RAT_RLL_FLTD` to prevent noise amplification.

Intuition: If the error is getting smaller very quickly (you are nearly at your target rate), the derivative is negative. This causes the D-term to subtract from the total motor power, "braking" the rotation so you don't overshoot.

The Engineer's View

This parameter is the `_kd` member of the `_pid_rate_roll` object.

It is executed in `AC_PID::update_all()`:

```
// AC_PID.cpp
if (is_positive(dt)) {
    float derivative = (error - _last_error) / dt;
    // ... filtering happens here ...
    _derivative = filtered_derivative;
}
return ... + _derivative * _kd;
```

Modern ArduPilot versions also feature **D-Term Tapering**, which reduces the D-gain at high throttle to prevent the increased motor noise at high RPM from causing "D-gain blowouts."

Tuning & Behavior

- **Default Value:** 0.0036 (Very sensitive!)
- **Range:** 0.0 - 0.05
- **Effect of Increasing:** Crisper stops, less bounce-back after flips, "robotic" handling.
- **Effect of Decreasing:** Softer handling, more "organic" movement, safer for motors on high-vibration frames.

Use Case Recommendations

- **FPV Freestyle: Increase to 0.005 - 0.008.** Essential for "snappy" flips and rolls where the drone must stop perfectly level.
- **Large Professional Drones (>15 inch props): Keep very low or set to 0.0.** Large props are too slow to react to derivative changes; D-gain here usually just creates heat and "prop wash" oscillations.
- **Cinematography: Keep at Default.** Provides just enough damping for a stable horizon without introducing high-frequency jitters.

Troubleshooting

- **Scenario:** You finish a roll, and the drone's wings "wobble" once or twice before settling.
  - **Diagnosis:** `ATC_RAT_RLL_D` is too low.
  - **Fix:** Increase `ATC_RAT_RLL_D` by 0.0005.
- **Scenario:** Motors sound like they are "grinding" or "chirping" even in a hover.
  - **Diagnosis:** `ATC_RAT_RLL_D` is too high (Vibration amplification).
  - **Fix:** Reduce `ATC_RAT_RLL_D` immediately.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





ATC_RAT_RLL_D_FF (Roll Axis Rate Derivative Feed-Forward)

DEFAULT: 0.0	RANGE: 0 0.02

Roll Axis Rate Derivative Feed-Forward (ATC_RAT_RLL_D_FF)

Description

`ATC_RAT_RLL_D_FF` is an advanced "predictive" term for roll control. While standard Feed-Forward (`ATC_RAT_RLL_FF`) looks at the target rotation speed, the **Derivative Feed-Forward (D_FF)** looks at how fast you are *moving the stick*.

Think of it as the "pre-acceleration" boost. If you slam the roll stick to the side, you are requesting a sudden jump in rotation speed. The D_FF term "senses" this rapid movement and instantly adds a massive burst of motor power to overcome the inertia of the propellers *before* the drone even starts to roll. It allows the drone to "lead" into a maneuver with zero lag.

- **Low Value (0.0):** Standard behavior. The drone accelerates as fast as its P-gain and base FF allow.
- **High Value:** Creates a very aggressive "snap" at the start of every roll.
- **Too High:** Can cause the motors to "bark" or click, and potentially lead to over-snapping where the drone overshoots the initial rotation.

The Mathematics

The D_FF term is proportional to the derivative of the target rotation rate (ω_target):

$$ \text{Out}*{D_FF} = \frac{d(\omega{target})}{dt} \cdot k_{D_FF} $$

Where:

- k_D_FF is `ATC_RAT_RLL_D_FF`.
- $\frac{d(\omega_target)}{dt}$ is the rate of change of your stick input (in deg/s^2).

This is added to the total motor demand alongside the standard P, I, D, and FF terms.

The Engineer's View

This parameter maps to `_kdff` in the `_pid_rate_roll` object.

It is executed in `AC_PID::update_all()`:

```
// AC_PID.cpp
if (is_positive(dt)) {
    float target_derivative = (target - _last_target) / dt;
    _pid_info.D_FF = target_derivative * _kdff;
}
```

Because it looks at the **Target** (software) rather than the **Gyro** (sensor), D_FF is immune to motor vibration noise. This makes it a much safer way to get "crisp" response than simply cranking up the standard D-gain.

Tuning & Behavior

- **Default Value:** 0.0 (Standard)
- **Range:** 0.0 - 0.02
- **Effect of Increasing:** Snappier, more instantaneous roll response. Reduces the "initial sag" at the start of a flip.
- **Effect of Decreasing:** Smoother, more natural-feeling rotations.

Use Case Recommendations

- **FPV Racing / Freestyle: Increase to 0.005 - 0.01.** Essential for the "robotic" and instant feel required to navigate tight technical courses.
- **Standard GPS Flying: Keep at 0.0.** The standard shaper and P-gain are sufficient for smooth flight.
- **High-Speed Tracking: Increase to 0.003.** Helps the drone follow rapid movements of a subject without falling behind the "lead."

Troubleshooting

- **Scenario:** Drone makes a loud "pop" or "snap" sound from the motors every time you move the stick.
  - **Diagnosis:** `ATC_RAT_RLL_D_FF` is too high.
  - **Fix:** Reduce to 0.0 or 0.002.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY

ATC_RAT_RLL_FF (Roll Axis Rate Feed-Forward Gain)

DEFAULT: 0.0	RANGE: 0.0 0.5

Roll Axis Rate Feed-Forward Gain (ATC_RAT_RLL_FF)

Description

`ATC_RAT_RLL_FF` is the "direct drive" of the flight controller. While the P, I, and D terms in the PID loop wait for the drone to make a mistake before they act, the Feed-Forward (FF) term acts the moment you move the stick.

If you command a 180 deg/s roll, the FF term instantly calculates a baseline motor power to start that rotation *before* the gyro even feels the movement. This removes the "mushy" feel found in many flight controllers and allows for a perfectly linear relationship between your stick position and the drone's rotation speed.

- **Low Value (0.0):** The drone relies entirely on error-correction. There will be a slight, perceived lag as the PID loop builds up authority to start the rotation.
- **High Value:** The drone feels "telepathic." It starts and stops rotating exactly when you move the sticks.
- **Too High:** The drone will "snap" too hard, potentially overshooting its target rate before the PID loop can catch it.

The Mathematics

The FF term is a simple multiplier of the target rotation rate:

$$ \text{Out}*{FF} = |omega*{target} \cdot k_{FF} $$

Where:

- ω_target is the desired roll rate (from the pilot or outer loop).
- k_FF is `ATC_RAT_RLL_FF`.

The total output sent to the motor mixer is the sum of PID and FF:
$$ \text{Out}*{total} = |text{Out}*{P} + \text{Out}*{I} + |text{Out}*{D} + \text{Out}_{FF} $$

The Engineer's View

This parameter is the `_kff` member of the `_pid_rate_roll` object.

It is applied in `AC_PID::update_all()`:

```
// AC_PID.cpp
_pid_info.FF = target * _kff;
// ...
return P + I + D + _pid_info.FF;
```

In the ArduPilot architecture, Feed-Forward is often the preferred way to get responsiveness on large frames because it does not create the high-frequency vibration risks associated with aggressive P and D gains.

Tuning & Behavior

- **Default Value:** 0.0 (Traditional Multirotor)
- **Range:** 0.0 - 0.5
- **Effect of Increasing:** Snappier, more immediate rotations. Better adherence to rapid 3D maneuvers.
- **Effect of Decreasing:** Smoother, softer handling.

Use Case Recommendations

- **FPV Racing / Freestyle: Increase to 0.15 - 0.25.** Essential for the "instant" feel needed to navigate tight gates or perform technical acro.
- **Heavy Lift / Commercial: Keep at 0.0.** Large drones are too heavy for instant rotation; the error-correction loops are safer for managing their massive inertia.
- **Cinematography: Set to 0.05.** A tiny amount of FF can help the drone start moving with less "slop" without making it feel jumpy.

Troubleshooting

- **Scenario:** When you start a roll, the drone "kicks" violently, but the rest of the rotation is smooth.
  - **Diagnosis:** `ATC_RAT_RLL_FF` is too high.
  - **Fix:** Reduce `ATC_RAT_RLL_FF` by 0.05.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





ATC_RAT_RLL_FLTD (Roll Axis Rate D-Term Filter)

DEFAULT: 0.0	RANGE: 0 100	UNITS: Hz

Roll Axis Rate D-Term Filter (ATC_RAT_RLL_FLTD)

Description

`ATC_RAT_RLL_FLTD` is arguably the most important safety filter on your drone. In a PID loop, the **Derivative (D)** term is exceptionally sensitive to noise because it looks at the *rate of change* between samples. Small high-frequency vibrations that the P-term ignores look like massive rotation rates to the D-term.

Without this filter, the D-gain would amplify motor noise so aggressively that your motors would "chirp," get hot, and potentially burn out within seconds of taking off. This filter specifically targets the signal used for D-gain, stripping out high-frequency "jitter" while leaving the drone's actual movements intact.

- **Higher Frequency (e.g., 40Hz):** Effective damping, stops flips instantly, but high risk of motor heat.
- **Lower Frequency (e.g., 10Hz):** Safe for noisy frames, but the D-gain will feel "laggy" and may not stop the drone from bouncing after a maneuver.
- **Disabled (0):** Defaults to half of the main error filter or a safe internal value.

The Mathematics

The D-term calculates the derivative of the error, then passes it through a single-pole low-pass filter:

$$D_filtered = D_prev + \alpha \cdot \left(\frac{E - E_prev}{dt} - D_prev \right)$$

Where the filter coefficient α is derived from `ATC_RAT_RLL_FLTD` (f_c):

$$\alpha = \frac{dt}{dt + \frac{1}{2\pi f_c}}$$

Engineer's Tip: The D-filter frequency should almost always be lower than (or equal to) the Error filter frequency (`ATC_RAT_RLL_FLTE`) because the derivative process itself creates high-frequency noise.

The Engineer's View

This parameter maps to `_filt_D_hz` in the `_pid_rate_roll` object.

It is applied in `AC_PID::update_all()`:

```
// AC_PID.cpp
if (is_positive(dt)) {
    float derivative = (error - _last_error) / dt;
    _derivative = _last_derivative + (dt / (_filt_D_hz + dt)) * (derivative - _last_derivative);
}
return ... + _derivative * _kd;
```

Modern ArduPilot versions run these filters at the loop rate (e.g., 400Hz), which allows for very precise noise rejection without introducing excessive control lag.

Tuning & Behavior

- **Default Value:** 0.0 (Internal auto-selection, usually 20-30Hz)
- **Range:** 0.0 - 100.0 Hz
- **Effect of Increasing:** Better "snap" at the end of maneuvers. Tighter control.
- **Effect of Decreasing:** Quieter motors, longer motor life, less susceptible to prop-wash oscillation.

Use Case Recommendations

- **FPV Racing (Rigid Frame): Set to 30.0 - 50.0 Hz.** High performance requires a high D-cutoff to stop rotations instantly.
- **10-inch Multirotor (Gimbal): Set to 15.0 - 20.0 Hz.** Provides a smooth, non-vibrating base for photography.
- **Large Heavy Lift (>22 inch props): Set to 5.0 - 10.0 Hz.** Low frequencies are required to avoid exciting the resonant "ringing" of long carbon fiber arms.

Troubleshooting

- **Scenario:** Motors are making a high-pitched "grinding" sound and coming down hot to the touch.
  - **Diagnosis:** D-term noise amplification.
  - **Fix:** Reduce `ATC_RAT_RLL_FLTD` by 5.0 Hz increments until the motors are cool.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY

ATC_RAT_RLL_FLTE (Roll Axis Rate Error Filter)

DEFAULT: 0.0	RANGE: 0 100	UNITS: Hz

Roll Axis Rate Error Filter (ATC_RAT_RLL_FLTE)

Description

`ATC_RAT_RLL_FLTE` is the primary "vibration shield" for your roll rate controller.

The Gyroscope on your drone is incredibly sensitive; it can feel every tiny vibration from the motors and every slight flex of the frame. If the flight controller tried to react to every single 100Hz vibration, the motors would be constantly surging, creating heat and noise without actually stabilizing the drone. This filter "washes out" that high-frequency noise, ensuring the PID loop only reacts to the "true" movement of the vehicle.

- **Higher Frequency (e.g., 40Hz):** Faster control response. The drone feels more "connected," but motor noise and heat will increase.
- **Lower Frequency (e.g., 10Hz):** Smoother flight and quieter motors, but the drone will feel slightly "loose" or "mushy."
- **Disabled (0):** The software uses the main gyro sample rate (often 400Hz or 1kHz). Highly risky for larger drones.

The Mathematics

This is a single-pole low-pass filter applied to the error signal ($E = \omega_target - \omega_gyro$):

$$E_filtered = E_prev + \alpha \cdot (E_raw - E_prev)$$

Where the smoothing factor α is derived from `ATC_RAT_RLL_FLTE` (f_c):

$$\alpha = \frac{dt}{dt + \frac{1}{2\pi f_c}}$$

Trade-off: Lowering f_c increases "Phase Lag." If the lag becomes too great, the drone will start to wobble at a low frequency because its corrections are arriving too late.

The Engineer's View

This parameter maps to `_filt_E_hz` in the `_pid_rate_roll` object.

It is applied in `AC_PID::update_all()`:

```
// AC_PID.cpp
if (_reset_filter) {
    _reset_filter = false;
    _error = target - measurement;
} else {
    _error += ((target - measurement) - _error) * get_filt_E_alpha(dt);
}
```

By filtering the error *before* the P, I, and D calculations, ArduPilot ensures that the entire rate loop is synchronized on a consistent, clean signal.

Tuning & Behavior

- **Default Value:** 0.0 (Internal auto-selection)
- **Range:** 0.0 - 100.0 Hz
- **Effect of Increasing:** Better stability in high winds. Drone feels more "locked" during flips.
- **Effect of Decreasing:** Quieter motor sound, less "jello" in the camera gimbal. Essential for large, flexible frames.

Use Case Recommendations

- **5-inch Racing Quad: Set to 30.0 - 40.0 Hz.** High performance and rigid frames can handle a very high filter cutoff for instant response.
- **10-15 inch Photography Drone: Set to 20.0 Hz.** Good compromise between stability and motor heat.
- **Large Professional Rig (>15 inch props): Set to 10.0 Hz.** Large props have high inertia; trying to control them with 30Hz data is pointless and inefficient.

Troubleshooting

- **Scenario:** Drone oscillates rapidly (like a cell phone vibrating) when taking off.
  - **Diagnosis:** Frame resonance passing through the PID loop.
  - **Fix:** Reduce `ATC_RAT_RLL_FLTE` by 5.0 Hz increments until the vibration stops.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





ATC_RAT_RLL_FLTT (Roll Axis Rate Target Filter)

DEFAULT: 0.0	RANGE: 0 100	UNITS: Hz

Roll Axis Rate Target Filter (ATC_RAT_RLL_FLTT)

Description

`ATC_RAT_RLL_FLTT` smooths the commands sent to the drone's roll rate controller. While other filters (like `FLTE`) clean up noise from the **Sensors**, this filter cleans up "jumps" from the **Software**.

When a pilot moves the stick or an autopilot changes the flight path, the target rotation rate might change in a sudden "step" (e.g., from 0 deg/s to 50 deg/s in one loop). These sharp edges can cause the motors to kick violently. This filter rounds off those corners, ensuring the drone accelerates into its rotation gracefully rather than with a mechanical shock.

- **Low Frequency (e.g., 10Hz):** Very smooth, "organic" flight feel. Commands are softened.
- **High Frequency (e.g., 40Hz):** Instant response. The drone follows every tiny twitch of the pilot's fingers.
- **Disabled (0):** The target rate is unfiltered. Maximum responsiveness, but potential for mechanical "ringing."

The Mathematics

This is a low-pass filter applied to the target rate (ω_target):

$$\omega_target_filtered = \omega_target_prev + \alpha \cdot (\omega_target_raw - \omega_target_prev)$$

Where α is derived from `ATC_RAT_RLL_FLTT` (f_c):

$$\alpha = \frac{dt}{dt + \frac{1}{2\pi f_c}}$$

Effect: This effectively acts as a "stick smoothing" parameter. A 20Hz filter frequency corresponds to a time constant of about 8 milliseconds—unnoticeable to most pilots but enough to save your frame from resonant vibration.

The Engineer's View

This parameter maps to `_filt_T_hz` in the `_pid_rate_roll` object.

It is applied at the entry point of the PID loop in `AC_PID::update_all()`:

```
// AC_PID.cpp
_target += (target - _target) * get_filt_T_alpha(dt);
```

By filtering the target *at the start* of the loop, ArduPilot ensures that the Feed-Forward (`RAT_RLL_FF`) and P-terms don't produce a massive, discontinuous spike when the command changes.

Tuning & Behavior

- **Default Value:** 0.0 (Disabled / Loop rate)
- **Range:** 0.0 - 100.0 Hz
- **Effect of Increasing:** Snappier, more "direct" stick feel. Better for racing.
- **Effect of Decreasing:** Smoother cinematic transitions. Essential for vehicles with very large, high-torque motors that can be damaged by sudden "kicks."

Use Case Recommendations

- **Standard build: Keep at 0.0.** The input shaping logic in ArduPilot usually handles command smoothing perfectly.
- **Heavy Lift / Long Endurance: Set to 10.0 Hz.** Reduces the "impact" of stick commands on the frame and battery, potentially increasing flight time and frame longevity.
- **FPV Freestyle: Keep at 0.0.** Pilots generally prefer the absolute minimum latency between their brain and the motors.

Troubleshooting

- **Scenario:** Drone makes a "clunk" or "knock" sound specifically when you suddenly move the roll stick to its limit.
  - **Diagnosis:** Discontinuous target rate is causing a motor current spike.
  - **Fix:** Set `ATC_RAT_RLL_FLTT` to 20.0 Hz.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY

ATC_RAT_RLL_I (Roll Axis Rate I Gain)

DEFAULT: 0.135	RANGE: 0.01 2.0

Roll Axis Rate I Gain (ATC_RAT_RLL_I)

Description

`ATC_RAT_RLL_I` is the "memory" of the roll controller. Its job is to fix persistent mistakes that the P-gain (`ATC_RAT_RLL_P`) isn't strong enough to handle.

If your drone is off-balance (e.g., the battery is mounted slightly to one side) or if there is a constant side-wind, the drone will naturally want to tilt. The I-term "notices" this persistent error and slowly builds up motor power on the weak side until the drone stays level. It is the parameter that ensures the drone actually *stays* where the sticks tell it to, rather than slowly drifting away.

- **Low Value:** The drone will feel "drifty." It won't hold its angle precisely, especially when moving fast or in wind.
- **High Value:** The drone holds its attitude very strictly.
- **Too High:** The drone will develop a slow, rhythmic "wobble" (oscillation) as the integrator over-corrects and then has to unwind itself.

The Mathematics

The I-term accumulates the rate error over time:

$$ \text{Out}*I += \int*0^t (\omega{target} - \omega_{actual}) \cdot k_I \cdot dt $$

Where:

- k_I is `ATC_RAT_RLL_I`.
- The output is capped by `ATC_RAT_RLL_IMAX` to prevent "Runaway" throttle.

Anti-Windup: ArduPilot includes advanced logic to stop the integrator from building up if the motors are already at 100% power, preventing the drone from flipping over if it hits an obstacle.

The Engineer's View

This parameter is the `_ki` member of the `_pid_rate_roll` object.

It is executed in `AC_PID::update_all()`:

```
// AC_PID.cpp
_integrator += (error * _ki * dt);
_integrator = constrain_float(_integrator, -_kimax, _kimax);
```

In `AC_AttitudeControl_Multi`, the integrator is also affected by the **Throttle Mix** logic (`ATC_THR_MIX_MAN`), which determines how much priority the I-term gets compared to the pilot's throttle request.

Tuning & Behavior

- **Default Value:** 0.135 (Matches P-gain usually)
- **Range:** 0.01 - 2.0
- **Effect of Increasing:** Better attitude hold, less drift in wind, drone feels "heavier" and more stable.
- **Effect of Decreasing:** Drone feels more agile and "loose," but requires more manual correction from the pilot.

Use Case Recommendations

- **Long-Distance Autonomy: Increase to 0.2 - 0.3.** Ensures the drone stays perfectly on its path for miles, even as weight changes (e.g., if carrying a liquid payload).
- **Cinematography: Keep Default.** Standard values prevent the slow "integrator wobble" that can ruin a long tracking shot.
- **FPV Freestyle: Decrease to 0.08 - 0.1.** Many freestyle pilots prefer a "looser" feel that allows the drone to follow its own momentum slightly more naturally during acrobatic flips.

Troubleshooting

- **Scenario:** Drone oscillates slowly (about 1-2 times per second) in the roll axis.
  - **Diagnosis:** `ATC_RAT_RLL_I` is too high (Integrator oscillation).
  - **Fix:** Reduce `ATC_RAT_RLL_I` by 0.02 until the wobble stops.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





ATC_RAT_RLL_ILMI (Roll Axis Rate I-Leak Minimum)

DEFAULT: 0.02	RANGE: 0 1.0

Roll Axis Rate I-Leak Minimum (ATC_RAT_RLL_ILMI)

Description

`ATC_RAT_RLL_ILMI` is a specialized "memory cleanup" feature for **Traditional Helicopters**.

In PID control, the **Integral (I) term** has a "memory"—it remembers past errors and builds up authority to fix them. On helicopters, this memory can be problematic. Because helis have complex mechanical linkagess and ground interactions, the I-term can build up "fake" authority while the heli is sitting on the ground or during a slow takeoff.

"Leakiness" allows this memory to slowly evaporate over time. `ATC_RAT_RLL_ILMI` sets the minimum speed of that evaporation. It ensures that the helicopter's attitude memory is always "fresh" and doesn't contain stale corrections from a maneuver you finished several seconds ago.

- **Higher Value:** The I-term decays faster. The heli feels more "natural" and less "locked-in" by software memory.
- **Lower Value:** The I-term stays around longer. Better for holding exact angles in steady-state hover.

Note: This parameter is unique to the `AC_HELI_PID` library and is not used by standard multirotors (Quads/Octos).

The Mathematics

The integrator (Int) is modified by a decay factor every time step (dt):

$$ \text{Int}_{new} = \text{Int}_{prev} \cdot (1.0 - \text{LeakRate} \cdot dt) $$

Where the Leak Rate is determined by `ATC_RAT_RLL_ILMI`.

This effectively acts as a "High-Pass Filter" on the integrator, ensuring that it only focuses on recent errors and ignores errors from the distant past.

The Engineer's View

This parameter is the `_leak_min` member in the `AC_HELI_PID` class.

Helicopters use a "Leaky PID" architecture because of the unique way their swashplates interact with the rotor head. Without a leak, the integrator would constantly fight the mechanical gyroscopic precession of the rotor, leading to "I-term stall" where the heli becomes unresponsive to small stick movements.

Tuning & Behavior

- **Default Value:** 0.02 (2% decay per second)
- **Range:** 0.0 - 1.0
- **Effect of Increasing:** The helicopter feels more "loose." It will be less prone to "I-term kickback" after a hard stop.
- **Effect of Decreasing:** Tighter attitude hold. The helicopter will hold its tilt angle more accurately in wind.

Use Case Recommendations

- **3D Aerobatics: Increase to 0.05 - 0.10.** Prevents the integrator from building up during rapid, repeating flips and rolls.
- **Scale / Civilian Helicopter: Keep at 0.02.** Provides the best balance between stability and mechanical safety.
- **Large Gas Helicopters: Set to 0.01.** Heavy rotor heads have massive inertia and need a very stable I-term to maintain level flight.

Troubleshooting

- **Scenario:** After performing a sharp roll and stopping, the helicopter slowly "drifts" back a few degrees in the opposite direction.
  - **Diagnosis:** I-term Windup. The "memory" of the maneuver is still pushing the heli.
  - **Fix:** Increase `ATC_RAT_RLL_ILMI` to 0.04.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY

ATC_RAT_RLL_IMAX (Roll Axis Rate I-Term Maximum)

DEFAULT: 0.5	RANGE: 0 1.0

Roll Axis Rate I-Term Maximum (ATC_RAT_RLL_IMAX)

Description

`ATC_RAT_RLL_IMAX` is a critical safety fuse for your drone's roll control. It limits the maximum power the **Integral (I) term** is allowed to accumulate.

The I-term is designed to "learn" offsets—like a heavy battery mounted to one side. However, if the drone is physically unable to roll (e.g., a prop is broken, or it's tangled in a branch), the I-term will keep increasing its demand until it reaches 100% motor power. This is called "windup." If the drone suddenly breaks free with a wound-up integrator, it will flip over instantly. `ATC_RAT_RLL_IMAX` prevents this by putting a hard ceiling on that "memory" authority.

- **Low Value:** Limits the drone's ability to fly if it is significantly off-balance.
- **High Value:** Allows for massive imbalance correction, but increases the risk of violent flips if the drone becomes stuck and then breaks free.
- **Default (0.5):** Allows the integrator to use up to 50% of the motor's remaining authority to correct for roll errors.

The Mathematics

The integrator ($extInt$) is accumulated every time step but is clamped by this value:

$$ ext{Int}_{new} = ext{constrain}(ext{Int}_{prev} + ext{error} \cdot k_I \cdot dt, -ext{IMAX}, ext{IMAX}) $$

Where:

- $extIMAX$ is `ATC_RAT_RLL_IMAX`.
- The output is a value between 0 and 1.0 representing motor mixer authority.

The Engineer's View

This parameter maps to `_kimax` in the `_pid_rate_roll` object.

It is applied in `AC_PID::update_all()`:

```
// AC_PID.cpp
_integrator = constrain_float(_integrator + (error * _ki * dt), -_kimax, _kimax);
```

ArduPilot also uses **Throttle-Limited Anti-Windup**: If the pilot is already at 100% throttle, the integrator is not allowed to increase further in the positive direction. This ensures that the drone always prioritizes manual throttle commands over the integrator's "learned" corrections when the motors are saturated.

Tuning & Behavior

- **Default Value:** 0.5 (50% authority)
- **Range:** 0.0 - 1.0
- **Effect of Increasing:** Greater ability to stabilize very off-balance drones (e.g., asymmetric payloads).
- **Effect of Decreasing:** Safer; reduces the chance of the drone "darting" or flipping after a crash or entanglement.

Use Case Recommendations

- **Standard GPS Photography Drone: Keep Default (0.5).** Balanced safety and stability.
- **Asymmetric Special Purpose Vehicles: Increase to 0.7 - 0.8.** If your drone is carrying a heavy sensor off-center, it needs more I-term authority to hold a stable hover.
- **Indoor / Education / Micro Drones: Decrease to 0.3.** Smaller drones have less inertia and flip faster; a lower IMAX makes them much safer for indoor mishaps.

Troubleshooting

- **Scenario:** Drone hovers perfectly but stays tilted at 10 degrees for several seconds after a crash or being caught in a branch.
  - **Diagnosis:** Integrator Windup.
  - **Fix:** Reduce `ATC_RAT_RLL_IMAX` to 0.4.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





ATC_RAT_RLL_NEF (Roll Axis Rate Error Notch Filter Index)

DEFAULT: 0	RANGE: 1 8

Roll Axis Rate Error Notch Filter Index (ATC_RAT_RLL_NEF)

Description

`ATC_RAT_RLL_NEF` is a specialized "anti-vibration" setting. While standard filters (like `RAT_RLL_FLTE`) remove all high-frequency noise, a **Notch Filter** is like a surgical knife—it removes noise at one very specific, problematic frequency.

On multirotors, the motors and propellers create intense vibration at specific frequencies (linked to their RPM). If these vibrations happen to match the "ringing" frequency of your drone's carbon fiber arms, the arms will start to oscillate. If the flight controller tries to fix these oscillations, it creates a feedback loop that makes the motors shriek and can even cause the drone to explode mid-air.

By setting this index, you tell the roll controller to use one of your pre-configured Harmonic Notch filters (linked to motor RPM or an FFT) to specifically "ignore" those frame resonances.

- **Set to 0:** No notch filter is applied to the roll rate error.
- **Set to 1-8:** Uses the corresponding Harmonic Notch filter (configured via `INS_HNTCH_*`).

The Mathematics

The notch filter (H_notch) is applied to the error signal (E) before it enters the PID logic:

$$ \vec{E}*{clean} = H*{notch}(\omega_{target} - \omega_{actual}) $$

The center frequency of the notch (f_c) is dynamic and moves with the motor speed:

$$f_c \propto \text{Motor_RPM}$$

This ensures that no matter how fast you are flying, the "noise" from the propellers is always perfectly filtered out.

The Engineer's View

This parameter maps to `_notch_E_filter` in the `_pid_rate_roll` object.

It is applied in `AC_PID::update_all()`:

```
// AC_PID.cpp
if (_notch_E_filter > 0) {
    // Passes the error through the INS library's harmonic notch pointer
    error = _notch_E_filter_ptr->apply(error);
}
```

By applying the notch to the **Error** signal (rather than the raw Gyro), ArduPilot ensures that the P, I, and D terms all receive the cleanest possible data, allowing for much higher gains and a "locked-in" feel.

Tuning & Behavior

- **Default Value:** 0 (Disabled)
- **Range:** 1 - 8 (Filter Index)
- **Effect of Enabling:** Allows for much higher `ATC_RAT_RLL_P` and `D` gains. Makes the drone sound significantly quieter and smoother.
- **Effect of Disabling:** Motors will be noisier and hotter. Increased risk of "D-gain blowouts."

Use Case Recommendations

- **High-Performance Multirotors: Always Set to 1 (after configuring `INS_HNTCH`).** Modern flight performance depends entirely on good notch filtering.
- **Large Frames (>15 inch): Set to 1.** Large arms are very prone to low-frequency resonance that standard filters can't catch without creating too much lag.
- **Small Whoops: Keep at 0.** Resonances are usually high enough that standard filters handle them.

Troubleshooting

- **Scenario:** I increased my D-gain and now the drone oscillates violently the moment I take off.
  - **Diagnosis:** Frame resonance is being amplified.
  - **Fix:** Ensure `INS_HNTCH` is configured and set `ATC_RAT_RLL_NEF` to 1.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





ATC_RAT_RLL_NTF (Roll Axis Rate Target Notch Filter Index)

DEFAULT: 0	RANGE: 1 8

Roll Axis Rate Target Notch Filter Index (ATC_RAT_RLL_NTF)

Description

`ATC_RAT_RLL_NTF` is the sister filter to the "Error Notch" filter. While the error notch cleans up noise from the **Sensors** (IMU), the target notch cleans up noise from the **Commands** (Software).

In a complex multi-layered control system, the "target rate" (how fast you want to roll) is the result of many calculations—GPS position planners, obstacle avoidance systems, and mission planners. Sometimes, these software processes can introduce rhythmic "ripples" or high-frequency pulses into the command. If these pulses happen to match the physical resonance of your drone's frame, they can cause the drone to vibrate or sound "grainy."

This parameter allows the flight controller to surgically remove those command-side frequencies before they ever reach the motors.

- **Set to 0:** No notch filter is applied to the roll target rate.
- **Set to 1-8:** Uses the corresponding Harmonic Notch filter (configured via `INS_HNTCH_*`).

The Mathematics

The notch filter (H_notch) is applied to the target rate signal (ω_target) at the very beginning of the PID calculation:

$$\omega_target_clean = H_notch(\omega_target_raw)$$

The clean target ω_target_clean is then used for the error calculation and the Feed-Forward path.

The Engineer's View

This parameter maps to `_notch_T_filter` in the `_pid_rate_roll` object.

It is applied in `AC_PID::update_all()`:

```
// AC_PID.cpp
if (_notch_T_filter > 0) {
    // Passes the target through the INS library's harmonic notch pointer
    target = _notch_T_filter_ptr->apply(target);
}
```

This is rarely needed on standard multirotors, but is an essential tool for **Heavy Lift** or **Experimental** frames where software path-planning jitter can excite mechanical arm resonances.

Tuning & Behavior

- **Default Value:** 0 (Disabled)
- **Range:** 1 - 8 (Filter Index)
- **Effect of Enabling:** Smoother motor sound during rapid maneuvers. Eliminates rhythmic vibrations caused by high-level software setpoints.
- **Effect of Disabling:** Maximum responsiveness to command changes.

Use Case Recommendations

- **Standard Multirotor: Keep at 0.** ArduPilot's standard input shapers provide plenty of smoothing for the command signal.
- **Flexible Wings / Long-Arm Octocopters: Consider setting to 1.** Large frames can act like tuning forks; a command-side notch ensures the software never accidentally "plucks" the arm at its resonant frequency.
- **Precision Surveying (RTK): Set to 1.** Very high-precision GPS position loops can sometimes introduce tiny high-frequency corrections that a notch filter can clean up.

Troubleshooting

- **Scenario:** Drone makes a faint "ringing" or "humming" sound specifically during Auto missions or Loiter, but is perfectly quiet in manual Stabilize.
  - **Diagnosis:** Command-side resonance.
  - **Fix:** Apply a notch filter at the resonant frequency via `ATC_RAT_RLL_NTF`.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





ATC_RAT_RLL_P (Roll Axis Rate P Gain)

DEFAULT: 0.135	RANGE: 0.01 0.5

Roll Axis Rate P Gain (ATC_RAT_RLL_P)

Description

`ATC_RAT_RLL_P` is the single most important parameter for your drone's stability. It sits in the **Innermost Loop** of the flight controller, responsible for managing the actual RPM of your motors to achieve a specific rotation speed.

If the drone is rotating at 10 deg/s but needs to be at 20 deg/s, `ATC_RAT_RLL_P` determines how much extra power is sent to the motors on one side to force that change.

- **Low Value:** The drone feels "heavy," sluggish, and will not be able to stabilize itself against wind. It may slowly tilt over and crash because it lacks the authority to fix itself.
- **High Value:** The drone feels very "tight" and precise.
- **Too High:** The drone will oscillate (vibrate) rapidly in the roll axis. This is often heard as a high-pitched "shrieking" or "warbling" sound from the motors.

The Mathematics

This is the Proportional (k_P) term of the rate controller. It calculates a motor demand ($extOut$) based on the **Rate Error**:

$$ \text{Out}*P = k*_P *cdot (*omega*{target} - \omega_{actual}) $$

Where:

- ω_target is the desired roll rate (deg/s) from the outer Angle loop.
- ω_actual is the actual roll rate measured by the Gyroscope.
- k_P is `ATC_RAT_RLL_P`.
- **Dimensionality:** This gain converts Rotation Rate (T^{-1}) into a dimensionless motor command (0.0 to 1.0).

The Engineer's View

This parameter is the `_kp` member of the `_pid_rate_roll` object (class `AC_PID`) inside `AC_AttitudeControl_Multi`.

It is executed hundreds of times per second (loop rate) in `AC_AttitudeControl_Multi::rate_controller_run_dt()`:

```
// AC_AttitudeControl_Multi.cpp
_motors.set_roll(get_rate_roll_pid().update_all(ang_vel_body.x, gyro.x, dt, ...) + ...);
```

The output of this loop is sent directly to the motor mixer, where it is added to the throttle to determine the final PWM signal for each ESC.

Tuning & Behavior

- **Default Value:** 0.135 (Varies by frame)
- **Range:** 0.01 - 0.50
- **Effect of Increasing:** Faster stabilization, tighter feel, better rejection of aerodynamic disturbances.
- **Effect of Decreasing:** Smoother flight, but increased risk of "washout" (losing control during aggressive moves) and poor wind handling.

Use Case Recommendations

- **Standard Quadcopter: Default (0.135) or run AutoTune.** Most modern 5-10 inch drones work well with values between 0.1 and 0.2.
- **Racing Drone (High Power-to-Weight): Decrease to 0.08 - 0.1.** High-power motors provide so much torque that a smaller P-gain is required to prevent oscillation.
- **Heavy Industrial / Large Props: Increase to 0.18 - 0.25.** Large props have high inertia and take longer to change speed; a higher P-gain is needed to force them to react.

Troubleshooting

- **Scenario:** High-frequency vibration ("shaking") in the roll axis, especially when at high throttle.
  - **Diagnosis:** `ATC_RAT_RLL_P` is too high (Rate loop oscillation).
  - **Fix:** Reduce `ATC_RAT_RLL_P` by 15% and re-test.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





ATC_RAT_RLL_RLL_PDMX (Roll Axis Rate PD Maximum)

DEFAULT: 0	RANGE: 0 1.0

Roll Axis Rate PD Maximum (ATC_RAT_RLL_PDMX)

Description

`ATC_RAT_RLL_PDMX` is a safety clamp for the "active" parts of the roll controller.

In a PID loop, the **P** and **D** terms are responsible for reacting to fast changes and noise. If the drone is hit by a massive gust or experiences an extreme impact, the P and D terms might compute a requested motor power of 300%—which is physically impossible and can cause the control software to behave erratically.

This parameter sets a ceiling on the combined power of P and D. It ensures that no matter how big the mistake is, the *immediate* rotation torque requested by the drone is kept within a reasonable range.

- **Low Value:** Limits the drone's "snap" or ability to fight hard against extreme turbulence.
- **High Value:** Allows the drone to use its full motor torque for immediate corrections.
- **Disabled (0):** The PD sum is unlimited (limited only by the physical constraints of the motors).

The Mathematics

The P and D terms are summed and then constrained by this value:

$$\text{PD_}output = \text{constrain}(P + D, -\text{PDMX}, \text{PDMX})$$

Where:

- P is the proportional contribution ($k_P \cdot \omega_error$).
- D is the derivative contribution ($k_D \cdot \frac{d}{dt}\omega_error$).
- PDMX is `ATC_RAT_RLL_PDMX`.

Note: This limit is applied *before* the Integral (I) term and Feed-Forward (FF) are added. This ensures that even if P and D are clamped, the drone can still maintain its long-term attitude through the I-term.

The Engineer's View

This parameter maps to `_kpdmax` in the `_pid_rate_roll` object.

It is applied in `AC_PID::update_all()`:

```
// AC_PID.cpp
if (is_positive(_kpdmax)) {
    float pd_sum = P + D;
    pd_sum = constrain_float(pd_sum, -_kpdmax, _kpdmax);
    // ... use pd_sum ...
}
```

This is particularly useful for very high-performance drones where the D-gain is very aggressive. It prevents a single noisy gyro sample from commanding a full-power motor reversal.

Tuning & Behavior

- **Default Value:** 0 (Unlimited)
- **Range:** 0 - 1.0 (0 to 100% motor authority)
- **Effect of Increasing:** Increases the drone's "recovery authority"—the ability to level itself after a massive disturbance.
- **Effect of Decreasing:** Makes the flight feel "safer" and less likely to have sudden motor torque transients.

Use Case Recommendations

- **Standard Multirotor: Keep at 0.**
- **Large Professional Rig (>10kg): Set to 0.5 (50%).** Prevents extreme software-driven torque from snapping a motor arm or stripping a propeller hub during a shock.
- **Research / Prototype flight: Set to 0.3.** A safe setting for testing experimental code where you want to limit how "hard" the controller can kick.

Troubleshooting

- **Scenario:** Drone feels sluggish to recover from a steep bank, even though `RAT_RLL_P` is high.
  - **Diagnosis:** `ATC_RAT_RLL_PDMX` is set too low, capping the controller's authority.
  - **Fix:** Increase `ATC_RAT_RLL_PDMX` or set it to 0.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





ATC_RAT_RLL_SMAX (Roll Axis Rate Slew Rate Limit)

DEFAULT: 0.0	RANGE: 0 200

Roll Axis Rate Slew Rate Limit (ATC_RAT_RLL_SMAX)

Description

`ATC_RAT_RLL_SMAX` is a protective "governor" for your drone's power delivery. A **Slew Rate** is a limit on how fast a signal can change over time.

In the roll controller, if the PID loop calculates that it needs to jump from 10% motor power to 100% motor power instantly (due to noise, a prop strike, or a violent gust), the motors and ESCs will experience a massive, destructive spike in current. This parameter rounds off those spikes, ensuring that the motor command always increases or decreases at a rate your hardware can safely handle.

- **Low Value:** Flight feels extremely soft and "buttery." Motors will sound very smooth.
- **High Value:** Allows the motors to reach their target power as fast as possible.
- **Too High:** Risk of "motor chirping" or ESC failure due to extreme current transients.
- **Disabled (0):** The software does not limit the rate of change of the motor command.

The Mathematics

The slew limit acts as a "speed limit" for the motor output signal (O):

$$|O_new - O_prev| \leq SMAX \cdot dt$$

If the calculated change in output exceeds $SMAX$, ArduPilot automatically reduces the P and D gains of the rate loop for that specific cycle to keep the command within the limit:

$$\text{Gain Reduction Factor} = \frac{SMAX}{\text{Calculated Rate}}$$

The Engineer's View

This parameter maps to `_slew_rate_max` in the `_pid_rate_roll` object.

It is executed in `AC_PID::update_all()`:

```
// AC_PID.cpp
if (is_positive(_slew_rate_max)) {
    _pid_info.slew_rate = (P + D - _last_out_pd) / dt;
    if (fabsf(_pid_info.slew_rate) > _slew_rate_max) {
        // Dynamically lower P and D for this loop cycle
        float scale = _slew_rate_max / fabsf(_pid_info.slew_rate);
        // ... apply scaling ...
    }
}
```

This **Dynamic Gain Reduction** is a sophisticated way to handle vibration. Instead of just clipping the output (which causes instability), it temporarily "softens" the drone until the vibration spike passes.

Tuning & Behavior

- **Default Value:** 0.0 (Disabled)
- **Range:** 0 - 200
- **Effect of Increasing:** Snappier, more immediate roll authority.
- **Effect of Decreasing:** Smoother motor response, longer motor/ESC life.

Use Case Recommendations

- **Standard build: Keep at 0.0.** Modern DShot ESCs and rigid frames generally handle raw commands well.
- **Large Heavy Lift (>15 inch props): Set to 50 - 100.** High rotating mass makes instant speed changes impossible; the slew limit prevents the ESCs from wasting energy and heat trying to fight physics.
- **Commercial VTOL: Set to 30.** Comfort and reliability are more important than razor-sharp FPV handling.

Troubleshooting

- **Scenario:** Motors make a high-pitched "chirp" or "screech" during fast maneuvers or in wind.
  - **Diagnosis:** High-frequency PID transients are exceeding the ESC's ability to sync.
  - **Fix:** Set `ATC_RAT_RLL_SMAX` to 100.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





ATC_RAT_YAW_D (Yaw Axis Rate D Gain)

DEFAULT: 0.0	RANGE: 0.0 0.02

Yaw Axis Rate D Gain (ATC_RAT_YAW_D)

Description

`ATC_RAT_YAW_D` is the "brakes" for your drone's rotation. In physics, the **Derivative (D)** term looks at how fast the rotation is changing and provides a counter-force to slow it down as it approaches the target.

In the yaw axis, this parameter is very rarely used. Because multirotors yaw using torque reaction (which is relatively weak), the air itself provides enough natural damping to stop the drone from spinning. For most multirotors, this parameter is kept at **0.0**. However, for specialized frames or vehicles with very high yaw authority (like Tricopters or single-motor helicopters), a small amount of D-gain can help stop a fast spin without any bounce-back.

- **Low Value (0.0):** Standard behavior. The drone relies on natural air resistance and the P-gain to stop yawing.
- **High Value:** The drone stops its yaw turns with mechanical precision. No overshoot.
- **Too High:** The drone will emit high-frequency "chatter" or buzzing from the tail, as the D-term over-reacts to tiny gyro vibrations. This can damage motor bearings.

The Mathematics

The D-term is proportional to the rate of change of the yaw rate error:

$$ \text{Out}*D = k*_D |*cdot |*frac{d(|*omega{yaw_target} - \omega_{yaw_actual})}{dt} $$

Where:

- k_D is `ATC_RAT_YAW_D`.
- The derivative is filtered by `ATC_RAT_YAW_FLTD`.

Intuition: If the tail is swinging very fast toward its heading, the D-term pushes *against* that speed to make the landing at the target heading as soft as possible.

The Engineer's View

This parameter is the `_kd` member of the `_pid_rate_yaw` object.

It is executed in `AC_PID::update_all()`:

```
// AC_PID.cpp
if (is_positive(dt)) {
    float derivative = (error - _last_error) / dt;
    _derivative = filtered_derivative;
}
return ... + _derivative * _kd;
```

In `AC_AttitudeControl_Multi`, the yaw D-term is often bypassed entirely if the gain is set to 0, which saves precious CPU cycles for the more critical Roll and Pitch loops.

Tuning & Behavior

- **Default Value:** 0.0 (Standard Quad)
- **Range:** 0.0 - 0.02
- **Effect of Increasing:** Crisper stops at the end of yaw turns. Better stability during high-speed forward flight.
- **Effect of Decreasing:** Maximum smoothness. Lower risk of motor vibration.

Use Case Recommendations

- **Standard Multirotor: Keep at 0.0.** It is almost never needed for basic stability.
- **Tricopters (Servo Yaw): Set to 0.005 - 0.01.** Servos are much faster and more powerful than torque reaction; they need damping to prevent the mechanical tail from "wagging" during stops.
- **Heavy Lift Octocopter: Increase to 0.002.** Massive frames have high yaw inertia and may "coast" past their heading; a tiny amount of D-gain helps bring them to a controlled stop.

Troubleshooting

- **Scenario:** You perform a fast 360-degree spin, and when you let go of the stick, the tail "overshoots" and then bounces back by 5 degrees.
  - **Diagnosis:** Inadequate yaw damping.
  - **Fix:** Increase `ATC_RAT_YAW_D` by 0.001 (start very small!).

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





ATC_RAT_YAW_D_FF (Yaw Axis Rate Derivative Feed-Forward)

DEFAULT: 0.0	RANGE: 0 0.02

Yaw Axis Rate Derivative Feed-Forward (ATC_RAT_YAW_D_FF)

Description

`ATC_RAT_YAW_D_FF` is an advanced "torque booster" for your drone's tail.

In multirotors, yaw is generated by motor torque reaction, which is a slow and weak physical force compared to the lift used for roll and pitch. If you want to perform a sudden, sharp heading change, the drone has to overcome the rotational inertia of its arms and motors.

Derivative Feed-Forward (D_FF) looks at how fast you are moving the yaw stick. If it sees a sudden command to spin, it instantly commands a massive differential torque to the motors *at the start* of the move. This provides the "snap" needed for high-speed heading changes and stops the tail from feeling like it's attached by a rubber band.

- **Low Value (0.0):** Standard behavior. The drone turns as fast as its motor torque and P-gain allow.
- **High Value:** The drone tail feels extremely "electric" and sharp. It starts and stops spinning instantly.
- **Too High:** Can cause the frame to "kick" or twist violently, potentially causing low-frequency vibrations in the other axes.

The Mathematics

The D_FF term is proportional to the rate of change of the yaw target (ω_yaw_target):

$$ \text{Out}*{D_FF} = \frac{d(\omega*{yaw_target})}{dt} \cdot k_{D_FF} $$

Where:

- k_D_FF is `ATC_RAT_YAW_D_FF`.
- $\frac{d(\omega_yaw_target)}{dt}$ is the acceleration of the yaw setpoint.

Benefit: Since this is calculated from the pilot's input, it doesn't amplify gyro noise. It is a "clean" way to get extreme tail performance without risking motor damage.

The Engineer's View

This parameter maps to `_kdff` in the `_pid_rate_yaw` object.

It is executed in `AC_PID::update_all()`:

```
// AC_PID.cpp
if (is_positive(dt)) {
    float target_derivative = (target - _last_target) / dt;
    _pid_info.D_FF = target_derivative * _kdff;
}
```

On large multirotors (like Octocopters), D_FF is often more effective than high P-gains for starting turns, because P-gains can only react once the drone has already "failed" to start spinning, whereas D_FF starts the spin immediately.

Tuning & Behavior

- **Default Value:** 0.0 (Standard)
- **Range:** 0.0 - 0.02
- **Effect of Increasing:** Snappier heading starts and stops. Tighter tracking during rapid 3D maneuvers.
- **Effect of Decreasing:** Smoother, more relaxed heading behavior.

Use Case Recommendations

- **FPV Freestyle (Yaw Spins): Increase to 0.005 - 0.01.** Essential for those "instant" yaw stops after a rapid rotation.
- **Tricopters (Servo Yaw): Set to 0.002.** Helps the tail servo "lead" the maneuver without needing extreme P-gains that cause servo chatter.
- **Professional Video: Keep at 0.0.** You want the tail to drift smoothly; any "snapping" from D_FF will be visible as a jerk in the camera gimbal.

Troubleshooting

- **Scenario:** Drone tail "vibrates" for a split second every time you move the yaw stick.
  - **Diagnosis:** `ATC_RAT_YAW_D_FF` is too high, creating a torque pulse that excites the arm resonance.
  - **Fix:** Reduce to 0.0 or 0.002.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





ATC_RAT_YAW_FF (Yaw Axis Rate Feed-Forward Gain)

DEFAULT: 0.0	RANGE: 0.0 0.5

Yaw Axis Rate Feed-Forward Gain (ATC_RAT_YAW_FF)

Description

`ATC_RAT_YAW_FF` is the "accelerator pedal" for the tail of your drone. While the P, I, and D terms wait for the drone to *fail* to reach a target rotation speed before they act, the Feed-Forward (FF) term acts pre-emptively.

When you push the yaw stick left or right, the FF term instantly calculates how much motor torque to add *at that exact moment* to start the turn. This removes the "elastic" or "mushy" feeling often found in the tail control of large drones, allowing for a perfectly linear relationship between your finger position and the drone's rotation rate.

- **Low Value (0.0):** The drone relies entirely on error-correction. There will be a slight delay as the tail "spools up" into the turn.
- **High Value:** The drone feels extremely sharp. It starts and stops spinning the exact millisecond you move the stick.
- **Too High:** The drone will "over-spin" when you first move the stick, potentially causing a jerky, high-frequency "snap" that stresses the frame.

The Mathematics

The FF term is a simple multiplier of the target rotation rate:

$$ \text{Out}*{FF} = \omega*{target} \cdot k_{FF} $$

Where:

- ω_target is the desired yaw rate (deg/s).
- k_FF is `ATC_RAT_YAW_FF`.

The total output is the sum of PID and FF contributions.

The Engineer's View

This parameter is the `_kff` member of the `_pid_rate_yaw` object.

It is applied in `AC_PID::update_all()`:

```
// AC_PID.cpp
_pid_info.FF = target * _kff;
// ...
return P + I + D + _pid_info.FF;
```

In the ArduPilot architecture, Yaw Feed-Forward is often crucial for **Octocopters** and other multi-rotor platforms where the torque reaction force is very small compared to the mass of the arms. It provides the "kick" needed to overcome the airframe's rotational inertia.

Tuning & Behavior

- **Default Value:** 0.0 (Standard)
- **Range:** 0.0 - 0.5
- **Effect of Increasing:** Snappier heading changes. Tighter tracking during autonomous flight.
- **Effect of Decreasing:** Smoother, more flowing yaw turns.

Use Case Recommendations

- **Cinematography (Pan Shots): Keep at 0.0.** You want the tail to drift smoothly into its turn to avoid "robotic" camera movement.
- **Tricopters (Servo Yaw): Set to 0.1 - 0.2.** Servos react instantly; adding FF allows the software to take full advantage of that mechanical speed.
- **FPV Freestyle / Racing: Increase to 0.2.** Essential for sharp "Yaw Spins" and clearing tight racing corners without any control lag.

Troubleshooting

- **Scenario:** When you start a yaw turn, the tail "snaps" or makes a loud motor sound for a fraction of a second before continuing smoothly.
  - **Diagnosis:** `ATC_RAT_YAW_FF` is too high.
  - **Fix:** Reduce `ATC_RAT_YAW_FF` by 0.05.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





ATC_RAT_YAW_FLTD (Yaw Axis Rate D-Term Filter)

DEFAULT: 0.0	RANGE: 5 50	UNITS: Hz

Yaw Axis Rate D-Term Filter (ATC_RAT_YAW_FLTD)

Description

`ATC_RAT_YAW_FLTD` is a specialized filter for the drone's yaw damping. In a PID loop, the **Derivative (D)** term reacts to the *rate of change* of rotation. In the yaw axis, this is a very high-noise signal because the Z-axis of the gyroscope is directly exposed to the mechanical vibration of the entire frame twisting.

- **Note:** Since most multirotors use a yaw D-gain (`ATC_RAT_YAW_D`) of **0.0**, this filter is typically inactive.
- **Purpose:** If you do use yaw damping (common on Tricopters), this filter prevents that damping from amplifying motor noise into a high-frequency "chatter."

The Mathematics

The D-term calculates the derivative of the yaw error and then passes it through this low-pass filter:

$$D_filtered = D_prev + \alpha \cdot \left(\frac{E - E_prev}{dt} - D_prev \right)$$

Where the filter coefficient α is derived from `ATC_RAT_YAW_FLTD` (f_c):

$$\alpha = \frac{dt}{dt + \frac{1}{2\pi f_c}}$$

The Engineer's View

This parameter maps to `_filt_D_hz` in the `_pid_rate_yaw` object.

It is applied in `AC_PID::update_all()`:

```
// AC_PID.cpp
if (is_positive(dt)) {
    float derivative = (error - _last_error) / dt;
    _derivative = _last_derivative + (dt / (_filt_D_hz + dt)) * (derivative - _last_derivative);
}
```

Because yaw torque is inherently slow to develop in multirotors, this filter can be set very low (5-10Hz) to ensure absolute silence in the motor output without significantly affecting the control feel.

Tuning & Behavior

- **Default Value:** 0.0 (Internal auto-selection, usually matching the error filter)
- **Range:** 5.0 - 50.0 Hz
- **Effect of Increasing:** Allows for more aggressive yaw damping (crisper turn stops).
- **Effect of Decreasing:** Quieter motor operation, but the yaw damper will be less effective at stopping "bounces" after a spin.

Use Case Recommendations

- **Standard Multirotor: Keep at 0.0.** No need to tune this if your yaw D-gain is 0.
- **Tricopters / Tilt-Rotors: Set to 10.0 - 15.0 Hz.** Mechanical yaw systems are much more sensitive to noise; a low D-filter frequency is essential for servo longevity.
- **Racing Drones: Set to 20.0 Hz.** If using yaw damping, racers prefer a higher filter frequency to keep the tail feeling "electric" and instant.

Troubleshooting

- **Scenario:** You increased `ATC_RAT_YAW_D` and now the drone's motors sound like they are "scratching" or "grinding" during yaw turns.
  - **Diagnosis:** D-term noise from the vertical axis.
  - **Fix:** Reduce `ATC_RAT_YAW_FLTD` to 10.0 Hz.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY

ATC_RAT_YAW_FLTE (Yaw Axis Rate Error Filter)

DEFAULT: 0.0	RANGE: 0 20	UNITS: Hz

Yaw Axis Rate Error Filter (ATC_RAT_YAW_FLTE)

Description

`ATC_RAT_YAW_FLTE` is the primary "tail-vibration shield" for your drone.

While Roll and Pitch have to deal with intense vibrations from the motor arms flexing, the Yaw axis (the vertical axis) typically feels a different kind of noise. Because multirotors yaw by changing torque, any rapid "pulsing" in the motor RPM can look like a yaw rotation to the gyroscope. This filter "washes out" that high-frequency motor noise, ensuring the tail controller only reacts to the drone's actual heading changes.

- **Higher Frequency (e.g., 10Hz):** Faster yaw response. The nose feels more "locked," but motor noise will increase.
- **Lower Frequency (e.g., 2Hz):** Smoother heading hold and quieter tail, but the drone may feel slightly "mushy" or take longer to stop a fast turn.
- **Default (0):** Internal auto-selection (usually very low, around 2-5Hz).

The Mathematics

This is a single-pole low-pass filter applied to the yaw error signal ($E = \omega_yaw_target - \omega_yaw_gyro$):

$$E_filtered = E_prev + \alpha \cdot (E_raw - E_prev)$$

Where α is derived from `ATC_RAT_YAW_FLTE` (f_c):

$$\alpha = \frac{dt}{dt + \frac{1}{2\pi f_c}}$$

Note: The yaw filter frequency is usually much lower than the roll/pitch filters because torque reaction is a slow physical process, and we want to avoid exciting the mechanical "twist" of the drone's frame.

The Engineer's View

This parameter maps to `_filt_E_hz` in the `_pid_rate_yaw` object.

It is applied in `AC_PID::update_all()`:

```
// AC_PID.cpp
if (_reset_filter) {
    _reset_filter = false;
    _error = target - measurement;
} else {
    _error += ((target - measurement) - _error) * get_filt_E_alpha(dt);
}
```

Filtering the error *before* it enters the PID loop ensures that the I-term (`ATC_RAT_YAW_I`) doesn't build up "learned corrections" based on momentary noise spikes.

Tuning & Behavior

- **Default Value:** 0.0 (Internal auto-selection, usually 2.0 - 5.0 Hz)
- **Range:** 0.0 - 20.0 Hz
- **Effect of Increasing:** Snappier tail hold. Faster corrections for motor-torque dips.
- **Effect of Decreasing:** Quieter motor operation, less "growling" sound from the propellers during hover.

Use Case Recommendations

- **Long Endurance Drones: Set to 2.0 Hz.** Minimizes wasted motor current on correcting high-frequency yaw jitter, increasing battery life.
- **Racing / FPV Chase Drones: Increase to 10.0 - 15.0 Hz.** Essential for the "instant tail" feel needed when tracking moving objects at high speeds.
- **Heavy Lift Multirotors: Set to 3.0 Hz.** Prevents the long, heavy arms from oscillating at their resonant twisting frequency.

Troubleshooting

- **Scenario:** Drone nose "shakes" rapidly from side to side in wind.
  - **Diagnosis:** Yaw loop resonance.
  - **Fix:** Reduce `ATC_RAT_YAW_FLTE` by 1.0 Hz increments.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





ATC_RAT_YAW_FLTT (Yaw Axis Rate Target Filter)

DEFAULT: 0.0	RANGE: 0 50	UNITS: Hz

Yaw Axis Rate Target Filter (ATC_RAT_YAW_FLTT)

Description

`ATC_RAT_YAW_FLTT` smooths the yaw commands sent into the drone's rate controller. While other filters (like `FLTE`) clean up noise from the **Sensors**, this filter cleans up "jumps" from the **Software**.

When a pilot moves the yaw stick or an autopilot changes the heading during a mission, the target rotation rate might change in a sudden "step." Since yaw torque is relatively weak in multirotors, these sudden steps can cause the drone to "lurch" or create a noticeable jerk in the tail. This filter rounds off those corners, ensuring the drone accelerates into its turn gracefully rather than with a mechanical shock.

- **Low Frequency (e.g., 5Hz):** Very smooth, "lazy" tail movement. Excellent for cinematography.
- **High Frequency (e.g., 20Hz):** Instant response to stick inputs.
- **Disabled (0):** The target rate is unfiltered. Maximum responsiveness, but potential for "robotic" and jerky tail movements.

The Mathematics

This is a single-pole low-pass filter applied to the target rate (ω_target):

$$\omega_target_filtered = \omega_target_prev + \alpha \cdot (\omega_target_raw - \omega_target_prev)$$

Where α is derived from `ATC_RAT_YAW_FLTT` (f_c):

$$\alpha = \frac{dt}{dt + \frac{1}{2\pi f_c}}$$

Effect: This filter effectively limits the "acceleration" of the yaw command. A 10Hz filter frequency corresponds to a time constant of about 16 milliseconds—enough to remove the perceived "jerk" from digital stick inputs.

The Engineer's View

This parameter maps to `_filt_T_hz` in the `_pid_rate_yaw` object.

It is applied at the very entry point of the PID loop in `AC_PID::update_all()`:

```
// AC_PID.cpp
_target += (target - _target) * get_filt_T_alpha(dt);
```

By filtering the target *at the start* of the loop, ArduPilot ensures that the Feed-Forward (`RAT_YAW_FF`) doesn't produce a massive, discontinuous torque spike when the pilot's stick position changes.

Tuning & Behavior

- **Default Value:** 0.0 (Disabled / Loop rate)
- **Range:** 0.0 - 50.0 Hz
- **Effect of Increasing:** Snappier, more "connected" feel to the yaw stick.
- **Effect of Decreasing:** Smoother cinematic panning. Reduces mechanical stress on large vehicles.

Use Case Recommendations

- **Standard Build: Keep at 0.0.** Standard ArduPilot input shaping handles this perfectly.
- **Professional Photography (Smooth Pans): Set to 5.0 - 10.0 Hz.** Ensures that every camera pan starts and ends with a professional-looking "ease-in" and "ease-out" effect.
- **Heavy Industrial / Cargo Drone: Set to 5.0 Hz.** Reduces the torsional stress on the long motor arms when initiating a turn.

Troubleshooting

- **Scenario:** Drone makes a faint "chirp" sound every time you suddenly move the yaw stick.
  - **Diagnosis:** Discontinuous target rate is causing a torque transient.
  - **Fix:** Set `ATC_RAT_YAW_FLTT` to 10.0 Hz.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY

ATC_RAT_YAW_I (Yaw Axis Rate I Gain)

DEFAULT: 0.018	RANGE: 0.01 1.0

Yaw Axis Rate I Gain (ATC_RAT_YAW_I)

Description

`ATC_RAT_YAW_I` is the "heading anchor" for your drone. Its job is to fix persistent rotation errors that the P-gain (`ATC_RAT_YAW_P`) isn't strong enough to stop.

If your drone has motors that aren't perfectly aligned, or if you have a propeller that is slightly damaged, the drone will naturally want to spin (yaw) on its own. The I-term "notices" this constant drift and slowly increases the counter-torque until the rotation stops. It is the parameter that ensures the nose stays exactly where you pointed it, even during long flights or in heavy wind.

- **Low Value:** The drone will have "Heading Creep." The nose will slowly drift left or right over several minutes. Turn completion will feel vague.
- **High Value:** Heading is held very strictly. The tail feels "locked."
- **Too High:** The drone will develop a slow tail oscillation or "hunting" behavior as the tail moves back and forth looking for the target heading.

The Mathematics

The I-term accumulates the yaw rate error over time:

$$ \text{Out}*I += \int_0^t (\omega*{target} - \omega_{actual}) \cdot k_I \cdot dt $$

Where:

- k_I is `ATC_RAT_YAW_I`.
- The result is constrained by `ATC_RAT_YAW_IMAX`.

Torque Balance: In multirotors, yaw I-term is essential because any difference in motor performance directly creates a yaw torque. This integrator is the only way the drone can automatically find its internal "zero torque" point.

The Engineer's View

This parameter is the `_ki` member of the `_pid_rate_yaw` object.

It is executed in `AC_PID::update_all()`:

```
// AC_PID.cpp
_integrator += (error * _ki * dt);
_integrator = constrain_float(_integrator, -_kimax, _kimax);
```

ArduPilot's **Yaw Re-centering** logic often resets this integrator when the pilot makes a deliberate large yaw command, allowing the drone to "start fresh" after a big turn without carrying over the old trim authority.

Tuning & Behavior

- **Default Value:** 0.018 (Standard)
- **Range:** 0.01 - 1.0
- **Effect of Increasing:** Better heading hold, especially at high speed or during vertical "punches."
- **Effect of Decreasing:** Smoother, more relaxed heading behavior.

Use Case Recommendations

- **Cinematography: Keep Default or set to 0.015.** You want the heading to be stable, but avoid any sudden "snapping" corrections that the gimbal might not be able to follow.
- **High Speed Racing: Increase to 0.05 - 0.1.** At high speeds, the aerodynamic forces on the drone are massive; a strong I-term is required to keep the nose pointed down the track.
- **Long-Range Survey: Keep Default.** Standard values provide a stable heading for mapping and navigation.

Troubleshooting

- **Scenario:** After you finish a turn, the drone's heading continues to "creep" for 5-10 degrees before finally stopping.
  - **Diagnosis:** `ATC_RAT_YAW_I` is too low.
  - **Fix:** Increase `ATC_RAT_YAW_I` by 0.005.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





ATC_RAT_YAW_ILMI (Yaw Axis Rate I-Leak Minimum)

DEFAULT: 0.02	RANGE: 0 1.0

Yaw Axis Rate I-Leak Minimum (ATC_RAT_YAW_ILMI)

Description

`ATC_RAT_YAW_ILMI` is a specialized "memory cleanup" feature for the tail (yaw axis) of **Traditional Helicopters**.

Most helicopters use a mechanical tail rotor or a servo-driven system to control heading. These systems often experience "Tail Stiction" or ground interaction. The I-term learns to fight these forces to keep the tail straight. However, if the tail "sticks" and then suddenly releases, a wound-up I-term can cause the heli to spin violently.

"Leakiness" allows the tail controller to "forget" its old corrections over time. `ATC_RAT_YAW_ILMI` determines the minimum speed of this forgetting process. It ensures the tail always feels "current" and doesn't hold onto stale motor authority.

- **Higher Value:** The tail memory clears faster. The heli feels more reactive.
- **Lower Value:** The tail memory persists longer. Better for maintaining a rock-solid heading during long, slow maneuvers.

Note: This is an `AC_HELI_PID` specific parameter and is not used on standard Multirotors.

The Mathematics

The yaw integrator (Int) is decayed every time step (dt):

$$ \text{Int}_{new} = \text{Int}_{prev} \cdot (1.0 - \text{LeakRate} \cdot dt) $$

Where the Leak Rate is determined by `ATC_RAT_YAW_ILMI`.

This provides a "High-Pass" effect, filtering out the DC-offset (permanent bias) of the integrator so that it doesn't become a hard mechanical limit.

The Engineer's View

This parameter is the `_leak_min` member in the `AC_HELI_PID` class.

Yaw leakiness is particularly important for helicopters because as the main rotor RPM changes, the required tail rotor torque changes instantly. A leaky integrator allows the software to quickly "forget" the old torque requirement and adapt to the new one, preventing the tail from "jumping" during throttle changes.

Tuning & Behavior

- **Default Value:** 0.02 (2% decay per second)
- **Range:** 0.0 - 1.0
- **Effect of Increasing:** Tail becomes more "relaxed." Better for learning scale-like turns.
- **Effect of Decreasing:** Tighter tail authority. The nose stays exactly where it is pointed, even if you are changing throttle rapidly.

Use Case Recommendations

- **Scale Flying (Realism): Increase to 0.05.** Allows the tail to drift slightly in a realistic way, requiring more pilot coordination.
- **Precision Industrial Mapping: Decrease to 0.01.** Heading accuracy is the priority; the I-term must be as persistent as possible.
- **General Hobby Flight: Keep at 0.02.**

Troubleshooting

- **Scenario:** After finishing a yaw turn, the tail "overshoots" and then slowly (over 2-3 seconds) moves back to the target heading.
  - **Diagnosis:** Yaw I-term windup.
  - **Fix:** Increase `ATC_RAT_YAW_ILMI` to 0.05.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY

ATC_RAT_YAW_IMAX (Yaw Axis Rate I-Term Maximum)

DEFAULT: 0.5	RANGE: 0 1.0

Yaw Axis Rate I-Term Maximum (ATC_RAT_YAW_IMAX)

Description

`ATC_RAT_YAW_IMAX` is a critical safety fuse for your drone's heading stability. It limits the maximum amount of torque the **Integral (I) term** is allowed to apply to fix a yaw error.

The I-term is designed to learn how to keep the nose straight. If one motor is tilted by 2 degrees, the drone will naturally want to spin. The I-term learns this and applies a constant counter-torque to stay straight. However, if the drone is physically stuck or if a motor is failing, the I-term could "wind up" to 100% authority. If the drone suddenly breaks free, it will spin uncontrollably (death spin). `ATC_RAT_YAW_IMAX` prevents this by putting a hard ceiling on how much authority the "learned memory" has.

- **Low Value:** Limits the drone's ability to fly if it has tilted motors or asymmetric aerodynamics.
- **High Value:** Allows the drone to handle major motor misalignments, but increases the risk of a "spin-out" after an entanglement or crash.
- **Default (0.5):** Allows the integrator to use up to 50% of the motor's counter-torque capacity to fix yaw.

The Mathematics

The yaw integrator (Int) is accumulated but clamped by this value:

$$ \text{Int}_{new} = \text{constrain}(\text{Int}_{prev} + \text{error} \cdot k_I \cdot dt, -\text{IMAX}, \text{IMAX}) $$

Where:

- IMAX is `ATC_RAT_YAW_IMAX`.
- The output is a 0.0 to 1.0 value sent to the motor torque mixer.

The Engineer's View

This parameter is the `_kimax` member of the `_pid_rate_yaw` object.

It is executed in `AC_PID::update_all()`:

```
// AC_PID.cpp
_integrator = constrain_float(_integrator + (error * _ki * dt), -_kimax, _kimax);
```

In `AC_AttitudeControl_Multi`, the yaw integrator has a unique safety feature: it is **Priority-Limited**. If the Roll or Pitch loops already require 100% of the motor power to stay level, the Yaw integrator will stop building up to avoid causing a crash.

Tuning & Behavior

- **Default Value:** 0.5 (50% authority)
- **Range:** 0.0 - 1.0
- **Effect of Increasing:** Better ability to hold heading on frames with severely tilted motors or asymmetric drag (like a drone carrying a large, non-aerodynamic sign).
- **Effect of Decreasing:** Safer; limits the force of a "death spin" if the drone's yaw system becomes saturated.

Use Case Recommendations

- **Standard Multirotor: Keep Default (0.5).** This is a proven, safe ceiling.
- **Delivery Drones (Large Props): Increase to 0.7.** Large propellers generate high yaw torque; if the weight is off-balance, the drone may need more than 50% authority to stay straight.
- **Indoor / Micro Drones: Decrease to 0.3.** Lower authority makes the drone safer and more predictable during indoor collisions.

Troubleshooting

- **Scenario:** Drone hovers well, but its tail constantly "creeps" to the left in wind, and increasing `ATC_RAT_YAW_I` doesn't help.
  - **Diagnosis:** `ATC_RAT_YAW_IMAX` is too low. The drone has reached its "correction limit."
  - **Fix:** Increase `ATC_RAT_YAW_IMAX` to 0.6 or 0.7.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY

ATC_RAT_YAW_NEF (Yaw Axis Rate Error Notch Filter Index)

DEFAULT: 0	RANGE: 1 8

Yaw Axis Rate Error Notch Filter Index (ATC_RAT_YAW_NEF)

Description

`ATC_RAT_YAW_NEF` is a vibration filter specifically for your drone's tail stability.

While Roll and Pitch filters deal with arm flexing, the Yaw axis filter deals with **Torsional Resonance**—the tendency of the whole drone frame to "twist" back and forth around its vertical center. On multirotors, yaw is generated by torque reaction, meaning any rapid change in motor speed causes a slight twisting force on the frame. If this twisting happens at the same frequency as your motors' RPM, the tail of the drone will start to wag or vibrate.

By applying a **Harmonic Notch Filter** to the yaw error, you tell the flight controller to ignore those twisting vibrations. This allows for a much more stable heading, especially in wind, and prevents the "yaw growl" sound often heard on large, high-torque octocopters.

- **Set to 0:** No notch filter is applied to the yaw rate error.
- **Set to 1-8:** Uses the corresponding Harmonic Notch filter (configured via `INS_HNTCH_*`).

The Mathematics

The notch filter (H_notch) is applied to the yaw rate error (E_yaw):

$$ \vec{E}*{yaw_clean} = H*{notch}(\omega_{yaw_target} - \omega_{yaw_actual}) $$

The filter moves its "rejection frequency" in real-time to track motor speed:

$$f_center \propto \text{Motor_RPM}$$

This ensures the filter always targets the high-intensity noise produced by the propellers.

The Engineer's View

This parameter maps to `_notch_E_filter` in the `_pid_rate_yaw` object.

It is applied in `AC_PID::update_all()`:

```
// AC_PID.cpp
if (_notch_E_filter > 0) {
    // Note: The actual filtering happens via a pointer to the
    // INS harmonic notch library.
    error = _notch_E_filter_ptr->apply(error);
}
```

Since yaw torque is a relatively weak force, the yaw loop is often the first to fail when high-frequency noise is present. Applying the notch filter here is a low-CPU way to significantly improve overall heading authority.

Tuning & Behavior

- **Default Value:** 0 (Disabled)
- **Range:** 1 - 8 (Filter Index)
- **Effect of Enabling:** Much more stable heading in wind. Reduces vertical altitude loss during aggressive yaw turns (since lift motors aren't busy fighting yaw noise).
- **Effect of Disabling:** Tail may wag or vibrate in wind. Heading hold feels "softer."

Use Case Recommendations

- **Large Multirotors (>15 inch props): Always Set to 1.** Large props generate massive torque pulses; a notch filter is essential to keep the long motor arms from twisting at their resonant frequency.
- **Tricopters: Set to 1.** Mechanical yaw servos are extremely prone to over-reacting to noise; a notch filter protects the servo from high-frequency chatter.
- **Standard 5-inch Racers: Keep at 0.** These frames are so rigid that torsional resonance is rarely an issue.

Troubleshooting

- **Scenario:** Drone tail "wags" rapidly (20-30Hz) during vertical punches.
  - **Diagnosis:** Torsional frame resonance is being amplified by the yaw controller.
  - **Fix:** Set `ATC_RAT_YAW_NEF` to 1 (after configuring the main notch).

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY

ATC_RAT_YAW_NTF (Yaw Axis Rate Target Notch Filter Index)

DEFAULT: 0	RANGE: 1 8

Yaw Axis Rate Target Notch Filter Index (ATC_RAT_YAW_NTF)

Description

`ATC_RAT_YAW_NTF` is a surgical filter for your drone's tail commands.

In a multirotor, yaw is the most mechanically flexible axis because the drone has to "twist" the frame to spin. On large drones with long motor arms, this torsional flexibility creates a specific resonant frequency—if the tail moves at just the right speed, the whole frame starts to vibrate.

Sometimes, the high-level flight software (like the GPS heading hold or an automated survey grid) can request tiny, rapid yaw changes that accidentally hit this resonance. This filter "scoops out" those problematic command frequencies before they reach the motors, ensuring the tail always moves with professional smoothness.

- **Set to 0:** No notch filter is applied to the yaw target.
- **Set to 1-8:** Uses the corresponding Harmonic Notch filter (configured via `INS_HNTCH_*`).

The Mathematics

The notch filter (H_notch) processes the target yaw rate (ω_yaw_target) before it enters the control loop:

$$\omega_yaw_clean = H_notch(\omega_yaw_target_raw)$$

The clean target ω_yaw_clean is then used for the PID calculation. Because the filter is dynamic ($f_c \propto \text{RPM}$), it perfectly ignores command noise that matches the physical propeller noise.

The Engineer's View

This parameter maps to `_notch_T_filter` in the `_pid_rate_yaw` object.

It is applied in `AC_PID::update_all()`:

```
// AC_PID.cpp
if (_notch_T_filter > 0) {
    // Passes the target through the dynamic notch filter
    target = _notch_T_filter_ptr->apply(target);
}
```

This is an essential feature for **Large Octocopters** or **Tethered Drones** where the physical heading torque is low and command-side jitter can easily cause a resonant "tail wag."

Tuning & Behavior

- **Default Value:** 0 (Disabled)
- **Range:** 1 - 8 (Filter Index)
- **Effect of Enabling:** Quieter motors during heading changes. Reduces the chance of "tail-induced" frame vibration during autonomous turns.
- **Effect of Disabling:** Maximum heading responsiveness.

Use Case Recommendations

- **High-Value Aerial Cinematography: Set to 1.** Ensures that every camera pan is perfectly clean and free from software-induced mikro-jitters.
- **Large Hybrid/Heavy Lift Drones: Set to 1.** Prevents the high-leverage motors from exciting the torsional modes of the long frame arms.
- **Standard Quadcopter: Keep at 0.** Standard low-pass filters are usually sufficient for command smoothing.

Troubleshooting

- **Scenario:** Drone tail "vibrates" or makes a high-pitched growl only when rotating in an Auto mission.
  - **Diagnosis:** Command-side torsional resonance.
  - **Fix:** Assign a notch filter to the yaw target via `ATC_RAT_YAW_NTF`.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





ATC_RAT_YAW_P (Yaw Axis Rate P Gain)

DEFAULT: 0.18	RANGE: 0.10 2.50

Yaw Axis Rate P Gain (ATC_RAT_YAW_P)

Description

`ATC_RAT_YAW_P` is the "tail hold" authority for your drone. It sits in the **Innermost Loop** of the yaw controller, managing the torque differences between clockwise and counter-clockwise spinning motors to achieve a specific rotation rate.

Unlike Roll and Pitch, which use powerful differential thrust, multirotors yaw by changing the balance of motor torque. This is a much weaker force. `ATC_RAT_YAW_P` determines how aggressively the flight controller fights to keep the nose pointed in the right direction.

- **Low Value:** The tail will feel "vague" or "lazy." When you stop a turn, the drone might continue to drift (coast) for a few degrees. It will also be easily blown around by the wind.
- **High Value:** The drone tracks heading commands precisely. Turns feel sharp and immediate.
- **Too High:** The drone will develop "Tail Wag"—the nose will rapidly oscillate left and right. This can cause the entire frame to vibrate and potentially lose altitude.

The Mathematics

This is the Proportional (k_P) term of the yaw rate controller:

$$ \text{Out}*P = k*_P |cdot (|omega*{target} - \omega_{actual}) $$

Where:

- ω_target is the desired yaw rate (deg/s).
- ω_actual is the actual rotation rate measured by the Z-axis of the Gyroscope.
- k_P is `ATC_RAT_YAW_P`.

Note: Yaw gains are typically much higher than Roll/Pitch gains (e.g., 0.5 to 1.0 vs 0.1) because the physical torque authority is so much lower.

The Engineer's View

This parameter is the `_kp` member of the `_pid_rate_yaw` object (class `AC_PID`) inside `AC_AttitudeControl_Multi`.

It is executed in `AC_AttitudeControl_Multi::rate_controller_run_dt()`:

```
// AC_AttitudeControl_Multi.cpp
_motors.set_yaw(get_rate_yaw_pid().update_all(ang_vel_body.z, gyro.z, dt, ...) + ...);
```

On some frames, high yaw demands can "starve" the other axes for power. ArduPilot's motor mixer includes logic to prioritize Roll and Pitch over Yaw to ensure the drone doesn't flip over just to fix its heading.

Tuning & Behavior

- **Default Value:** 0.18 (Standard)
- **Range:** 0.10 - 2.50
- **Effect of Increasing:** Sharper, more precise yaw response. Better "locking" of the heading in wind.
- **Effect of Decreasing:** Smoother, more cinematic turns. Less stress on the motor bearings.

Use Case Recommendations

- **Professional Video (Smooth Pans): Decrease to 0.12 - 0.15.** Creates very graceful, slow yaw movements that look great on camera.
- **Yaw-Heavy Freestyle (Yaw Spins): Increase to 0.5 - 1.0.** Needed to stop extreme rotations instantly and provide the "locked" feel during technical maneuvers.
- **Tricopters: Rely on specific tuning.** Tricopters use a servo for yaw, which provides much higher authority than torque reaction. Their `ATC_RAT_YAW_P` values are usually much lower.

Troubleshooting

- **Scenario:** The drone's tail constantly "wags" left and right about 2-3 times per second during a hover.
  - **Diagnosis:** `ATC_RAT_YAW_P` is too high.
  - **Fix:** Reduce `ATC_RAT_YAW_P` by 0.05.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





ATC_RAT_YAW_PDMX (Yaw Axis Rate PD Maximum)

DEFAULT: 0	RANGE: 0 1.0

Yaw Axis Rate PD Maximum (ATC_RAT_YAW_PDMX)

Description

`ATC_RAT_YAW_PDMX` is a safety governor for your drone's tail authority.

Because yaw is generated by torque reaction, it is the most limited axis on a multirotor. If the flight controller commands a 100% yaw torque change instantly, it creates a massive "torque shock" through the frame. This can cause the drone's arms to twist and can even destabilize the Roll and Pitch loops as the lift motors are suddenly prioritized for rotation.

`ATC_RAT_YAW_PDMX` sets a limit on the combined power of the P and D terms. It ensures that the *instantaneous* heading correction requested by the software never exceeds a certain percentage of motor power, protecting the airframe and preserving lift for the other axes.

- **Low Value:** Limits the tail's "snap" or ability to stop spins instantly.
- **High Value:** Allows the tail to use its full motor torque for immediate corrections.
- **Disabled (0):** The PD sum is unlimited (limited only by motor saturation).

The Mathematics

The P and D terms for yaw are summed and then constrained:

$$\text{PD_}yaw = \text{constrain}(P + D, -\text{PDMX}, \text{PDMX})$$

Where:

- $P = k_P \cdot \omega_yaw_error$
- $D = k_D \cdot \frac{d}{dt}\omega_yaw_error$
- PDMX is `ATC_RAT_YAW_PDMX`.

Note: The Yaw Integral (I) term and Feed-Forward (FF) are added *after* this clamp. This ensures the tail can still hold a steady heading (via the I-term) even if the "fast" reactions are clamped.

The Engineer's View

This parameter maps to `_kpdmax` in the `_pid_rate_yaw` object.

It is applied in `AC_PID::update_all()`:

```
// AC_PID.cpp
if (is_positive(_kpdmax)) {
    float pd_sum = P + D;
    pd_sum = constrain_float(pd_sum, -_kpdmax, _kpdmax);
}
```

Since the yaw axis is often the most prone to torsional resonance, capping the PD sum is a powerful way to "quiet down" a noisy tail without sacrificing long-term heading hold accuracy.

Tuning & Behavior

- **Default Value:** 0 (Unlimited)
- **Range:** 0 - 1.0 (0 to 100% motor authority)
- **Effect of Increasing:** Snappier heading corrections. Better "snap" at the end of a pirouette.
- **Effect of Decreasing:** Smoother tail behavior. Reduces frame twisting and motor "chirp" during yaw maneuvers.

Use Case Recommendations

- **Large Octocopter / X8: Set to 0.4.** High motor counts generate massive yaw torque; capping the PD sum prevents the drone from "twisting itself apart" during aggressive heading moves.
- **Precision Aerial Panning: Set to 0.2.** Ensures that any software-induced heading correction is gentle and unnoticeable to the camera.
- **Tricopters: Keep at 0.** Tricopters have high mechanical authority via a servo; capping the PD sum would only increase lag in their already-complex yaw physics.

Troubleshooting

- **Scenario:** Drone heading "wavers" or is slow to stop spinning, but `RAT_YAW_P` is already high.
  - **Diagnosis:** `ATC_RAT_YAW_PDMX` is set too low, bottlenecking the yaw authority.
  - **Fix:** Increase `ATC_RAT_YAW_PDMX` or set it to 0.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





ATC_RAT_YAW_SMAX (Yaw Axis Rate Slew Rate Limit)

DEFAULT: 0.0	RANGE: 0 200

Yaw Axis Rate Slew Rate Limit (ATC_RAT_YAW_SMAX)

Description

`ATC_RAT_YAW_SMAX` is a protective mechanism for your drone's torsional stability. A **Slew Rate** is a limit on how fast a command signal can change over time.

In the yaw controller, if the PID loop calculates that it needs to jump from 10% motor torque to 90% motor torque instantly (due to a violent gust or a sensor spike), the ESCs and motors will experience a massive, potentially destructive surge in current. This parameter rounds off those spikes, ensuring that the motor command always increases or decreases at a rate your hardware can safely handle.

- **Low Value:** Yaw response feels extremely smooth and "damped." Motors will sound very quiet.
- **High Value:** Allows the yaw authority to reach its target power as fast as possible.
- **Too High:** Risk of "motor chirping" or ESC failure due to extreme current transients during fast yaw turns.
- **Disabled (0):** The software does not limit the rate of change of the yaw motor command.

The Mathematics

The slew limit acts as a "speed limit" for the change in the yaw motor command (O_yaw):

$$|O_new - O_prev| \leq SMAX \cdot dt$$

If the calculated change in output exceeds $SMAX$, ArduPilot automatically reduces the P and D gains of the yaw rate loop for that specific cycle to keep the command within the limit:

$$\text{Gain Scale Factor} = \frac{SMAX}{\text{Calculated Change}}$$

The Engineer's View

This parameter maps to `_slew_rate_max` in the `_pid_rate_yaw` object.

It is executed in `AC_PID::update_all()`:

```
// AC_PID.cpp
if (is_positive(_slew_rate_max)) {
    _pid_info.slew_rate = (P + D - _last_out_pd) / dt;
    if (fabsf(_pid_info.slew_rate) > _slew_rate_max) {
        // Dynamically lower authority for this loop cycle to prevent mechanical shock
        float scale = _slew_rate_max / fabsf(_pid_info.slew_rate);
        // ... apply scaling ...
    }
}
```

This **Dynamic Gain Reduction** is especially useful for preventing "Yaw Jumps" at the end of high-speed maneuvers, where aerodynamic forces can cause large momentary rate errors.

Tuning & Behavior

- **Default Value:** 0.0 (Disabled)
- **Range:** 0 - 200
- **Effect of Increasing:** Snappier, more immediate yaw authority. Essential for high-speed tracking.
- **Effect of Decreasing:** Smoother motor response, longer motor/ESC life.

Use Case Recommendations

- **Large Prop Drones (>18 inch): Set to 50 - 80.** Large arms have low torsional rigidity; limiting the slew rate prevents the arms from "twisting" and vibrating.
- **Industrial VTOL: Set to 40.** Heading precision is less important than reliability and mechanical longevity.
- **FPV Freestyle: Keep at 0.0.** Pilots generally prefer the absolute minimum latency in their yaw response.

Troubleshooting

- **Scenario:** Drone's tail makes a "snap" or "pop" sound during sharp turns.
  - **Diagnosis:** High-frequency yaw transients are exceeding the frame's torsional rigidity.
  - **Fix:** Set `ATC_RAT_YAW_SMAX` to 100.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





ATC_SLEW_YAW (Yaw Target Slew Rate)

DEFAULT: 6000	RANGE: 500 18000	UNITS: cdeg/s

Yaw Target Slew Rate (ATC_SLEW_YAW)

Description

`ATC_SLEW_YAW` is the "comfort setting" for your drone's tail during autonomous flight. While other parameters control how fast the drone *can* spin, this parameter controls how fast the software *wants* to change the goal heading.

When a drone reaches a waypoint and needs to turn 90 degrees to head to the next one, the software calculates a new heading target. Without a slew limit, the target would "jump" instantly, causing the drone to whip its tail around violently. `ATC_SLEW_YAW` forces that change to be gradual, making the drone turn smoothly and predictably.

- **Low Value (e.g. 1000):** The drone turns very slowly and elegantly during missions.
- **High Value (e.g. 10000):** The drone snaps to its new heading as fast as possible.
- **Default (6000):** 60 degrees per second. A standard, safe value for general flight.

The Mathematics

This parameter acts as a rate limiter for the heading setpoint (ψ_target):

$$\left| \frac{\psi_target(t) - \psi_target(t - dt)}{dt} \right| \leq \text{SLEW_YAW}$$

Where:

- SLEW_YAW is in centidegrees per second (e.g., 6000 = 60 deg/s).

Note: This is an *input* slew limit. It is independent of the *output* limits like `ATC_RATE_Y_MAX`. Even if the drone is physically capable of spinning at 200 deg/s, this parameter will force it to follow a 60 deg/s "ramp" during autonomous moves.

The Engineer's View

This parameter is the `_slew_yaw` member in `AC_AttitudeControl`.

It is applied in `AC_AttitudeControl::get_slew_yaw_max_degs()`:

```
// AC_AttitudeControl.cpp
float AC_AttitudeControl::get_slew_yaw_max_degs() const
{
    if (!is_positive(_ang_vel_yaw_max)) {
        return _slew_yaw * 0.01;
    }
    return MIN(_ang_vel_yaw_max, _slew_yaw * 0.01);
}
```

By capping the "Slew" at the same value as the "Max Rate," ArduPilot ensures that the path planner never requests a turn that the physical motors can't track, resulting in zero-overshoot turns during Auto missions.

Tuning & Behavior

- **Default Value:** 6000 cdeg/s (60 deg/s)
- **Range:** 500 - 18,000
- **Effect of Increasing:** Faster waypoint-to-waypoint transitions. Better for high-speed missions where every second counts.
- **Effect of Decreasing:** Much smoother video during auto-missions. Reduces the "jerky tail" look of autonomous flight.

Use Case Recommendations

- **Cinematography (Auto-Paths): Decrease to 1500 - 3000.** Ensures the tail move is so slow that it looks like a professional motion-control rig.
- **Industrial Survey / Mapping: Keep Default (6000).** Standard turns are efficient and keep the camera stabilized.
- **Package Delivery: Increase to 10000.** You want the drone to spend as little time as possible turning, to maximize forward flight time and battery life.

Troubleshooting

- **Scenario:** During an Auto mission, the drone starts its forward flight to the next waypoint before it has finished turning its nose, resulting in a "banana-shaped" flight path.
  - **Diagnosis:** `ATC_SLEW_YAW` is too low for the drone's mission speed.
  - **Fix:** Increase `ATC_SLEW_YAW` to 8000.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





ATC_SPEED_D (Speed Control D Gain)

DEFAULT: 0.0	RANGE: 0.0 0.4

Speed Control D Gain (ATC_SPEED_D)

Description

`ATC_SPEED_D` stops the rover from overshooting its target speed.

Tuning & Behavior

- **Default Value:** 0.0.
- **Recommendation:** Usually left at **0** for rovers. Friction provides enough natural damping.

ATC_SPEED_I (Speed Control I Gain)

DEFAULT: 0.2	RANGE: 0.0 2.0

Speed Control I Gain (ATC_SPEED_I)

Description

`ATC_SPEED_I` is the "Cruise Control."

If `ATC_SPEED_P` isn't enough to hold the speed (e.g., going up a steep hill), the I-term slowly increases the throttle until the error is gone.

Tuning & Behavior

- **Default Value:** 0.20.
- **Recommendation:** Set equal to `ATC_SPEED_P`.

ATC_SPEED_P (Speed Control P Gain)

DEFAULT: 0.2	RANGE: 0.01 2.0

Speed Control P Gain (ATC_SPEED_P)

Description

`ATC_SPEED_P` is the main "Throttle" knob.

If the rover is driving 1 m/s slower than it should be, this parameter determines how much extra throttle to apply.

- **Low Value:** Sluggish acceleration. The rover struggles to reach top speed uphill.
- **High Value:** Aggressive acceleration. May oscillate (surge) speed.

The Mathematics

$$ \text{Throttle} = k_P \cdot (\text{Speed}*{target} - |*text{Speed}*{actual}) $$

Tuning & Behavior

- **Default Value:** 0.20.
- **Tuning:** Increase P until the rover surges/pulses speed, then back off by 20%.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





ATC_STOP_SPEED (Speed Control Stop Speed)

DEFAULT: 0.1	RANGE: 0 0.5	UNITS: m/s

Speed Control Stop Speed (ATC_STOP_SPEED)

Description

`ATC_STOP_SPEED` is the "Zero Threshold."

When the rover slows down, at some point the autopilot has to decide "Okay, we are stopped now" and cut the throttle completely.

Tuning & Behavior

- **Default Value:** 0.1 m/s.
- **Recommendation:** If the rover creeps or jitters at a standstill, increase this to **0.2**.

ATC_STR_ACC_MAX (Steering Control Angular Acceleration Maximum)

DEFAULT: 120	RANGE: 0 1000	UNITS: deg/s^2

Steering Control Angular Acceleration Maximum (ATC_STR_ACC_MAX)

Description

`ATC_STR_ACC_MAX` controls the "jerkiness" of the steering. It limits how fast the turn rate can change.

- **Low Value:** The steering wheel turns slowly. The rover eases into turns.
- **High Value:** The steering snaps to the desired angle.

Tuning & Behavior

- **Default Value:** 120 deg/s^2.
- **Recommendation:**
  - **Cars:** Set to 180-360 to ensure the servo moves fast enough to catch slides.
  - **Large Tanks:** Set lower (e.g. 60) to prevent current spikes when reversing the motors for a pivot turn.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





ATC_STR_RAT_MAX (Steering Control Rotation Rate Maximum)

DEFAULT: 120	RANGE: 0 1000	UNITS: deg/s

Steering Control Rotation Rate Maximum (ATC_STR_RAT_MAX)

Description

`ATC_STR_RAT_MAX` limits how fast your rover turns.

It acts as a speed governor for the yaw axis. Even if the navigation controller demands a sharp 90-degree turn, the steering controller will not command a rotation rate higher than this value.

- **Low Value (e.g. 60):** The rover turns slowly and gracefully. Good for large, top-heavy vehicles to prevent rollovers.
- **High Value (e.g. 360):** The rover turns instantly. Good for small, skid-steering bots.

The Mathematics

The desired turn rate ω_target is constrained:

$$\omega_target = \text{constrain}(\omega_requested, -\text{ATC_STR_RAT_MAX}, \text{ATC_STR_RAT_MAX})$$

The Engineer's View

Defined in `AR_AttitudeControl.cpp`.
Used in `get_steering_out_rate()`.
This limit is applied *after* the lateral acceleration limit (`ATC_TURN_MAX_G`). If the G-force limit allows a turn rate of 200 deg/s, but `ATC_STR_RAT_MAX` is 100 deg/s, the turn rate will be capped at 100 deg/s.

Tuning & Behavior

- **Default Value:** 120 deg/s.
- **Recommendation:**
  - **Ackermann Steering (Car):** Set to match the physical turning circle speed (usually 60-90 deg/s).
  - **Skid Steering (Tank):** Set to 180-360 deg/s for snappy pivots.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY

ATC_THR_G_BOOST (Throttle-Gain Boost)

DEFAULT: 0.0	RANGE: 0 1.0

Throttle-Gain Boost (ATC_THR_G_BOOST)

Description

`ATC_THR_G_BOOST` is a dynamic "authority booster" for high-performance flight.

Drones are most unstable when their motor speeds are changing rapidly—specifically during a sudden vertical "punch" (full throttle) or a rapid descent (low throttle). During these transients, the aerodynamics around the propellers are chaotic, and the drone needs extra "stiffness" to keep from wobbling.

`ATC_THR_G_BOOST` monitors how fast your throttle is changing and automatically "turns up the volume" on your Roll and Pitch PID gains for a few milliseconds to ensure the drone stays perfectly level through the shock.

- **Set to 0 (Default):** No boosting is applied. Gains stay constant.
- **Set to 1.0:** The drone will double its attitude authority during rapid throttle movements.

The Mathematics

The controller calculates the **Throttle Slew Rate** (how fast the PWM is changing). If the slew rate exceeds a threshold, the attitude gains are scaled:

$$\text{Boost_Scale} = 1.0 + (\text{ATC_THR_G_BOOST} \cdot \text{Normalized_Slew})$$

Effect: If you have `ATC_THR_G_BOOST` set to 1.0 and you slam the throttle to 100%, your `ATC_RAT_RLL_P` and `ATC_ANG_RLL_P` might temporarily double in value to "lock" the drone's orientation during the acceleration.

The Engineer's View

This parameter is the `_throttle_gain_boost` member in `AC_AttitudeControl_Multi`.

It is applied in `AC_AttitudeControl_Multi::update_throttle_gain_boost()`:

```
// AC_AttitudeControl_Multi.cpp
if (_motors.get_throttle_slew_rate() > AC_ATTITUDE_CONTROL_THR_G_BOOST_THRESH) {
    const float pd_boost = constrain_float(_throttle_gain_boost + 1.0f, 1.0, 2.0);
    set_PD_scale_mult(Vector3f(pd_boost, pd_boost, 1.0f));

    const float angle_p_boost = constrain_float((_throttle_gain_boost + 1.0f) * (_throttle_gain_boost + 1.0f), 1.0, 4.0);
    set_angle_P_scale_mult(Vector3f(angle_p_boost, angle_p_boost, 1.0f));
}
```

Notice the **Quadratic Scaling** on the Angle P-gain (`angle_p_boost`). This is designed to provide massive "restoring force" during vertical maneuvers, keeping the drone perfectly flat even when the air is extremely turbulent.

Tuning & Behavior

- **Default Value:** 0.0 (Disabled)
- **Range:** 0.0 - 1.0
- **Effect of Increasing:** Much more stable "vertical punches." Reduced "nose-dipping" during rapid acceleration.
- **Effect of Decreasing:** Smoother, more natural power transitions. Lower risk of high-frequency "chirping" during aggressive flight.

Use Case Recommendations

- **FPV Racing / Freestyle: Set to 0.5 - 1.0.** Essential for maintaining attitude precision during rapid directional changes and "split-S" maneuvers.
- **Heavy Lift Multirotors: Set to 0.2.** Large drones have sluggish motor response; a small boost helps them "dig in" when the pilot requests more power.
- **Cinematography: Keep at 0.0.** You want the drone to be smooth; sudden gain increases can cause tiny visible twitches in the video.

Troubleshooting

- **Scenario:** Drone "oscillates" or vibrates specifically during the first 0.5 seconds of a full-throttle climb.
  - **Diagnosis:** `ATC_THR_G_BOOST` is set too high, pushing the gains past the point of stability during the power-up.
  - **Fix:** Reduce `ATC_THR_G_BOOST` to 0.25 or 0.0.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





ATC_THR_MIX_MAN (Throttle Mix Manual)

DEFAULT: 0.1	RANGE: 0.1 0.9

Throttle Mix Manual (ATC_THR_MIX_MAN)

Description

`ATC_THR_MIX_MAN` is the manual counterpart to `ATC_THR_MIX_MAX`. It defines how much authority the drone's stabilization loops have when you are flying in "Old School" manual modes like **Stabilize** or **Acro**.

In manual modes, the pilot is in direct control of the throttle. If you push the stick to 100%, you expect 100% power. However, the drone still needs to stay level. `ATC_THR_MIX_MAN` determines how much the drone should "ignore" your 100% throttle request to ensure it doesn't flip over.

- **Low Value (0.1):** The drone prioritizes **Pilot Intent**. If you push 100% throttle, you get almost 100% throttle. This makes the drone feel extremely powerful, but it may "wash out" (lose control) during full-throttle climbs.
- **High Value (0.5+):** The drone prioritizes **Stability**. It will automatically lower your effective throttle to ensure it has enough motor range to keep the attitude perfectly locked.

The Mathematics

The Mixing Ratio (M) for manual modes is directly set by this parameter:

$$\text{Effective_Throttle} = \text{constrain}(\text{Pilot_Throttle}, \text{Min}, \text{Max} - \text{M} \cdot \text{Headroom})$$

Physical Meaning: A value of 0.5 means that even if you hold full throttle, the drone will reduce its average lift by 50% if that's what it takes to stop it from tilting or tumbling.

The Engineer's View

This parameter is the `_thr_mix_man` member in `AC_AttitudeControl_Multi`.

It is particularly important for high-power-to-weight vehicles (like 5-inch racing quads). On these vehicles, the motors are so powerful that even a 10% torque difference can create a massive roll rate. If `ATC_THR_MIX_MAN` is too low, the drone will "torque roll" (unintentionally tilt) when you floor the throttle.

Tuning & Behavior

- **Default Value:** 0.1 (10% reservation)
- **Range:** 0.1 - 0.9
- **Effect of Increasing:** Much more stable full-throttle climbs. Drone feels more "locked" during punch-outs.
- **Effect of Decreasing:** Maximum vertical power. Drone feels "raw" and "unfiltered."

Use Case Recommendations

- **Racing Drone (Standard Setup): Increase to 0.5.** Provides the best balance between speed and control during high-speed gate navigation.
- **FPV Freestyle: Keep at 0.1.** Freestyle pilots often use "Air Mode" (which handles mixing differently) and prefer the raw power of the motors for maneuvers like "Matty Flips."
- **Heavy Cargo Drone (Manual Recovery): Increase to 0.5.** If you ever have to switch to manual to save a heavy drone, you want it to prioritize staying level above all else.

Troubleshooting

- **Scenario:** In Stabilize mode, if I punch the throttle, the drone "climbs" but also tilts violently to the right.
  - **Diagnosis:** Torque Roll. The motors on the left can't spin any faster to correct the roll because they are already at 100%.
  - **Fix:** Increase `ATC_THR_MIX_MAN` to 0.5.
- **Scenario:** My drone feels "weak" in manual mode, even at 100% throttle.
  - **Diagnosis:** `ATC_THR_MIX_MAN` is too high, and the drone is excessively reserving power for stability.
  - **Fix:** Reduce `ATC_THR_MIX_MAN` to 0.2.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY

ATC_THR_MIX_MAX (Throttle Mix Maximum)

DEFAULT: 0.5	RANGE: 0.5 0.9

Throttle Mix Maximum (ATC_THR_MIX_MAX)

Description

`ATC_THR_MIX_MAX` handles the most dangerous moment for a flight controller: **Motor Saturation**.

If you punch the throttle to 100% to avoid an obstacle, all your motors spin up to their absolute limit. At that moment, the drone has no "extra" power left to roll or pitch. If a gust of wind hits it, the drone will tilt over and crash because the motors can't spin any faster to correct the attitude.

`ATC_THR_MIX_MAX` solves this by forcing the drone to "sacrifice" some vertical lift to save its attitude. It caps the priority of throttle, ensuring that a certain percentage of motor power is *always* reserved for staying level, even if that means the drone doesn't climb as fast as you asked.

- **Low Value (0.5):** Very safe. The drone will always keep 50% of its motor range available for attitude control. It will never flip over at full throttle, but its "vertical punch" will feel weak.
- **High Value (0.9):** Maximum performance. The drone uses 90% of its power for climbing. It will have a massive vertical "pop," but it is at high risk of losing attitude control (and crashing) during that climb.

The Mathematics

The controller calculates how much "headroom" is needed for the RPY (Roll/Pitch/Yaw) loops. `ATC_THR_MIX_MAX` sets the upper limit for this priority:

$$\text{Headroom_Reserved} = \text{MixRatio} \cdot \text{Max_Authority}$$

Physical Meaning: If `ATC_THR_MIX_MAX` is 0.5, and you are at 100% throttle, the flight controller will actually *reduce* the average throttle of all motors to 50% to give the Roll and Pitch loops the room they need to increase/decrease individual motor speeds.

The Engineer's View

This parameter is the `_thr_mix_max` member in `AC_AttitudeControl_Multi`.

It is enforced in the motor mixer:

```
// AC_AttitudeControl_Multi.cpp
_throttle_rpy_mix_desired = (1.0f - ratio) * _thr_mix_min + ratio * _thr_mix_max;
```

This is the reason why ArduPilot drones sometimes sound like they are "stuttering" or reducing power when you do a high-speed maneuver—the software is dynamically lowering the throttle to ensure it can still meet the roll/pitch torque requirements.

Tuning & Behavior

- **Default Value:** 0.5 (50% headroom)
- **Range:** 0.5 - 0.9
- **Effect of Increasing:** Much faster vertical acceleration. The drone "rockets" up.
- **Effect of Decreasing:** Maximum stability. The drone will never lose its level orientation, regardless of how hard you push it.

Use Case Recommendations

- **Standard Multirotor: Keep Default (0.5).** 50% is a very safe value that prevents "washout" in turns.
- **Racing Drone: Increase to 0.7 - 0.8.** Needed for that "snappy" vertical authority, but requires a very rigid frame and perfectly balanced props.
- **Heavy Lift / Expensive Payload: Decrease to 0.45.** Never sacrifice the safety of the payload for a faster climb rate.

Troubleshooting

- **Scenario:** When I do a 100% throttle "punch out," the drone starts to tilt or "tumble" to one side.
  - **Diagnosis:** Throttle Saturation. The motors are at 100%, and there isn't enough headroom left for the attitude controller to stay level.
  - **Fix:** Reduce `ATC_THR_MIX_MAX` to 0.5 or 0.45.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY

ATC_THR_MIX_MIN (Throttle Mix Minimum)

DEFAULT: 0.1	RANGE: 0.1 0.25

Throttle Mix Minimum (ATC_THR_MIX_MIN)

Description

`ATC_THR_MIX_MIN` is a "priority setting" for your drone's motors. It handles the difficult moment of **Landing**.

When you land a drone, you pull the throttle stick to zero. If the drone is tilted slightly as it touches the ground, the flight controller will try to level it by spinning up some motors. However, if the motors spin up too hard while the drone is on the ground, it can create a "bounce" or "ground resonance" that makes the drone flip over.

`ATC_THR_MIX_MIN` limits how much "authority" the attitude controller has to overrule your low throttle command.

- **Low Value (0.1):** The drone prioritizes your throttle command. If you pull the stick to zero, the motors will stay very low, even if the drone is tilted. This makes for safer, "stickier" landings.
- **High Value (0.25):** The drone prioritizes staying level. Even at zero throttle, it will spin the motors aggressively to stay flat. This can cause the drone to bounce back into the air during a touchdown.

The Mathematics

The motor mixer calculates a priority ratio (R) between attitude correction and vertical lift. `ATC_THR_MIX_MIN` sets the lower bound of this ratio:

$$\text{MixRatio} = \text{constrain}(\text{Current_Priority}, \text{THR_MIX_MIN}, \text{THR_MIX_MAX})$$

Effect: If the drone is landing and the Calculated Priority is 0.05, it will be "boosted" to 0.1 (the minimum). This ensures that at least 10% of the motor authority is always reserved for leveling the drone, regardless of how low the throttle is.

The Engineer's View

This parameter is the `_thr_mix_min` member in `AC_AttitudeControl_Multi`.

It is used in `AC_AttitudeControl_Multi::update_throttle_rpy_mix()`:

```
// AC_AttitudeControl_Multi.cpp
_throttle_rpy_mix = (1.0f - ratio) * _thr_mix_min + ratio * _thr_mix_max;
```

This dynamic mixing allows the drone to smoothly transition from "High Control" during flight to "Low Authority" during landing, preventing the feedback loops that cause "Ground Wobble."

Tuning & Behavior

- **Default Value:** 0.1 (10% Authority)
- **Range:** 0.1 - 0.25
- **Effect of Increasing:** The drone fights harder to stay level during landing. Safer for landing in high wind on a slope.
- **Effect of Decreasing:** The drone is more likely to stay on the ground once it touches down. Reduces "bouncing."

Use Case Recommendations

- **Top-Heavy Drones: Increase to 0.15.** Drones with high centers of gravity (e.g., those with tall antennas or top-mounted batteries) are unstable on the ground and need more authority to stay upright during touchdown.
- **Underpowered Drones: Keep at 0.1.** You want the motors to stop as soon as possible to prevent a tip-over.
- **Autonomous Landing (RTL): Keep Default.** 0.1 is the most tested value for reliable auto-landings.

Troubleshooting

- **Scenario:** Every time I land, the drone touches the ground and then "jumps" back up 20cm before finally disarming.
  - **Diagnosis:** `ATC_THR_MIX_MIN` is too high. The drone is over-correcting for the impact and "powering" itself back into the air.
  - **Fix:** Reduce `ATC_THR_MIX_MIN` to 0.1.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY

ATC_TURN_MAX_G (Turning maximum G force)

DEFAULT: 0.6	RANGE: 0.1 10.0	UNITS: gravities

Turning maximum G force (ATC_TURN_MAX_G)

Description

`ATC_TURN_MAX_G` is the "Anti-Rollover" setting for rovers and boats.

When a vehicle turns at high speed, centrifugal force pushes it sideways. If this force is too high, a car might flip over, or a boat might slide (drift) uncontrollably. This parameter caps that force. If the rover is driving fast, the autopilot will automatically widen its turns to ensure the lateral force never exceeds this limit.

- **Low Value (e.g. 0.3):** Safe, slow turns. Good for top-heavy trucks or tractors.
- **High Value (e.g. 1.5):** Aggressive, sharp turns. Good for low-slung racing cars with high-grip tires.

The Mathematics

The maximum turn rate (ω_max) is calculated from the current speed (V) and the G-limit (A_max_g):

$$\omega_max = \frac{A_max_g \cdot g}{V}$$

As speed V increases, the allowed turn rate ω must decrease to stay under the G-limit.

Tuning & Behavior

- **Default Value:** 0.6 G.
- **Recommendation:**
  - **Ackermann Car:** Set to the point where the tires just start to squeal or the car feels "tippy."
  - **Boat:** Set to 0.3-0.5 to prevent massive heel angles.
  - **Skid-Steer:** Usually set high (1.0+) as they are typically low-profile and stable.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **AUTO** parameter group configures the behavior of the **Auto** flight mode, where the vehicle follows a pre-programmed mission (waypoints) uploaded from the Ground Control Station (GCS).

Key Concepts

1. Mission Behavior (`AUTO_OPTIONS`)

A bitmask that modifies how the vehicle executes missions.

- **Allow Arming:** Whether the vehicle can be armed directly into Auto mode.
- **Takeoff behavior:** specific behaviors for VTOL takeoff.

2. Vehicle Specifics

- **Rover:** `AUTO_KICKSTART` helps overcome static friction (stiction) by momentarily boosting throttle at the start of motion.
- **Plane:** `AUTO_FBW_STEER` (sometimes aliased) might control steering assist.

Parameter Breakdown

- `AUTO_OPTIONS` : Bitmask of options.
- `AUTO_KICKSTART` : (Rover) Throttle boost pulse duration/intensity.
- `AUTO_TRIGGER_PIN` : Input pin to start the mission.

Integration Guide

- **Planning:** Use the "Plan" tab in Mission Planner or QGroundControl to create a mission.
- **Upload:** "Write Waypoints" to the flight controller.
- **Execution:** Switch to Auto mode. Ensure you have a GPS Lock (Green LED).

Developer Notes

- **Logic:** Handled by `ModeAuto` class in each vehicle directory (`ArduCopter/mode_auto.cpp` , `Rover/mode_auto.cpp`).

AUTO_CONFIG (GPS Auto Configuration)

DEFAULT: 1	RANGE: 0 1

GPS Auto Configuration (AUTO_CONFIG)

Description

Allows ArduPilot to automatically send configuration commands to the GPS module on boot. This ensures the GPS is outputting the correct MAVLink-compatible sentences at the optimal rate.

Tuning & Behavior

- **Default Value:** 1 (Enabled)
- **Only disable this if using a specialized GPS module that requires a static, pre-programmed configuration.**

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





AUTO_KICKSTART (Auto Kickstart Acceleration)

DEFAULT: 0	RANGE: 0 20.0	UNITS: m/s^2

Auto Kickstart Acceleration (AUTO_KICKSTART)

Description

`AUTO_KICKSTART` allows you to delay the start of an autonomous mission until the vehicle is physically pushed or launched.

Normally, when you switch to AUTO mode, the vehicle starts its mission immediately. If you set this parameter to a value like **2.0**, the vehicle will sit still until it detects an acceleration of 2.0 m/s^2 (a firm push). This is useful for starting missions where you want to manually launch the rover or ensuring the motors only start once the vehicle is on its way.

- **0 (Default):** Mission starts immediately upon entering AUTO mode.
- **Value:** Acceleration threshold to trigger the start.

Tuning & Behavior

- **Usage:** Common in racing rovers or hand-launched small vehicles where a "Push-to-Start" behavior is desired for safety or convenience.

AUTO_ONLY (Camera Auto Mode Only)

DEFAULT: 0	RANGE: 0 1

Camera Auto Mode Only (AUTO_ONLY)

Description

Restricts camera triggering functionality to autonomous mission modes. This prevents accidental triggers during manual flight or while on the ground in manual modes.

Tuning & Behavior

- **Default Value:** 0 (Disabled - Can trigger in any mode)
- **Setting to 1 ensures camera missions only run during the actual mission legs.**

AUTO_OPTIONS (Auto mode options)

DEFAULT: 0

Auto mode options (AUTO_OPTIONS)

Description

Configuration bitmask for specific autonomous flight behaviors.

Tuning & Behavior

- **Default Value:** 0
- **Bitmask Examples:**
  - 0: Allow Arming (in Auto)
  - 1: Allow Takeoff Without Raising Throttle
  - 2: Ignore pilot yaw
  - 7: Allow weathervaning

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





AUTO_SWITCH (GPS Auto Switch)

DEFAULT: 1	RANGE: 0 4

GPS Auto Switch (AUTO_SWITCH)

Description

Logic for choosing the active GPS sensor when multiple modules are connected. It can prioritize the module with the best "fix" or blend the data from both modules.

Tuning & Behavior

- **Default Value:** 1 (Use Best)
- **Values:** 0:Disabled, 1:Use Best, 2:Blend, 3:Use Primary, 4:Use Secondary

AUTO_TRIGGER_PIN (Auto Mode Trigger Pin)

DEFAULT: -1	RANGE: -1 103

Auto Mode Trigger Pin (AUTO_TRIGGER_PIN)

Description

`AUTO_TRIGGER_PIN` allows a physical button or external signal to start the autonomous mission.

AUTO_TRIM (Automatic servo trim)

DEFAULT: 0	RANGE: 0 1

Automatic servo trim (AUTO_TRIM)

Description

Continuously learns and saves the required servo trim offsets while flying in stabilized modes. This compensates for slight physical imbalances in the airframe without requiring manual trim adjustments.

Tuning & Behavior

- **Default Value:** 0 (Disabled)
- **Only active in stabilized modes (e.g., FBWA) when the pilot is not providing stick inputs.**

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **AUTOLAND** parameter group appears to contain legacy or alias parameters related to the Automatic Landing sequence in ArduPlane.

Note: In modern ArduPilot firmware, most landing configuration is handled by:

- `LAND_` group: For Copter landing speeds.
- `TECS_` group: For Plane glide slope and descent rates.
- `RTL_AUTOLAND`: To trigger landing after RTL.
- **Mission Items:** `DO_LAND_START`, `NAV_LAND` waypoints.

Key Parameters (Legacy/Reference)

- `AUTOLAND_WP_ALT`: Likely referred to the altitude of the final approach waypoint.
- `AUTOLAND_OPTIONS`: Bitmask for behavior (e.g., flare logic).

Integration Guide

For setting up Auto Landing in modern firmware:

1. **Plane:** Use a mission with `DO_LAND_START` → `NAV_LOITER_TO_ALT` → `NAV_LAND`.
2. **Copter:** Use `LAND_SPEED` and `LAND_SPEED_HIGH` to tune the descent.

Developer Notes

- These parameters may not exist in the current `ardupilot` codebase and might be artifacts of older documentation generation.

AUTOLAND_OPTIONS (Landing options bitmask)

DEFAULT: 0

Landing options bitmask (AUTOLAND_OPTIONS)

Description

Configuration bitmask for various landing behaviors.

Tuning & Behavior

- **Default Value:** 0
- **Bitmask:**
  - 0: honor min throttle during landing flare
  - 1: Increase Target landing airspeed constraint From Trim Airspeed to AIRSPEED_MAX

AUTOLAND_WP_ALT (Takeoff mode altitude)

DEFAULT: 50	RANGE: 0 200	UNITS: m

Takeoff mode altitude (AUTOLAND_WP_ALT)

Description

Defines the target altitude for the aircraft to reach when in "Takeoff" mode.

Tuning & Behavior

- **Default Value:** 50 m
- **Range:** 0 to 200 m

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





AUTOLAND_WP_DIST (Takeoff mode distance)

DEFAULT: 200	RANGE: 0 500	UNITS: m

Takeoff mode distance (AUTOLAND_WP_DIST)

Description

Defines the horizontal distance from the takeoff point to the designated loiter point where the aircraft will wait after completing the initial climb.

Tuning & Behavior

- **Default Value:** 200 m
- **Range:** 0 to 500 m

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





AUTOTUNE CONFIGURATION (AUTOTUNE)

Overview

The **AUTOTUNE** parameter group configures the **AutoTune** flight mode. This is an automated system-identification routine where the autopilot rapidly twitches the vehicle in Roll, Pitch, and Yaw to measure its physical response (inertia, authority, damping) and automatically calculates the optimal PID gains (`ATC_` group).

Key Concepts

1. Aggressiveness (`AUTOTUNE_AGGR`)

This parameter controls how "sharp" or "locked-in" the resulting tune will be.

- **0.1 (Default):** Standard crisp tune. Good for most racers and camera drones.
- **0.075:** Softer tune. Better for large cinematic builds, older frames, or if the default tune causes oscillation.
- **0.05:** Very soft/damped.

2. Axes Selection (`AUTOTUNE_AXES`)

Selects which axes to tune during the session.

- **1:** Roll only.
- **2:** Pitch only.
- **4:** Yaw only.
- **7 (Default):** Roll, Pitch, and Yaw.

Parameter Breakdown

- `AUTOTUNE_AGGR`: Aggressiveness target.
- `AUTOTUNE_AXES`: Bitmask of axes to tune.
- `AUTOTUNE_MIN_D`: Minimum D-gain to ensure damping.

Integration Guide

Running AutoTune

1. **Setup:** Map a switch to AutoTune mode (or use `AUTOTUNE_AXES`).
2. **Safety:** Find a large open area on a calm day.
3. **Process:**
    - Take off and establish a stable hover (AltHold or Loiter).
    - Switch to AutoTune.
    - The vehicle will start twitching (Roll → Pitch → Yaw).
    - **Pilot Input:** If the vehicle drifts too far, use the sticks to gently bring it back. The tuning pauses while sticks are moved and resumes when centered.
4. **Completion:** The twitching stops. Land and Disarm to save the new PIDs. Switch out of AutoTune *before* disarming to discard them.

Developer Notes

- **Library:** `ArduCopter/mode_autotune.cpp`
- **Algorithm:** Iterative Step Response optimization. It tries to match the requested Rate vs Actual Rate with minimal overshoot.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





AUTOTUNE_ACC_MAX (AutoTune maximum allowable angular acceleration)

DEFAULT: 0	RANGE: 1 4000	UNITS: deg/s^2

AutoTune maximum allowable angular acceleration (AUTOTUNE_ACC_MAX)

Description

Limits the intensity of the "twitches" used by the Helicopter AutoTune process by capping the maximum angular acceleration.

Tuning & Behavior

- **Default Value:** 0 (Calculated automatically)
- **Range:** 1 to 4000 deg/s^2

AUTOTUNE_AGGR (Autotune aggressiveness)

DEFAULT: 0.075	RANGE: 0.05 0.10

Autotune aggressiveness (AUTOTUNE_AGGR)

Description

Determines how "sharp" the resulting PID gains will be. Higher aggressiveness results in a more responsive aircraft but increases the risk of oscillation.

Tuning & Behavior

- **Default Value:** 0.075
- **Range:** 0.05 to 0.10 (Typical)
- **Lower values (e.g., 0.05) produce a "softer" tune suitable for cinematic flight.**
- **Higher values (e.g., 0.10) produce a "stiffer" tune suitable for racing or high-precision flight.**

AUTOTUNE_AXES (Autotune axis bitmask)

DEFAULT: 7

Autotune axis bitmask (AUTOTUNE_AXES)

Description

Selects which flight axes (Roll, Pitch, Yaw) will be tested and tuned during the next AutoTune session.

Tuning & Behavior

- **Default Value:** 7 (Roll, Pitch, and Yaw)
- **Bitmask:**
  - Bit 0: Roll
  - Bit 1: Pitch
  - Bit 2: Yaw
- **Tuning one axis at a time can be safer for large or unbalanced aircraft.**

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





AUTOTUNE_FRQ_MAX (AutoTune maximum sweep frequency)

DEFAULT: 70.0	RANGE: 50 120	UNITS: Hz

AutoTune maximum sweep frequency (AUTOTUNE_FRQ_MAX)

Description

The ending frequency for the automated frequency response testing used in Helicopter AutoTune.

Tuning & Behavior

- **Default Value:** 70.0 Hz
- **Range:** 50 to 120 Hz

AUTOTUNE_FRQ_MIN (AutoTune minimum sweep frequency)

DEFAULT: 10.0	RANGE: 10 30	UNITS: Hz

AutoTune minimum sweep frequency (AUTOTUNE_FRQ_MIN)

Description

The starting frequency for the automated frequency response testing used in Helicopter AutoTune.

Tuning & Behavior

- **Default Value:** 10.0 Hz
- **Range:** 10 to 30 Hz

AUTOTUNE_GN_MAX (AutoTune maximum response gain)

DEFAULT: 1.0	RANGE: 1 2.5

AutoTune maximum response gain (AUTOTUNE_GN_MAX)

Description

Target closed-loop gain for the Helicopter rate controller during the tuning process.

Tuning & Behavior

- **Default Value:** 1.0
- **Range:** 1.0 to 2.5

AUTOTUNE_MIN_D (AutoTune minimum D)

DEFAULT: 0.0005	RANGE: 0.0001 0.005

AutoTune minimum D (AUTOTUNE_MIN_D)

Description

A safety floor for the Rate D gain. AutoTune will not lower the D gain below this value, even if the physics of the aircraft suggest it.

Tuning & Behavior

- **Default Value:** 0.0005
- **Range:** 0.0001 to 0.005

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





AUTOTUNE_OPTIONS (AutoTune options)

DEFAULT: 0

AutoTune options (AUTOTUNE_OPTIONS)

Description

Bitmask for configuring specific AutoTune behaviors, such as whether to automatically save gains upon completion or use specific tuning logic.

Tuning & Behavior

- **Default Value:** 0

AUTOTUNE_RAT_MAX (Autotune maximum allowable angular rate)

DEFAULT: 0	RANGE: 0 500	UNITS: deg/s

Autotune maximum allowable angular rate (AUTOTUNE_RAT_MAX)

Description

Limits the maximum angular velocity reached during the Helicopter AutoTune process.

Tuning & Behavior

- **Default Value:** 0 (Calculated automatically)
- **Range:** 0 to 500 deg/s

AUTOTUNE_SEQ (AutoTune Sequence Bitmask)

DEFAULT: 3

AutoTune Sequence Bitmask (AUTOTUNE_SEQ)

Description

Controls the sequence of tests performed during Helicopter AutoTune (e.g., FeedForward, Rate D/P, Angle P).

Tuning & Behavior

- **Default Value:** 3
- **Bitmask:**
  - 1: VFF Only
  - 2: Rate D/Rate P Only
  - 4: Angle P Only
  - 8: Max Gain Only
  - 16: Tune Check

AUTOTUNE_VELXY_P (AutoTune velocity xy P gain)

DEFAULT: 0.1	RANGE: 0 1

AutoTune velocity xy P gain (AUTOTUNE_VELXY_P)

Description

Proportional gain for the position-hold logic active during Helicopter AutoTune. This helps keep the aircraft from drifting significantly while it is being "twitched" for frequency response analysis.

Tuning & Behavior

- **Default Value:** 0.1
- **Range:** 0 to 1

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **AVD** parameter group configures the **ADSB Collision Avoidance** system. This library uses data from the ADS-B Receiver (`ADSB` group) to detect manned aircraft on a collision course and automatically take evasive action.

Note: Do not confuse this with `AVOID`, which handles obstacle avoidance (walls/trees) using Lidar/cameras.

Key Concepts

1. Failsafe vs. Warning

- **Warning (`W_` params):** When an intruder is detected within the "Warning" radius/time, the GCS triggers an alert ("Traffic!").
- **Failsafe (`F_` params):** When an intruder breaches the critical "Failsafe" radius/time, the vehicle takes autonomous action.

2. Actions (`AVD_F_ACTION`)

What to do when a collision is imminent?

- **0:** Disable.
- **1:** Report Only.
- **2:** Climb or Descend (Plane only).
- **3:** RTL.
- **4:** Hover.
- **5:** Loiter.

Parameter Breakdown

- `AVD_ENABLE`: Master switch.
- `AVD_W_DIST_XY`: Horizontal distance threshold for Warning.
- `AVD_F_DIST_XY`: Horizontal distance threshold for Failsafe action.
- `AVD_F_TIME`: Time to impact (seconds) threshold for Failsafe.

Integration Guide

1. **Enable:** Set `AVD_ENABLE = 1`.
2. **Define Zones:** Set your Warning zone (e.g., 2000m) and Failsafe zone (e.g., 1000m).
3. **Select Action:** For Copter, `RTL` or `Loiter` is common. For Plane, `Climb/Descend` tries to change altitude to clear the traffic.

Developer Notes

- **Library:** `libraries/AP_Avoidance`
- **Prefix:** `AVD_`

AVD_ENABLE (Enable Avoidance using ADSB)

DEFAULT: 0	RANGE: 0 1

Enable Avoidance using ADSB (AVD_ENABLE)

Description

Global master switch for the ADS-B based collision avoidance system.

Tuning & Behavior

- **Default Value:** 0 (Disabled)
- **Values:** 0:Disabled, 1:Enabled
- **Requires a functional ADS-B receiver (see ADSB_TYPE).**

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





AVD_F_ACTION (Collision Avoidance Behavior (Fail))

DEFAULT: 1	RANGE: 0 6

Collision Avoidance Behavior (AVD_F_ACTION)

Description

Defines the automatic maneuver the aircraft will perform when it determines that a collision is likely within the F_TIME horizon.

Tuning & Behavior

- **Default Value:** 1 (Report)
- **Values:** 0:None, 1:Report, 2:Climb Or Descend, 3:Move Horizontally, 4:Move Perpendicularly in 3D, 5:RTL, 6:Hover

AVD_F_ALT_MIN (ADS-B avoidance minimum altitude)

DEFAULT: 0	UNITS: m

ADS-B avoidance minimum altitude (AVD_F_ALT_MIN)

Description

Safety floor for ADS-B avoidance. Avoidance maneuvers are disabled below this altitude to prevent collisions with ground structures or during landing.

Tuning & Behavior

- **Default Value:** 0 m (Disabled)
- **Recommended to be set above the local tree line or building height.**

AVD_F_DIST_XY (Distance Fail XY)

DEFAULT: 300	RANGE: 1 2000	UNITS: m

Distance Fail XY (AVD_F_DIST_XY)

Description

Horizontal separation threshold for the "Fail" threat level.

Tuning & Behavior

- **Default Value:** 300 m (Plane), 100 m (Copter)
- **Range:** 1 to 2000 m

AVD_F_DIST_Z (Distance Fail Z)

DEFAULT: 100	RANGE: 1 500	UNITS: m

Distance Fail Z (AVD_F_DIST_Z)

Description

Vertical separation threshold for the "Fail" threat level.

Tuning & Behavior

- **Default Value:** 100 m
- **Range:** 1 to 500 m

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





AVD_F_RCVRY (Recovery behaviour after a fail event)

DEFAULT: 2	RANGE: 0 3

Recovery behaviour after a fail event (AVD_F_RCVRY)

Description

Specifies the flight mode or action the aircraft transitions to once a previously detected collision threat has been cleared.

Tuning & Behavior

- **Default Value:** 2 (RTL) for Copter, 3 (Resume/Loiter) for Plane
- **Values:** 0:Remain in AVOID_ADSB, 1:Resume previous flight mode, 2:RTL, 3:Resume if AUTO else Loiter

AVD_F_TIME (Time Horizon Fail)

DEFAULT: 30	RANGE: 1 120	UNITS: s

Time Horizon Fail (AVD_F_TIME)

Description

The look-ahead time for the "Fail" threat level. If a collision is predicted to occur within this many seconds, the AVD_F_ACTION will be triggered.

Tuning & Behavior

- **Default Value:** 30 s
- **A larger value provides more time to react but may lead to more false alarms.**

AVD_OBS_MAX (Maximum number of obstacles to track)

DEFAULT: 20	RANGE: 1 50

Maximum number of obstacles to track (AVD_OBS_MAX)

Description

Limits the memory and CPU resources spent on tracking distant ADS-B vehicles.

Tuning & Behavior

- **Default Value:** 20
- **Range:** 1 to 50

AVD_W_ACTION (Collision Avoidance Behavior - Warn)

DEFAULT: 1	RANGE: 0 1

Collision Avoidance Behavior - Warn (AVD_W_ACTION)

Description

Defines the action taken when a potential collision is detected within the W_TIME horizon.

Tuning & Behavior

- **Default Value:** 1 (Report)
- **Values:** 0:None, 1:Report
- **Usually set to "Report" to provide situational awareness without taking automatic flight control actions.**

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





AVD_W_DIST_XY (Distance Warn XY)

DEFAULT: 1000	RANGE: 1 5000	UNITS: m

Distance Warn XY (AVD_W_DIST_XY)

Description

Horizontal separation threshold for the "Warn" threat level.

Tuning & Behavior

- **Default Value:** 1000 m (Plane), 300 m (Copter)
- **Range:** 1 to 5000 m

AVD_W_DIST_Z (Distance Warn Z)

DEFAULT: 300	RANGE: 1 1000	UNITS: m

Distance Warn Z (AVD_W_DIST_Z)

Description

Vertical separation threshold for the "Warn" threat level.

Tuning & Behavior

- **Default Value:** 300 m
- **Range:** 1 to 1000 m

AVD_W_TIME (Time Horizon Warn)

DEFAULT: 30	RANGE: 1 120	UNITS: s

Time Horizon Warn (AVD_W_TIME)

Description

The look-ahead time for the "Warn" threat level. This provides an earlier notification than the "Fail" level.

Tuning & Behavior

- **Default Value:** 30 s
- **Range:** 1 to 120 s

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

Key Concepts

AVOID_ACCEL_MAX (Avoidance Max Acceleration)

DEFAULT: 3.0	RANGE: 0 10	UNITS: m/s^2

Avoidance Max Acceleration (AVOID_ACCEL_MAX)

Description

`AVOID_ACCEL_MAX` limits how hard the drone brakes when it sees a wall.

If a proximity sensor detects an object within the `AVOID_DIST_MAX` limit, the autopilot commands the drone to stop. This parameter ensures the drone doesn't pitch back so violently that it crashes or destabilizes the payload.

Tuning & Behavior

- **Default Value:** 3.0 m/s^2.
- **Recommendation:** Set to **1.0 or 2.0** for smoother braking on large, heavy cinema drones. Keep at **3.0** for smaller, agile quads.

AVOID_ALT_MIN (Avoidance Minimum Altitude)

DEFAULT: 0	RANGE: 0 10	UNITS: m

Avoidance Minimum Altitude (AVOID_ALT_MIN)

Description

`AVOID_ALT_MIN` prevents the drone from freaking out near the ground.

Proximity sensors (like 360 Lidar) often detect the ground as an "obstacle" when landing or taking off. If avoidance stays active, the drone might refuse to land because it thinks the floor is a wall. This parameter sets a floor altitude where avoidance is turned off.

- **Default Value:** 0 (Disabled).
- **Recommendation:** Set to **1.0 or 2.0 meters**.

The Engineer's View

Used in `AC_Avoid::adjust_velocity`. If `current_alt < alt_min`, the velocity modification logic is bypassed.

Tuning & Behavior

- **Safety:** Always set this if you have downward-angled sensors or if you fly in tall grass.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





AVOID_ANGLE_MAX (Avoidance Max Lean Angle)

DEFAULT: 1000	RANGE: 0 4500	UNITS: cdeg

Avoidance Max Lean Angle (AVOID_ANGLE_MAX)

Description

`AVOID_ANGLE_MAX` limits how aggressively the drone can tilt away from a threat.

- **1000 (Default):** 10 degrees. Gentle avoidance.
- **3000:** 30 degrees. Aggressive dodge.

Tuning & Behavior

- **Default Value:** 1000 (10°).
- **Recommendation:** Keep low (10-15°) to ensure smooth video and prevent the drone from over-reacting to sensor noise.

AVOID_BACKUP_DZ (Avoidance Backup Deadzone)

DEFAULT: 0.1	RANGE: 0 1	UNITS: m

Avoidance Backup Deadzone (AVOID_BACKUP_DZ)

Description

`AVOID_BACKUP_DZ` prevents the drone from "jittering" back and forth.

If the drone is exactly at the `AVOID_DIST_MAX` limit (e.g. 2.0m from a wall), sensor noise might report 1.99m then 2.01m. Without a deadzone, the drone would twitch forward and backward. This parameter adds a buffer.

Tuning & Behavior

- **Default Value:** 0.1 m.
- **Recommendation:** Leave at **0.1** or increase to **0.2** if your Lidar is noisy.

AVOID_BACKUP_SPD (Avoidance Backup Speed)

DEFAULT: 0.75	RANGE: 0 2	UNITS: m/s

Avoidance Backup Speed (AVOID_BACKUP_SPD)

Description

`AVOID_BACKUP_SPD` controls how fast the drone retreats.

If an obstacle moves towards the drone (or the drone drifts into it), the autopilot can actively fly backwards to maintain the `AVOID_DIST_MAX` clearance.

Tuning & Behavior

- **Default Value:** 0.75 m/s.
- **Recommendation:** Keep low (0.5 - 1.0 m/s). You don't want the drone to bolt backwards into *another* obstacle that its rear sensors haven't seen yet.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





AVOID_BACKZ_SPD (Avoidance Backup Vertical Speed)

DEFAULT: 0.75	RANGE: 0 2	UNITS: m/s

Avoidance Backup Vertical Speed (AVOID_BACKZ_SPD)

Description

`AVOID_BACKZ_SPD` is the same as `AVOID_BACKUP_SPD` but for up/down movement.

If you fly too close to the ceiling (with an upward facing Lidar), this controls how fast the drone descends to regain clearance.

Tuning & Behavior

- **Default Value:** 0.75 m/s.
- **Recommendation:** Keep conservative.

AVOID_BEHAVE (Avoidance behaviour)

DEFAULT: 0	RANGE: 0 1

Avoidance behaviour (AVOID_BEHAVE)

Description

`AVOID_BEHAVE` determines the drone's "Personality" when it sees an object like a wall or a tree.

- **0: Slide (Default for Copter).** The drone will try to "slip" past the obstacle. If you fly straight at a corner, it will veer slightly to one side to maintain speed while keeping a safe distance (`AVOID_MARGIN`).
- **1: Stop (Default for Rover).** The drone will simply come to a full stop in front of the obstacle. It will not try to go around it.

The Mathematics

In **Slide (0)** mode, the velocity vector $\vec{V}$ is projected onto the tangent of the obstacle boundary:

$$\vec{V}_safe = \vec{V} - (\vec{V} \cdot \hat{n})\hat{n}$$

where $\hat{n}$ is the normal vector pointing from the obstacle.

In **Stop (1)** mode, the component of velocity toward the obstacle is simply zeroed.

Tuning & Behavior

- **Default Value:** 0 (Slide)
- **Recommendation:**
  - **Multicopters:** Use **0 (Slide)**. It feels much more natural and prevents the drone from "stuttering" when flying near complex structures.
  - **Rovers:** Use **1 (Stop)** to prevent the vehicle from accidentally sliding into a ditch while trying to "veer" around a rock.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY

AVOID_DIST_MAX (Avoidance Max Distance)

DEFAULT: 5	RANGE: 1 20	UNITS: m

Avoidance Max Distance (AVOID_DIST_MAX)

Description

`AVOID_DIST_MAX` defines the "Reaction Zone."

When an object is further away than this, the drone ignores it. Once it enters this range, the drone starts calculating if it needs to brake. This is effectively the range of your sensor that you *trust* for avoidance logic.

Tuning & Behavior

- **Default Value:** 5 meters.
- **Recommendation:** Set to **80%** of your Lidar's reliable range.
  - **TFmini:** 4m.
  - **LIDAR-Lite:** 10m.
  - **Lightware SF40:** 20m.

AVOID_ENABLE (Avoidance control enable/disable)

DEFAULT: 3	RANGE: 0 7

Avoidance control enable/disable (AVOID_ENABLE)

Description

`AVOID_ENABLE` activates the logic that prevents the drone from hitting things. While `FENCE` prevents the drone from leaving a large area, `AVOID` handles "Local Obstacles" detected by sensors (like Lidars) or boundaries (like the Fence).

Key Bits:

- **Bit 0 (1): Use Fence.** The drone will treat the geofence as a solid wall it cannot fly into.
- **Bit 1 (2): Use Proximity Sensor.** Enables avoidance using 360-degree Lidars or rangefinders.
- **Bit 2 (4): Use Beacon Fence.** Treat beacon-defined boundaries as obstacles.

The Engineer's View

Used in `AC_Avoid::adjust_velocity()`.
When enabled, the avoidance library intercepts the pilot's velocity commands and "pushes back" if they lead toward an obstacle. It effectively creates a virtual "Repulsive Field" around the drone.

Tuning & Behavior

- **Default Value:** 3 (Use Fence + Proximity) on most systems.
- **Recommendation:** Keep at **3** if you have any avoidance hardware installed.
- **Safety:** This logic is active in **Loiter**, **Circle**, and **Auto** modes. In **Stabilize** or **Acro**, the drone will NOT automatically stop for obstacles.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY

AVOID_MARGIN (Avoidance distance margin)

DEFAULT: 2.0	RANGE: 1.0 10.0	UNITS: m

Avoidance distance margin (AVOID_MARGIN)

Description

`AVOID_MARGIN` is the "Social Distancing" rule for your drone. It defines how close the drone is allowed to get to a wall or obstacle before the avoidance logic kicks in.

- **2.0 (Default):** The drone will try to stay at least 2 meters away from everything.

The Mathematics

The desired velocity is scaled by the distance to the obstacle d using a square-root deceleration curve:

$$v_max = \sqrt{2 \cdot \text{AVOID_ACCEL_MAX} \cdot (d - \text{AVOID_MARGIN})}$$

If $d \leq \text{AVOID_MARGIN}$, $v_max = 0$ (The drone stops).

Tuning & Behavior

- **Default Value:** 2.0 meters
- **Recommendation:**
  - **Indoors:** Decrease to **0.5m or 1.0m** to allow flying through doorways.
  - **Outdoors:** Keep at **2.0m**. This provides enough reaction time for the Lidar to see small objects like branches.
- **Latency:** If your sensor has high latency (e.g., a slow spinning Lidar), you must increase this margin.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **AXIS** parameter group currently contains parameters related to **Custom Controller** axis masking.

Key Parameters

- `AXIS_MASK` (or `CC_AXIS_MASK`): Determines which flight axes (Roll, Pitch, Yaw) are handled by a user-defined custom control backend instead of the standard ArduPilot PID loops.

See Also

- CC - Custom Controller configuration.

AXIS_MASK (Custom Controller Axis Mask)

`DEFAULT: 0`

Custom Controller Axis Mask (AXIS_MASK)

Description

Defines which flight axes (Roll, Pitch, Yaw) are managed by a custom user-defined controller instead of the standard ArduPilot attitude control logic.

Tuning & Behavior

- **Default Value:** 0 (Standard controllers active on all axes)
- **Bitmask:**
  - Bit 0: Roll
  - Bit 1: Pitch
  - Bit 2: Yaw

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY

Overview

The **BACKEND** parameter group (specifically `BACKEND_TYPE`) configures the **AP_Logger** storage destinations. In the full parameter list, this is usually seen as `LOG_BACKEND_TYPE`.

Key Concepts

DataFlash Logging

ArduPilot records high-frequency flight data (IMU, EKF, GPS, Motors) to a "DataFlash" log (`.BIN` file). This log is essential for post-flight analysis and tuning.

Backend Types

- **File (1):** Standard SD Card logging.
- **MAVLink (2):** Streams log data over the MAVLink telemetry link to the GCS (requires high bandwidth).
- **Block (4):** Logs to onboard SPI Flash memory (common on smaller boards without SD slots).

Parameter Breakdown

- `BACKEND_TYPE`: Bitmask of active logging destinations.

Integration Guide

- **Standard:** Set to 1 (File) for most Pixhawk-class controllers.
- **No SD Card:** Set to 4 (Block) if your board uses internal flash.

BACKEND_TYPE (Logging Backend Type)

`DEFAULT: 1`

Logging Backend Type (BACKEND_TYPE)

Description

Selects the storage or transmission medium for dataflash logs.

Tuning & Behavior

- **Default Value:** 1 (File)
- **Bitmask:**
  - Bit 0: File (SD Card)
  - Bit 1: MAVLink (Streaming to GCS)
  - Bit 2: Block (Internal Flash)

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **BAL** parameter group contains specific tuning parameters for **Balance Bots** (two-wheeled self-balancing robots) running **ArduRover**.

Unlike standard Rovers, a Balance Bot uses a pitch controller to stay upright. This controller drives the wheels forward/backward to correct pitch errors.

Key Concepts

1. Pitch Control

The robot must constantly adjust its wheel speed to maintain a target pitch angle (usually 0, or leaned forward for acceleration).

- `BAL_PITCH_MAX`: Limits how far the robot is allowed to lean. If it leans past this, it may fall over or be unable to recover.
- `BAL_PIT_FF`: Pitch FeedForward. This is critical. It provides the immediate throttle response needed to counter gravity when the robot starts falling.

Parameter Breakdown

- `BAL_PITCH_MAX`: Maximum pitch angle (deg).
- `BAL_PITCH_TRIM`: Pitch angle offset for "Level" (balance point).
- `BAL_PIT_FF`: Feed-forward gain.

Integration Guide

Tuning a Balance Bot

1. **Level:** Calibrate the accelerometer with the robot perfectly balanced.
2. **Trim:** Use `BAL_PITCH_TRIM` to find the exact balance point where the motors are quiet.
3. **PID:** Tune the `ATC_` or `PSC_` pitch controllers (depending on firmware version) alongside `BAL_PIT_FF`.

BAL_CELL1_PIN (Battery balance cell 1 pin)

DEFAULT: 1

Battery balance cell 1 pin (BAL_CELL1_PIN)

Description

Specifies the hardware pin used to measure the voltage of the first cell in a multi-cell battery configuration for smart battery management.

BAL_LIM_TC (Balance Bot Pitch Limit Time Constant)

DEFAULT: 0.5	RANGE: 0.1 2.0	UNITS: s

Balance Bot Pitch Limit Time Constant (BAL_LIM_TC)

Description

Smoothing factor for the pitch limiting logic in Rover Balance Bots.

BAL_LIM_THR (Balance Bot Pitch Limit Throttle Threshold)

DEFAULT: 0.7	RANGE: 0.1 1.0

Balance Bot Pitch Limit Throttle Threshold (BAL_LIM_THR)

Description

Defines the throttle level at which the balance bot begins to prioritize maintaining its upright balance over following the speed command.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





BAL_PITCH_MAX (Balance Bot Maximum Pitch Angle)

DEFAULT: 10	**RANGE:** 1 45	**UNITS:** deg

Balance Bot Maximum Pitch Angle (BAL_PITCH_MAX)

Description

Safety limit for the maximum lean angle of a two-wheeled balance bot.

BAL_PITCH_TRIM (Balance Bot Pitch Trim)

DEFAULT: 0	**RANGE:** -5 5	**UNITS:** deg

Balance Bot Pitch Trim (BAL_PITCH_TRIM)

Description

Fine-tuning adjustment to the "center of gravity" for a balance bot, used to ensure it stays perfectly stationary when no movement is commanded.

BAL_PIT_FF (Balance Bot Pitch to Throttle FeedForward)

DEFAULT: 0.5	**RANGE:** 0 1

Balance Bot Pitch to Throttle FeedForward (BAL_PIT_FF)

Description

Feed-forward gain that couples pitch error directly to motor throttle in balance bot configurations, improving stability.

BAL_RATE (Balance Bot Controlled Rate (Invalid/Legacy))

DEFAULT: 0	**RANGE:** 1 360	**UNITS:** deg/s

Balance Bot Controlled Rate (BAL_RATE)

Description

This parameter is not found as BAL_RATE in the main codebase.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **BARO** parameter group configures the **Barometric Pressure Sensors** (Barometers). These sensors measure atmospheric pressure to estimate the vehicle's **Altitude** relative to the launch point (AGL).

Barometers are the primary altitude source for most flight modes (AltHold, Loiter, Auto). They are sensitive to:

1. **Light:** Sunlight striking the sensor element can cause jumps (cover with open-cell foam).
2. **Wind/Prop Wash:** Airflow over the sensor creates pressure variations (Bernoulli effect), leading to altitude errors.
3. **Temperature:** Drift occurs as the board heats up.

Key Concepts

1. Primary Sensor (`BARO_PRIMARY`)

ArduPilot supports up to 3 redundant barometers. This parameter selects which one is trusted by the EKF (State Estimator) by default.

- **0:** First detected (Internal).
- **1:** Second detected.
- **2:** Third detected.

2. Ground Reference (`BARO_GND_TEMP`)

The barometer calculates altitude based on the temperature and pressure measured on the ground during arming.

- `BARO_FIELD_ELV`: Allows manually setting the takeoff elevation (AMSL) if known, to improve absolute altitude reporting.

3. Wind Compensation (`BARO1_WCF_...`)

These parameters provide **Wind Coefficient corrections**. If your drone's altitude reading changes significantly when flying forward vs. hovering (due to dynamic pressure build-up inside the fuselage), you can tune these coefficients to subtract the speed-induced pressure error.

Parameter Breakdown

- `BARO_ENABLE`: Master switch.
- `BARO_PRIMARY`: Active sensor selector.
- `BARO_ALT_OFFSET`: Calibration offset (tuning).
- `BARO_ALTERR_MAX`: Maximum allowed difference between GPS and Baro altitude before pre-arm failure.
- `BARO_PROBE_EXT`: Probe for external I2C barometers (BMP280, DPS310, etc.).

Integration Guide

Hardware Setup

- **Foam:** ALWAYS cover the barometer chip on the flight controller with open-cell foam and a breathable case to shield it from light and direct airflow.
- **External:** If the internal baro is noisy or subject to heat, use an external DroneCAN or I2C barometer mounted in a better location.

Troubleshooting "Baro Drift"

- If altitude rises while the drone is sitting on the bench: **Thermal Drift**. Wait for the board to warm up before flying.
- If altitude drops when you throttle up: **Prop Wash**. Improve isolation/foam.
- If altitude changes when flying fast: **Dynamic Pressure**. Tune `WCF` params or move the static port.

Developer Notes

- **Library:** `libraries/AP_Baro`
- **Singleton:** `AP_Baro::get_singleton()`
- **Drivers:** Supports MS5611, BMP280, DPS310, ICP10100, and many others.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





BARO1_DEVID (Barometer 1 Device ID)

DEFAULT: 0

Barometer 1 Device ID (BARO1_DEVID)

Description

`BARO1_DEVID` is a **Read-Only** parameter that identifies the specific barometer hardware detected by the autopilot as the primary sensor. It includes information about the sensor type (e.g., BMP280, MS5611), the communication bus (I2C/SPI), and the bus address.

- **Role:** Diagnostic tool to verify that the expected barometer is being used.
- **Usage:** If this is 0, the autopilot has failed to detect any barometer on the expected pins/buses.

The Engineer's View

The ID is encoded using the standard ArduPilot `DeviceID` structure.
It is populated during the `AP_Baro::init()` sequence when the driver successfully probes the hardware.

Tuning & Behavior

- **Default:** 0 (Detection failed or not yet performed).
- **Action:** If your drone has multiple barometers, you can use this ID to identify which one is which in the logs.

BARO1_GND_PRESS (Ground Pressure (Sensor 1))

DEFAULT: 0	RANGE: 0 110000	UNITS: Pa

Ground Pressure (BARO1_GND_PRESS)

Description

`BARO1_GND_PRESS` stores the atmospheric pressure measured when the drone was last "Zeroed" (usually at arming or during the initial power-up).

The autopilot uses this as the reference (Sea Level equivalent) to calculate relative altitude.

- **Units:** Pascals (Pa).
- **Standard Pressure:** Approximately 101325 Pa.

The Engineer's View

Stored in `AP_Baro::sensors[0].ground_pressure`.
It is updated during the ground calibration phase. The EKF uses this value to initialize the altitude state.

Tuning & Behavior

- **Default:** 0 (Updated automatically).
- **Maintenance:** You should not need to edit this manually. If your drone reports a large altitude offset immediately after arming, ensure the barometer is not covered by foam or exposed to direct sunlight/wind.

BARO1_WCF_BCK (Baro Wind Compensation Backward (Sensor 1))

DEFAULT: 0	RANGE: -1.0 1.0

Baro Wind Compensation Backward (BARO1_WCF_BCK)

Description

`BARO1_WCF_BCK` compensates for pressure errors during backward flight. See BARO1_WCF_FWD for more details.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





BARO1_WCF_DN (Baro Wind Compensation Down (Sensor 1))

DEFAULT: 0	RANGE: -1.0 1.0

Baro Wind Compensation Down (BARO1_WCF_DN)

Description

BARO1_WCF_DN compensates for pressure errors during downward (vertical) flight. See BARO1_WCF_FWD for more details.

BARO1_WCF_FWD (Baro Wind Compensation Forward (Sensor 1))

DEFAULT: 0	RANGE: -1.0 1.0

Baro Wind Compensation Forward (BARO1_WCF_FWD)

Description

BARO1_WCF_FWD compensates for the "Pressure Hull" effect.

When a drone flies forward at high speed, air can pile up or create a vacuum around the barometer sensor (depending on the fuselage shape and venting). This creates a false pressure reading, making the drone think it is climbing or diving when it is actually level. This coefficient allows the EKF to use airspeed/acceleration data to cancel out this error.

- **Positive Value:** Corrects for high pressure (compression).
- **Negative Value:** Corrects for low pressure (suction).

Tuning & Behavior

- **Default:** 0.
- **Tuning:** Fly at high speed in a straight line. If the drone loses altitude as it speeds up (thinking it is higher than it is), adjust this coefficient until the altitude remains steady.
- **Dependencies:** Requires a functional EKF3 and correctly configured drag coefficients.

BARO1_WCF_LFT (Baro Wind Compensation Left (Sensor 1))

DEFAULT: 0	RANGE: -1.0 1.0

Baro Wind Compensation Left (BARO1_WCF_LFT)

Description

BARO1_WCF_LFT compensates for pressure errors during leftward (sideways) flight. See BARO1_WCF_FWD for more details.

BARO1_WCF_RGT (Baro Wind Compensation Right (Sensor 1))

DEFAULT: 0	RANGE: -1.0 1.0

Baro Wind Compensation Right (BARO1_WCF_RGT)

Description

BARO1_WCF_RGT compensates for pressure errors during rightward (sideways) flight. See BARO1_WCF_FWD for more details.

BARO1_WCF_UP (Baro Wind Compensation Up (Sensor 1))

DEFAULT: 0	RANGE: -1.0 1.0

Baro Wind Compensation Up (BARO1_WCF_UP)

Description

BARO1_WCF_UP compensates for pressure errors during upward (vertical) flight. See BARO1_WCF_FWD for more details.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





BARO2_DEVID (Baro ID2)

DEFAULT: 0	RANGE: null

Baro ID2

Note: This parameter functions identically to BARO1_DEVID.

BARO2_GND_PRESS (Ground Pressure)

DEFAULT: 0	RANGE: null	UNITS: Pa

Ground Pressure

Note: This parameter functions identically to BARO1_GND_PRESS.

BARO3_DEVID (Baro ID3)

DEFAULT: 0	RANGE: null

Baro ID3

Note: This parameter functions identically to BARO1_DEVID.

BARO3_GND_PRESS (Absolute Pressure)

DEFAULT: 0	RANGE: null	UNITS: Pa

Absolute Pressure

Note: This parameter functions identically to BARO1_GND_PRESS.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





BARO_ALTERR_MAX (Altitude error maximum)

DEFAULT: 2000	RANGE: 0 5000	UNITS: m

Altitude error maximum (BARO_ALTERR_MAX)

Description

This parameter is a pre-arm safety check that compares the altitude reported by the GPS to the altitude calculated from the barometer. It is designed to catch "Hardware Substitutions" or faulty sensors before you take off.

A common issue in the hardware market is vendors substituting one barometer model (like the MS5607) for another (like the MS5611) without updating the firmware settings. This causes the altitude to be calculated incorrectly (often off by thousands of meters). This check prevents the vehicle from arming if such a massive discrepancy is detected.

The Mathematics

The autopilot compares the GPS altitude (Alt_AMSL) to the barometric pressure altitude calculated against the Standard Sea Level (SSL) atmosphere.

$$Error = |Alt_GPS - Alt_Baro_SSL|$$

If $Error > BARO_ALTERR_MAX$, the arming check fails.

The Engineer's View

In `AP_Baro::arming_checks()` (`libraries/AP_Baro/AP_Baro.cpp`):

1. The code verifies a valid 3D GPS fix is available.
2. It calculates `alt_pressure = get_altitude_difference(SSL_AIR_PRESSURE, get_pressure())`.
3. It compares this to `gps.location().alt`.
4. If the gap is too large, it returns the error message: `"GPS alt error [X]m (see BARO_ALTERR_MAX)"`.

Tuning & Behavior

- **Default Value:** 2000 meters.
- **Usage:** You should not typically need to tune this. If you are getting this error, it is a strong signal that your barometer is either faulty or misconfigured.
- **0:** Disables this safety check.
- **Hardware Fix:** If you are certain your hardware is an MS5607 being treated as an MS5611, refer to the `BARO_OPTIONS` parameter to force the correct driver behavior.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





BARO_ALT_OFFSET (Altitude offset)

DEFAULT: 0	RANGE: null	UNITS: m

Altitude offset (BARO_ALT_OFFSET)

Description

This parameter allows for a manual bias (offset) to be added to the calculated barometric altitude. It is primarily used by Ground Control Stations (GCS) to synchronize the aircraft's altitude with a known ground reference point or a GCS-integrated barometer.

It essentially "shifts" the entire altitude profile up or down by the specified number of meters.

The Mathematics

The offset is added directly to the final altitude calculation for every sensor instance.

$$Alt_reported = Alt_raw + BARO_ALT_OFFSET$$

To prevent "jerking" the flight controllers (which could upset the EKF), ArduPilot slews this value slowly over time rather than applying it as a step function.

The Engineer's View

In `AP_Baro::update()`, the code checks if `_alt_offset` has changed:
`_alt_offset_active = (0.98f*_alt_offset_active) + (0.02f*_alt_offset);`
This implementation acts as a low-pass filter (98% old, 2% new) to smoothly transition to a new offset.
The value is reset to **0** every time a full barometer calibration is performed (e.g., on reboot).

Tuning & Behavior

- **Default Value:** 0 meters.
- **Usage:** Typically modified via MAVLink by a GCS rather than tuned manually by a user.
- **Persistence:** This value is saved to EEPROM but is automatically reset to 0 during the `AP_Baro::calibrate()` call on boot.

BARO_ENABLE (Barometer Enable)

DEFAULT: 1	RANGE: 0 1	UNITS: Option

Barometer Enable (BARO_ENABLE)

Description

This parameter is the master switch for the barometer driver. It is primarily used on CAN-based peripheral nodes (`AP_Periph`) to enable or disable the onboard pressure sensor.

If your peripheral node has a physical barometer installed, setting this to **1** allows the node to start measuring pressure and reporting it to the main flight controller over the CAN bus.

The Mathematics

This is a binary switch:

- **0:** Disabled (Driver not initialized)
- **1:** Enabled (Driver initialized and reporting)

The Engineer's View

In `AP_Periph` code, this parameter maps to `g.baro_enable`. It is used during the boot sequence to decide whether to call `baro.init()`. If disabled, no memory is allocated for barometer drivers, and no DroneCAN barometer messages are transmitted.

Tuning & Behavior

- **Default Value:** 1 (Enabled).
- **Usage:** Only set to 0 if your hardware lacks a barometer or if you are using the node for a different purpose (like a dedicated GPS) and want to reduce CAN bus traffic.
- **Reboot Required:** Yes. Driver initialization happens only at startup.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





BARO_EXP (Baro Temperature Correction Exponent)

DEFAULT: 0	RANGE: null	UNITS: Exponent

Baro Temperature Correction Exponent (BARO_EXP)

Description

This parameter is part of ArduPilot's advanced temperature calibration system. Many barometric sensors exhibit a "drift" where the reported pressure changes simply because the sensor is heating up, even if the altitude remains constant. This is a common issue with sensors like the ICM-20789.

`BARO_EXP` stores a learned mathematical exponent that models this drift. When temperature calibration is enabled, the autopilot uses this value to "cancel out" the temperature-induced pressure changes, resulting in a much more stable altitude estimate during long flights or in varying weather conditions.

The Mathematics

The correction applied to the raw pressure (P_raw) is a power function based on the temperature difference from a reference point (T_zero, typically 20°C):

$$P_corrected = P_raw + \max(T - T_zero, 0)^{BARO_EXP}$$

This model is chosen for its robustness and its ability to accurately fit the physical characteristics of high-drift barometers.

The Engineer's View

In `AP_TempCalibration.cpp`, this parameter is defined as `_BARO_EXP` (internal name `baro_exponent`).

- **Learning Phase:** When `TCAL_ENABLED` is set to **2 (EnableAndLearn)**, the system monitors pressure vs. temperature while the vehicle is sitting still on the ground. It uses a least-squares error minimization approach to find the exponent that best flattens the pressure curve.
- **Application:** Once learned, the exponent is saved to this parameter. In subsequent flights (with `TCAL_ENABLED=1`), `AP_TempCalibration::apply_calibration()` uses this exponent to provide a real-time `p_correction` to the `AP_Baro` library.

Tuning & Behavior

- **Default Value:** 0 (No correction).
- **How to Set:** Do not set this manually. Instead, set `TCAL_ENABLED=2`, power on your vehicle, and let it sit still while the internal electronics warm up (about 10-15 minutes). The system will automatically calculate and save the optimal `BARO_EXP`.
- **Range:** Typically between 0.0 and 2.0.
- **Effect:** A successful calibration will result in the "Baro Altitude" remaining near 0.0m even as the flight controller gets hot.

BARO_EXT_BUS (External Barometer I2C Bus)

DEFAULT: -1	RANGE: -1 6

External Barometer I2C Bus (BARO_EXT_BUS)

Description

`BARO_EXT_BUS` is used if you are connecting an external high-precision barometer to your flight controller via an I2C cable.

- **-1 (Disabled/Auto):** The flight controller will probe all external buses automatically.
- **0-6:** Forces the flight controller to look on a specific I2C bus for the sensor.

The Engineer's View

Used in `AP_Baro::init()`.
Manually setting the bus can speed up the boot process and prevent conflicts if multiple devices on different buses have the same I2C address.

Tuning & Behavior

- **Default Value:** -1
- **Recommendation:** Leave at **-1** unless you have a specific hardware conflict or are using an unusual board configuration.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





BARO_FIELD_ELV (Field elevation)

DEFAULT: 0	UNITS: m

Field elevation (BARO_FIELD_ELV)

Description

This parameter allows the user to manually enter the known elevation of the takeoff location (field elevation) in meters above mean sea level (AMSL). This provides a known reference altitude for the EKF and barometer before a GPS fix is obtained, ensuring more accurate absolute altitude reporting from startup.

The Mathematics

$$ \text{Altitude}*{\text{AMSL}} \approx \text{Altitude}*{\text{Baro}} + \text{BARO_FIELD_ELV} $$

If `BARO_FIELD_ELV` is 0, the system attempts to initialize the origin altitude from GPS or other sources once available.

The Engineer's View

Defined in `libraries/AP_Baro/AP_Baro.cpp`. The parameter `_field_elevation` is used to set `_field_elevation_active`. It is non-persistent across reboots in typical usage (resets to 0) to prevent taking off from a different location with an incorrect fixed elevation, though the parameter definition itself allows storage. The code logic often resets it or prioritizes GPS data once valid.

Tuning & Behavior

- **Default Value:** 0 m
- **Set to:** Known altitude of the launch site (e.g., 100m).
- **Note:** This is primarily for advanced users or operations in GPS-denied environments where absolute altitude reference is required from power-on.

BARO_FLTR_RNG (Filter Range)

DEFAULT: 0	RANGE: 0 100	UNITS: %

Filter Range (BARO_FLTR_RNG)

Description

This parameter acts as a "Noise Gate" for barometric pressure samples. It defines a percentage-based threshold around the current moving average. Any new pressure reading that falls outside of this window is rejected as "glitchy" noise and ignored.

This is extremely useful for protecting the altitude estimate from electromagnetic interference (EMI) or voltage spikes on the I2C bus, which can cause sudden, unrealistic jumps in pressure readings.

The Mathematics

A sample (P_new) is only accepted if it is within $\pm X\%$ of the current smoothed average (P_avg):

$$|P_new - P_avg| \leq P_avg \times \frac{BARO_FLTR_RNG}{100}$$

If the new sample is outside this range, the system discards it and waits for the next sample.

The Engineer's View

In `AP_Baro.cpp`, this parameter maps to `_filter_range`.
It is primarily used in sensor backends (like `AP_Baro_MS56XX.cpp` or `AP_Baro_BMP280.cpp`) to validate raw pressure data before it enters the altitude conversion logic.

- **0 (Default):** Disables the check. All samples are accepted.
- **Setting it too tight:** Can cause the barometer to "freeze" during rapid climbs or descents because the actual pressure change is faster than the allowed window.

Tuning & Behavior

- **Default Value:** 0 (Filter disabled).
- **Recommended Value:** If you experience altitude "glitches" or sudden jumps, try a value of **10** (10% window).
- **Symptom of too Low:** Barometric altitude stops updating during high-performance maneuvers.
- **Symptom of too High:** The sensor remains sensitive to electrical noise spikes.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





BARO_GND_TEMP (Ground Temperature)

DEFAULT: 0	RANGE: -50 60	UNITS: degC

Ground Temperature (BARO_GND_TEMP)

Description

`BARO_GND_TEMP` allows you to manually specify the temperature at the takeoff location.

ArduPilot uses an "International Standard Atmosphere" (ISA) model to translate pressure changes into altitude. However, air density changes with temperature. If it is very hot or very cold, the standard model will have a significant error (the drone might think it is at 100m when it's really at 110m). Providing the actual ground temperature improves this calculation.

- **0 (Default):** The autopilot uses the temperature measured by its own internal sensor.
- **Non-Zero:** Overrides the internal sensor with your manual reading.

Tuning & Behavior

- **Default:** 0.
- **Usage:** For most pilots, leave at **0**. Only change this if you are performing precision survey work or high-altitude balloon flights where temperature gradients are critical.
- **Note:** This parameter is reset to 0 upon reboot.

BARO_OPTIONS (Barometer options)

DEFAULT: 0	RANGE: null	UNITS: Bitmask

Barometer options (BARO_OPTIONS)

Description

This parameter is a bitmask used to enable specialized workarounds or features for barometric sensors.

Its primary current use is to solve a common "Hardware Identity Crisis": some manufacturers sell flight controllers labeled as containing an MS5611 barometer, but actually install an MS5607 (which is cheaper and looks identical). If the software treats an MS5607 as an MS5611, your altitude will be wildly incorrect. Enabling the workaround bit here forces the software to use the correct math for the substituted part.

The Mathematics

This parameter is a bitmask (B):

- **Bit 0 (Value 1):** Treat MS5611 as MS5607.

If Bit 0 is set, the `AP_Baro_MS56XX` driver swaps the internal PROM coefficients and scaling constants (C_1 to C_6) to match the MS5607 physics model, ensuring correct pressure-to-altitude conversion.

The Engineer's View

In `AP_Baro.cpp`, this is defined as `_options`.
In `AP_Baro_MS56XX::probe()`, the driver checks the global `AP_Baro` options. If the "Treat as MS5607" bit is set, it overrides the auto-detected sensor type. This is necessary because the MS5611 and MS5607 have the same I2C/SPI identifiers and cannot always be distinguished purely by electronic signatures.

Tuning & Behavior

- **Bit 0 (Value 1):** Enable the MS5611 → MS5607 workaround.
- **Usage:** Only enable this if you are getting a `BARO_ALTERR_MAX` (GPS altitude discrepancy) error on a new flight controller and you suspect the manufacturer substituted the barometer.
- **Reboot Required:** Yes. Barometer driver initialization happens on boot.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





BARO_PRIMARY (Primary barometer)

DEFAULT: 0	RANGE: 0 2	UNITS: Option

Primary barometer (BARO_PRIMARY)

Description

This parameter selects the "Master" barometer instance to be used for altitude estimation. Many modern flight controllers have multiple barometers (one internal and one or more external via CAN or I2C) for redundancy.

The primary barometer is the one whose data is prioritized by the EKF (Extended Kalman Filter) and reported to the pilot as the official barometric altitude.

The Mathematics

This is a simple index selection (i):

- **0:** 1st Baro (Instance 0)
- **1:** 2nd Baro (Instance 1)
- **2:** 3rd Baro (Instance 2)

The Engineer's View

In `AP_Baro::update()`, the logic selects the index stored in `_primary_baro`.
If the selected barometer becomes unhealthy during flight, ArduPilot has a failover mechanism that will automatically switch to the first healthy sensor it finds. However, `BARO_PRIMARY` defines the starting preference.

Tuning & Behavior

- **Default Value:** 0.
- **Recommendation:** If you have an external, high-precision barometer (e.g., inside a GPS module mounted away from wind/prop wash), set this to that instance number (often 1).
- **Detection:** You can check which barometer is which by looking at the `HWID` or `ID` logs, or by blowing on the sensors and watching the `BARO[x]` altitude response in your Ground Control Station's status tab.

BARO_PROBE_EXT (External Barometers to Probe)

DEFAULT: 0	RANGE: 0 16383

External Barometers to Probe (BARO_PROBE_EXT)

Description

`BARO_PROBE_EXT` tells ArduPilot which specific barometer hardware to look for on the external I2C bus.

By default, the autopilot attempts to detect the most common sensors automatically. However, if you are using an unusual or newer sensor that isn't being found, you may need to enable its bit in this parameter.

- **Bit 0:** BMP085
- **Bit 1:** BMP280
- **Bit 2:** MS5611
- **Bit 3:** MS5607
- **Bit 10:** BMP388
- **Bit 11:** SPL06

Tuning & Behavior

- **Default:** 0 (Standard auto-detection).
- **Usage:** Only modify this if your external barometer is not being detected (Instance 2 or 3 is missing in the GCS).
- **Reboot Required:** Yes.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





BARO_SPEC_GRAV (Fluid Specific Gravity)

DEFAULT: 1.0	RANGE: 0.5 2.0

Fluid Specific Gravity (BARO_SPEC_GRAV)

Description

`BARO_SPEC_GRAV` is used by **ArduSub** (underwater ROVs) to calculate depth.

When a pressure sensor is used underwater, the autopilot translates the pressure into depth based on the density of the water. Fresh water and salt water have different densities, and this parameter allows you to calibrate the depth reading for your specific operating environment.

- **1.00:** Fresh Water.
- **1.025:** Standard Salt Water (Sea water).

The Mathematics

$$\text{Depth} = \frac{\text{Pressure Difference}}{\rho \times g}$$

where ρ is the water density derived from `BARO_SPEC_GRAV`.

Tuning & Behavior

- **Default:** 1.0.
- **Recommendation:** If you are diving in the ocean, change this to **1.025** for the most accurate depth telemetry.
- **Note:** This parameter is generally ignored for aerial vehicles.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

Key Concepts

BAROALT_OFFSET (Altitude offset)

DEFAULT: 0	UNITS: m

Altitude offset (BAROALT_OFFSET)

Description

A manual or GCS-provided offset added to the calculated barometric altitude. It is typically used to align the vehicle's altitude with a known ground reference.

Tuning & Behavior

- **Default Value:** 0 m
- **It is automatically reset to 0 when the barometer is calibrated on reboot.**

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

Key Concepts

BAROALTERR_MAX (Altitude error maximum)

DEFAULT: 2000	RANGE: 0 5000	UNITS: m

Altitude error maximum (BAROALTERR_MAX)

Description

Safety check during arming that compares barometric altitude with GPS altitude.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

Key Concepts

BAROEXT_BUS (External baro bus)

`DEFAULT: -1`

External baro bus (BAROEXT_BUS)

Description

Specifies a particular I2C bus to search for an external barometer.

Tuning & Behavior

- **Default Value:** -1 (Probe all buses)
- **Values:** -1:Disabled/Auto, 0:Bus0, 1:Bus1, 6:Bus6

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

Key Concepts

BAROFIELD_ELV (field elevation)

DEFAULT: 0	UNITS: m

field elevation (BAROFIELD_ELV)

Description

Manual input for the airfield's elevation above sea level. This improves the absolute altitude accuracy by providing a known ground reference.

Tuning & Behavior

- **Default Value:** 0 m
- **This parameter is not persistent and will be reset to 0 every time the vehicle is rebooted.**

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

Key Concepts

BAROFLTR_RNG (Range in which sample is accepted)

DEFAULT: 0	RANGE: 0 100	UNITS: %

Range in which sample is accepted (BAROFLTR_RNG)

Description

A noise filter that rejects barometer samples that deviate too far from the rolling average.

Tuning & Behavior

- **Default Value:** 0 (Disabled)
- **Range:** 0 to 100 %
- **Useful for mitigating spikes in altitude data caused by electrical interference or long cable runs.**

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

Key Concepts

BAROGND_TEMP (ground temperature)

DEFAULT: 0	UNITS: degC

ground temperature (BAROGND_TEMP)

Description

Manual input for the ambient temperature at the ground level. ArduPilot uses this to improve the barometric altitude calculation, which is temperature-dependent.

Tuning & Behavior

- **Default Value:** 0 degC (Use internal sensor measurement)
- **This parameter is not persistent and will be reset to 0 every time the vehicle is rebooted.**

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

Key Concepts

BAROPROBE_EXT (External barometers to probe)

DEFAULT: 0

External barometers to probe (BAROPROBE_EXT)

Description

Bitmask selecting which specific barometer hardware types ArduPilot should search for on the I2C bus.

Tuning & Behavior

- **Default Value:** 0
- **Bitmask Examples:**
  - Bit 0: BMP085
  - Bit 1: BMP280
  - Bit 2: MS5611
  - Bit 10: BMP388
  - Bit 11: SPL06

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

Key Concepts

BAROSPEC_GRAV (Specific Gravity)

DEFAULT: 1.0

Specific Gravity (BAROSPEC_GRAV)

Description

Defines the density of the surrounding fluid for ArduSub vehicles. This is used to convert pressure readings into accurate depth measurements.

Tuning & Behavior

- **Default Value:** 1.0 (Freshwater)
- **Common Values:** 1.0:Freshwater, 1.024:Saltwater

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





BATTERY MONITOR CONFIGURATION (BATT)

Overview

The **BATT** parameter group configures the power monitoring subsystem. This is essential for:

1. **Safety:** Triggering a failsafe (RTL/Land) before the battery is exhausted.
2. **Telemetry:** Reporting Voltage (V), Current (A), and Consumed Energy (mAh) to the pilot.
3. **Estimation:** The autopilot uses voltage levels to scale motor outputs (voltage compensation) for consistent performance as the battery drains.

Key Concepts

1. Monitor Type (`BATT_MONITOR`)

Selects the driver/hardware interface.

- **3 (Analog Voltage Only):** Simple voltage divider.
- **4 (Analog Voltage & Current):** Standard power module (Mauch, Holybro, generic).
- **7 (SMBus-I2C):** Smart batteries (Maxell, Solo, some generic I2C BMS).
- **8 (DroneCAN):** Digital power modules (Zubax, Mauch CAN).

2. Calibration (Analog)

For standard analog sensors (`Type = 4`), you must tune the scaling factors.

- `BATT_VOLT_MULT`: Scales the 0-3.3V ADC reading to actual battery voltage.
- `BATT_AMP_PERVLT`: Defines how many Amps correspond to 1V on the current pin.

3. Failsafes (`BATT_FS_...`)

ArduPilot supports "Low" and "Critical" battery failsafes.

- **Triggers:** Low Voltage (`BATT_LOW_VOLT`), Low Capacity (`BATT_LOW_MAH`).
- **Actions:** Report Only, Land, RTL, SmartRTL, Terminate.

4. Multiple Batteries (`BATT2_`, `BATT3_` ...)

ArduPilot supports up to 16 battery monitors. The data can be:

- **Summed:** Total voltage/current reported to GCS.
- **Individual:** Reported separately.
- **Redundant:** Automatic switchover if one fails (rare).

Parameter Breakdown

- `BATT_MONITOR`: Driver selector.
- `BATT_CAPACITY`: Battery pack capacity in mAh (used for % remaining calculation).
- `BATT_VOLT_PIN` / `CURR_PIN`: Hardware ADC pin assignment.
- `BATT_ARM_VOLT`: Minimum voltage required to arm (prevents taking off with a dead battery).

Integration Guide

Calibrating a Standard Power Module

1. **Connect:** Plug in the battery and connect via USB.
2. **Measure:** Use a multimeter to read the actual voltage at the XT60 connector.
3. **Adjust:** Modify `BATT_VOLT_MULT` until the GCS voltage matches your multimeter.
4. **Current:** Apply a known load (motors spinning) and adjust `BATT_AMP_PERVLT` to match a wattmeter.

Smart Battery Setup

1. Set `BATT_MONITOR = 7` (SMBus).
2. Set `BATT_I2C_BUS` to the correct bus (usually 0 or 1).
3. Reboot.

Developer Notes

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





- **Library:** `libraries/AP_BattMonitor`
- **Singleton:** `AP_BattMonitor::get_singleton()`
- **Scaling:** Voltage compensation logic is handled in `AP_Motors`.

BATT2_AMP_OFFSET (AMP offset)

DEFAULT: AP_BATT_CURR_AMP_OFFSET_DEFAULT	RANGE: null	UNITS: V

AMP offset

Note: This parameter functions identically to BATT_AMP_OFFSET.

BATT2_AMP_PERVLT (Amps per volt)

DEFAULT: AP_BATT_CURR_AMP_PERVOLT_DEFAULT	RANGE: null	UNITS: A/V

Amps per volt

Note: This parameter functions identically to BATT_AMP_PERVLT.

BATT2_ARM_MAH (Required arming remaining capacity)

DEFAULT: 0	RANGE: null	UNITS: mAh

Required arming remaining capacity

Note: This parameter functions identically to BATT_ARM_MAH.

BATT2_ARM_VOLT (Required arming voltage)

DEFAULT: 0	RANGE: null	UNITS: V

Required arming voltage

Note: This parameter functions identically to BATT_ARM_VOLT.

BATT2_CAPACITY (Battery capacity)

DEFAULT: AP_BATT_MONITOR_BATTERY_CAPACITY	RANGE: null	UNITS: mAh

Battery capacity

Note: This parameter functions identically to BATT_CAPACITY.

BATT2_CRT_MAH (Battery critical capacity)

DEFAULT: 0	RANGE: null	UNITS: mAh

Battery critical capacity

Note: This parameter functions identically to BATT_CRT_MAH.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





BATT2_CRT_VOLT (Critical battery voltage)

DEFAULT: 0	RANGE: null	UNITS: V

Critical battery voltage

Note: This parameter functions identically to BATT_CRT_VOLT.

BATT2_CURR_MULT (Scales reported power monitor current)

DEFAULT: 1.0	RANGE: .1 10

Scales reported power monitor current

Note: This parameter functions identically to BATT_CURR_MULT.

BATT2_CURR_PIN (Battery Current sensing pin)

DEFAULT: AP_BATT_CURR_PIN	RANGE: null

Battery Current sensing pin

Note: This parameter functions identically to BATT_CURR_PIN.

BATT2_ESC_INDEX (ESC Telemetry Index to write to)

DEFAULT: 0	RANGE: 0 10

ESC Telemetry Index to write to

Note: This parameter functions identically to BATT_ESC_INDEX.

BATT2_FS_CRT_ACT (Critical battery failsafe action)

DEFAULT: 0	RANGE: null

Critical battery failsafe action

Note: This parameter functions identically to BATT_FS_CRT_ACT.

BATT2_FS_LOW_ACT (Low battery failsafe action)

DEFAULT: 0	RANGE: null

Low battery failsafe action

Note: This parameter functions identically to BATT_FS_LOW_ACT.

BATT2_FS_VOLTSRC (Failsafe voltage source)

DEFAULT: BattMonitor_LowVoltageSource_Raw	RANGE: null

Failsafe voltage source

Note: This parameter functions identically to BATT_FS_VOLTSRC.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





BATT2_I2C_ADDR (Battery monitor I2C address)

DEFAULT: AP_BATTMONITOR_SMBUS_I2C_ADDR	RANGE: 0 127

Battery monitor I2C address

Note: This parameter functions identically to BATT_I2C_ADDR.

BATT2_I2C_BUS (Battery monitor I2C bus number)

DEFAULT: 0	RANGE: 0 3

Battery monitor I2C bus number

Note: This parameter functions identically to BATT_I2C_BUS.

BATT2_LOW_MAH (Low battery capacity)

DEFAULT: 0	RANGE: null	UNITS: mAh

Low battery capacity

Note: This parameter functions identically to BATT_LOW_MAH.

BATT2_LOW_TIMER (Low voltage timeout)

DEFAULT: 10	RANGE: 0 120	UNITS: s

Low voltage timeout

Note: This parameter functions identically to BATT_LOW_TIMER.

BATT2_LOW_VOLT (Low battery voltage)

DEFAULT: DEFAULT_LOW_BATTERY_VOLTAGE	RANGE: null	UNITS: V

Low battery voltage

Note: This parameter functions identically to BATT_LOW_VOLT.

BATT2_MONITOR (Battery monitoring)

DEFAULT: int8_t(AP_BattMonitor::Type::NONE	RANGE: null

Battery monitoring

Note: This parameter functions identically to BATT_MONITOR.

BATT2_OPTIONS (Battery monitor options)

DEFAULT: 0	RANGE: null

Battery monitor options

Note: This parameter functions identically to BATT_OPTIONS.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





BATT2_SERIAL_NUM (Battery serial number)

DEFAULT: AP_BATT_SERIAL_NUMBER_DEFAULT	RANGE: null

Battery serial number

Note: This parameter functions identically to BATT_SERIAL_NUM.

BATT2_SUM_MASK (Battery Sum mask)

DEFAULT: 0	RANGE: null

Battery Sum mask

Note: This parameter functions identically to BATT_SUM_MASK.

BATT2_VLT_OFFSET (Voltage offset)

DEFAULT: 0	RANGE: null	UNITS: V

Voltage offset

Note: This parameter functions identically to BATT_VLT_OFFSET.

BATT2_VOLT_MULT (Voltage Multiplier)

DEFAULT: AP_BATT_VOLTDIVIDER_DEFAULT	RANGE: null

Voltage Multiplier

Note: This parameter functions identically to BATT_VOLT_MULT.

BATT2_VOLT_PIN (Battery Voltage sensing pin)

DEFAULT: AP_BATT_VOLT_PIN	RANGE: null

Battery Voltage sensing pin

Note: This parameter functions identically to BATT_VOLT_PIN.

BATT2_WATT_MAX (Maximum allowed power (Watts))

DEFAULT: 0	RANGE: null	UNITS: W

Maximum allowed power (Watts)

Note: This parameter functions identically to BATT_WATT_MAX.

BATT3_AMP_OFFSET (AMP offset)

DEFAULT: AP_BATT_CURR_AMP_OFFSET_DEFAULT	RANGE: null	UNITS: V

AMP offset

Note: This parameter functions identically to BATT_AMP_OFFSET.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





BATT3_AMP_PERVLT (Amps per volt)

DEFAULT: AP_BATT_CURR_AMP_PERVOLT_DEFAULT	RANGE: null	UNITS: A/V

Amps per volt

Note: This parameter functions identically to BATT_AMP_PERVLT.

BATT3_ARM_MAH (Required arming remaining capacity)

DEFAULT: 0	RANGE: null	UNITS: mAh

Required arming remaining capacity

Note: This parameter functions identically to BATT_ARM_MAH.

BATT3_ARM_VOLT (Required arming voltage)

DEFAULT: 0	RANGE: null	UNITS: V

Required arming voltage

Note: This parameter functions identically to BATT_ARM_VOLT.

BATT3_CAPACITY (Battery capacity)

DEFAULT: AP_BATT_MONITOR_BATTERY_CAPACITY	RANGE: null	UNITS: mAh

Battery capacity

Note: This parameter functions identically to BATT_CAPACITY.

BATT3_CRT_MAH (Battery critical capacity)

DEFAULT: 0	RANGE: null	UNITS: mAh

Battery critical capacity

Note: This parameter functions identically to BATT_CRT_MAH.

BATT3_CRT_VOLT (Critical battery voltage)

DEFAULT: 0	RANGE: null	UNITS: V

Critical battery voltage

Note: This parameter functions identically to BATT_CRT_VOLT.

BATT3_CURR_MULT (Scales reported power monitor current)

DEFAULT: 1.0	RANGE: .1 10

Scales reported power monitor current

Note: This parameter functions identically to BATT_CURR_MULT.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





BATT3_CURR_PIN (Battery Current sensing pin)

DEFAULT: AP_BATT_CURR_PIN	RANGE: null

Battery Current sensing pin

Note: This parameter functions identically to BATT_CURR_PIN.

BATT3_ESC_INDEX (ESC Telemetry Index to write to)

DEFAULT: 0	RANGE: 0 10

ESC Telemetry Index to write to

Note: This parameter functions identically to BATT_ESC_INDEX.

BATT3_FS_CRT_ACT (Critical battery failsafe action)

DEFAULT: 0	RANGE: null

Critical battery failsafe action

Note: This parameter functions identically to BATT_FS_CRT_ACT.

BATT3_FS_LOW_ACT (Low battery failsafe action)

DEFAULT: 0	RANGE: null

Low battery failsafe action

Note: This parameter functions identically to BATT_FS_LOW_ACT.

BATT3_FS_VOLTSRC (Failsafe voltage source)

DEFAULT: BattMonitor_LowVoltageSource_Raw	RANGE: null

Failsafe voltage source

Note: This parameter functions identically to BATT_FS_VOLTSRC.

BATT3_I2C_ADDR (Battery monitor I2C address)

DEFAULT: AP_BATTMONITOR_SMBUS_I2C_ADDR	RANGE: 0 127

Battery monitor I2C address

Note: This parameter functions identically to BATT_I2C_ADDR.

BATT3_I2C_BUS (Battery monitor I2C bus number)

DEFAULT: 0	RANGE: 0 3

Battery monitor I2C bus number

Note: This parameter functions identically to BATT_I2C_BUS.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





BATT3_LOW_MAH (Low battery capacity)

DEFAULT: 0	RANGE: null	UNITS: mAh

Low battery capacity

Note: This parameter functions identically to BATT_LOW_MAH.

BATT3_LOW_TIMER (Low voltage timeout)

DEFAULT: 10	RANGE: 0 120	UNITS: s

Low voltage timeout

Note: This parameter functions identically to BATT_LOW_TIMER.

BATT3_LOW_VOLT (Low battery voltage)

DEFAULT: DEFAULT_LOW_BATTERY_VOLTAGE	RANGE: null	UNITS: V

Low battery voltage

Note: This parameter functions identically to BATT_LOW_VOLT.

BATT3_MONITOR (Battery monitoring)

DEFAULT: int8_t(AP_BattMonitor::Type::NONE	RANGE: null

Battery monitoring

Note: This parameter functions identically to BATT_MONITOR.

BATT3_OPTIONS (Battery monitor options)

DEFAULT: 0	RANGE: null

Battery monitor options

Note: This parameter functions identically to BATT_OPTIONS.

BATT3_SERIAL_NUM (Battery serial number)

DEFAULT: AP_BATT_SERIAL_NUMBER_DEFAULT	RANGE: null

Battery serial number

Note: This parameter functions identically to BATT_SERIAL_NUM.

BATT3_SUM_MASK (Battery Sum mask)

DEFAULT: 0	RANGE: null

Battery Sum mask

Note: This parameter functions identically to BATT_SUM_MASK.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





BATT3_VLT_OFFSET (Voltage offset)

DEFAULT: 0	RANGE: null	UNITS: V

Voltage offset

Note: This parameter functions identically to BATT_VLT_OFFSET.

BATT3_VOLT_MULT (Voltage Multiplier)

DEFAULT: AP_BATT_VOLTDIVIDER_DEFAULT	RANGE: null

Voltage Multiplier

Note: This parameter functions identically to BATT_VOLT_MULT.

BATT3_VOLT_PIN (Battery Voltage sensing pin)

DEFAULT: AP_BATT_VOLT_PIN	RANGE: null

Battery Voltage sensing pin

Note: This parameter functions identically to BATT_VOLT_PIN.

BATT3_WATT_MAX (Maximum allowed power (Watts))

DEFAULT: 0	RANGE: null	UNITS: W

Maximum allowed power (Watts)

Note: This parameter functions identically to BATT_WATT_MAX.

BATT4_AMP_OFFSET (AMP offset)

DEFAULT: AP_BATT_CURR_AMP_OFFSET_DEFAULT	RANGE: null	UNITS: V

AMP offset

Note: This parameter functions identically to BATT_AMP_OFFSET.

BATT4_AMP_PERVLT (Amps per volt)

DEFAULT: AP_BATT_CURR_AMP_PERVOLT_DEFAULT	RANGE: null	UNITS: A/V

Amps per volt

Note: This parameter functions identically to BATT_AMP_PERVLT.

BATT4_ARM_MAH (Required arming remaining capacity)

DEFAULT: 0	RANGE: null	UNITS: mAh

Required arming remaining capacity

Note: This parameter functions identically to BATT_ARM_MAH.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





BATT4_ARM_VOLT (Required arming voltage)

DEFAULT: 0	RANGE: null	UNITS: V

Required arming voltage

Note: This parameter functions identically to BATT_ARM_VOLT.

BATT4_CAPACITY (Battery capacity)

DEFAULT: AP_BATT_MONITOR_BATTERY_CAPACITY	RANGE: null	UNITS: mAh

Battery capacity

Note: This parameter functions identically to BATT_CAPACITY.

BATT4_CRT_MAH (Battery critical capacity)

DEFAULT: 0	RANGE: null	UNITS: mAh

Battery critical capacity

Note: This parameter functions identically to BATT_CRT_MAH.

BATT4_CRT_VOLT (Critical battery voltage)

DEFAULT: 0	RANGE: null	UNITS: V

Critical battery voltage

Note: This parameter functions identically to BATT_CRT_VOLT.

BATT4_CURR_MULT (Scales reported power monitor current)

DEFAULT: 1.0	RANGE: .1 10

Scales reported power monitor current

Note: This parameter functions identically to BATT_CURR_MULT.

BATT4_CURR_PIN (Battery Current sensing pin)

DEFAULT: AP_BATT_CURR_PIN	RANGE: null

Battery Current sensing pin

Note: This parameter functions identically to BATT_CURR_PIN.

BATT4_ESC_INDEX (ESC Telemetry Index to write to)

DEFAULT: 0	RANGE: 0 10

ESC Telemetry Index to write to

Note: This parameter functions identically to BATT_ESC_INDEX.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





BATT4_FS_CRT_ACT (Critical battery failsafe action)

DEFAULT: 0	RANGE: null

Critical battery failsafe action

Note: This parameter functions identically to BATT_FS_CRT_ACT.

BATT4_FS_LOW_ACT (Low battery failsafe action)

DEFAULT: 0	RANGE: null

Low battery failsafe action

Note: This parameter functions identically to BATT_FS_LOW_ACT.

BATT4_FS_VOLTSRC (Failsafe voltage source)

DEFAULT: BattMonitor_LowVoltageSource_Raw	RANGE: null

Failsafe voltage source

Note: This parameter functions identically to BATT_FS_VOLTSRC.

BATT4_I2C_ADDR (Battery monitor I2C address)

DEFAULT: AP_BATTMONITOR_SMBUS_I2C_ADDR	RANGE: 0 127

Battery monitor I2C address

Note: This parameter functions identically to BATT_I2C_ADDR.

BATT4_I2C_BUS (Battery monitor I2C bus number)

DEFAULT: 0	RANGE: 0 3

Battery monitor I2C bus number

Note: This parameter functions identically to BATT_I2C_BUS.

BATT4_LOW_MAH (Low battery capacity)

DEFAULT: 0	RANGE: null	UNITS: mAh

Low battery capacity

Note: This parameter functions identically to BATT_LOW_MAH.

BATT4_LOW_TIMER (Low voltage timeout)

DEFAULT: 10	RANGE: 0 120	UNITS: s

Low voltage timeout

Note: This parameter functions identically to BATT_LOW_TIMER.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





BATT4_LOW_VOLT (Low battery voltage)

DEFAULT: DEFAULT_LOW_BATTERY_VOLTAGE	RANGE: null	UNITS: V

Low battery voltage

Note: This parameter functions identically to BATT_LOW_VOLT.

BATT4_MONITOR (Battery monitoring)

DEFAULT: int8_t(AP_BattMonitor::Type::NONE	RANGE: null

Battery monitoring

Note: This parameter functions identically to BATT_MONITOR.

BATT4_OPTIONS (Battery monitor options)

DEFAULT: 0	RANGE: null

Battery monitor options

Note: This parameter functions identically to BATT_OPTIONS.

BATT4_SERIAL_NUM (Battery serial number)

DEFAULT: AP_BATT_SERIAL_NUMBER_DEFAULT	RANGE: null

Battery serial number

Note: This parameter functions identically to BATT_SERIAL_NUM.

BATT4_SUM_MASK (Battery Sum mask)

DEFAULT: 0	RANGE: null

Battery Sum mask

Note: This parameter functions identically to BATT_SUM_MASK.

BATT4_VLT_OFFSET (Voltage offset)

DEFAULT: 0	RANGE: null	UNITS: V

Voltage offset

Note: This parameter functions identically to BATT_VLT_OFFSET.

BATT4_VOLT_MULT (Voltage Multiplier)

DEFAULT: AP_BATT_VOLTDIVIDER_DEFAULT	RANGE: null

Voltage Multiplier

Note: This parameter functions identically to BATT_VOLT_MULT.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





BATT4_VOLT_PIN (Battery Voltage sensing pin)

DEFAULT: AP_BATT_VOLT_PIN	RANGE: null

Battery Voltage sensing pin

Note: This parameter functions identically to BATT_VOLT_PIN.

BATT4_WATT_MAX (Maximum allowed power (Watts))

DEFAULT: 0	RANGE: null	UNITS: W

Maximum allowed power (Watts)

Note: This parameter functions identically to BATT_WATT_MAX.

BATT5_AMP_OFFSET (AMP offset)

DEFAULT: AP_BATT_CURR_AMP_OFFSET_DEFAULT	RANGE: null	UNITS: V

AMP offset

Note: This parameter functions identically to BATT_AMP_OFFSET.

BATT5_AMP_PERVLT (Amps per volt)

DEFAULT: AP_BATT_CURR_AMP_PERVOLT_DEFAULT	RANGE: null	UNITS: A/V

Amps per volt

Note: This parameter functions identically to BATT_AMP_PERVLT.

BATT5_ARM_MAH (Required arming remaining capacity)

DEFAULT: 0	RANGE: null	UNITS: mAh

Required arming remaining capacity

Note: This parameter functions identically to BATT_ARM_MAH.

BATT5_ARM_VOLT (Required arming voltage)

DEFAULT: 0	RANGE: null	UNITS: V

Required arming voltage

Note: This parameter functions identically to BATT_ARM_VOLT.

BATT5_CAPACITY (Battery capacity)

DEFAULT: AP_BATT_MONITOR_BATTERY_CAPACITY	RANGE: null	UNITS: mAh

Battery capacity

Note: This parameter functions identically to BATT_CAPACITY.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





BATT5_CRT_MAH (Battery critical capacity)

DEFAULT: 0	RANGE: null	UNITS: mAh

Battery critical capacity

Note: This parameter functions identically to BATT_CRT_MAH.

BATT5_CRT_VOLT (Critical battery voltage)

DEFAULT: 0	RANGE: null	UNITS: V

Critical battery voltage

Note: This parameter functions identically to BATT_CRT_VOLT.

BATT5_CURR_MULT (Scales reported power monitor current)

DEFAULT: 1.0	RANGE: .1 10

Scales reported power monitor current

Note: This parameter functions identically to BATT_CURR_MULT.

BATT5_CURR_PIN (Battery Current sensing pin)

DEFAULT: AP_BATT_CURR_PIN	RANGE: null

Battery Current sensing pin

Note: This parameter functions identically to BATT_CURR_PIN.

BATT5_ESC_INDEX (ESC Telemetry Index to write to)

DEFAULT: 0	RANGE: 0 10

ESC Telemetry Index to write to

Note: This parameter functions identically to BATT_ESC_INDEX.

BATT5_FS_CRT_ACT (Critical battery failsafe action)

DEFAULT: 0	RANGE: null

Critical battery failsafe action

Note: This parameter functions identically to BATT_FS_CRT_ACT.

BATT5_FS_LOW_ACT (Low battery failsafe action)

DEFAULT: 0	RANGE: null

Low battery failsafe action

Note: This parameter functions identically to BATT_FS_LOW_ACT.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





BATT5_FS_VOLTSRC (Failsafe voltage source)

DEFAULT: BattMonitor_LowVoltageSource_Raw	RANGE: null

Failsafe voltage source

Note: This parameter functions identically to BATT_FS_VOLTSRC.

BATT5_I2C_ADDR (Battery monitor I2C address)

DEFAULT: AP_BATTMONITOR_SMBUS_I2C_ADDR	RANGE: 0 127

Battery monitor I2C address

Note: This parameter functions identically to BATT_I2C_ADDR.

BATT5_I2C_BUS (Battery monitor I2C bus number)

DEFAULT: 0	RANGE: 0 3

Battery monitor I2C bus number

Note: This parameter functions identically to BATT_I2C_BUS.

BATT5_LOW_MAH (Low battery capacity)

DEFAULT: 0	RANGE: null	UNITS: mAh

Low battery capacity

Note: This parameter functions identically to BATT_LOW_MAH.

BATT5_LOW_TIMER (Low voltage timeout)

DEFAULT: 10	RANGE: 0 120	UNITS: s

Low voltage timeout

Note: This parameter functions identically to BATT_LOW_TIMER.

BATT5_LOW_VOLT (Low battery voltage)

DEFAULT: DEFAULT_LOW_BATTERY_VOLTAGE	RANGE: null	UNITS: V

Low battery voltage

Note: This parameter functions identically to BATT_LOW_VOLT.

BATT5_MONITOR (Battery monitoring)

DEFAULT: int8_t(AP_BattMonitor::Type::NONE	RANGE: null

Battery monitoring

Note: This parameter functions identically to BATT_MONITOR.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





BATT5_OPTIONS (Battery monitor options)

DEFAULT: 0	RANGE: null

Battery monitor options

Note: This parameter functions identically to BATT_OPTIONS.

BATT5_SERIAL_NUM (Battery serial number)

DEFAULT: AP_BATT_SERIAL_NUMBER_DEFAULT	RANGE: null

Battery serial number

Note: This parameter functions identically to BATT_SERIAL_NUM.

BATT5_SUM_MASK (Battery Sum mask)

DEFAULT: 0	RANGE: null

Battery Sum mask

Note: This parameter functions identically to BATT_SUM_MASK.

BATT5_VLT_OFFSET (Voltage offset)

DEFAULT: 0	RANGE: null	UNITS: V

Voltage offset

Note: This parameter functions identically to BATT_VLT_OFFSET.

BATT5_VOLT_MULT (Voltage Multiplier)

DEFAULT: AP_BATT_VOLTDIVIDER_DEFAULT	RANGE: null

Voltage Multiplier

Note: This parameter functions identically to BATT_VOLT_MULT.

BATT5_VOLT_PIN (Battery Voltage sensing pin)

DEFAULT: AP_BATT_VOLT_PIN	RANGE: null

Battery Voltage sensing pin

Note: This parameter functions identically to BATT_VOLT_PIN.

BATT5_WATT_MAX (Maximum allowed power (Watts))

DEFAULT: 0	RANGE: null	UNITS: W

Maximum allowed power (Watts)

Note: This parameter functions identically to BATT_WATT_MAX.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





BATT6_AMP_OFFSET (AMP offset)

DEFAULT: AP_BATT_CURR_AMP_OFFSET_DEFAULT	RANGE: null	UNITS: V

AMP offset

Note: This parameter functions identically to BATT_AMP_OFFSET.

BATT6_AMP_PERVLT (Amps per volt)

DEFAULT: AP_BATT_CURR_AMP_PERVOLT_DEFAULT	RANGE: null	UNITS: A/V

Amps per volt

Note: This parameter functions identically to BATT_AMP_PERVLT.

BATT6_ARM_MAH (Required arming remaining capacity)

DEFAULT: 0	RANGE: null	UNITS: mAh

Required arming remaining capacity

Note: This parameter functions identically to BATT_ARM_MAH.

BATT6_ARM_VOLT (Required arming voltage)

DEFAULT: 0	RANGE: null	UNITS: V

Required arming voltage

Note: This parameter functions identically to BATT_ARM_VOLT.

BATT6_CAPACITY (Battery capacity)

DEFAULT: AP_BATT_MONITOR_BATTERY_CAPACITY	RANGE: null	UNITS: mAh

Battery capacity

Note: This parameter functions identically to BATT_CAPACITY.

BATT6_CRT_MAH (Battery critical capacity)

DEFAULT: 0	RANGE: null	UNITS: mAh

Battery critical capacity

Note: This parameter functions identically to BATT_CRT_MAH.

BATT6_CRT_VOLT (Critical battery voltage)

DEFAULT: 0	RANGE: null	UNITS: V

Critical battery voltage

Note: This parameter functions identically to BATT_CRT_VOLT.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





BATT6_CURR_MULT (Scales reported power monitor current)

DEFAULT: 1.0	RANGE: .1 10

Scales reported power monitor current

Note: This parameter functions identically to BATT_CURR_MULT.

BATT6_CURR_PIN (Battery Current sensing pin)

DEFAULT: AP_BATT_CURR_PIN	RANGE: null

Battery Current sensing pin

Note: This parameter functions identically to BATT_CURR_PIN.

BATT6_ESC_INDEX (ESC Telemetry Index to write to)

DEFAULT: 0	RANGE: 0 10

ESC Telemetry Index to write to

Note: This parameter functions identically to BATT_ESC_INDEX.

BATT6_FS_CRT_ACT (Critical battery failsafe action)

DEFAULT: 0	RANGE: null

Critical battery failsafe action

Note: This parameter functions identically to BATT_FS_CRT_ACT.

BATT6_FS_LOW_ACT (Low battery failsafe action)

DEFAULT: 0	RANGE: null

Low battery failsafe action

Note: This parameter functions identically to BATT_FS_LOW_ACT.

BATT6_FS_VOLTSRC (Failsafe voltage source)

DEFAULT: BattMonitor_LowVoltageSource_Raw	RANGE: null

Failsafe voltage source

Note: This parameter functions identically to BATT_FS_VOLTSRC.

BATT6_I2C_ADDR (Battery monitor I2C address)

DEFAULT: AP_BATTMONITOR_SMBUS_I2C_ADDR	RANGE: 0 127

Battery monitor I2C address

Note: This parameter functions identically to BATT_I2C_ADDR.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





BATT6_I2C_BUS (Battery monitor I2C bus number)

DEFAULT: 0	RANGE: 0 3

Battery monitor I2C bus number

Note: This parameter functions identically to BATT_I2C_BUS.

BATT6_LOW_MAH (Low battery capacity)

DEFAULT: 0	RANGE: null	UNITS: mAh

Low battery capacity

Note: This parameter functions identically to BATT_LOW_MAH.

BATT6_LOW_TIMER (Low voltage timeout)

DEFAULT: 10	RANGE: 0 120	UNITS: s

Low voltage timeout

Note: This parameter functions identically to BATT_LOW_TIMER.

BATT6_LOW_VOLT (Low battery voltage)

DEFAULT: DEFAULT_LOW_BATTERY_VOLTAGE	RANGE: null	UNITS: V

Low battery voltage

Note: This parameter functions identically to BATT_LOW_VOLT.

BATT6_MONITOR (Battery monitoring)

DEFAULT: int8_t(AP_BattMonitor::Type::NONE	RANGE: null

Battery monitoring

Note: This parameter functions identically to BATT_MONITOR.

BATT6_OPTIONS (Battery monitor options)

DEFAULT: 0	RANGE: null

Battery monitor options

Note: This parameter functions identically to BATT_OPTIONS.

BATT6_SERIAL_NUM (Battery serial number)

DEFAULT: AP_BATT_SERIAL_NUMBER_DEFAULT	RANGE: null

Battery serial number

Note: This parameter functions identically to BATT_SERIAL_NUM.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





BATT6_SUM_MASK (Battery Sum mask)

DEFAULT: 0	RANGE: null

Battery Sum mask

Note: This parameter functions identically to BATT_SUM_MASK.

BATT6_VLT_OFFSET (Voltage offset)

DEFAULT: 0	RANGE: null	UNITS: V

Voltage offset

Note: This parameter functions identically to BATT_VLT_OFFSET.

BATT6_VOLT_MULT (Voltage Multiplier)

DEFAULT: AP_BATT_VOLTDIVIDER_DEFAULT	RANGE: null

Voltage Multiplier

Note: This parameter functions identically to BATT_VOLT_MULT.

BATT6_VOLT_PIN (Battery Voltage sensing pin)

DEFAULT: AP_BATT_VOLT_PIN	RANGE: null

Battery Voltage sensing pin

Note: This parameter functions identically to BATT_VOLT_PIN.

BATT6_WATT_MAX (Maximum allowed power (Watts))

DEFAULT: 0	RANGE: null	UNITS: W

Maximum allowed power (Watts)

Note: This parameter functions identically to BATT_WATT_MAX.

BATT7_AMP_OFFSET (AMP offset)

DEFAULT: AP_BATT_CURR_AMP_OFFSET_DEFAULT	RANGE: null	UNITS: V

AMP offset

Note: This parameter functions identically to BATT_AMP_OFFSET.

BATT7_AMP_PERVLT (Amps per volt)

DEFAULT: AP_BATT_CURR_AMP_PERVOLT_DEFAULT	RANGE: null	UNITS: A/V

Amps per volt

Note: This parameter functions identically to BATT_AMP_PERVLT.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





BATT7_ARM_MAH (Required arming remaining capacity)

DEFAULT: 0	RANGE: null	UNITS: mAh

Required arming remaining capacity

Note: This parameter functions identically to BATT_ARM_MAH.

BATT7_ARM_VOLT (Required arming voltage)

DEFAULT: 0	RANGE: null	UNITS: V

Required arming voltage

Note: This parameter functions identically to BATT_ARM_VOLT.

BATT7_CAPACITY (Battery capacity)

DEFAULT: AP_BATT_MONITOR_BATTERY_CAPACITY	RANGE: null	UNITS: mAh

Battery capacity

Note: This parameter functions identically to BATT_CAPACITY.

BATT7_CRT_MAH (Battery critical capacity)

DEFAULT: 0	RANGE: null	UNITS: mAh

Battery critical capacity

Note: This parameter functions identically to BATT_CRT_MAH.

BATT7_CRT_VOLT (Critical battery voltage)

DEFAULT: 0	RANGE: null	UNITS: V

Critical battery voltage

Note: This parameter functions identically to BATT_CRT_VOLT.

BATT7_CURR_MULT (Scales reported power monitor current)

DEFAULT: 1.0	RANGE: .1 10

Scales reported power monitor current

Note: This parameter functions identically to BATT_CURR_MULT.

BATT7_CURR_PIN (Battery Current sensing pin)

DEFAULT: AP_BATT_CURR_PIN	RANGE: null

Battery Current sensing pin

Note: This parameter functions identically to BATT_CURR_PIN.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





BATT7_ESC_INDEX (ESC Telemetry Index to write to)

DEFAULT: 0	RANGE: 0 10

ESC Telemetry Index to write to

Note: This parameter functions identically to BATT_ESC_INDEX.

BATT7_FS_CRT_ACT (Critical battery failsafe action)

DEFAULT: 0	RANGE: null

Critical battery failsafe action

Note: This parameter functions identically to BATT_FS_CRT_ACT.

BATT7_FS_LOW_ACT (Low battery failsafe action)

DEFAULT: 0	RANGE: null

Low battery failsafe action

Note: This parameter functions identically to BATT_FS_LOW_ACT.

BATT7_FS_VOLTSRC (Failsafe voltage source)

DEFAULT: BattMonitor_LowVoltageSource_Raw	RANGE: null

Failsafe voltage source

Note: This parameter functions identically to BATT_FS_VOLTSRC.

BATT7_I2C_ADDR (Battery monitor I2C address)

DEFAULT: AP_BATTMONITOR_SMBUS_I2C_ADDR	RANGE: 0 127

Battery monitor I2C address

Note: This parameter functions identically to BATT_I2C_ADDR.

BATT7_I2C_BUS (Battery monitor I2C bus number)

DEFAULT: 0	RANGE: 0 3

Battery monitor I2C bus number

Note: This parameter functions identically to BATT_I2C_BUS.

BATT7_LOW_MAH (Low battery capacity)

DEFAULT: 0	RANGE: null	UNITS: mAh

Low battery capacity

Note: This parameter functions identically to BATT_LOW_MAH.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





BATT7_LOW_TIMER (Low voltage timeout)

DEFAULT: 10	RANGE: 0 120	UNITS: s

Low voltage timeout

Note: This parameter functions identically to BATT_LOW_TIMER.

BATT7_LOW_VOLT (Low battery voltage)

DEFAULT: DEFAULT_LOW_BATTERY_VOLTAGE	RANGE: null	UNITS: V

Low battery voltage

Note: This parameter functions identically to BATT_LOW_VOLT.

BATT7_MONITOR (Battery monitoring)

DEFAULT: int8_t(AP_BattMonitor::Type::NONE	RANGE: null

Battery monitoring

Note: This parameter functions identically to BATT_MONITOR.

BATT7_OPTIONS (Battery monitor options)

DEFAULT: 0	RANGE: null

Battery monitor options

Note: This parameter functions identically to BATT_OPTIONS.

BATT7_SERIAL_NUM (Battery serial number)

DEFAULT: AP_BATT_SERIAL_NUMBER_DEFAULT	RANGE: null

Battery serial number

Note: This parameter functions identically to BATT_SERIAL_NUM.

BATT7_SUM_MASK (Battery Sum mask)

DEFAULT: 0	RANGE: null

Battery Sum mask

Note: This parameter functions identically to BATT_SUM_MASK.

BATT7_VLT_OFFSET (Voltage offset)

DEFAULT: 0	RANGE: null	UNITS: V

Voltage offset

Note: This parameter functions identically to BATT_VLT_OFFSET.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





BATT7_VOLT_MULT (Voltage Multiplier)

DEFAULT: AP_BATT_VOLTDIVIDER_DEFAULT	RANGE: null

Voltage Multiplier

Note: This parameter functions identically to BATT_VOLT_MULT.

BATT7_VOLT_PIN (Battery Voltage sensing pin)

DEFAULT: AP_BATT_VOLT_PIN	RANGE: null

Battery Voltage sensing pin

Note: This parameter functions identically to BATT_VOLT_PIN.

BATT7_WATT_MAX (Maximum allowed power (Watts))

DEFAULT: 0	RANGE: null	UNITS: W

Maximum allowed power (Watts)

Note: This parameter functions identically to BATT_WATT_MAX.

BATT8_AMP_OFFSET (AMP offset)

DEFAULT: AP_BATT_CURR_AMP_OFFSET_DEFAULT	RANGE: null	UNITS: V

AMP offset

Note: This parameter functions identically to BATT_AMP_OFFSET.

BATT8_AMP_PERVLT (Amps per volt)

DEFAULT: AP_BATT_CURR_AMP_PERVOLT_DEFAULT	RANGE: null	UNITS: A/V

Amps per volt

Note: This parameter functions identically to BATT_AMP_PERVLT.

BATT8_ARM_MAH (Required arming remaining capacity)

DEFAULT: 0	RANGE: null	UNITS: mAh

Required arming remaining capacity

Note: This parameter functions identically to BATT_ARM_MAH.

BATT8_ARM_VOLT (Required arming voltage)

DEFAULT: 0	RANGE: null	UNITS: V

Required arming voltage

Note: This parameter functions identically to BATT_ARM_VOLT.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





BATT8_CAPACITY (Battery capacity)

DEFAULT: AP_BATT_MONITOR_BATTERY_CAPACITY	RANGE: null	UNITS: mAh

Battery capacity

Note: This parameter functions identically to BATT_CAPACITY.

BATT8_CRT_MAH (Battery critical capacity)

DEFAULT: 0	RANGE: null	UNITS: mAh

Battery critical capacity

Note: This parameter functions identically to BATT_CRT_MAH.

BATT8_CRT_VOLT (Critical battery voltage)

DEFAULT: 0	RANGE: null	UNITS: V

Critical battery voltage

Note: This parameter functions identically to BATT_CRT_VOLT.

BATT8_CURR_MULT (Scales reported power monitor current)

DEFAULT: 1.0	RANGE: .1 10

Scales reported power monitor current

Note: This parameter functions identically to BATT_CURR_MULT.

BATT8_CURR_PIN (Battery Current sensing pin)

DEFAULT: AP_BATT_CURR_PIN	RANGE: null

Battery Current sensing pin

Note: This parameter functions identically to BATT_CURR_PIN.

BATT8_ESC_INDEX (ESC Telemetry Index to write to)

DEFAULT: 0	RANGE: 0 10

ESC Telemetry Index to write to

Note: This parameter functions identically to BATT_ESC_INDEX.

BATT8_FS_CRT_ACT (Critical battery failsafe action)

DEFAULT: 0	RANGE: null

Critical battery failsafe action

Note: This parameter functions identically to BATT_FS_CRT_ACT.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





BATT8_FS_LOW_ACT (Low battery failsafe action)

DEFAULT: 0	RANGE: null

Low battery failsafe action

Note: This parameter functions identically to BATT_FS_LOW_ACT.

BATT8_FS_VOLTSRC (Failsafe voltage source)

DEFAULT: BattMonitor_LowVoltageSource_Raw	RANGE: null

Failsafe voltage source

Note: This parameter functions identically to BATT_FS_VOLTSRC.

BATT8_I2C_ADDR (Battery monitor I2C address)

DEFAULT: AP_BATTMONITOR_SMBUS_I2C_ADDR	RANGE: 0 127

Battery monitor I2C address

Note: This parameter functions identically to BATT_I2C_ADDR.

BATT8_I2C_BUS (Battery monitor I2C bus number)

DEFAULT: 0	RANGE: 0 3

Battery monitor I2C bus number

Note: This parameter functions identically to BATT_I2C_BUS.

BATT8_LOW_MAH (Low battery capacity)

DEFAULT: 0	RANGE: null	UNITS: mAh

Low battery capacity

Note: This parameter functions identically to BATT_LOW_MAH.

BATT8_LOW_TIMER (Low voltage timeout)

DEFAULT: 10	RANGE: 0 120	UNITS: s

Low voltage timeout

Note: This parameter functions identically to BATT_LOW_TIMER.

BATT8_LOW_VOLT (Low battery voltage)

DEFAULT: DEFAULT_LOW_BATTERY_VOLTAGE	RANGE: null	UNITS: V

Low battery voltage

Note: This parameter functions identically to BATT_LOW_VOLT.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





BATT8_MONITOR (Battery monitoring)

DEFAULT: int8_t(AP_BattMonitor::Type::NONE	RANGE: null

Battery monitoring

Note: This parameter functions identically to BATT_MONITOR.

BATT8_OPTIONS (Battery monitor options)

DEFAULT: 0	RANGE: null

Battery monitor options

Note: This parameter functions identically to BATT_OPTIONS.

BATT8_SERIAL_NUM (Battery serial number)

DEFAULT: AP_BATT_SERIAL_NUMBER_DEFAULT	RANGE: null

Battery serial number

Note: This parameter functions identically to BATT_SERIAL_NUM.

BATT8_SUM_MASK (Battery Sum mask)

DEFAULT: 0	RANGE: null

Battery Sum mask

Note: This parameter functions identically to BATT_SUM_MASK.

BATT8_VLT_OFFSET (Voltage offset)

DEFAULT: 0	RANGE: null	UNITS: V

Voltage offset

Note: This parameter functions identically to BATT_VLT_OFFSET.

BATT8_VOLT_MULT (Voltage Multiplier)

DEFAULT: AP_BATT_VOLTDIVIDER_DEFAULT	RANGE: null

Voltage Multiplier

Note: This parameter functions identically to BATT_VOLT_MULT.

BATT8_VOLT_PIN (Battery Voltage sensing pin)

DEFAULT: AP_BATT_VOLT_PIN	RANGE: null

Battery Voltage sensing pin

Note: This parameter functions identically to BATT_VOLT_PIN.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





BATT8_WATT_MAX (Maximum allowed power (Watts))

DEFAULT: 0	RANGE: null	UNITS: W

Maximum allowed power (Watts)

Note: This parameter functions identically to BATT_WATT_MAX.

BATT9_AMP_OFFSET (AMP offset)

DEFAULT: AP_BATT_CURR_AMP_OFFSET_DEFAULT	RANGE: null	UNITS: V

AMP offset

Note: This parameter functions identically to BATT_AMP_OFFSET.

BATT9_AMP_PERVLT (Amps per volt)

DEFAULT: AP_BATT_CURR_AMP_PERVOLT_DEFAULT	RANGE: null	UNITS: A/V

Amps per volt

Note: This parameter functions identically to BATT_AMP_PERVLT.

BATT9_ARM_MAH (Required arming remaining capacity)

DEFAULT: 0	RANGE: null	UNITS: mAh

Required arming remaining capacity

Note: This parameter functions identically to BATT_ARM_MAH.

BATT9_ARM_VOLT (Required arming voltage)

DEFAULT: 0	RANGE: null	UNITS: V

Required arming voltage

Note: This parameter functions identically to BATT_ARM_VOLT.

BATT9_CAPACITY (Battery capacity)

DEFAULT: AP_BATT_MONITOR_BATTERY_CAPACITY	RANGE: null	UNITS: mAh

Battery capacity

Note: This parameter functions identically to BATT_CAPACITY.

BATT9_CRT_MAH (Battery critical capacity)

DEFAULT: 0	RANGE: null	UNITS: mAh

Battery critical capacity

Note: This parameter functions identically to BATT_CRT_MAH.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





BATT9_CRT_VOLT (Critical battery voltage)

DEFAULT: 0	RANGE: null	UNITS: V

Critical battery voltage

Note: This parameter functions identically to BATT_CRT_VOLT.

BATT9_CURR_MULT (Scales reported power monitor current)

DEFAULT: 1.0	RANGE: .1 10

Scales reported power monitor current

Note: This parameter functions identically to BATT_CURR_MULT.

BATT9_CURR_PIN (Battery Current sensing pin)

DEFAULT: AP_BATT_CURR_PIN	RANGE: null

Battery Current sensing pin

Note: This parameter functions identically to BATT_CURR_PIN.

BATT9_ESC_INDEX (ESC Telemetry Index to write to)

DEFAULT: 0	RANGE: 0 10

ESC Telemetry Index to write to

Note: This parameter functions identically to BATT_ESC_INDEX.

BATT9_FS_CRT_ACT (Critical battery failsafe action)

DEFAULT: 0	RANGE: null

Critical battery failsafe action

Note: This parameter functions identically to BATT_FS_CRT_ACT.

BATT9_FS_LOW_ACT (Low battery failsafe action)

DEFAULT: 0	RANGE: null

Low battery failsafe action

Note: This parameter functions identically to BATT_FS_LOW_ACT.

BATT9_FS_VOLTSRC (Failsafe voltage source)

DEFAULT: BattMonitor_LowVoltageSource_Raw	RANGE: null

Failsafe voltage source

Note: This parameter functions identically to BATT_FS_VOLTSRC.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





BATT9_I2C_ADDR (Battery monitor I2C address)

DEFAULT: AP_BATTMONITOR_SMBUS_I2C_ADDR	RANGE: 0 127

Battery monitor I2C address

Note: This parameter functions identically to BATT_I2C_ADDR.

BATT9_I2C_BUS (Battery monitor I2C bus number)

DEFAULT: 0	RANGE: 0 3

Battery monitor I2C bus number

Note: This parameter functions identically to BATT_I2C_BUS.

BATT9_LOW_MAH (Low battery capacity)

DEFAULT: 0	RANGE: null	UNITS: mAh

Low battery capacity

Note: This parameter functions identically to BATT_LOW_MAH.

BATT9_LOW_TIMER (Low voltage timeout)

DEFAULT: 10	RANGE: 0 120	UNITS: s

Low voltage timeout

Note: This parameter functions identically to BATT_LOW_TIMER.

BATT9_LOW_VOLT (Low battery voltage)

DEFAULT: DEFAULT_LOW_BATTERY_VOLTAGE	RANGE: null	UNITS: V

Low battery voltage

Note: This parameter functions identically to BATT_LOW_VOLT.

BATT9_MONITOR (Battery monitoring)

DEFAULT: int8_t(AP_BattMonitor::Type::NONE	RANGE: null

Battery monitoring

Note: This parameter functions identically to BATT_MONITOR.

BATT9_OPTIONS (Battery monitor options)

DEFAULT: 0	RANGE: null

Battery monitor options

Note: This parameter functions identically to BATT_OPTIONS.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





BATT9_SERIAL_NUM (Battery serial number)

DEFAULT: AP_BATT_SERIAL_NUMBER_DEFAULT	RANGE: null

Battery serial number

Note: This parameter functions identically to BATT_SERIAL_NUM.

BATT9_SUM_MASK (Battery Sum mask)

DEFAULT: 0	RANGE: null

Battery Sum mask

Note: This parameter functions identically to BATT_SUM_MASK.

BATT9_VLT_OFFSET (Voltage offset)

DEFAULT: 0	RANGE: null	UNITS: V

Voltage offset

Note: This parameter functions identically to BATT_VLT_OFFSET.

BATT9_VOLT_MULT (Voltage Multiplier)

DEFAULT: AP_BATT_VOLTDIVIDER_DEFAULT	RANGE: null

Voltage Multiplier

Note: This parameter functions identically to BATT_VOLT_MULT.

BATT9_VOLT_PIN (Battery Voltage sensing pin)

DEFAULT: AP_BATT_VOLT_PIN	RANGE: null

Battery Voltage sensing pin

Note: This parameter functions identically to BATT_VOLT_PIN.

BATT9_WATT_MAX (Maximum allowed power (Watts))

DEFAULT: 0	RANGE: null	UNITS: W

Maximum allowed power (Watts)

Note: This parameter functions identically to BATT_WATT_MAX.

BATTA_AMP_OFFSET (AMP offset)

DEFAULT: AP_BATT_CURR_AMP_OFFSET_DEFAULT	RANGE: null	UNITS: V

AMP offset

Note: This parameter functions identically to BATT_AMP_OFFSET.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





BATTA_AMP_PERVLT (Amps per volt)

DEFAULT: AP_BATT_CURR_AMP_PERVOLT_DEFAULT	RANGE: null	UNITS: A/V

Amps per volt

Note: This parameter functions identically to BATT_AMP_PERVLT.

BATTA_ARM_MAH (Required arming remaining capacity)

DEFAULT: 0	RANGE: null	UNITS: mAh

Required arming remaining capacity

Note: This parameter functions identically to BATT_ARM_MAH.

BATTA_ARM_VOLT (Required arming voltage)

DEFAULT: 0	RANGE: null	UNITS: V

Required arming voltage

Note: This parameter functions identically to BATT_ARM_VOLT.

BATTA_CAPACITY (Battery capacity)

DEFAULT: AP_BATT_MONITOR_BATTERY_CAPACITY	RANGE: null	UNITS: mAh

Battery capacity

Note: This parameter functions identically to BATT_CAPACITY.

BATTA_CRT_MAH (Battery critical capacity)

DEFAULT: 0	RANGE: null	UNITS: mAh

Battery critical capacity

Note: This parameter functions identically to BATT_CRT_MAH.

BATTA_CRT_VOLT (Critical battery voltage)

DEFAULT: 0	RANGE: null	UNITS: V

Critical battery voltage

Note: This parameter functions identically to BATT_CRT_VOLT.

BATTA_CURR_MULT (Scales reported power monitor current)

DEFAULT: 1.0	RANGE: .1 10

Scales reported power monitor current

Note: This parameter functions identically to BATT_CURR_MULT.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





BATTA_CURR_PIN (Battery Current sensing pin)

DEFAULT: AP_BATT_CURR_PIN	RANGE: null

Battery Current sensing pin

Note: This parameter functions identically to BATT_CURR_PIN.

BATTA_ESC_INDEX (ESC Telemetry Index to write to)

DEFAULT: 0	RANGE: 0 10

ESC Telemetry Index to write to

Note: This parameter functions identically to BATT_ESC_INDEX.

BATTA_FS_CRT_ACT (Critical battery failsafe action)

DEFAULT: 0	RANGE: null

Critical battery failsafe action

Note: This parameter functions identically to BATT_FS_CRT_ACT.

BATTA_FS_LOW_ACT (Low battery failsafe action)

DEFAULT: 0	RANGE: null

Low battery failsafe action

Note: This parameter functions identically to BATT_FS_LOW_ACT.

BATTA_FS_VOLTSRC (Failsafe voltage source)

DEFAULT: BattMonitor_LowVoltageSource_Raw	RANGE: null

Failsafe voltage source

Note: This parameter functions identically to BATT_FS_VOLTSRC.

BATTA_I2C_ADDR (Battery monitor I2C address)

DEFAULT: AP_BATTMONITOR_SMBUS_I2C_ADDR	RANGE: 0 127

Battery monitor I2C address

Note: This parameter functions identically to BATT_I2C_ADDR.

BATTA_I2C_BUS (Battery monitor I2C bus number)

DEFAULT: 0	RANGE: 0 3

Battery monitor I2C bus number

Note: This parameter functions identically to BATT_I2C_BUS.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





BATTA_LOW_MAH (Low battery capacity)

DEFAULT: 0	RANGE: null	UNITS: mAh

Low battery capacity

Note: This parameter functions identically to BATT_LOW_MAH.

BATTA_LOW_TIMER (Low voltage timeout)

DEFAULT: 10	RANGE: 0 120	UNITS: s

Low voltage timeout

Note: This parameter functions identically to BATT_LOW_TIMER.

BATTA_LOW_VOLT (Low battery voltage)

DEFAULT: DEFAULT_LOW_BATTERY_VOLTAGE	RANGE: null	UNITS: V

Low battery voltage

Note: This parameter functions identically to BATT_LOW_VOLT.

BATTA_MONITOR (Battery monitoring)

DEFAULT: int8_t(AP_BattMonitor::Type::NONE	RANGE: null

Battery monitoring

Note: This parameter functions identically to BATT_MONITOR.

BATTA_OPTIONS (Battery monitor options)

DEFAULT: 0	RANGE: null

Battery monitor options

Note: This parameter functions identically to BATT_OPTIONS.

BATTA_SERIAL_NUM (Battery serial number)

DEFAULT: AP_BATT_SERIAL_NUMBER_DEFAULT	RANGE: null

Battery serial number

Note: This parameter functions identically to BATT_SERIAL_NUM.

BATTA_SUM_MASK (Battery Sum mask)

DEFAULT: 0	RANGE: null

Battery Sum mask

Note: This parameter functions identically to BATT_SUM_MASK.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





BATTA_VLT_OFFSET (Voltage offset)

DEFAULT: 0	RANGE: null	UNITS: V

Voltage offset

Note: This parameter functions identically to BATT_VLT_OFFSET.

BATTA_VOLT_MULT (Voltage Multiplier)

DEFAULT: AP_BATT_VOLTDIVIDER_DEFAULT	RANGE: null

Voltage Multiplier

Note: This parameter functions identically to BATT_VOLT_MULT.

BATTA_VOLT_PIN (Battery Voltage sensing pin)

DEFAULT: AP_BATT_VOLT_PIN	RANGE: null

Battery Voltage sensing pin

Note: This parameter functions identically to BATT_VOLT_PIN.

BATTA_WATT_MAX (Maximum allowed power (Watts))

DEFAULT: 0	RANGE: null	UNITS: W

Maximum allowed power (Watts)

Note: This parameter functions identically to BATT_WATT_MAX.

BATTB_AMP_OFFSET (AMP offset)

DEFAULT: AP_BATT_CURR_AMP_OFFSET_DEFAULT	RANGE: null	UNITS: V

AMP offset

Note: This parameter functions identically to BATT_AMP_OFFSET.

BATTB_AMP_PERVLT (Amps per volt)

DEFAULT: AP_BATT_CURR_AMP_PERVOLT_DEFAULT	RANGE: null	UNITS: A/V

Amps per volt

Note: This parameter functions identically to BATT_AMP_PERVLT.

BATTB_ARM_MAH (Required arming remaining capacity)

DEFAULT: 0	RANGE: null	UNITS: mAh

Required arming remaining capacity

Note: This parameter functions identically to BATT_ARM_MAH.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





BATTB_ARM_VOLT (Required arming voltage)

DEFAULT: 0	RANGE: null	UNITS: V

Required arming voltage

Note: This parameter functions identically to BATT_ARM_VOLT.

BATTB_CAPACITY (Battery capacity)

DEFAULT: AP_BATT_MONITOR_BATTERY_CAPACITY	RANGE: null	UNITS: mAh

Battery capacity

Note: This parameter functions identically to BATT_CAPACITY.

BATTB_CRT_MAH (Battery critical capacity)

DEFAULT: 0	RANGE: null	UNITS: mAh

Battery critical capacity

Note: This parameter functions identically to BATT_CRT_MAH.

BATTB_CRT_VOLT (Critical battery voltage)

DEFAULT: 0	RANGE: null	UNITS: V

Critical battery voltage

Note: This parameter functions identically to BATT_CRT_VOLT.

BATTB_CURR_MULT (Scales reported power monitor current)

DEFAULT: 1.0	RANGE: .1 10

Scales reported power monitor current

Note: This parameter functions identically to BATT_CURR_MULT.

BATTB_CURR_PIN (Battery Current sensing pin)

DEFAULT: AP_BATT_CURR_PIN	RANGE: null

Battery Current sensing pin

Note: This parameter functions identically to BATT_CURR_PIN.

BATTB_ESC_INDEX (ESC Telemetry Index to write to)

DEFAULT: 0	RANGE: 0 10

ESC Telemetry Index to write to

Note: This parameter functions identically to BATT_ESC_INDEX.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





BATTB_FS_CRT_ACT (Critical battery failsafe action)

DEFAULT: 0	RANGE: null

Critical battery failsafe action

Note: This parameter functions identically to BATT_FS_CRT_ACT.

BATTB_FS_LOW_ACT (Low battery failsafe action)

DEFAULT: 0	RANGE: null

Low battery failsafe action

Note: This parameter functions identically to BATT_FS_LOW_ACT.

BATTB_FS_VOLTSRC (Failsafe voltage source)

DEFAULT: BattMonitor_LowVoltageSource_Raw	RANGE: null

Failsafe voltage source

Note: This parameter functions identically to BATT_FS_VOLTSRC.

BATTB_I2C_ADDR (Battery monitor I2C address)

DEFAULT: AP_BATTMONITOR_SMBUS_I2C_ADDR	RANGE: 0 127

Battery monitor I2C address

Note: This parameter functions identically to BATT_I2C_ADDR.

BATTB_I2C_BUS (Battery monitor I2C bus number)

DEFAULT: 0	RANGE: 0 3

Battery monitor I2C bus number

Note: This parameter functions identically to BATT_I2C_BUS.

BATTB_LOW_MAH (Low battery capacity)

DEFAULT: 0	RANGE: null	UNITS: mAh

Low battery capacity

Note: This parameter functions identically to BATT_LOW_MAH.

BATTB_LOW_TIMER (Low voltage timeout)

DEFAULT: 10	RANGE: 0 120	UNITS: s

Low voltage timeout

Note: This parameter functions identically to BATT_LOW_TIMER.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





BATTB_LOW_VOLT (Low battery voltage)

DEFAULT: DEFAULT_LOW_BATTERY_VOLTAGE	RANGE: null	UNITS: V

Low battery voltage

Note: This parameter functions identically to BATT_LOW_VOLT.

BATTB_MONITOR (Battery monitoring)

DEFAULT: int8_t(AP_BattMonitor::Type::NONE	RANGE: null

Battery monitoring

Note: This parameter functions identically to BATT_MONITOR.

BATTB_OPTIONS (Battery monitor options)

DEFAULT: 0	RANGE: null

Battery monitor options

Note: This parameter functions identically to BATT_OPTIONS.

BATTB_SERIAL_NUM (Battery serial number)

DEFAULT: AP_BATT_SERIAL_NUMBER_DEFAULT	RANGE: null

Battery serial number

Note: This parameter functions identically to BATT_SERIAL_NUM.

BATTB_SUM_MASK (Battery Sum mask)

DEFAULT: 0	RANGE: null

Battery Sum mask

Note: This parameter functions identically to BATT_SUM_MASK.

BATTB_VLT_OFFSET (Voltage offset)

DEFAULT: 0	RANGE: null	UNITS: V

Voltage offset

Note: This parameter functions identically to BATT_VLT_OFFSET.

BATTB_VOLT_MULT (Voltage Multiplier)

DEFAULT: AP_BATT_VOLTDIVIDER_DEFAULT	RANGE: null

Voltage Multiplier

Note: This parameter functions identically to BATT_VOLT_MULT.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





BATTB_VOLT_PIN (Battery Voltage sensing pin)

DEFAULT: AP_BATT_VOLT_PIN	RANGE: null

Battery Voltage sensing pin

Note: This parameter functions identically to BATT_VOLT_PIN.

BATTB_WATT_MAX (Maximum allowed power (Watts))

DEFAULT: 0	RANGE: null	UNITS: W

Maximum allowed power (Watts)

Note: This parameter functions identically to BATT_WATT_MAX.

BATTC_AMP_OFFSET (AMP offset)

DEFAULT: AP_BATT_CURR_AMP_OFFSET_DEFAULT	RANGE: null	UNITS: V

AMP offset

Note: This parameter functions identically to BATT_AMP_OFFSET.

BATTC_AMP_PERVLT (Amps per volt)

DEFAULT: AP_BATT_CURR_AMP_PERVOLT_DEFAULT	RANGE: null	UNITS: A/V

Amps per volt

Note: This parameter functions identically to BATT_AMP_PERVLT.

BATTC_ARM_MAH (Required arming remaining capacity)

DEFAULT: 0	RANGE: null	UNITS: mAh

Required arming remaining capacity

Note: This parameter functions identically to BATT_ARM_MAH.

BATTC_ARM_VOLT (Required arming voltage)

DEFAULT: 0	RANGE: null	UNITS: V

Required arming voltage

Note: This parameter functions identically to BATT_ARM_VOLT.

BATTC_CAPACITY (Battery capacity)

DEFAULT: AP_BATT_MONITOR_BATTERY_CAPACITY	RANGE: null	UNITS: mAh

Battery capacity

Note: This parameter functions identically to BATT_CAPACITY.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





BATTC_CRT_MAH (Battery critical capacity)

DEFAULT: 0	RANGE: null	UNITS: mAh

Battery critical capacity

Note: This parameter functions identically to BATT_CRT_MAH.

BATTC_CRT_VOLT (Critical battery voltage)

DEFAULT: 0	RANGE: null	UNITS: V

Critical battery voltage

Note: This parameter functions identically to BATT_CRT_VOLT.

BATTC_CURR_MULT (Scales reported power monitor current)

DEFAULT: 1.0	RANGE: .1 10

Scales reported power monitor current

Note: This parameter functions identically to BATT_CURR_MULT.

BATTC_CURR_PIN (Battery Current sensing pin)

DEFAULT: AP_BATT_CURR_PIN	RANGE: null

Battery Current sensing pin

Note: This parameter functions identically to BATT_CURR_PIN.

BATTC_ESC_INDEX (ESC Telemetry Index to write to)

DEFAULT: 0	RANGE: 0 10

ESC Telemetry Index to write to

Note: This parameter functions identically to BATT_ESC_INDEX.

BATTC_FS_CRT_ACT (Critical battery failsafe action)

DEFAULT: 0	RANGE: null

Critical battery failsafe action

Note: This parameter functions identically to BATT_FS_CRT_ACT.

BATTC_FS_LOW_ACT (Low battery failsafe action)

DEFAULT: 0	RANGE: null

Low battery failsafe action

Note: This parameter functions identically to BATT_FS_LOW_ACT.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





BATTC_FS_VOLTSRC (Failsafe voltage source)

DEFAULT: BattMonitor_LowVoltageSource_Raw	RANGE: null

Failsafe voltage source

Note: This parameter functions identically to BATT_FS_VOLTSRC.

BATTC_I2C_ADDR (Battery monitor I2C address)

DEFAULT: AP_BATTMONITOR_SMBUS_I2C_ADDR	RANGE: 0 127

Battery monitor I2C address

Note: This parameter functions identically to BATT_I2C_ADDR.

BATTC_I2C_BUS (Battery monitor I2C bus number)

DEFAULT: 0	RANGE: 0 3

Battery monitor I2C bus number

Note: This parameter functions identically to BATT_I2C_BUS.

BATTC_LOW_MAH (Low battery capacity)

DEFAULT: 0	RANGE: null	UNITS: mAh

Low battery capacity

Note: This parameter functions identically to BATT_LOW_MAH.

BATTC_LOW_TIMER (Low voltage timeout)

DEFAULT: 10	RANGE: 0 120	UNITS: s

Low voltage timeout

Note: This parameter functions identically to BATT_LOW_TIMER.

BATTC_LOW_VOLT (Low battery voltage)

DEFAULT: DEFAULT_LOW_BATTERY_VOLTAGE	RANGE: null	UNITS: V

Low battery voltage

Note: This parameter functions identically to BATT_LOW_VOLT.

BATTC_MONITOR (Battery monitoring)

DEFAULT: int8_t(AP_BattMonitor::Type::NONE	RANGE: null

Battery monitoring

Note: This parameter functions identically to BATT_MONITOR.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





BATTC_OPTIONS (Battery monitor options)

DEFAULT: 0	RANGE: null

Battery monitor options

Note: This parameter functions identically to BATT_OPTIONS.

BATTC_SERIAL_NUM (Battery serial number)

DEFAULT: AP_BATT_SERIAL_NUMBER_DEFAULT	RANGE: null

Battery serial number

Note: This parameter functions identically to BATT_SERIAL_NUM.

BATTC_SUM_MASK (Battery Sum mask)

DEFAULT: 0	RANGE: null

Battery Sum mask

Note: This parameter functions identically to BATT_SUM_MASK.

BATTC_VLT_OFFSET (Voltage offset)

DEFAULT: 0	RANGE: null	UNITS: V

Voltage offset

Note: This parameter functions identically to BATT_VLT_OFFSET.

BATTC_VOLT_MULT (Voltage Multiplier)

DEFAULT: AP_BATT_VOLTDIVIDER_DEFAULT	RANGE: null

Voltage Multiplier

Note: This parameter functions identically to BATT_VOLT_MULT.

BATTC_VOLT_PIN (Battery Voltage sensing pin)

DEFAULT: AP_BATT_VOLT_PIN	RANGE: null

Battery Voltage sensing pin

Note: This parameter functions identically to BATT_VOLT_PIN.

BATTC_WATT_MAX (Maximum allowed power (Watts))

DEFAULT: 0	RANGE: null	UNITS: W

Maximum allowed power (Watts)

Note: This parameter functions identically to BATT_WATT_MAX.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





BATTD_AMP_OFFSET (AMP offset)

DEFAULT: AP_BATT_CURR_AMP_OFFSET_DEFAULT	RANGE: null	UNITS: V

AMP offset

Note: This parameter functions identically to BATT_AMP_OFFSET.

BATTD_AMP_PERVLT (Amps per volt)

DEFAULT: AP_BATT_CURR_AMP_PERVOLT_DEFAULT	RANGE: null	UNITS: A/V

Amps per volt

Note: This parameter functions identically to BATT_AMP_PERVLT.

BATTD_ARM_MAH (Required arming remaining capacity)

DEFAULT: 0	RANGE: null	UNITS: mAh

Required arming remaining capacity

Note: This parameter functions identically to BATT_ARM_MAH.

BATTD_ARM_VOLT (Required arming voltage)

DEFAULT: 0	RANGE: null	UNITS: V

Required arming voltage

Note: This parameter functions identically to BATT_ARM_VOLT.

BATTD_CAPACITY (Battery capacity)

DEFAULT: AP_BATT_MONITOR_BATTERY_CAPACITY	RANGE: null	UNITS: mAh

Battery capacity

Note: This parameter functions identically to BATT_CAPACITY.

BATTD_CRT_MAH (Battery critical capacity)

DEFAULT: 0	RANGE: null	UNITS: mAh

Battery critical capacity

Note: This parameter functions identically to BATT_CRT_MAH.

BATTD_CRT_VOLT (Critical battery voltage)

DEFAULT: 0	RANGE: null	UNITS: V

Critical battery voltage

Note: This parameter functions identically to BATT_CRT_VOLT.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





BATTD_CURR_MULT (Scales reported power monitor current)

DEFAULT: 1.0	RANGE: .1 10

Scales reported power monitor current

Note: This parameter functions identically to BATT_CURR_MULT.

BATTD_CURR_PIN (Battery Current sensing pin)

DEFAULT: AP_BATT_CURR_PIN	RANGE: null

Battery Current sensing pin

Note: This parameter functions identically to BATT_CURR_PIN.

BATTD_ESC_INDEX (ESC Telemetry Index to write to)

DEFAULT: 0	RANGE: 0 10

ESC Telemetry Index to write to

Note: This parameter functions identically to BATT_ESC_INDEX.

BATTD_FS_CRT_ACT (Critical battery failsafe action)

DEFAULT: 0	RANGE: null

Critical battery failsafe action

Note: This parameter functions identically to BATT_FS_CRT_ACT.

BATTD_FS_LOW_ACT (Low battery failsafe action)

DEFAULT: 0	RANGE: null

Low battery failsafe action

Note: This parameter functions identically to BATT_FS_LOW_ACT.

BATTD_FS_VOLTSRC (Failsafe voltage source)

DEFAULT: BattMonitor_LowVoltageSource_Raw	RANGE: null

Failsafe voltage source

Note: This parameter functions identically to BATT_FS_VOLTSRC.

BATTD_I2C_ADDR (Battery monitor I2C address)

DEFAULT: AP_BATTMONITOR_SMBUS_I2C_ADDR	RANGE: 0 127

Battery monitor I2C address

Note: This parameter functions identically to BATT_I2C_ADDR.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





BATTD_I2C_BUS (Battery monitor I2C bus number)

DEFAULT: 0	RANGE: 0 3

Battery monitor I2C bus number

Note: This parameter functions identically to BATT_I2C_BUS.

BATTD_LOW_MAH (Low battery capacity)

DEFAULT: 0	RANGE: null	UNITS: mAh

Low battery capacity

Note: This parameter functions identically to BATT_LOW_MAH.

BATTD_LOW_TIMER (Low voltage timeout)

DEFAULT: 10	RANGE: 0 120	UNITS: s

Low voltage timeout

Note: This parameter functions identically to BATT_LOW_TIMER.

BATTD_LOW_VOLT (Low battery voltage)

DEFAULT: DEFAULT_LOW_BATTERY_VOLTAGE	RANGE: null	UNITS: V

Low battery voltage

Note: This parameter functions identically to BATT_LOW_VOLT.

BATTD_MONITOR (Battery monitoring)

DEFAULT: int8_t(AP_BattMonitor::Type::NONE	RANGE: null

Battery monitoring

Note: This parameter functions identically to BATT_MONITOR.

BATTD_OPTIONS (Battery monitor options)

DEFAULT: 0	RANGE: null

Battery monitor options

Note: This parameter functions identically to BATT_OPTIONS.

BATTD_SERIAL_NUM (Battery serial number)

DEFAULT: AP_BATT_SERIAL_NUMBER_DEFAULT	RANGE: null

Battery serial number

Note: This parameter functions identically to BATT_SERIAL_NUM.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





BATTD_SUM_MASK (Battery Sum mask)

DEFAULT: 0	RANGE: null

Battery Sum mask

Note: This parameter functions identically to BATT_SUM_MASK.

BATTD_VLT_OFFSET (Voltage offset)

DEFAULT: 0	RANGE: null	UNITS: V

Voltage offset

Note: This parameter functions identically to BATT_VLT_OFFSET.

BATTD_VOLT_MULT (Voltage Multiplier)

DEFAULT: AP_BATT_VOLTDIVIDER_DEFAULT	RANGE: null

Voltage Multiplier

Note: This parameter functions identically to BATT_VOLT_MULT.

BATTD_VOLT_PIN (Battery Voltage sensing pin)

DEFAULT: AP_BATT_VOLT_PIN	RANGE: null

Battery Voltage sensing pin

Note: This parameter functions identically to BATT_VOLT_PIN.

BATTD_WATT_MAX (Maximum allowed power (Watts))

DEFAULT: 0	RANGE: null	UNITS: W

Maximum allowed power (Watts)

Note: This parameter functions identically to BATT_WATT_MAX.

BATTE_AMP_OFFSET (AMP offset)

DEFAULT: AP_BATT_CURR_AMP_OFFSET_DEFAULT	RANGE: null	UNITS: V

AMP offset

Note: This parameter functions identically to BATT_AMP_OFFSET.

BATTE_AMP_PERVLT (Amps per volt)

DEFAULT: AP_BATT_CURR_AMP_PERVOLT_DEFAULT	RANGE: null	UNITS: A/V

Amps per volt

Note: This parameter functions identically to BATT_AMP_PERVLT.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





BATTE_ARM_MAH (Required arming remaining capacity)

DEFAULT: 0	RANGE: null	UNITS: mAh

Required arming remaining capacity

Note: This parameter functions identically to BATT_ARM_MAH.

BATTE_ARM_VOLT (Required arming voltage)

DEFAULT: 0	RANGE: null	UNITS: V

Required arming voltage

Note: This parameter functions identically to BATT_ARM_VOLT.

BATTE_CAPACITY (Battery capacity)

DEFAULT: AP_BATT_MONITOR_BATTERY_CAPACITY	RANGE: null	UNITS: mAh

Battery capacity

Note: This parameter functions identically to BATT_CAPACITY.

BATTE_CRT_MAH (Battery critical capacity)

DEFAULT: 0	RANGE: null	UNITS: mAh

Battery critical capacity

Note: This parameter functions identically to BATT_CRT_MAH.

BATTE_CRT_VOLT (Critical battery voltage)

DEFAULT: 0	RANGE: null	UNITS: V

Critical battery voltage

Note: This parameter functions identically to BATT_CRT_VOLT.

BATTE_CURR_MULT (Scales reported power monitor current)

DEFAULT: 1.0	RANGE: .1 10

Scales reported power monitor current

Note: This parameter functions identically to BATT_CURR_MULT.

BATTE_CURR_PIN (Battery Current sensing pin)

DEFAULT: AP_BATT_CURR_PIN	RANGE: null

Battery Current sensing pin

Note: This parameter functions identically to BATT_CURR_PIN.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





BATTE_ESC_INDEX (ESC Telemetry Index to write to)

DEFAULT: 0	RANGE: 0 10

ESC Telemetry Index to write to

Note: This parameter functions identically to BATT_ESC_INDEX.

BATTE_FS_CRT_ACT (Critical battery failsafe action)

DEFAULT: 0	RANGE: null

Critical battery failsafe action

Note: This parameter functions identically to BATT_FS_CRT_ACT.

BATTE_FS_LOW_ACT (Low battery failsafe action)

DEFAULT: 0	RANGE: null

Low battery failsafe action

Note: This parameter functions identically to BATT_FS_LOW_ACT.

BATTE_FS_VOLTSRC (Failsafe voltage source)

DEFAULT: BattMonitor_LowVoltageSource_Raw	RANGE: null

Failsafe voltage source

Note: This parameter functions identically to BATT_FS_VOLTSRC.

BATTE_I2C_ADDR (Battery monitor I2C address)

DEFAULT: AP_BATTMONITOR_SMBUS_I2C_ADDR	RANGE: 0 127

Battery monitor I2C address

Note: This parameter functions identically to BATT_I2C_ADDR.

BATTE_I2C_BUS (Battery monitor I2C bus number)

DEFAULT: 0	RANGE: 0 3

Battery monitor I2C bus number

Note: This parameter functions identically to BATT_I2C_BUS.

BATTE_LOW_MAH (Low battery capacity)

DEFAULT: 0	RANGE: null	UNITS: mAh

Low battery capacity

Note: This parameter functions identically to BATT_LOW_MAH.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





BATTE_LOW_TIMER (Low voltage timeout)

DEFAULT: 10	RANGE: 0 120	UNITS: s

Low voltage timeout

Note: This parameter functions identically to BATT_LOW_TIMER.

BATTE_LOW_VOLT (Low battery voltage)

DEFAULT: DEFAULT_LOW_BATTERY_VOLTAGE	RANGE: null	UNITS: V

Low battery voltage

Note: This parameter functions identically to BATT_LOW_VOLT.

BATTE_MONITOR (Battery monitoring)

DEFAULT: int8_t(AP_BattMonitor::Type::NONE	RANGE: null

Battery monitoring

Note: This parameter functions identically to BATT_MONITOR.

BATTE_OPTIONS (Battery monitor options)

DEFAULT: 0	RANGE: null

Battery monitor options

Note: This parameter functions identically to BATT_OPTIONS.

BATTE_SERIAL_NUM (Battery serial number)

DEFAULT: AP_BATT_SERIAL_NUMBER_DEFAULT	RANGE: null

Battery serial number

Note: This parameter functions identically to BATT_SERIAL_NUM.

BATTE_SUM_MASK (Battery Sum mask)

DEFAULT: 0	RANGE: null

Battery Sum mask

Note: This parameter functions identically to BATT_SUM_MASK.

BATTE_VLT_OFFSET (Voltage offset)

DEFAULT: 0	RANGE: null	UNITS: V

Voltage offset

Note: This parameter functions identically to BATT_VLT_OFFSET.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





BATTE_VOLT_MULT (Voltage Multiplier)

DEFAULT: AP_BATT_VOLTDIVIDER_DEFAULT	RANGE: null

Voltage Multiplier

Note: This parameter functions identically to BATT_VOLT_MULT.

BATTE_VOLT_PIN (Battery Voltage sensing pin)

DEFAULT: AP_BATT_VOLT_PIN	RANGE: null

Battery Voltage sensing pin

Note: This parameter functions identically to BATT_VOLT_PIN.

BATTE_WATT_MAX (Maximum allowed power (Watts))

DEFAULT: 0	RANGE: null	UNITS: W

Maximum allowed power (Watts)

Note: This parameter functions identically to BATT_WATT_MAX.

BATTF_AMP_OFFSET (AMP offset)

DEFAULT: AP_BATT_CURR_AMP_OFFSET_DEFAULT	RANGE: null	UNITS: V

AMP offset

Note: This parameter functions identically to BATT_AMP_OFFSET.

BATTF_AMP_PERVLT (Amps per volt)

DEFAULT: AP_BATT_CURR_AMP_PERVOLT_DEFAULT	RANGE: null	UNITS: A/V

Amps per volt

Note: This parameter functions identically to BATT_AMP_PERVLT.

BATTF_ARM_MAH (Required arming remaining capacity)

DEFAULT: 0	RANGE: null	UNITS: mAh

Required arming remaining capacity

Note: This parameter functions identically to BATT_ARM_MAH.

BATTF_ARM_VOLT (Required arming voltage)

DEFAULT: 0	RANGE: null	UNITS: V

Required arming voltage

Note: This parameter functions identically to BATT_ARM_VOLT.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





BATTF_CAPACITY (Battery capacity)

DEFAULT: AP_BATT_MONITOR_BATTERY_CAPACITY	RANGE: null	UNITS: mAh

Battery capacity

Note: This parameter functions identically to BATT_CAPACITY.

BATTF_CRT_MAH (Battery critical capacity)

DEFAULT: 0	RANGE: null	UNITS: mAh

Battery critical capacity

Note: This parameter functions identically to BATT_CRT_MAH.

BATTF_CRT_VOLT (Critical battery voltage)

DEFAULT: 0	RANGE: null	UNITS: V

Critical battery voltage

Note: This parameter functions identically to BATT_CRT_VOLT.

BATTF_CURR_MULT (Scales reported power monitor current)

DEFAULT: 1.0	RANGE: .1 10

Scales reported power monitor current

Note: This parameter functions identically to BATT_CURR_MULT.

BATTF_CURR_PIN (Battery Current sensing pin)

DEFAULT: AP_BATT_CURR_PIN	RANGE: null

Battery Current sensing pin

Note: This parameter functions identically to BATT_CURR_PIN.

BATTF_ESC_INDEX (ESC Telemetry Index to write to)

DEFAULT: 0	RANGE: 0 10

ESC Telemetry Index to write to

Note: This parameter functions identically to BATT_ESC_INDEX.

BATTF_FS_CRT_ACT (Critical battery failsafe action)

DEFAULT: 0	RANGE: null

Critical battery failsafe action

Note: This parameter functions identically to BATT_FS_CRT_ACT.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





BATTF_FS_LOW_ACT (Low battery failsafe action)

DEFAULT: 0	RANGE: null

Low battery failsafe action

Note: This parameter functions identically to BATT_FS_LOW_ACT.

BATTF_FS_VOLTSRC (Failsafe voltage source)

DEFAULT: BattMonitor_LowVoltageSource_Raw	RANGE: null

Failsafe voltage source

Note: This parameter functions identically to BATT_FS_VOLTSRC.

BATTF_I2C_ADDR (Battery monitor I2C address)

DEFAULT: AP_BATTMONITOR_SMBUS_I2C_ADDR	RANGE: 0 127

Battery monitor I2C address

Note: This parameter functions identically to BATT_I2C_ADDR.

BATTF_I2C_BUS (Battery monitor I2C bus number)

DEFAULT: 0	RANGE: 0 3

Battery monitor I2C bus number

Note: This parameter functions identically to BATT_I2C_BUS.

BATTF_LOW_MAH (Low battery capacity)

DEFAULT: 0	RANGE: null	UNITS: mAh

Low battery capacity

Note: This parameter functions identically to BATT_LOW_MAH.

BATTF_LOW_TIMER (Low voltage timeout)

DEFAULT: 10	RANGE: 0 120	UNITS: s

Low voltage timeout

Note: This parameter functions identically to BATT_LOW_TIMER.

BATTF_LOW_VOLT (Low battery voltage)

DEFAULT: DEFAULT_LOW_BATTERY_VOLTAGE	RANGE: null	UNITS: V

Low battery voltage

Note: This parameter functions identically to BATT_LOW_VOLT.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





BATTF_MONITOR (Battery monitoring)

DEFAULT: int8_t(AP_BattMonitor::Type::NONE	RANGE: null

Battery monitoring

Note: This parameter functions identically to BATT_MONITOR.

BATTF_OPTIONS (Battery monitor options)

DEFAULT: 0	RANGE: null

Battery monitor options

Note: This parameter functions identically to BATT_OPTIONS.

BATTF_SERIAL_NUM (Battery serial number)

DEFAULT: AP_BATT_SERIAL_NUMBER_DEFAULT	RANGE: null

Battery serial number

Note: This parameter functions identically to BATT_SERIAL_NUM.

BATTF_SUM_MASK (Battery Sum mask)

DEFAULT: 0	RANGE: null

Battery Sum mask

Note: This parameter functions identically to BATT_SUM_MASK.

BATTF_VLT_OFFSET (Voltage offset)

DEFAULT: 0	RANGE: null	UNITS: V

Voltage offset

Note: This parameter functions identically to BATT_VLT_OFFSET.

BATTF_VOLT_MULT (Voltage Multiplier)

DEFAULT: AP_BATT_VOLTDIVIDER_DEFAULT	RANGE: null

Voltage Multiplier

Note: This parameter functions identically to BATT_VOLT_MULT.

BATTF_VOLT_PIN (Battery Voltage sensing pin)

DEFAULT: AP_BATT_VOLT_PIN	RANGE: null

Battery Voltage sensing pin

Note: This parameter functions identically to BATT_VOLT_PIN.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





BATTF_WATT_MAX (Maximum allowed power (Watts))

DEFAULT: 0	RANGE: null	UNITS: W

Maximum allowed power (Watts)

Note: This parameter functions identically to BATT_WATT_MAX.

BATTG_AMP_OFFSET (AMP offset)

DEFAULT: AP_BATT_CURR_AMP_OFFSET_DEFAULT	RANGE: null	UNITS: V

AMP offset

Note: This parameter functions identically to BATT_AMP_OFFSET.

BATTG_AMP_PERVLT (Amps per volt)

DEFAULT: AP_BATT_CURR_AMP_PERVOLT_DEFAULT	RANGE: null	UNITS: A/V

Amps per volt

Note: This parameter functions identically to BATT_AMP_PERVLT.

BATTG_ARM_MAH (Required arming remaining capacity)

DEFAULT: 0	RANGE: null	UNITS: mAh

Required arming remaining capacity

Note: This parameter functions identically to BATT_ARM_MAH.

BATTG_ARM_VOLT (Required arming voltage)

DEFAULT: 0	RANGE: null	UNITS: V

Required arming voltage

Note: This parameter functions identically to BATT_ARM_VOLT.

BATTG_CAPACITY (Battery capacity)

DEFAULT: AP_BATT_MONITOR_BATTERY_CAPACITY	RANGE: null	UNITS: mAh

Battery capacity

Note: This parameter functions identically to BATT_CAPACITY.

BATTG_CRT_MAH (Battery critical capacity)

DEFAULT: 0	RANGE: null	UNITS: mAh

Battery critical capacity

Note: This parameter functions identically to BATT_CRT_MAH.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





BATTG_CRT_VOLT (Critical battery voltage)

DEFAULT: 0	RANGE: null	UNITS: V

Critical battery voltage

Note: This parameter functions identically to BATT_CRT_VOLT.

BATTG_CURR_MULT (Scales reported power monitor current)

DEFAULT: 1.0	RANGE: .1 10

Scales reported power monitor current

Note: This parameter functions identically to BATT_CURR_MULT.

BATTG_CURR_PIN (Battery Current sensing pin)

DEFAULT: AP_BATT_CURR_PIN	RANGE: null

Battery Current sensing pin

Note: This parameter functions identically to BATT_CURR_PIN.

BATTG_ESC_INDEX (ESC Telemetry Index to write to)

DEFAULT: 0	RANGE: 0 10

ESC Telemetry Index to write to

Note: This parameter functions identically to BATT_ESC_INDEX.

BATTG_FS_CRT_ACT (Critical battery failsafe action)

DEFAULT: 0	RANGE: null

Critical battery failsafe action

Note: This parameter functions identically to BATT_FS_CRT_ACT.

BATTG_FS_LOW_ACT (Low battery failsafe action)

DEFAULT: 0	RANGE: null

Low battery failsafe action

Note: This parameter functions identically to BATT_FS_LOW_ACT.

BATTG_FS_VOLTSRC (Failsafe voltage source)

DEFAULT: BattMonitor_LowVoltageSource_Raw	RANGE: null

Failsafe voltage source

Note: This parameter functions identically to BATT_FS_VOLTSRC.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





BATTG_I2C_ADDR (Battery monitor I2C address)

DEFAULT: AP_BATTMONITOR_SMBUS_I2C_ADDR	RANGE: 0 127

Battery monitor I2C address

Note: This parameter functions identically to BATT_I2C_ADDR.

BATTG_I2C_BUS (Battery monitor I2C bus number)

DEFAULT: 0	RANGE: 0 3

Battery monitor I2C bus number

Note: This parameter functions identically to BATT_I2C_BUS.

BATTG_LOW_MAH (Low battery capacity)

DEFAULT: 0	RANGE: null	UNITS: mAh

Low battery capacity

Note: This parameter functions identically to BATT_LOW_MAH.

BATTG_LOW_TIMER (Low voltage timeout)

DEFAULT: 10	RANGE: 0 120	UNITS: s

Low voltage timeout

Note: This parameter functions identically to BATT_LOW_TIMER.

BATTG_LOW_VOLT (Low battery voltage)

DEFAULT: DEFAULT_LOW_BATTERY_VOLTAGE	RANGE: null	UNITS: V

Low battery voltage

Note: This parameter functions identically to BATT_LOW_VOLT.

BATTG_MONITOR (Battery monitoring)

DEFAULT: int8_t(AP_BattMonitor::Type::NONE	RANGE: null

Battery monitoring

Note: This parameter functions identically to BATT_MONITOR.

BATTG_OPTIONS (Battery monitor options)

DEFAULT: 0	RANGE: null

Battery monitor options

Note: This parameter functions identically to BATT_OPTIONS.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





BATTG_SERIAL_NUM (Battery serial number)

DEFAULT: AP_BATT_SERIAL_NUMBER_DEFAULT	RANGE: null

Battery serial number

Note: This parameter functions identically to BATT_SERIAL_NUM.

BATTG_SUM_MASK (Battery Sum mask)

DEFAULT: 0	RANGE: null

Battery Sum mask

Note: This parameter functions identically to BATT_SUM_MASK.

BATTG_VLT_OFFSET (Voltage offset)

DEFAULT: 0	RANGE: null	UNITS: V

Voltage offset

Note: This parameter functions identically to BATT_VLT_OFFSET.

BATTG_VOLT_MULT (Voltage Multiplier)

DEFAULT: AP_BATT_VOLTDIVIDER_DEFAULT	RANGE: null

Voltage Multiplier

Note: This parameter functions identically to BATT_VOLT_MULT.

BATTG_VOLT_PIN (Battery Voltage sensing pin)

DEFAULT: AP_BATT_VOLT_PIN	RANGE: null

Battery Voltage sensing pin

Note: This parameter functions identically to BATT_VOLT_PIN.

BATTG_WATT_MAX (Maximum allowed power (Watts))

DEFAULT: 0	RANGE: null	UNITS: W

Maximum allowed power (Watts)

Note: This parameter functions identically to BATT_WATT_MAX.

BATT_AMP_OFFSET (AMP offset)

DEFAULT: 0	UNITS: V

AMP offset (BATT_AMP_OFFSET)

Description

The voltage reported by the current sensor when zero current is flowing. This value is subtracted from the reading before scaling to ensure accuracy at low loads.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





BATT_AMP_PERVLT (Amps per volt)

DEFAULT: 17.0	UNITS: A/V

Amps per volt (BATT_AMP_PERVLT)

Description

Scaling factor for the current ADC reading. It defines the relationship between the voltage output of the current sensor and the actual amperage flowing through it.

BATT_ARM_MAH (Required arming remaining capacity)

DEFAULT: 0	UNITS: mAh

Required arming remaining capacity (BATT_ARM_MAH)

Description

A pre-arm safety check. The vehicle will refuse to arm if the calculated remaining capacity is below this value.

Tuning & Behavior

- **Default Value:** 0 (Disabled)
- **Note: Rebooting usually resets this estimate for non-smart batteries.**

BATT_ARM_VOLT (Required arming voltage)

DEFAULT: 0	UNITS: V

Required arming voltage (BATT_ARM_VOLT)

Description

A pre-arm safety check. The vehicle will refuse to arm if the battery voltage is below this value.

Tuning & Behavior

- **Default Value:** 0 (Disabled)
- **Prevents starting a flight with a partially discharged battery.**

BATT_BAL_CELL1_PIN (Battery Balance Cell 1 Pin)

DEFAULT: -1	RANGE: -1 127

Battery Balance Cell 1 Pin (BATT_BAL_CELL1_PIN)

Description

`BATT_BAL_CELL1_PIN` is used by specialized battery monitoring hardware to track the voltage of individual cells in a LiPo pack.

Instead of just seeing the "Total" voltage of the battery, this parameter (and the pins that follow it) allows the autopilot to monitor each cell. This is critical for detecting a single bad cell that might be failing even if the total voltage looks acceptable.

Tuning & Behavior

- **Hardware:** This requires a dedicated "Cell Balancer" circuit or a custom wiring harness connected to several analog pins on the flight controller.
- **Sequential Logic:** You only define the pin for **Cell 1**. The software automatically assumes that Cell 2 is on the next physical pin, Cell 3 on the next, and so on.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





BATT_BAL_ID (Battery Balancer DroneCAN ID)

DEFAULT: 0	RANGE: 0 127

Battery Balancer DroneCAN ID (BATT_BAL_ID)

Description

`BATT_BAL_ID` identifies the battery balancer unit on the DroneCAN bus.

BATT_BAL_NUM_CELLS (Battery Balancer Cell Count)

DEFAULT: 0	RANGE: 0 16	UNITS: Cells

Battery Balancer Cell Count (BATT_BAL_NUM_CELLS)

Description

`BATT_BAL_NUM_CELLS` defines the configuration of your battery pack for the internal balancing system.

This is only used if you have specialized hardware (like a dedicated balancer board) that allows the flight controller to monitor or equalize individual cell voltages.

- **Default:** 0.
- **Usage:** Set to the number of cells in series (e.g. 4 for 4S, 6 for 6S).

BATT_BAL_RATE (Battery Balancer Update Rate)

DEFAULT: 10	RANGE: 1 50	UNITS: Hz

Battery Balancer Update Rate (BATT_BAL_RATE)

Description

`BATT_BAL_RATE` defines how often individual cell voltages are sampled and reported over the communication bus.

A higher rate provides more immediate detection of an individual cell failing or sagging under load, which is critical for high-performance lithium battery management.

BATT_CAPACITY (Battery capacity)

DEFAULT: 3300	UNITS: mAh

Battery capacity (BATT_CAPACITY)

Description

Defines the total usable capacity of the battery pack. This is used to calculate the "Battery Remaining" percentage and trigger capacity-based failsafes.

Tuning & Behavior

- **Default Value:** 3300 mAh
- **Set this to the value printed on your battery label, or slightly lower (e.g., 90%) for a safety margin.**

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





BATT_CHANNEL (Battery Channel (Invalid/Legacy))

DEFAULT: 0	RANGE: 0 16

Battery Channel (BATT_CHANNEL)

Description

This parameter is not found as BATT_CHANNEL in the main codebase. It appears to be a miscategorized RSSI parameter.

BATT_CRT_MAH (Battery critical capacity)

DEFAULT: 0	UNITS: mAh

Battery critical capacity (BATT_CRT_MAH)

Description

The emergency capacity threshold for the "Battery Critical" failsafe. Triggers BATT_FS_CRT_ACT when the remaining mAh drops below this value.

Tuning & Behavior

- **Default Value:** 0 (Disabled)
- **Typically set to 10% of the total BATT_CAPACITY.**

BATT_CRT_VOLT (Critical battery voltage)

DEFAULT: 0	UNITS: V

Critical battery voltage (BATT_CRT_VOLT)

Description

The emergency voltage threshold for the "Battery Critical" failsafe. Triggers BATT_FS_CRT_ACT immediately when the voltage stays below this level for BATT_LOW_TIMER.

Tuning & Behavior

- **Default Value:** 0 (Disabled)
- **Typically set lower than BATT_LOW_VOLT (e.g., 3.3V/cell).**

BATT_CURR_MULT (Current Multiplier (Invalid/Legacy))

DEFAULT: 1.0

Current Multiplier (BATT_CURR_MULT)

Description

This parameter is not found as BATT_CURR_MULT in the main codebase.

BATT_CURR_PIN (Battery Current sensing pin)

DEFAULT: -1

Battery Current sensing pin (BATT_CURR_PIN)

Description

Specifies the hardware analog-to-digital (ADC) pin connected to the current sensor on the power module.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





BATT_ESC_INDEX (ESC Telemetry Index)

DEFAULT: 0	RANGE: 0 10

ESC Telemetry Index (BATT_ESC_INDEX)

Description

Links the battery monitor instance to a specific ESC telemetry channel. This allows the FC to log battery data alongside individual ESC metrics.

BATT_ESC_MASK (ESC Mask (Invalid/Legacy))

DEFAULT: 0

ESC Mask (BATT_ESC_MASK)

Description

This parameter is not found as BATT_ESC_MASK in the main codebase.

BATT_FL_FF (Fuel Level First Order Coefficient)

DEFAULT: 1.0

Fuel Level First Order Coefficient (BATT_FL_FF)

Description

Part of a polynomial fit used to linearize non-linear fuel level sensors (e.g., irregularly shaped tanks).

BATT_FL_FLTR (Fuel Level Filter Frequency)

DEFAULT: 0.3	UNITS: Hz

Fuel Level Filter Frequency (BATT_FL_FLTR)

Description

Low-pass filter frequency for smoothing fuel level readings, preventing spikes caused by sloshing.

BATT_FL_FS (Fuel Level Second Order Coefficient)

DEFAULT: 0.0

Fuel Level Second Order Coefficient (BATT_FL_FS)

Description

Polynomial coefficient for fuel level linearization.

BATT_FL_FT (Fuel Level Third Order Coefficient)

DEFAULT: 0.0

Fuel Level Third Order Coefficient (BATT_FL_FT)

Description

Polynomial coefficient for fuel level linearization.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





BATT_FL_OFF (Fuel Level Offset)

DEFAULT: 0.0

Fuel Level Offset (BATT_FL_OFF)

Description

Linear offset for the fuel level calculation.

BATT_FL_PIN (Fuel Level Pin)

DEFAULT: -1

Fuel Level Pin (BATT_FL_PIN)

Description

Specifies the hardware pin connected to the fuel level sensor.

BATT_FL_VLT_MIN (Fuel Level Empty Voltage)

DEFAULT: 0.5	UNITS: V

Fuel Level Empty Voltage (BATT_FL_VLT_MIN)

Description

Calibration point representing the sensor voltage when the fuel tank is empty.

BATT_FL_V_MULT (Fuel Level Voltage Multiplier)

DEFAULT: 0.5

Fuel Level Voltage Multiplier (BATT_FL_V_MULT)

Description

Scaling factor for the fuel level sensor's analog voltage output.

BATT_FS_CRT_ACT (Critical battery failsafe action)

DEFAULT: 0	RANGE: 0 7

Critical battery failsafe action (BATT_FS_CRT_ACT)

Description

Specifies the flight behavior when the battery reaches the "Critical" threshold.

Tuning & Behavior

- **Default Value:** 0 (None)
- **Common Values:** 1:RTL, 2:Land, 5:Terminate
- **Usually set to "Land" to force an immediate landing to save the battery and airframe.**

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





BATT_FS_LOW_ACT (Low battery failsafe action)

DEFAULT: 0	RANGE: 0 7

Low battery failsafe action (BATT_FS_LOW_ACT)

Description

Specifies the flight behavior when the battery reaches the "Low" threshold.

Tuning & Behavior

- **Default Value:** 0 (None)
- **Common Values:** 1:RTL, 2:Land, 3:SmartRTL
- **Ensure your chosen action is appropriate for your flying environment.**

BATT_FS_VOLTSRC (Failsafe voltage source)

DEFAULT: 0	RANGE: 0 1

Failsafe voltage source (BATT_FS_VOLTSRC)

Description

Selects whether the failsafe logic uses the "Raw" measured voltage or the "Sag Compensated" voltage (which accounts for voltage drops under load).

Tuning & Behavior

- **Default Value:** 0 (Raw Voltage)
- **Values:** 0:Raw Voltage, 1:Sag Compensated Voltage

BATT_HIDE_MASK (Battery hide mask)

DEFAULT: 0

Battery hide mask (BATT_HIDE_MASK)

Description

Allows an AP_Periph node to monitor multiple battery inputs internally while only reporting a subset (or a single SUM) to the primary flight controller via CAN.

BATT_I2C_ADDR (Battery monitor I2C address)

DEFAULT: 0	RANGE: 0 127

Battery monitor I2C address (BATT_I2C_ADDR)

Description

The I2C device address of the smart battery. A value of 0 usually triggers auto-detection.

BATT_I2C_BUS (Battery monitor I2C bus)

DEFAULT: 0	RANGE: 0 3

Battery monitor I2C bus (BATT_I2C_BUS)

Description

Specifies the physical I2C bus where the smart battery (SMBus/I2C) is connected.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





BATT_LOW_MAH (Low battery capacity)

DEFAULT: 0	UNITS: mAh

Low battery capacity (BATT_LOW_MAH)

Description

The primary capacity threshold for the "Battery Low" failsafe. Triggers BATT_FS_LOW_ACT when the remaining mAh drops below this value.

Tuning & Behavior

- **Default Value:** 0 (Disabled)
- **Typically set to 20% of the total BATT_CAPACITY.**

BATT_LOW_TIMER (Low voltage timeout)

DEFAULT: 10	RANGE: 0 120	UNITS: s

Low voltage timeout (BATT_LOW_TIMER)

Description

Defines how long the battery voltage must remain below the BATT_LOW_VOLT threshold before a failsafe is triggered. This prevents false triggers caused by brief voltage sags during high-throttle maneuvers.

Tuning & Behavior

- **Default Value:** 10 s
- **Range:** 0 to 120 s

BATT_LOW_VOLT (Low battery voltage)

DEFAULT: 10.5	UNITS: V

Low battery voltage (BATT_LOW_VOLT)

Description

The primary voltage threshold for the "Battery Low" failsafe. When the voltage stays below this value for BATT_LOW_TIMER, the action defined in BATT_FS_LOW_ACT is taken.

Tuning & Behavior

- **Default Value:** 10.5 V (Copter), 0.0 V (Plane)
- **For 3S LiPo, 10.5V (3.5V/cell) is a common safe threshold.**

BATT_MAX_AMPS (Maximum current)

DEFAULT: 90	UNITS: A

Maximum current (BATT_MAX_AMPS)

Description

The hardware measurement limit for the current sensor (e.g., for INA2xx or similar sensors).

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





BATT_MAX_VOLT (Maximum battery voltage (Invalid/Legacy))

DEFAULT: 0	UNITS: V

Maximum battery voltage (BATT_MAX_VOLT)

Description

This parameter is not found as BATT_MAX_VOLT in the main codebase.

BATT_MONITOR (Battery monitoring)

DEFAULT: 0

Battery monitoring (BATT_MONITOR)

Description

Enables the battery monitoring system and selects the sensor type (Analog, SMBus, DroneCAN, etc.).

Tuning & Behavior

- **Default Value:** 0 (Disabled)
- **Common Values:** 3:Analog Voltage Only, 4:Analog Voltage and Current, 8:DroneCAN, 9:ESC
- **Requires reboot after changing to initialize the specific driver.**

BATT_OPTIONS (Battery monitor options)

DEFAULT: 0

Battery monitor options (BATT_OPTIONS)

Description

Configuration bitmask for advanced battery features like MPPT reporting or special DroneCAN behaviors.

Tuning & Behavior

- **Default Value:** 0
- **Bitmask:**
  - Bit 6: Send resistance compensated voltage to GCS
  - Bit 8: Battery is for internal autopilot use only
  - Bit 9: Sum monitor measures minimum voltage instead of average

BATT_SERIAL_NUM (Battery serial number)

DEFAULT: -1

Battery serial number (BATT_SERIAL_NUM)

Description

Unique serial number or ID assigned to the battery pack. Primarily used for distinguishing between multiple "Smart" or DroneCAN batteries.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





BATT_SHUNT (Shunt resistance)

DEFAULT: 0.0005	UNITS: Ohm

Shunt resistance (BATT_SHUNT)

Description

The resistance value of the physical shunt resistor used for current measurement. Critical for accurate amperage reporting on certain sensor types.

BATT_SUM_MASK (Battery Sum Mask)

DEFAULT: 0

Battery Sum Mask (BATT_SUM_MASK)

Description

Defines which battery monitor instances should be mathematically combined (summed) to create a single virtual battery monitor. This is useful for vehicles with multiple independent batteries powering the same rail.

BATT_VLT_OFFSET (Voltage offset)

DEFAULT: 0	UNITS: V

Voltage offset (BATT_VLT_OFFSET)

Description

A static voltage offset subtracted from the ADC reading before the multiplier is applied. Commonly used to account for voltage drops across protective diodes.

BATT_VOLT_MULT (Voltage Multiplier)

DEFAULT: 10.1

Voltage Multiplier (BATT_VOLT_MULT)

Description

Scaling factor for the voltage ADC reading. This value depends on the resistor divider ratio in your power module.

BATT_VOLT_PIN (Battery Voltage sensing pin)

DEFAULT: -1

Battery Voltage sensing pin (BATT_VOLT_PIN)

Description

Specifies the hardware analog-to-digital (ADC) pin connected to the battery voltage divider on the power module.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





BATT_WATT_MAX (Maximum allowed power (Watts))

DEFAULT: 0	UNITS: W

Maximum allowed power (BATT_WATT_MAX)

Description

A power-limiting safety feature for ArduPlane. If the instantaneous power consumption exceeds this threshold, the flight controller will automatically cap the maximum throttle to protect the battery and ESCs.

Tuning & Behavior

- **Default Value:** 0 (Disabled)
- **Prevents over-discharging low-C rated batteries or overheating the powertrain.**

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

Key Concepts

BATTA_AMP_OFFSET (Airspeed offset)

DEFAULT: 0	RANGE: null	UNITS: null

Airspeed offset

Note: This parameter is functionally identical to BATT_AMP_OFFSET.

Please refer to the documentation for the primary instance for detailed configuration instructions.

BATTA_AMP_PERVLT (Amps per volt)

DEFAULT: 0	RANGE: null	UNITS: A/V

Amps per volt

Note: This parameter is functionally identical to BATT_AMP_PERVLT.

Please refer to the documentation for the primary instance for detailed configuration instructions.

BATTA_ARM_MAH (Required arming remaining capacity)

DEFAULT: 0	RANGE: null	UNITS: mAh

Required arming remaining capacity

Note: This parameter configures Battery Monitor A. It functions identically to BATT_ARM_MAH.

BATTA_ARM_VOLT (Required arming voltage)

DEFAULT: 0	RANGE: null	UNITS: V

Required arming voltage

Note: This parameter is functionally identical to BATT_ARM_VOLT.

Please refer to the documentation for the primary instance for detailed configuration instructions.

BATTA_CAPACITY (Battery capacity)

DEFAULT: AP_BATT_MONITOR_BATTERY_CAPACITY	RANGE: null	UNITS: mAh

Battery capacity

Note: This parameter is functionally identical to BATT_CAPACITY.

Please refer to the documentation for the primary instance for detailed configuration instructions.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





BATTA_CHANNEL (Receiver RSSI channel number)

DEFAULT: 0	RANGE: 0 16	UNITS: null

Receiver RSSI channel number

Note: This parameter configures instance A. It functions identically to BATT_CHANNEL.

BATTA_CRT_MAH (Battery critical capacity)

DEFAULT: 0	RANGE: null	UNITS: mAh

Battery critical capacity

Note: This parameter is functionally identical to BATT_CRT_MAH.

Please refer to the documentation for the primary instance for detailed configuration instructions.

BATTA_CRT_VOLT (Critical battery voltage)

DEFAULT: 0	RANGE: null	UNITS: V

Critical battery voltage

Note: This parameter is functionally identical to BATT_CRT_VOLT.

Please refer to the documentation for the primary instance for detailed configuration instructions.

BATTA_CURR_MULT (Scales reported power monitor current)

DEFAULT: 1.0	RANGE: .1 10	UNITS: null

Scales reported power monitor current

Note: This parameter configures instance A. It functions identically to BATT_CURR_MULT.

BATTA_CURR_PIN (Airspeed pin)

DEFAULT: 0	RANGE: null	UNITS: null

Airspeed pin

Note: This parameter is functionally identical to BATT_CURR_PIN.

Please refer to the documentation for the primary instance for detailed configuration instructions.

BATTA_ESC_INDEX (ESC Telemetry Index to write to)

DEFAULT: 0	RANGE: 0 10	UNITS: null

ESC Telemetry Index to write to

Note: This parameter configures Battery Monitor A. It functions identically to BATT_ESC_INDEX.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





BATTA_ESC_MASK (Tiltrotor mask)

DEFAULT: 0	RANGE: null	UNITS: null

Tiltrotor mask

Note: This parameter configures Battery Monitor A. It functions identically to BATT_ESC_MASK.

BATTA_FL_FF (FF FeedForward Gain)

DEFAULT: 0	RANGE: null	UNITS: null

FF FeedForward Gain

Note: This parameter configures instance A. It functions identically to BATT_FL_FF.

BATTA_FL_FLTR (Fuel level filter frequency)

DEFAULT: 0.3	RANGE: -1 1	UNITS: Hz

Fuel level filter frequency

Note: This parameter configures instance A. It functions identically to BATT_FL_FLTR.

BATTA_FL_FS (Second order term)

DEFAULT: 0	RANGE: 0 10	UNITS: null

Second order term

Note: This parameter configures instance A. It functions identically to BATT_FL_FS.

BATTA_FL_FT (Third order term)

DEFAULT: 0	RANGE: 0 10	UNITS: null

Third order term

Note: This parameter configures instance A. It functions identically to BATT_FL_FT.

BATTA_FL_OFF (Offset term)

DEFAULT: 0	RANGE: 0 10	UNITS: null

Offset term

Note: This parameter configures instance A. It functions identically to BATT_FL_OFF.

BATTA_FL_PIN (Airspeed pin)

DEFAULT: 0	RANGE: null	UNITS: null

Airspeed pin

Note: This parameter configures instance A. It functions identically to BATT_FL_PIN.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





BATTA_FL_VLT_MIN (Empty fuel level voltage)

DEFAULT: 0.5	RANGE: 0.01 10	UNITS: V

Empty fuel level voltage

Note: This parameter configures instance A. It functions identically to BATT_FL_VLT_MIN.

BATTA_FL_V_MULT (Fuel level voltage multiplier)

DEFAULT: 0.5	RANGE: 0.01 10	UNITS: null

Fuel level voltage multiplier

Note: This parameter configures instance A. It functions identically to BATT_FL_V_MULT.

BATTA_FS_CRT_ACT (Critical battery failsafe action)

DEFAULT: 0	RANGE: null	UNITS: null

Critical battery failsafe action

Note: This parameter configures Battery Monitor A. It functions identically to BATT_FS_CRT_ACT.

BATTA_FS_LOW_ACT (Low battery failsafe action)

DEFAULT: 0	RANGE: null	UNITS: null

Low battery failsafe action

Note: This parameter is functionally identical to BATT_FS_LOW_ACT.

Please refer to the documentation for the primary instance for detailed configuration instructions.

BATTA_FS_VOLTSRC (Failsafe voltage source)

DEFAULT: BattMonitor_LowVoltageSource_Raw	RANGE: null	UNITS: null

Failsafe voltage source

Note: This parameter configures instance A. It functions identically to BATT_FS_VOLTSRC.

BATTA_I2C_ADDR (Battery monitor I2C address)

DEFAULT: HAL_BATTMON_INA2XX_ADDR	RANGE: 0 127	UNITS: null

Battery monitor I2C address

Note: This parameter configures Battery Monitor A. It functions identically to BATT_I2C_ADDR.

BATTA_I2C_BUS (Sensor Bus)

DEFAULT: -1	RANGE: null	UNITS: null

Sensor Bus

Note: This parameter configures Battery Monitor A. It functions identically to BATT_I2C_BUS.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





BATTA_LOW_MAH (Low battery capacity)

DEFAULT: 0	RANGE: null	UNITS: mAh

Low battery capacity

Note: This parameter is functionally identical to BATT_LOW_MAH.

Please refer to the documentation for the primary instance for detailed configuration instructions.

BATTA_LOW_TIMER (Low voltage timeout)

DEFAULT: 10	RANGE: 0 120	UNITS: s

Low voltage timeout

Note: This parameter is functionally identical to BATT_LOW_TIMER.

Please refer to the documentation for the primary instance for detailed configuration instructions.

BATTA_LOW_VOLT (Low battery voltage)

DEFAULT: DEFAULT_LOW_BATTERY_VOLTAGE	RANGE: null	UNITS: V

Low battery voltage

Note: This parameter is functionally identical to BATT_LOW_VOLT.

Please refer to the documentation for the primary instance for detailed configuration instructions.

BATTA_MAX_AMPS (Battery monitor max current)

DEFAULT: HAL_BATTMON_INA239_MAX_CURRENT	RANGE: 1 400	UNITS: A

Battery monitor max current

Note: This parameter configures Battery Monitor A. It functions identically to BATT_MAX_AMPS.

BATTA_MAX_VOLT (Maximum Battery Voltage)

DEFAULT: 12.6	RANGE: 7 100	UNITS: null

Maximum Battery Voltage

Note: This parameter configures Battery Monitor A. It functions identically to BATT_MAX_VOLT.

BATTA_MONITOR (Battery monitoring)

DEFAULT: int8_t(AP_BattMonitor::Type::NONE	RANGE: null	UNITS: null

Battery monitoring

Note: This parameter is functionally identical to BATT_MONITOR.

Please refer to the documentation for the primary instance for detailed configuration instructions.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





BATTA_OPTIONS (quadplane options)

DEFAULT: 0	RANGE: null	UNITS: null

quadplane options

Note: This parameter configures Battery Monitor A. It functions identically to BATT_OPTIONS.

BATTA_SERIAL_NUM (Battery serial number)

DEFAULT: AP_BATT_SERIAL_NUMBER_DEFAULT	RANGE: null	UNITS: null

Battery serial number

Note: This parameter configures Battery Monitor A. It functions identically to BATT_SERIAL_NUM.

BATTA_SHUNT (Battery monitor shunt resistor)

DEFAULT: HAL_BATTMON_INA239_SHUNT_RESISTANCE	RANGE: 0.0001 0.01	UNITS: Ohm

Battery monitor shunt resistor

Note: This parameter configures Battery Monitor A. It functions identically to BATT_SHUNT.

BATTA_SUM_MASK (Tiltrotor mask)

DEFAULT: 0	RANGE: null	UNITS: null

Tiltrotor mask

Note: This parameter is functionally identical to BATT_SUM_MASK.

Please refer to the documentation for the primary instance for detailed configuration instructions.

BATTA_VLT_OFFSET (Airspeed offset)

DEFAULT: 0	RANGE: null	UNITS: null

Airspeed offset

Note: This parameter configures Battery Monitor A. It functions identically to BATT_VLT_OFFSET.

BATTA_VOLT_MULT (Voltage Multiplier)

DEFAULT: 0	RANGE: null	UNITS: null

Voltage Multiplier

Note: This parameter is functionally identical to BATT_VOLT_MULT.

Please refer to the documentation for the primary instance for detailed configuration instructions.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





BATTA_VOLT_PIN (Airspeed pin)

DEFAULT: 0	RANGE: null	UNITS: null

Airspeed pin

Note: This parameter is functionally identical to BATT_VOLT_PIN.

Please refer to the documentation for the primary instance for detailed configuration instructions.

BATTA_WATT_MAX (Tiltrotor maximum VTOL angle)

DEFAULT: 45	RANGE: 20 80	UNITS: deg

Tiltrotor maximum VTOL angle

Note: This parameter configures Battery Monitor A. It functions identically to BATT_WATT_MAX.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY

Overview

Key Concepts

BATTB_AMP_OFFSET (Airspeed offset)

DEFAULT: 0	RANGE: null	UNITS: null

Airspeed offset

Note: This parameter is functionally identical to BATT_AMP_OFFSET.

Please refer to the documentation for the primary instance for detailed configuration instructions.

BATTB_AMP_PERVLT (Amps per volt)

DEFAULT: 0	RANGE: null	UNITS: A/V

Amps per volt

Note: This parameter is functionally identical to BATT_AMP_PERVLT.

Please refer to the documentation for the primary instance for detailed configuration instructions.

BATTB_ARM_MAH (Required arming remaining capacity)

DEFAULT: 0	RANGE: null	UNITS: mAh

Required arming remaining capacity

Note: This parameter configures Battery Monitor B. It functions identically to BATT_ARM_MAH.

BATTB_ARM_VOLT (Required arming voltage)

DEFAULT: 0	RANGE: null	UNITS: V

Required arming voltage

Note: This parameter is functionally identical to BATT_ARM_VOLT.

Please refer to the documentation for the primary instance for detailed configuration instructions.

BATTB_CAPACITY (Battery capacity)

DEFAULT: AP_BATT_MONITOR_BATTERY_CAPACITY	RANGE: null	UNITS: mAh

Battery capacity

Note: This parameter is functionally identical to BATT_CAPACITY.

Please refer to the documentation for the primary instance for detailed configuration instructions.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





BATTB_CHANNEL (Receiver RSSI channel number)

DEFAULT: 0	RANGE: 0 16	UNITS: null

Receiver RSSI channel number

Note: This parameter configures instance B. It functions identically to BATT_CHANNEL.

BATTB_CRT_MAH (Battery critical capacity)

DEFAULT: 0	RANGE: null	UNITS: mAh

Battery critical capacity

Note: This parameter is functionally identical to BATT_CRT_MAH.

Please refer to the documentation for the primary instance for detailed configuration instructions.

BATTB_CRT_VOLT (Critical battery voltage)

DEFAULT: 0	RANGE: null	UNITS: V

Critical battery voltage

Note: This parameter is functionally identical to BATT_CRT_VOLT.

Please refer to the documentation for the primary instance for detailed configuration instructions.

BATTB_CURR_MULT (Scales reported power monitor current)

DEFAULT: 1.0	RANGE: .1 10	UNITS: null

Scales reported power monitor current

Note: This parameter configures instance B. It functions identically to BATT_CURR_MULT.

BATTB_CURR_PIN (Airspeed pin)

DEFAULT: 0	RANGE: null	UNITS: null

Airspeed pin

Note: This parameter is functionally identical to BATT_CURR_PIN.

Please refer to the documentation for the primary instance for detailed configuration instructions.

BATTB_ESC_INDEX (ESC Telemetry Index to write to)

DEFAULT: 0	RANGE: 0 10	UNITS: null

ESC Telemetry Index to write to

Note: This parameter configures Battery Monitor B. It functions identically to BATT_ESC_INDEX.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





BATTB_ESC_MASK (Tiltrotor mask)

DEFAULT: 0	RANGE: null	UNITS: null

Tiltrotor mask

Note: This parameter configures Battery Monitor B. It functions identically to BATT_ESC_MASK.

BATTB_FL_FF (FF FeedForward Gain)

DEFAULT: 0	RANGE: null	UNITS: null

FF FeedForward Gain

Note: This parameter configures instance B. It functions identically to BATT_FL_FF.

BATTB_FL_FLTR (Fuel level filter frequency)

DEFAULT: 0.3	RANGE: -1 1	UNITS: Hz

Fuel level filter frequency

Note: This parameter configures instance B. It functions identically to BATT_FL_FLTR.

BATTB_FL_FS (Second order term)

DEFAULT: 0	RANGE: 0 10	UNITS: null

Second order term

Note: This parameter configures instance B. It functions identically to BATT_FL_FS.

BATTB_FL_FT (Third order term)

DEFAULT: 0	RANGE: 0 10	UNITS: null

Third order term

Note: This parameter configures instance B. It functions identically to BATT_FL_FT.

BATTB_FL_OFF (Offset term)

DEFAULT: 0	RANGE: 0 10	UNITS: null

Offset term

Note: This parameter configures instance B. It functions identically to BATT_FL_OFF.

BATTB_FL_PIN (Airspeed pin)

DEFAULT: 0	RANGE: null	UNITS: null

Airspeed pin

Note: This parameter configures instance B. It functions identically to BATT_FL_PIN.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





BATTB_FL_VLT_MIN (Empty fuel level voltage)

DEFAULT: 0.5	RANGE: 0.01 10	UNITS: V

Empty fuel level voltage

Note: This parameter configures instance B. It functions identically to BATT_FL_VLT_MIN.

BATTB_FL_V_MULT (Fuel level voltage multiplier)

DEFAULT: 0.5	RANGE: 0.01 10	UNITS: null

Fuel level voltage multiplier

Note: This parameter configures instance B. It functions identically to BATT_FL_V_MULT.

BATTB_FS_CRT_ACT (Critical battery failsafe action)

DEFAULT: 0	RANGE: null	UNITS: null

Critical battery failsafe action

Note: This parameter configures Battery Monitor B. It functions identically to BATT_FS_CRT_ACT.

BATTB_FS_LOW_ACT (Low battery failsafe action)

DEFAULT: 0	RANGE: null	UNITS: null

Low battery failsafe action

Note: This parameter is functionally identical to BATT_FS_LOW_ACT.

Please refer to the documentation for the primary instance for detailed configuration instructions.

BATTB_FS_VOLTSRC (Failsafe voltage source)

DEFAULT: BattMonitor_LowVoltageSource_Raw	RANGE: null	UNITS: null

Failsafe voltage source

Note: This parameter configures instance B. It functions identically to BATT_FS_VOLTSRC.

BATTB_I2C_ADDR (Battery monitor I2C address)

DEFAULT: HAL_BATTMON_INA2XX_ADDR	RANGE: 0 127	UNITS: null

Battery monitor I2C address

Note: This parameter configures Battery Monitor B. It functions identically to BATT_I2C_ADDR.

BATTB_I2C_BUS (Sensor Bus)

DEFAULT: -1	RANGE: null	UNITS: null

Sensor Bus

Note: This parameter configures Battery Monitor B. It functions identically to BATT_I2C_BUS.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





BATTB_LOW_MAH (Low battery capacity)

DEFAULT: 0	RANGE: null	UNITS: mAh

Low battery capacity

Note: This parameter is functionally identical to BATT_LOW_MAH.

Please refer to the documentation for the primary instance for detailed configuration instructions.

BATTB_LOW_TIMER (Low voltage timeout)

DEFAULT: 10	RANGE: 0 120	UNITS: s

Low voltage timeout

Note: This parameter is functionally identical to BATT_LOW_TIMER.

Please refer to the documentation for the primary instance for detailed configuration instructions.

BATTB_LOW_VOLT (Low battery voltage)

DEFAULT: DEFAULT_LOW_BATTERY_VOLTAGE	RANGE: null	UNITS: V

Low battery voltage

Note: This parameter is functionally identical to BATT_LOW_VOLT.

Please refer to the documentation for the primary instance for detailed configuration instructions.

BATTB_MAX_AMPS (Battery monitor max current)

DEFAULT: HAL_BATTMON_INA239_MAX_CURRENT	RANGE: 1 400	UNITS: A

Battery monitor max current

Note: This parameter configures Battery Monitor B. It functions identically to BATT_MAX_AMPS.

BATTB_MAX_VOLT (Maximum Battery Voltage)

DEFAULT: 12.6	RANGE: 7 100	UNITS: null

Maximum Battery Voltage

Note: This parameter configures Battery Monitor B. It functions identically to BATT_MAX_VOLT.

BATTB_MONITOR (Battery monitoring)

DEFAULT: int8_t(AP_BattMonitor::Type::NONE	RANGE: null	UNITS: null

Battery monitoring

Note: This parameter is functionally identical to BATT_MONITOR.

Please refer to the documentation for the primary instance for detailed configuration instructions.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





BATTB_OPTIONS (quadplane options)

DEFAULT: 0	RANGE: null	UNITS: null

quadplane options

Note: This parameter configures Battery Monitor B. It functions identically to BATT_OPTIONS.

BATTB_SERIAL_NUM (Battery serial number)

DEFAULT: AP_BATT_SERIAL_NUMBER_DEFAULT	RANGE: null	UNITS: null

Battery serial number

Note: This parameter configures Battery Monitor B. It functions identically to BATT_SERIAL_NUM.

BATTB_SHUNT (Battery monitor shunt resistor)

DEFAULT: HAL_BATTMON_INA239_SHUNT_RESISTANCE	RANGE: 0.0001 0.01	UNITS: Ohm

Battery monitor shunt resistor

Note: This parameter configures Battery Monitor B. It functions identically to BATT_SHUNT.

BATTB_SUM_MASK (Tiltrotor mask)

DEFAULT: 0	RANGE: null	UNITS: null

Tiltrotor mask

Note: This parameter is functionally identical to BATT_SUM_MASK.

Please refer to the documentation for the primary instance for detailed configuration instructions.

BATTB_VLT_OFFSET (Airspeed offset)

DEFAULT: 0	RANGE: null	UNITS: null

Airspeed offset

Note: This parameter configures Battery Monitor B. It functions identically to BATT_VLT_OFFSET.

BATTB_VOLT_MULT (Voltage Multiplier)

DEFAULT: 0	RANGE: null	UNITS: null

Voltage Multiplier

Note: This parameter is functionally identical to BATT_VOLT_MULT.

Please refer to the documentation for the primary instance for detailed configuration instructions.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





BATTB_VOLT_PIN (Airspeed pin)

DEFAULT: 0	RANGE: null	UNITS: null

Airspeed pin

Note: This parameter is functionally identical to BATT_VOLT_PIN.

Please refer to the documentation for the primary instance for detailed configuration instructions.

BATTB_WATT_MAX (Tiltrotor maximum VTOL angle)

DEFAULT: 45	RANGE: 20 80	UNITS: deg

Tiltrotor maximum VTOL angle

Note: This parameter configures Battery Monitor B. It functions identically to BATT_WATT_MAX.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY

Overview

Key Concepts

BATTC_AMP_OFFSET (Airspeed offset)

DEFAULT: 0	RANGE: null	UNITS: null

Airspeed offset

Note: This parameter is functionally identical to BATT_AMP_OFFSET.

Please refer to the documentation for the primary instance for detailed configuration instructions.

BATTC_AMP_PERVLT (Amps per volt)

DEFAULT: 0	RANGE: null	UNITS: A/V

Amps per volt

Note: This parameter is functionally identical to BATT_AMP_PERVLT.

Please refer to the documentation for the primary instance for detailed configuration instructions.

BATTC_ARM_MAH (Required arming remaining capacity)

DEFAULT: 0	RANGE: null	UNITS: mAh

Required arming remaining capacity

Note: This parameter configures Battery Monitor C. It functions identically to BATT_ARM_MAH.

BATTC_ARM_VOLT (Required arming voltage)

DEFAULT: 0	RANGE: null	UNITS: V

Required arming voltage

Note: This parameter is functionally identical to BATT_ARM_VOLT.

Please refer to the documentation for the primary instance for detailed configuration instructions.

BATTC_CAPACITY (Battery capacity)

DEFAULT: AP_BATT_MONITOR_BATTERY_CAPACITY	RANGE: null	UNITS: mAh

Battery capacity

Note: This parameter is functionally identical to BATT_CAPACITY.

Please refer to the documentation for the primary instance for detailed configuration instructions.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





BATTC_CHANNEL (Receiver RSSI channel number)

DEFAULT: 0	RANGE: 0 16	UNITS: null

Receiver RSSI channel number

Note: This parameter configures instance C. It functions identically to BATT_CHANNEL.

BATTC_CRT_MAH (Battery critical capacity)

DEFAULT: 0	RANGE: null	UNITS: mAh

Battery critical capacity

Note: This parameter is functionally identical to BATT_CRT_MAH.

Please refer to the documentation for the primary instance for detailed configuration instructions.

BATTC_CRT_VOLT (Critical battery voltage)

DEFAULT: 0	RANGE: null	UNITS: V

Critical battery voltage

Note: This parameter is functionally identical to BATT_CRT_VOLT.

Please refer to the documentation for the primary instance for detailed configuration instructions.

BATTC_CURR_MULT (Scales reported power monitor current)

DEFAULT: 1.0	RANGE: .1 10	UNITS: null

Scales reported power monitor current

Note: This parameter configures instance C. It functions identically to BATT_CURR_MULT.

BATTC_CURR_PIN (Airspeed pin)

DEFAULT: 0	RANGE: null	UNITS: null

Airspeed pin

Note: This parameter is functionally identical to BATT_CURR_PIN.

Please refer to the documentation for the primary instance for detailed configuration instructions.

BATTC_ESC_INDEX (ESC Telemetry Index to write to)

DEFAULT: 0	RANGE: 0 10	UNITS: null

ESC Telemetry Index to write to

Note: This parameter configures Battery Monitor C. It functions identically to BATT_ESC_INDEX.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





BATTC_ESC_MASK (Tiltrotor mask)

DEFAULT: 0	RANGE: null	UNITS: null

Tiltrotor mask

Note: This parameter configures Battery Monitor C. It functions identically to BATT_ESC_MASK.

BATTC_FL_FF (FF FeedForward Gain)

DEFAULT: 0	RANGE: null	UNITS: null

FF FeedForward Gain

Note: This parameter configures instance C. It functions identically to BATT_FL_FF.

BATTC_FL_FLTR (Fuel level filter frequency)

DEFAULT: 0.3	RANGE: -1 1	UNITS: Hz

Fuel level filter frequency

Note: This parameter configures instance C. It functions identically to BATT_FL_FLTR.

BATTC_FL_FS (Second order term)

DEFAULT: 0	RANGE: 0 10	UNITS: null

Second order term

Note: This parameter configures instance C. It functions identically to BATT_FL_FS.

BATTC_FL_FT (Third order term)

DEFAULT: 0	RANGE: 0 10	UNITS: null

Third order term

Note: This parameter configures instance C. It functions identically to BATT_FL_FT.

BATTC_FL_OFF (Offset term)

DEFAULT: 0	RANGE: 0 10	UNITS: null

Offset term

Note: This parameter configures instance C. It functions identically to BATT_FL_OFF.

BATTC_FL_PIN (Airspeed pin)

DEFAULT: 0	RANGE: null	UNITS: null

Airspeed pin

Note: This parameter configures instance C. It functions identically to BATT_FL_PIN.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





BATTC_FL_VLT_MIN (Empty fuel level voltage)

DEFAULT: 0.5	RANGE: 0.01 10	UNITS: V

Empty fuel level voltage

Note: This parameter configures instance C. It functions identically to BATT_FL_VLT_MIN.

BATTC_FL_V_MULT (Fuel level voltage multiplier)

DEFAULT: 0.5	RANGE: 0.01 10	UNITS: null

Fuel level voltage multiplier

Note: This parameter configures instance C. It functions identically to BATT_FL_V_MULT.

BATTC_FS_CRT_ACT (Critical battery failsafe action)

DEFAULT: 0	RANGE: null	UNITS: null

Critical battery failsafe action

Note: This parameter configures Battery Monitor C. It functions identically to BATT_FS_CRT_ACT.

BATTC_FS_LOW_ACT (Low battery failsafe action)

DEFAULT: 0	RANGE: null	UNITS: null

Low battery failsafe action

Note: This parameter is functionally identical to BATT_FS_LOW_ACT.

Please refer to the documentation for the primary instance for detailed configuration instructions.

BATTC_FS_VOLTSRC (Failsafe voltage source)

DEFAULT: BattMonitor_LowVoltageSource_Raw	RANGE: null	UNITS: null

Failsafe voltage source

Note: This parameter configures instance C. It functions identically to BATT_FS_VOLTSRC.

BATTC_I2C_ADDR (Battery monitor I2C address)

DEFAULT: HAL_BATTMON_INA2XX_ADDR	RANGE: 0 127	UNITS: null

Battery monitor I2C address

Note: This parameter configures Battery Monitor C. It functions identically to BATT_I2C_ADDR.

BATTC_I2C_BUS (Sensor Bus)

DEFAULT: -1	RANGE: null	UNITS: null

Sensor Bus

Note: This parameter configures Battery Monitor C. It functions identically to BATT_I2C_BUS.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





BATTC_LOW_MAH (Low battery capacity)

DEFAULT: 0	RANGE: null	UNITS: mAh

Low battery capacity

Note: This parameter is functionally identical to BATT_LOW_MAH.

Please refer to the documentation for the primary instance for detailed configuration instructions.

BATTC_LOW_TIMER (Low voltage timeout)

DEFAULT: 10	RANGE: 0 120	UNITS: s

Low voltage timeout

Note: This parameter is functionally identical to BATT_LOW_TIMER.

Please refer to the documentation for the primary instance for detailed configuration instructions.

BATTC_LOW_VOLT (Low battery voltage)

DEFAULT: DEFAULT_LOW_BATTERY_VOLTAGE	RANGE: null	UNITS: V

Low battery voltage

Note: This parameter is functionally identical to BATT_LOW_VOLT.

Please refer to the documentation for the primary instance for detailed configuration instructions.

BATTC_MAX_AMPS (Battery monitor max current)

DEFAULT: HAL_BATTMON_INA239_MAX_CURRENT	RANGE: 1 400	UNITS: A

Battery monitor max current

Note: This parameter configures Battery Monitor C. It functions identically to BATT_MAX_AMPS.

BATTC_MAX_VOLT (Maximum Battery Voltage)

DEFAULT: 12.6	RANGE: 7 100	UNITS: null

Maximum Battery Voltage

Note: This parameter configures Battery Monitor C. It functions identically to BATT_MAX_VOLT.

BATTC_MONITOR (Battery monitoring)

DEFAULT: int8_t(AP_BattMonitor::Type::NONE	RANGE: null	UNITS: null

Battery monitoring

Note: This parameter is functionally identical to BATT_MONITOR.

Please refer to the documentation for the primary instance for detailed configuration instructions.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





BATTC_OPTIONS (quadplane options)

DEFAULT: 0	RANGE: null	UNITS: null

quadplane options

Note: This parameter configures Battery Monitor C. It functions identically to BATT_OPTIONS.

BATTC_SERIAL_NUM (Battery serial number)

DEFAULT: AP_BATT_SERIAL_NUMBER_DEFAULT	RANGE: null	UNITS: null

Battery serial number

Note: This parameter configures Battery Monitor C. It functions identically to BATT_SERIAL_NUM.

BATTC_SHUNT (Battery monitor shunt resistor)

DEFAULT: HAL_BATTMON_INA239_SHUNT_RESISTANCE	RANGE: 0.0001 0.01	UNITS: Ohm

Battery monitor shunt resistor

Note: This parameter configures Battery Monitor C. It functions identically to BATT_SHUNT.

BATTC_SUM_MASK (Tiltrotor mask)

DEFAULT: 0	RANGE: null	UNITS: null

Tiltrotor mask

Note: This parameter is functionally identical to BATT_SUM_MASK.

Please refer to the documentation for the primary instance for detailed configuration instructions.

BATTC_VLT_OFFSET (Airspeed offset)

DEFAULT: 0	RANGE: null	UNITS: null

Airspeed offset

Note: This parameter configures Battery Monitor C. It functions identically to BATT_VLT_OFFSET.

BATTC_VOLT_MULT (Voltage Multiplier)

DEFAULT: 0	RANGE: null	UNITS: null

Voltage Multiplier

Note: This parameter is functionally identical to BATT_VOLT_MULT.

Please refer to the documentation for the primary instance for detailed configuration instructions.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





BATTC_VOLT_PIN (Airspeed pin)

DEFAULT: 0	RANGE: null	UNITS: null

Airspeed pin

Note: This parameter is functionally identical to BATT_VOLT_PIN.

Please refer to the documentation for the primary instance for detailed configuration instructions.

BATTC_WATT_MAX (Tiltrotor maximum VTOL angle)

DEFAULT: 45	RANGE: 20 80	UNITS: deg

Tiltrotor maximum VTOL angle

Note: This parameter configures Battery Monitor C. It functions identically to BATT_WATT_MAX.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY

Overview

Key Concepts

BATTD_AMP_OFFSET (Airspeed offset)

DEFAULT: 0	RANGE: null	UNITS: null

Airspeed offset

Note: This parameter is functionally identical to BATT_AMP_OFFSET.

Please refer to the documentation for the primary instance for detailed configuration instructions.

BATTD_AMP_PERVLT (Amps per volt)

DEFAULT: 0	RANGE: null	UNITS: A/V

Amps per volt

Note: This parameter is functionally identical to BATT_AMP_PERVLT.

Please refer to the documentation for the primary instance for detailed configuration instructions.

BATTD_ARM_MAH (Required arming remaining capacity)

DEFAULT: 0	RANGE: null	UNITS: mAh

Required arming remaining capacity

Note: This parameter configures Battery Monitor D. It functions identically to BATT_ARM_MAH.

BATTD_ARM_VOLT (Required arming voltage)

DEFAULT: 0	RANGE: null	UNITS: V

Required arming voltage

Note: This parameter is functionally identical to BATT_ARM_VOLT.

Please refer to the documentation for the primary instance for detailed configuration instructions.

BATTD_CAPACITY (Battery capacity)

DEFAULT: AP_BATT_MONITOR_BATTERY_CAPACITY	RANGE: null	UNITS: mAh

Battery capacity

Note: This parameter is functionally identical to BATT_CAPACITY.

Please refer to the documentation for the primary instance for detailed configuration instructions.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





BATTD_CHANNEL (Receiver RSSI channel number)

DEFAULT: 0	RANGE: 0 16	UNITS: null

Receiver RSSI channel number

Note: This parameter configures instance D. It functions identically to BATT_CHANNEL.

BATTD_CRT_MAH (Battery critical capacity)

DEFAULT: 0	RANGE: null	UNITS: mAh

Battery critical capacity

Note: This parameter is functionally identical to BATT_CRT_MAH.

Please refer to the documentation for the primary instance for detailed configuration instructions.

BATTD_CRT_VOLT (Critical battery voltage)

DEFAULT: 0	RANGE: null	UNITS: V

Critical battery voltage

Note: This parameter is functionally identical to BATT_CRT_VOLT.

Please refer to the documentation for the primary instance for detailed configuration instructions.

BATTD_CURR_MULT (Scales reported power monitor current)

DEFAULT: 1.0	RANGE: .1 10	UNITS: null

Scales reported power monitor current

Note: This parameter configures instance D. It functions identically to BATT_CURR_MULT.

BATTD_CURR_PIN (Airspeed pin)

DEFAULT: 0	RANGE: null	UNITS: null

Airspeed pin

Note: This parameter is functionally identical to BATT_CURR_PIN.

Please refer to the documentation for the primary instance for detailed configuration instructions.

BATTD_ESC_INDEX (ESC Telemetry Index to write to)

DEFAULT: 0	RANGE: 0 10	UNITS: null

ESC Telemetry Index to write to

Note: This parameter configures Battery Monitor D. It functions identically to BATT_ESC_INDEX.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





BATTD_ESC_MASK (Tiltrotor mask)

DEFAULT: 0	RANGE: null	UNITS: null

Tiltrotor mask

Note: This parameter configures Battery Monitor D. It functions identically to BATT_ESC_MASK.

BATTD_FL_FF (FF FeedForward Gain)

DEFAULT: 0	RANGE: null	UNITS: null

FF FeedForward Gain

Note: This parameter configures instance D. It functions identically to BATT_FL_FF.

BATTD_FL_FLTR (Fuel level filter frequency)

DEFAULT: 0.3	RANGE: -1 1	UNITS: Hz

Fuel level filter frequency

Note: This parameter configures instance D. It functions identically to BATT_FL_FLTR.

BATTD_FL_FS (Second order term)

DEFAULT: 0	RANGE: 0 10	UNITS: null

Second order term

Note: This parameter configures instance D. It functions identically to BATT_FL_FS.

BATTD_FL_FT (Third order term)

DEFAULT: 0	RANGE: 0 10	UNITS: null

Third order term

Note: This parameter configures instance D. It functions identically to BATT_FL_FT.

BATTD_FL_OFF (Offset term)

DEFAULT: 0	RANGE: 0 10	UNITS: null

Offset term

Note: This parameter configures instance D. It functions identically to BATT_FL_OFF.

BATTD_FL_PIN (Airspeed pin)

DEFAULT: 0	RANGE: null	UNITS: null

Airspeed pin

Note: This parameter configures instance D. It functions identically to BATT_FL_PIN.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





BATTD_FL_VLT_MIN (Empty fuel level voltage)

DEFAULT: 0.5	RANGE: 0.01 10	UNITS: V

Empty fuel level voltage

Note: This parameter configures instance D. It functions identically to BATT_FL_VLT_MIN.

BATTD_FL_V_MULT (Fuel level voltage multiplier)

DEFAULT: 0.5	RANGE: 0.01 10	UNITS: null

Fuel level voltage multiplier

Note: This parameter configures instance D. It functions identically to BATT_FL_V_MULT.

BATTD_FS_CRT_ACT (Critical battery failsafe action)

DEFAULT: 0	RANGE: null	UNITS: null

Critical battery failsafe action

Note: This parameter configures Battery Monitor D. It functions identically to BATT_FS_CRT_ACT.

BATTD_FS_LOW_ACT (Low battery failsafe action)

DEFAULT: 0	RANGE: null	UNITS: null

Low battery failsafe action

Note: This parameter is functionally identical to BATT_FS_LOW_ACT.

Please refer to the documentation for the primary instance for detailed configuration instructions.

BATTD_FS_VOLTSRC (Failsafe voltage source)

DEFAULT: BattMonitor_LowVoltageSource_Raw	RANGE: null	UNITS: null

Failsafe voltage source

Note: This parameter configures instance D. It functions identically to BATT_FS_VOLTSRC.

BATTD_I2C_ADDR (Battery monitor I2C address)

DEFAULT: HAL_BATTMON_INA2XX_ADDR	RANGE: 0 127	UNITS: null

Battery monitor I2C address

Note: This parameter configures Battery Monitor D. It functions identically to BATT_I2C_ADDR.

BATTD_I2C_BUS (Sensor Bus)

DEFAULT: -1	RANGE: null	UNITS: null

Sensor Bus

Note: This parameter configures Battery Monitor D. It functions identically to BATT_I2C_BUS.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





BATTD_LOW_MAH (Low battery capacity)

DEFAULT: 0	RANGE: null	UNITS: mAh

Low battery capacity

Note: This parameter is functionally identical to BATT_LOW_MAH.

Please refer to the documentation for the primary instance for detailed configuration instructions.

BATTD_LOW_TIMER (Low voltage timeout)

DEFAULT: 10	RANGE: 0 120	UNITS: s

Low voltage timeout

Note: This parameter is functionally identical to BATT_LOW_TIMER.

Please refer to the documentation for the primary instance for detailed configuration instructions.

BATTD_LOW_VOLT (Low battery voltage)

DEFAULT: DEFAULT_LOW_BATTERY_VOLTAGE	RANGE: null	UNITS: V

Low battery voltage

Note: This parameter is functionally identical to BATT_LOW_VOLT.

Please refer to the documentation for the primary instance for detailed configuration instructions.

BATTD_MAX_AMPS (Battery monitor max current)

DEFAULT: HAL_BATTMON_INA239_MAX_CURRENT	RANGE: 1 400	UNITS: A

Battery monitor max current

Note: This parameter configures Battery Monitor D. It functions identically to BATT_MAX_AMPS.

BATTD_MAX_VOLT (Maximum Battery Voltage)

DEFAULT: 12.6	RANGE: 7 100	UNITS: null

Maximum Battery Voltage

Note: This parameter configures Battery Monitor D. It functions identically to BATT_MAX_VOLT.

BATTD_MONITOR (Battery monitoring)

DEFAULT: int8_t(AP_BattMonitor::Type::NONE	RANGE: null	UNITS: null

Battery monitoring

Note: This parameter is functionally identical to BATT_MONITOR.

Please refer to the documentation for the primary instance for detailed configuration instructions.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





BATTD_OPTIONS (quadplane options)

DEFAULT: 0	RANGE: null	UNITS: null

quadplane options

Note: This parameter configures Battery Monitor D. It functions identically to BATT_OPTIONS.

BATTD_SERIAL_NUM (Battery serial number)

DEFAULT: AP_BATT_SERIAL_NUMBER_DEFAULT	RANGE: null	UNITS: null

Battery serial number

Note: This parameter configures Battery Monitor D. It functions identically to BATT_SERIAL_NUM.

BATTD_SHUNT (Battery monitor shunt resistor)

DEFAULT: HAL_BATTMON_INA239_SHUNT_RESISTANCE	RANGE: 0.0001 0.01	UNITS: Ohm

Battery monitor shunt resistor

Note: This parameter configures Battery Monitor D. It functions identically to BATT_SHUNT.

BATTD_SUM_MASK (Tiltrotor mask)

DEFAULT: 0	RANGE: null	UNITS: null

Tiltrotor mask

Note: This parameter is functionally identical to BATT_SUM_MASK.

Please refer to the documentation for the primary instance for detailed configuration instructions.

BATTD_VLT_OFFSET (Airspeed offset)

DEFAULT: 0	RANGE: null	UNITS: null

Airspeed offset

Note: This parameter configures Battery Monitor D. It functions identically to BATT_VLT_OFFSET.

BATTD_VOLT_MULT (Voltage Multiplier)

DEFAULT: 0	RANGE: null	UNITS: null

Voltage Multiplier

Note: This parameter is functionally identical to BATT_VOLT_MULT.

Please refer to the documentation for the primary instance for detailed configuration instructions.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





BATTD_VOLT_PIN (Airspeed pin)

DEFAULT: 0	RANGE: null	UNITS: null

Airspeed pin

Note: This parameter is functionally identical to BATT_VOLT_PIN.

Please refer to the documentation for the primary instance for detailed configuration instructions.

BATTD_WATT_MAX (Tiltrotor maximum VTOL angle)

DEFAULT: 45	RANGE: 20 80	UNITS: deg

Tiltrotor maximum VTOL angle

Note: This parameter configures Battery Monitor D. It functions identically to BATT_WATT_MAX.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

Key Concepts

BATTE_AMP_OFFSET (Airspeed offset)

DEFAULT: 0	RANGE: null	UNITS: null

Airspeed offset

Note: This parameter is functionally identical to BATT_AMP_OFFSET.

Please refer to the documentation for the primary instance for detailed configuration instructions.

BATTE_AMP_PERVLT (Amps per volt)

DEFAULT: 0	RANGE: null	UNITS: A/V

Amps per volt

Note: This parameter is functionally identical to BATT_AMP_PERVLT.

Please refer to the documentation for the primary instance for detailed configuration instructions.

BATTE_ARM_MAH (Required arming remaining capacity)

DEFAULT: 0	RANGE: null	UNITS: mAh

Required arming remaining capacity

Note: This parameter configures Battery Monitor E. It functions identically to BATT_ARM_MAH.

BATTE_ARM_VOLT (Required arming voltage)

DEFAULT: 0	RANGE: null	UNITS: V

Required arming voltage

Note: This parameter is functionally identical to BATT_ARM_VOLT.

Please refer to the documentation for the primary instance for detailed configuration instructions.

BATTE_CAPACITY (Battery capacity)

DEFAULT: AP_BATT_MONITOR_BATTERY_CAPACITY	RANGE: null	UNITS: mAh

Battery capacity

Note: This parameter is functionally identical to BATT_CAPACITY.

Please refer to the documentation for the primary instance for detailed configuration instructions.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





BATTE_CHANNEL (Receiver RSSI channel number)

DEFAULT: 0	RANGE: 0 16	UNITS: null

Receiver RSSI channel number

Note: This parameter configures instance E. It functions identically to BATT_CHANNEL.

BATTE_CRT_MAH (Battery critical capacity)

DEFAULT: 0	RANGE: null	UNITS: mAh

Battery critical capacity

Note: This parameter is functionally identical to BATT_CRT_MAH.

Please refer to the documentation for the primary instance for detailed configuration instructions.

BATTE_CRT_VOLT (Critical battery voltage)

DEFAULT: 0	RANGE: null	UNITS: V

Critical battery voltage

Note: This parameter is functionally identical to BATT_CRT_VOLT.

Please refer to the documentation for the primary instance for detailed configuration instructions.

BATTE_CURR_MULT (Scales reported power monitor current)

DEFAULT: 1.0	RANGE: .1 10	UNITS: null

Scales reported power monitor current

Note: This parameter configures instance E. It functions identically to BATT_CURR_MULT.

BATTE_CURR_PIN (Airspeed pin)

DEFAULT: 0	RANGE: null	UNITS: null

Airspeed pin

Note: This parameter is functionally identical to BATT_CURR_PIN.

Please refer to the documentation for the primary instance for detailed configuration instructions.

BATTE_ESC_INDEX (ESC Telemetry Index to write to)

DEFAULT: 0	RANGE: 0 10	UNITS: null

ESC Telemetry Index to write to

Note: This parameter configures Battery Monitor E. It functions identically to BATT_ESC_INDEX.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





BATTE_ESC_MASK (Tiltrotor mask)

DEFAULT: 0	RANGE: null	UNITS: null

Tiltrotor mask

Note: This parameter configures Battery Monitor E. It functions identically to BATT_ESC_MASK.

BATTE_FL_FF (FF FeedForward Gain)

DEFAULT: 0	RANGE: null	UNITS: null

FF FeedForward Gain

Note: This parameter configures instance E. It functions identically to BATT_FL_FF.

BATTE_FL_FLTR (Fuel level filter frequency)

DEFAULT: 0.3	RANGE: -1 1	UNITS: Hz

Fuel level filter frequency

Note: This parameter configures instance E. It functions identically to BATT_FL_FLTR.

BATTE_FL_FS (Second order term)

DEFAULT: 0	RANGE: 0 10	UNITS: null

Second order term

Note: This parameter configures instance E. It functions identically to BATT_FL_FS.

BATTE_FL_FT (Third order term)

DEFAULT: 0	RANGE: 0 10	UNITS: null

Third order term

Note: This parameter configures instance E. It functions identically to BATT_FL_FT.

BATTE_FL_OFF (Offset term)

DEFAULT: 0	RANGE: 0 10	UNITS: null

Offset term

Note: This parameter configures instance E. It functions identically to BATT_FL_OFF.

BATTE_FL_PIN (Airspeed pin)

DEFAULT: 0	RANGE: null	UNITS: null

Airspeed pin

Note: This parameter configures instance E. It functions identically to BATT_FL_PIN.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





BATTE_FL_VLT_MIN (Empty fuel level voltage)

DEFAULT: 0.5	RANGE: 0.01 10	UNITS: V

Empty fuel level voltage

Note: This parameter configures instance E. It functions identically to BATT_FL_VLT_MIN.

BATTE_FL_V_MULT (Fuel level voltage multiplier)

DEFAULT: 0.5	RANGE: 0.01 10	UNITS: null

Fuel level voltage multiplier

Note: This parameter configures instance E. It functions identically to BATT_FL_V_MULT.

BATTE_FS_CRT_ACT (Critical battery failsafe action)

DEFAULT: 0	RANGE: null	UNITS: null

Critical battery failsafe action

Note: This parameter configures Battery Monitor E. It functions identically to BATT_FS_CRT_ACT.

BATTE_FS_LOW_ACT (Low battery failsafe action)

DEFAULT: 0	RANGE: null	UNITS: null

Low battery failsafe action

Note: This parameter is functionally identical to BATT_FS_LOW_ACT.

Please refer to the documentation for the primary instance for detailed configuration instructions.

BATTE_FS_VOLTSRC (Failsafe voltage source)

DEFAULT: BattMonitor_LowVoltageSource_Raw	RANGE: null	UNITS: null

Failsafe voltage source

Note: This parameter configures instance E. It functions identically to BATT_FS_VOLTSRC.

BATTE_I2C_ADDR (Battery monitor I2C address)

DEFAULT: HAL_BATTMON_INA2XX_ADDR	RANGE: 0 127	UNITS: null

Battery monitor I2C address

Note: This parameter configures Battery Monitor E. It functions identically to BATT_I2C_ADDR.

BATTE_I2C_BUS (Sensor Bus)

DEFAULT: -1	RANGE: null	UNITS: null

Sensor Bus

Note: This parameter configures Battery Monitor E. It functions identically to BATT_I2C_BUS.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





BATTE_LOW_MAH (Low battery capacity)

DEFAULT: 0	RANGE: null	UNITS: mAh

Low battery capacity

Note: This parameter is functionally identical to BATT_LOW_MAH.

Please refer to the documentation for the primary instance for detailed configuration instructions.

BATTE_LOW_TIMER (Low voltage timeout)

DEFAULT: 10	RANGE: 0 120	UNITS: s

Low voltage timeout

Note: This parameter is functionally identical to BATT_LOW_TIMER.

Please refer to the documentation for the primary instance for detailed configuration instructions.

BATTE_LOW_VOLT (Low battery voltage)

DEFAULT: DEFAULT_LOW_BATTERY_VOLTAGE	RANGE: null	UNITS: V

Low battery voltage

Note: This parameter is functionally identical to BATT_LOW_VOLT.

Please refer to the documentation for the primary instance for detailed configuration instructions.

BATTE_MAX_AMPS (Battery monitor max current)

DEFAULT: HAL_BATTMON_INA239_MAX_CURRENT	RANGE: 1 400	UNITS: A

Battery monitor max current

Note: This parameter configures Battery Monitor E. It functions identically to BATT_MAX_AMPS.

BATTE_MAX_VOLT (Maximum Battery Voltage)

DEFAULT: 12.6	RANGE: 7 100	UNITS: null

Maximum Battery Voltage

Note: This parameter configures Battery Monitor E. It functions identically to BATT_MAX_VOLT.

BATTE_MONITOR (Battery monitoring)

DEFAULT: int8_t(AP_BattMonitor::Type::NONE	RANGE: null	UNITS: null

Battery monitoring

Note: This parameter is functionally identical to BATT_MONITOR.

Please refer to the documentation for the primary instance for detailed configuration instructions.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





BATTE_OPTIONS (quadplane options)

DEFAULT: 0	RANGE: null	UNITS: null

quadplane options

Note: This parameter configures Battery Monitor E. It functions identically to BATT_OPTIONS.

BATTE_SERIAL_NUM (Battery serial number)

DEFAULT: AP_BATT_SERIAL_NUMBER_DEFAULT	RANGE: null	UNITS: null

Battery serial number

Note: This parameter configures Battery Monitor E. It functions identically to BATT_SERIAL_NUM.

BATTE_SHUNT (Battery monitor shunt resistor)

DEFAULT: HAL_BATTMON_INA239_SHUNT_RESISTANCE	RANGE: 0.0001 0.01	UNITS: Ohm

Battery monitor shunt resistor

Note: This parameter configures Battery Monitor E. It functions identically to BATT_SHUNT.

BATTE_SUM_MASK (Tiltrotor mask)

DEFAULT: 0	RANGE: null	UNITS: null

Tiltrotor mask

Note: This parameter is functionally identical to BATT_SUM_MASK.

Please refer to the documentation for the primary instance for detailed configuration instructions.

BATTE_VLT_OFFSET (Airspeed offset)

DEFAULT: 0	RANGE: null	UNITS: null

Airspeed offset

Note: This parameter configures Battery Monitor E. It functions identically to BATT_VLT_OFFSET.

BATTE_VOLT_MULT (Voltage Multiplier)

DEFAULT: 0	RANGE: null	UNITS: null

Voltage Multiplier

Note: This parameter is functionally identical to BATT_VOLT_MULT.

Please refer to the documentation for the primary instance for detailed configuration instructions.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





BATTE_VOLT_PIN (Airspeed pin)

DEFAULT: 0	RANGE: null	UNITS: null

Airspeed pin

Note: This parameter is functionally identical to BATT_VOLT_PIN.

Please refer to the documentation for the primary instance for detailed configuration instructions.

BATTE_WATT_MAX (Tiltrotor maximum VTOL angle)

DEFAULT: 45	RANGE: 20 80	UNITS: deg

Tiltrotor maximum VTOL angle

Note: This parameter configures Battery Monitor E. It functions identically to BATT_WATT_MAX.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

Key Concepts

BATTF_AMP_OFFSET (Airspeed offset)

DEFAULT: 0	RANGE: null	UNITS: null

Airspeed offset

Note: This parameter is functionally identical to BATT_AMP_OFFSET.

Please refer to the documentation for the primary instance for detailed configuration instructions.

BATTF_AMP_PERVLT (Amps per volt)

DEFAULT: 0	RANGE: null	UNITS: A/V

Amps per volt

Note: This parameter is functionally identical to BATT_AMP_PERVLT.

Please refer to the documentation for the primary instance for detailed configuration instructions.

BATTF_ARM_MAH (Required arming remaining capacity)

DEFAULT: 0	RANGE: null	UNITS: mAh

Required arming remaining capacity

Note: This parameter configures Battery Monitor F. It functions identically to BATT_ARM_MAH.

BATTF_ARM_VOLT (Required arming voltage)

DEFAULT: 0	RANGE: null	UNITS: V

Required arming voltage

Note: This parameter is functionally identical to BATT_ARM_VOLT.

Please refer to the documentation for the primary instance for detailed configuration instructions.

BATTF_CAPACITY (Battery capacity)

DEFAULT: AP_BATT_MONITOR_BATTERY_CAPACITY	RANGE: null	UNITS: mAh

Battery capacity

Note: This parameter is functionally identical to BATT_CAPACITY.

Please refer to the documentation for the primary instance for detailed configuration instructions.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





BATTF_CHANNEL (Receiver RSSI channel number)

DEFAULT: 0	RANGE: 0 16	UNITS: null

Receiver RSSI channel number

Note: This parameter configures instance F. It functions identically to BATT_CHANNEL.

BATTF_CRT_MAH (Battery critical capacity)

DEFAULT: 0	RANGE: null	UNITS: mAh

Battery critical capacity

Note: This parameter is functionally identical to BATT_CRT_MAH.

Please refer to the documentation for the primary instance for detailed configuration instructions.

BATTF_CRT_VOLT (Critical battery voltage)

DEFAULT: 0	RANGE: null	UNITS: V

Critical battery voltage

Note: This parameter is functionally identical to BATT_CRT_VOLT.

Please refer to the documentation for the primary instance for detailed configuration instructions.

BATTF_CURR_MULT (Scales reported power monitor current)

DEFAULT: 1.0	RANGE: .1 10	UNITS: null

Scales reported power monitor current

Note: This parameter configures instance F. It functions identically to BATT_CURR_MULT.

BATTF_CURR_PIN (Airspeed pin)

DEFAULT: 0	RANGE: null	UNITS: null

Airspeed pin

Note: This parameter is functionally identical to BATT_CURR_PIN.

Please refer to the documentation for the primary instance for detailed configuration instructions.

BATTF_ESC_INDEX (ESC Telemetry Index to write to)

DEFAULT: 0	RANGE: 0 10	UNITS: null

ESC Telemetry Index to write to

Note: This parameter configures Battery Monitor F. It functions identically to BATT_ESC_INDEX.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





BATTF_ESC_MASK (Tiltrotor mask)

DEFAULT: 0	RANGE: null	UNITS: null

Tiltrotor mask

Note: This parameter configures Battery Monitor F. It functions identically to BATT_ESC_MASK.

BATTF_FL_FF (FF FeedForward Gain)

DEFAULT: 0	RANGE: null	UNITS: null

FF FeedForward Gain

Note: This parameter configures instance F. It functions identically to BATT_FL_FF.

BATTF_FL_FLTR (Fuel level filter frequency)

DEFAULT: 0.3	RANGE: -1 1	UNITS: Hz

Fuel level filter frequency

Note: This parameter configures instance F. It functions identically to BATT_FL_FLTR.

BATTF_FL_FS (Second order term)

DEFAULT: 0	RANGE: 0 10	UNITS: null

Second order term

Note: This parameter configures instance F. It functions identically to BATT_FL_FS.

BATTF_FL_FT (Third order term)

DEFAULT: 0	RANGE: 0 10	UNITS: null

Third order term

Note: This parameter configures instance F. It functions identically to BATT_FL_FT.

BATTF_FL_OFF (Offset term)

DEFAULT: 0	RANGE: 0 10	UNITS: null

Offset term

Note: This parameter configures instance F. It functions identically to BATT_FL_OFF.

BATTF_FL_PIN (Airspeed pin)

DEFAULT: 0	RANGE: null	UNITS: null

Airspeed pin

Note: This parameter configures instance F. It functions identically to BATT_FL_PIN.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





BATTF_FL_VLT_MIN (Empty fuel level voltage)

DEFAULT: 0.5	RANGE: 0.01 10	UNITS: V

Empty fuel level voltage

Note: This parameter configures instance F. It functions identically to BATT_FL_VLT_MIN.

BATTF_FL_V_MULT (Fuel level voltage multiplier)

DEFAULT: 0.5	RANGE: 0.01 10	UNITS: null

Fuel level voltage multiplier

Note: This parameter configures instance F. It functions identically to BATT_FL_V_MULT.

BATTF_FS_CRT_ACT (Critical battery failsafe action)

DEFAULT: 0	RANGE: null	UNITS: null

Critical battery failsafe action

Note: This parameter configures Battery Monitor F. It functions identically to BATT_FS_CRT_ACT.

BATTF_FS_LOW_ACT (Low battery failsafe action)

DEFAULT: 0	RANGE: null	UNITS: null

Low battery failsafe action

Note: This parameter is functionally identical to BATT_FS_LOW_ACT.

Please refer to the documentation for the primary instance for detailed configuration instructions.

BATTF_FS_VOLTSRC (Failsafe voltage source)

DEFAULT: BattMonitor_LowVoltageSource_Raw	RANGE: null	UNITS: null

Failsafe voltage source

Note: This parameter configures instance F. It functions identically to BATT_FS_VOLTSRC.

BATTF_I2C_ADDR (Battery monitor I2C address)

DEFAULT: HAL_BATTMON_INA2XX_ADDR	RANGE: 0 127	UNITS: null

Battery monitor I2C address

Note: This parameter configures Battery Monitor F. It functions identically to BATT_I2C_ADDR.

BATTF_I2C_BUS (Sensor Bus)

DEFAULT: -1	RANGE: null	UNITS: null

Sensor Bus

Note: This parameter configures Battery Monitor F. It functions identically to BATT_I2C_BUS.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





BATTF_LOW_MAH (Low battery capacity)

DEFAULT: 0	RANGE: null	UNITS: mAh

Low battery capacity

Note: This parameter is functionally identical to BATT_LOW_MAH.

Please refer to the documentation for the primary instance for detailed configuration instructions.

BATTF_LOW_TIMER (Low voltage timeout)

DEFAULT: 10	RANGE: 0 120	UNITS: s

Low voltage timeout

Note: This parameter is functionally identical to BATT_LOW_TIMER.

Please refer to the documentation for the primary instance for detailed configuration instructions.

BATTF_LOW_VOLT (Low battery voltage)

DEFAULT: DEFAULT_LOW_BATTERY_VOLTAGE	RANGE: null	UNITS: V

Low battery voltage

Note: This parameter is functionally identical to BATT_LOW_VOLT.

Please refer to the documentation for the primary instance for detailed configuration instructions.

BATTF_MAX_AMPS (Battery monitor max current)

DEFAULT: HAL_BATTMON_INA239_MAX_CURRENT	RANGE: 1 400	UNITS: A

Battery monitor max current

Note: This parameter configures Battery Monitor F. It functions identically to BATT_MAX_AMPS.

BATTF_MAX_VOLT (Maximum Battery Voltage)

DEFAULT: 12.6	RANGE: 7 100	UNITS: null

Maximum Battery Voltage

Note: This parameter configures Battery Monitor F. It functions identically to BATT_MAX_VOLT.

BATTF_MONITOR (Battery monitoring)

DEFAULT: int8_t(AP_BattMonitor::Type::NONE	RANGE: null	UNITS: null

Battery monitoring

Note: This parameter is functionally identical to BATT_MONITOR.

Please refer to the documentation for the primary instance for detailed configuration instructions.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





BATTF_OPTIONS (quadplane options)

DEFAULT: 0	RANGE: null	UNITS: null

quadplane options

Note: This parameter configures Battery Monitor F. It functions identically to BATT_OPTIONS.

BATTF_SERIAL_NUM (Battery serial number)

DEFAULT: AP_BATT_SERIAL_NUMBER_DEFAULT	RANGE: null	UNITS: null

Battery serial number

Note: This parameter configures Battery Monitor F. It functions identically to BATT_SERIAL_NUM.

BATTF_SHUNT (Battery monitor shunt resistor)

DEFAULT: HAL_BATTMON_INA239_SHUNT_RESISTANCE	RANGE: 0.0001 0.01	UNITS: Ohm

Battery monitor shunt resistor

Note: This parameter configures Battery Monitor F. It functions identically to BATT_SHUNT.

BATTF_SUM_MASK (Tiltrotor mask)

DEFAULT: 0	RANGE: null	UNITS: null

Tiltrotor mask

Note: This parameter is functionally identical to BATT_SUM_MASK.

Please refer to the documentation for the primary instance for detailed configuration instructions.

BATTF_VLT_OFFSET (Airspeed offset)

DEFAULT: 0	RANGE: null	UNITS: null

Airspeed offset

Note: This parameter configures Battery Monitor F. It functions identically to BATT_VLT_OFFSET.

BATTF_VOLT_MULT (Voltage Multiplier)

DEFAULT: 0	RANGE: null	UNITS: null

Voltage Multiplier

Note: This parameter is functionally identical to BATT_VOLT_MULT.

Please refer to the documentation for the primary instance for detailed configuration instructions.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





BATTF_VOLT_PIN (Airspeed pin)

DEFAULT: 0	RANGE: null	UNITS: null

Airspeed pin

Note: This parameter is functionally identical to BATT_VOLT_PIN.

Please refer to the documentation for the primary instance for detailed configuration instructions.

BATTF_WATT_MAX (Tiltrotor maximum VTOL angle)

DEFAULT: 45	RANGE: 20 80	UNITS: deg

Tiltrotor maximum VTOL angle

Note: This parameter configures Battery Monitor F. It functions identically to BATT_WATT_MAX.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

Key Concepts

BATTG_AMP_OFFSET (Airspeed offset)

DEFAULT: 0	RANGE: null	UNITS: null

Airspeed offset

Note: This parameter is functionally identical to BATT_AMP_OFFSET.

Please refer to the documentation for the primary instance for detailed configuration instructions.

BATTG_AMP_PERVLT (Amps per volt)

DEFAULT: 0	RANGE: null	UNITS: A/V

Amps per volt

Note: This parameter is functionally identical to BATT_AMP_PERVLT.

Please refer to the documentation for the primary instance for detailed configuration instructions.

BATTG_ARM_MAH (Required arming remaining capacity)

DEFAULT: 0	RANGE: null	UNITS: mAh

Required arming remaining capacity

Note: This parameter configures Battery Monitor G. It functions identically to BATT_ARM_MAH.

BATTG_ARM_VOLT (Required arming voltage)

DEFAULT: 0	RANGE: null	UNITS: V

Required arming voltage

Note: This parameter is functionally identical to BATT_ARM_VOLT.

Please refer to the documentation for the primary instance for detailed configuration instructions.

BATTG_CAPACITY (Battery capacity)

DEFAULT: AP_BATT_MONITOR_BATTERY_CAPACITY	RANGE: null	UNITS: mAh

Battery capacity

Note: This parameter is functionally identical to BATT_CAPACITY.

Please refer to the documentation for the primary instance for detailed configuration instructions.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





BATTG_CHANNEL (Receiver RSSI channel number)

DEFAULT: 0	RANGE: 0 16	UNITS: null

Receiver RSSI channel number

Note: This parameter configures instance G. It functions identically to BATT_CHANNEL.

BATTG_CRT_MAH (Battery critical capacity)

DEFAULT: 0	RANGE: null	UNITS: mAh

Battery critical capacity

Note: This parameter is functionally identical to BATT_CRT_MAH.

Please refer to the documentation for the primary instance for detailed configuration instructions.

BATTG_CRT_VOLT (Critical battery voltage)

DEFAULT: 0	RANGE: null	UNITS: V

Critical battery voltage

Note: This parameter is functionally identical to BATT_CRT_VOLT.

Please refer to the documentation for the primary instance for detailed configuration instructions.

BATTG_CURR_MULT (Scales reported power monitor current)

DEFAULT: 1.0	RANGE: .1 10	UNITS: null

Scales reported power monitor current

Note: This parameter configures instance G. It functions identically to BATT_CURR_MULT.

BATTG_CURR_PIN (Airspeed pin)

DEFAULT: 0	RANGE: null	UNITS: null

Airspeed pin

Note: This parameter is functionally identical to BATT_CURR_PIN.

Please refer to the documentation for the primary instance for detailed configuration instructions.

BATTG_ESC_INDEX (ESC Telemetry Index to write to)

DEFAULT: 0	RANGE: 0 10	UNITS: null

ESC Telemetry Index to write to

Note: This parameter configures Battery Monitor G. It functions identically to BATT_ESC_INDEX.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





BATTG_ESC_MASK (Tiltrotor mask)

DEFAULT: 0	RANGE: null	UNITS: null

Tiltrotor mask

Note: This parameter configures Battery Monitor G. It functions identically to BATT_ESC_MASK.

BATTG_FL_FF (FF FeedForward Gain)

DEFAULT: 0	RANGE: null	UNITS: null

FF FeedForward Gain

Note: This parameter configures instance G. It functions identically to BATT_FL_FF.

BATTG_FL_FLTR (Fuel level filter frequency)

DEFAULT: 0.3	RANGE: -1 1	UNITS: Hz

Fuel level filter frequency

Note: This parameter configures instance G. It functions identically to BATT_FL_FLTR.

BATTG_FL_FS (Second order term)

DEFAULT: 0	RANGE: 0 10	UNITS: null

Second order term

Note: This parameter configures instance G. It functions identically to BATT_FL_FS.

BATTG_FL_FT (Third order term)

DEFAULT: 0	RANGE: 0 10	UNITS: null

Third order term

Note: This parameter configures instance G. It functions identically to BATT_FL_FT.

BATTG_FL_OFF (Offset term)

DEFAULT: 0	RANGE: 0 10	UNITS: null

Offset term

Note: This parameter configures instance G. It functions identically to BATT_FL_OFF.

BATTG_FL_PIN (Airspeed pin)

DEFAULT: 0	RANGE: null	UNITS: null

Airspeed pin

Note: This parameter configures instance G. It functions identically to BATT_FL_PIN.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





BATTG_FL_VLT_MIN (Empty fuel level voltage)

DEFAULT: 0.5	RANGE: 0.01 10	UNITS: V

Empty fuel level voltage

Note: This parameter configures instance G. It functions identically to BATT_FL_VLT_MIN.

BATTG_FL_V_MULT (Fuel level voltage multiplier)

DEFAULT: 0.5	RANGE: 0.01 10	UNITS: null

Fuel level voltage multiplier

Note: This parameter configures instance G. It functions identically to BATT_FL_V_MULT.

BATTG_FS_CRT_ACT (Critical battery failsafe action)

DEFAULT: 0	RANGE: null	UNITS: null

Critical battery failsafe action

Note: This parameter configures Battery Monitor G. It functions identically to BATT_FS_CRT_ACT.

BATTG_FS_LOW_ACT (Low battery failsafe action)

DEFAULT: 0	RANGE: null	UNITS: null

Low battery failsafe action

Note: This parameter is functionally identical to BATT_FS_LOW_ACT.

Please refer to the documentation for the primary instance for detailed configuration instructions.

BATTG_FS_VOLTSRC (Failsafe voltage source)

DEFAULT: BattMonitor_LowVoltageSource_Raw	RANGE: null	UNITS: null

Failsafe voltage source

Note: This parameter configures instance G. It functions identically to BATT_FS_VOLTSRC.

BATTG_I2C_ADDR (Battery monitor I2C address)

DEFAULT: HAL_BATTMON_INA2XX_ADDR	RANGE: 0 127	UNITS: null

Battery monitor I2C address

Note: This parameter configures Battery Monitor G. It functions identically to BATT_I2C_ADDR.

BATTG_I2C_BUS (Sensor Bus)

DEFAULT: -1	RANGE: null	UNITS: null

Sensor Bus

Note: This parameter configures Battery Monitor G. It functions identically to BATT_I2C_BUS.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





BATTG_LOW_MAH (Low battery capacity)

DEFAULT: 0	RANGE: null	UNITS: mAh

Low battery capacity

Note: This parameter is functionally identical to BATT_LOW_MAH.

Please refer to the documentation for the primary instance for detailed configuration instructions.

BATTG_LOW_TIMER (Low voltage timeout)

DEFAULT: 10	RANGE: 0 120	UNITS: s

Low voltage timeout

Note: This parameter is functionally identical to BATT_LOW_TIMER.

Please refer to the documentation for the primary instance for detailed configuration instructions.

BATTG_LOW_VOLT (Low battery voltage)

DEFAULT: DEFAULT_LOW_BATTERY_VOLTAGE	RANGE: null	UNITS: V

Low battery voltage

Note: This parameter is functionally identical to BATT_LOW_VOLT.

Please refer to the documentation for the primary instance for detailed configuration instructions.

BATTG_MAX_AMPS (Battery monitor max current)

DEFAULT: HAL_BATTMON_INA239_MAX_CURRENT	RANGE: 1 400	UNITS: A

Battery monitor max current

Note: This parameter configures Battery Monitor G. It functions identically to BATT_MAX_AMPS.

BATTG_MAX_VOLT (Maximum Battery Voltage)

DEFAULT: 12.6	RANGE: 7 100	UNITS: null

Maximum Battery Voltage

Note: This parameter configures Battery Monitor G. It functions identically to BATT_MAX_VOLT.

BATTG_MONITOR (Battery monitoring)

DEFAULT: int8_t(AP_BattMonitor::Type::NONE	RANGE: null	UNITS: null

Battery monitoring

Note: This parameter is functionally identical to BATT_MONITOR.

Please refer to the documentation for the primary instance for detailed configuration instructions.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





BATTG_OPTIONS (quadplane options)

DEFAULT: 0	RANGE: null	UNITS: null

quadplane options

Note: This parameter configures Battery Monitor G. It functions identically to BATT_OPTIONS.

BATTG_SERIAL_NUM (Battery serial number)

DEFAULT: AP_BATT_SERIAL_NUMBER_DEFAULT	RANGE: null	UNITS: null

Battery serial number

Note: This parameter configures Battery Monitor G. It functions identically to BATT_SERIAL_NUM.

BATTG_SHUNT (Battery monitor shunt resistor)

DEFAULT: HAL_BATTMON_INA239_SHUNT_RESISTANCE	RANGE: 0.0001 0.01	UNITS: Ohm

Battery monitor shunt resistor

Note: This parameter configures Battery Monitor G. It functions identically to BATT_SHUNT.

BATTG_SUM_MASK (Tiltrotor mask)

DEFAULT: 0	RANGE: null	UNITS: null

Tiltrotor mask

Note: This parameter is functionally identical to BATT_SUM_MASK.

Please refer to the documentation for the primary instance for detailed configuration instructions.

BATTG_VLT_OFFSET (Airspeed offset)

DEFAULT: 0	RANGE: null	UNITS: null

Airspeed offset

Note: This parameter configures Battery Monitor G. It functions identically to BATT_VLT_OFFSET.

BATTG_VOLT_MULT (Voltage Multiplier)

DEFAULT: 0	RANGE: null	UNITS: null

Voltage Multiplier

Note: This parameter is functionally identical to BATT_VOLT_MULT.

Please refer to the documentation for the primary instance for detailed configuration instructions.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





BATTG_VOLT_PIN (Airspeed pin)

DEFAULT: 0	RANGE: null	UNITS: null

Airspeed pin

Note: This parameter is functionally identical to BATT_VOLT_PIN.

Please refer to the documentation for the primary instance for detailed configuration instructions.

BATTG_WATT_MAX (Tiltrotor maximum VTOL angle)

DEFAULT: 45	RANGE: 20 80	UNITS: deg

Tiltrotor maximum VTOL angle

Note: This parameter configures Battery Monitor G. It functions identically to BATT_WATT_MAX.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





BEACON POSITIONING (BCN)

Overview

The **BCN** parameter group configures the **Indoor Beacon** positioning system. This driver allows ArduPilot to use range-based beacon systems (like Pozyx, Marvelmind, or Nooploop) as a substitute for GPS in indoor or GPS-denied environments.

The EKF (State Estimator) fuses distance measurements from the vehicle to known fixed beacons to calculate position (`PosHold`, `Loiter`, `Auto`).

Key Concepts

1. Beacon Origin

Unlike GPS, beacons operate in a local coordinate frame (X, Y, Z relative to a "Master" beacon). To fly missions, you often need to anchor this local frame to a global Geodetic coordinate (Lat/Lon).

- `BCN_LATITUDE` / `BCN_LONGITUDE` / `BCN_ALT`: Sets the global position of the system origin (usually the 0,0,0 point of the beacon network).

2. Alignment (`BCN_ORIENT_YAW`)

You must tell the autopilot how the beacon network's "North" aligns with True North (or the frame's heading 0).

- If the beacon system's X-axis points East, you must set the orientation offset so the EKF aligns correctly.

Parameter Breakdown

- `BCN_TYPE`: Driver selection (0=Disabled, 1=Pozyx, 2=Marvelmind, etc.).
- `BCN_ORIENT_YAW`: Rotation of the beacon network relative to True North.

Integration Guide

Setup

1. **Place Beacons:** Set up your anchors/beacons in the room.
2. **Map:** Use the manufacturer's software to map the beacon positions and freeze the map.
3. **Config:** Set `BCN_TYPE` and `EK3_SRC1_POSXY` = Beacon.
4. **Align:** Use a compass or known heading to align the vehicle and check if the reported position on the GCS map moves in the correct direction when you move the drone.

Developer Notes

- **Library:** `libraries/AP_Beacon`
- **Usage:** Fused by `AP_NavEKF3` as a range or position source.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





BCN_ALT (Beacon origin's altitude above sealevel in meters)

DEFAULT: 0	RANGE: 0 10000	UNITS: m

Beacon origin's altitude above sealevel in meters (BCN_ALT)

Description

This parameter sets the Altitude above Mean Sea Level (AMSL) in meters for the beacon system's local origin $(0, 0, 0)$.

This establishes the vertical reference ("Z" axis anchor) for the indoor positioning system. By knowing the real-world altitude of the floor (or wherever the beacon zero-point is defined), the autopilot can report accurate AMSL altitude to the Ground Control Station and other systems.

The Mathematics

$$Alt_global = BCN_ALT + Z_local$$

(Note: The sign of Z_local depends on whether the beacon system uses NED (North-East-Down) or NEU (North-East-Up) coordinates, but the driver typically handles this normalization).

The Engineer's View

In `AP_Beacon::get_origin()`, this value is multiplied by 100 to convert it to centimeters (`int32_t` cm) used by ArduPilot's `Location` class for altitude.

Tuning & Behavior

- **Default Value:** 0
- **Range:** 0 to 10000 meters
- **Setting:** Set this to the known AMSL altitude of the beacon origin.
- **Dependencies:** Works in tandem with `BCN_LATITUDE` and `BCN_LONGITUDE`.

BCN_LATITUDE (Beacon origin's latitude)

DEFAULT: 0	RANGE: -90 90	UNITS: deg

Beacon origin's latitude (BCN_LATITUDE)

Description

This parameter sets the global Latitude (in degrees) that corresponds to the local origin $(0, 0, 0)$ of the beacon system.

Beacon systems typically operate in a local coordinate frame (X/Y in meters). By providing the real-world GPS coordinates of the beacon system's origin, the autopilot can convert these local measurements into global Latitude and Longitude. This allows you to fly autonomous missions using GPS waypoints even when navigating purely via indoor beacons.

The Mathematics

The autopilot uses this value as the reference point for a Local-to-Global projection (typically an Equirectangular projection or WGS84 calculation depending on the library used).

$$Lat_global = BCN_LAT + \Delta Lat(Y_local)$$

The Engineer's View

In `AP_Beacon::get_origin()`, this value is multiplied by 1.0×10^7 to convert it to the internal integer format (`int32_t` scaled by 1e7) used by ArduPilot's `Location` class.

It is critical that this value is non-zero if you want `AP_Beacon::get_origin()` to return true, which is often a prerequisite for the EKF to accept the beacon data as an absolute position source.

Tuning & Behavior

- **Default Value:** 0
- **Range:** -90 to +90 Degrees
- **Setting:** Set this to the precise latitude of the Beacon System's (0,0) point. You can measure this by placing a GPS-equipped phone or drone at the origin point outdoors, or by using Google Maps satellite view.
- **Dependencies:** Works in tandem with `BCN_LONGITUDE` and `BCN_ALT`.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





BCN_LONGITUDE (Beacon origin's longitude)

DEFAULT: 0	RANGE: -180 180	UNITS: deg

Beacon origin's longitude (BCN_LONGITUDE)

Description

This parameter sets the global Longitude (in degrees) that corresponds to the local origin $(0, 0, 0)$ of the beacon system.

Together with `BCN_LATITUDE`, it anchors the local Cartesian coordinate system (X, Y) of the indoor positioning system to the Earth's geodetic frame. This allows the EKF (Extended Kalman Filter) to output a global position estimate even when relying on local beacons.

The Mathematics

The autopilot uses this value as the reference point for a Local-to-Global projection.

$$Lon_global = BCN_LONG + \Delta Lon(X_local)$$

Note that longitude scaling varies with latitude ($\cos(Lat)$), which the internal projection functions handle automatically.

The Engineer's View

In `AP_Beacon::get_origin()`, this value is multiplied by 1.0×10^7 to convert it to the internal integer format (`int32_t` scaled by 1e7) used by ArduPilot's `Location` class.

Tuning & Behavior

- **Default Value:** 0
- **Range:** -180 to +180 Degrees
- **Setting:** Set this to the precise longitude of the Beacon System's (0,0) point.
- **Dependencies:** Works in tandem with `BCN_LATITUDE` and `BCN_ALT`.

BCN_ORIENT_YAW (Beacon systems rotation from north in degrees)

DEFAULT: 0	RANGE: -180 +180	UNITS: deg

Beacon systems rotation from north in degrees (BCN_ORIENT_YAW)

Description

This parameter defines the orientation of the beacon system's local "X" axis relative to True North.

If your beacon system is set up such that its "Forward" (X-axis) points East instead of North, you would set this to 90 degrees. This aligns the local coordinate grid with the global compass directions, ensuring that "Flying North" in the beacon frame corresponds to "Flying North" in the real world.

The Mathematics

The autopilot applies a 2D rotation matrix to the beacon's local (X, Y) coordinates before adding them to the global origin.

$$\begin{bmatrix} X_aligned \\ Y_aligned \end{bmatrix} = \begin{bmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{bmatrix} \begin{bmatrix} X_raw \\ Y_raw \end{bmatrix}$$

Where θ is `BCN_ORIENT_YAW`.

The Engineer's View

In `AP_Beacon_Backend::correct_position()`, this yaw rotation is applied to the raw position reported by the driver. It allows for arbitrary alignment of the physical beacon anchors without needing to reinstall them to face exactly North.

Tuning & Behavior

- **Default Value:** 0 (Beacon X-axis points North)
- **Range:** -180 to +180 Degrees
- **Positive Value:** Rotates the system Clockwise (e.g., +90 means X points East).
- **Negative Value:** Rotates the system Counter-Clockwise (e.g., -90 means X points West).
- **Tuning:** If your vehicle flies sideways when you command it forward (in Loiter/PosHold), or if the "Map" position moves incorrectly, this parameter is likely 90 or 180 degrees off.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





BCN_TYPE (Beacon based position estimation device type)

DEFAULT: 0	RANGE: 0 10	UNITS: Option

Beacon based position estimation device type (BCN_TYPE)

Description

This parameter enables the Beacon driver and selects the specific hardware vendor connected to the autopilot. Beacon systems (often called Indoor GPS) provide high-precision position data (X, Y, Z) for environments where standard GPS is unavailable, such as indoors or under heavy canopy.

Setting this parameter allocates the necessary driver memory and starts the communication backend.

The Mathematics

The value maps directly to a driver backend instantiation factory:

- **0:** None (Disabled)
- **1:** Pozyx (UWB)
- **2:** Marvelmind (Ultrasound)
- **3:** Nooploop (UWB)
- **10:** SITL (Software In The Loop simulation)

The Engineer's View

In `AP_Beacon::init()`, this parameter is read to switch-case the construction of `_driver`.

- `Type::Pozyx` → `AP_Beacon_Pozyx`
- `Type::Marvelmind` → `AP_Beacon_Marvelmind`
- etc.

If set to **0**, `AP_Beacon::enabled()` returns `false`, effectively bypassing the entire library.

Tuning & Behavior

- **Reboot Required:** Yes. Changing this parameter requires a reboot to initialize the correct driver and allocate resources.
- **Dependencies:** Enabling this is the first step. You must also configure the beacon's Origin (`BCN_LATITUDE`, etc.) if you want the vehicle to report global coordinates, or use `EK3_SRCx` parameters to tell the EKF to use Beacons as a position source.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **BLEND** parameter group (specifically `GPS_BLEND_MASK`) configures the **Dual GPS Blending** logic.

When two GPS units are installed, ArduPilot can "blend" their outputs to create a single, smoother, and more robust position solution for the EKF. This is distinct from "GPS Switching" (Best Available).

Key Concepts

1. Weighting Logic

The blended solution is a weighted average of the two sensors. The weights are calculated based on the reported accuracy (uncertainty) of each receiver.

$$P_\text{blend} = \frac{P_1 \cdot W_1 + P_2 \cdot W_2}{W_1 + W_2}$$

where $W = 1/\text{Accuracy}^2$.

2. Blending Mask (`GPS_BLEND_MASK`)

This bitmask determines *which* accuracy metrics are used to calculate the weights.

- **Bit 0:** Horizontal Position Accuracy.
- **Bit 1:** Vertical Position Accuracy.
- **Bit 2:** Speed Accuracy.

Parameter Breakdown

- `GPS_BLEND_MASK`: Controls weight calculation.
  - **5 (Default):** Use Horizontal Pos + Speed accuracy. Vertical accuracy is often ignored because GPS altitude is notoriously noisy.

Integration Guide

- **Requirement:** Two GPS units (`GPS_TYPE` and `GPS_TYPE2` set).
- **Activation:** Set `GPS_AUTO_SWITCH = 2` (Blend).
- **Verification:** Check `GPS2` status in GCS. The "Blended" solution is what the EKF consumes.

Developer Notes

- **Library:** `libraries/AP_GPS/AP_GPS_Blended.cpp`
- **Prefix:** `GPS_BLEND_` (often aliased or grouped here).

BLEND_MASK (GPS Blending Mask)

`DEFAULT: 5`

GPS Blending Mask (BLEND_MASK)

Description

Defines which GPS receivers are included in the multi-GPS "blending" logic.

Tuning & Behavior

- **Default Value:** 5 (GPS1 and GPS2)
- **Bitmask:**
  - Bit 0: GPS1
  - Bit 1: GPS2
  - Bit 2: GPS3

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

Key Concepts

BLK_RATEMAX (Block Logging Max Rate)

DEFAULT: 0	UNITS: Hz

Block Logging Max Rate (BLK_RATEMAX)

Description

Limits the frequency of data logging to internal flash (block storage) to conserve bandwidth and CPU.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **BRAKE** parameter group configures the braking behavior for specific flight modes, notably **FlowHold** in ArduCopter.

Note: The standard "Brake" flight mode uses `LOIT_BRK_` parameters. The parameter here (`BRAKE_RATE`) specifically applies to the **FlowHold** mode, which is an optical-flow-based loiter that allows the vehicle to drift when sticks are released unless braking is engaged.

Key Concepts

1. FlowHold Braking

In FlowHold mode, the vehicle tracks features on the ground. When the pilot releases the sticks:

- The vehicle attempts to stop its drift.
- `BRAKE_RATE` limits the maximum pitch/roll angle (or rate) used to achieve this stop.

Parameter Breakdown

- `BRAKE_RATE`: Maximum braking rate (deg/s) for FlowHold. Higher values mean sharper stops (more like Loiter). Lower values mean smoother, driftier stops.

Integration Guide

- **FlowHold:** Requires an Optical Flow sensor (e.g., HereFlow, CX-OF).
- **Tuning:** If the vehicle stops too abruptly in FlowHold, reduce `BRAKE_RATE`.

Developer Notes

- **Library:** `ArduCopter/mode_flowhold.cpp`
- **Context:** Unique to the FlowHold mode state machine.

BRAKE_RATE (FlowHold Brake Rate)

DEFAULT: 8	RANGE: 1 50	UNITS: deg/s

FlowHold Brake Rate (BRAKE_RATE)

Description

Defines the maximum angular rate used for automatic braking in FlowHold mode.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **BRD** parameter group contains **Hardware Abstraction Layer (HAL)** settings. These parameters configure the low-level behavior of the flight controller board itself, independent of the vehicle type (Copter/Plane/Rover).

It handles IO processor communication, safety switches, internal heating, SD card speeds, and board-specific voltage monitoring.

Key Concepts

1. Safety Switch (`BRD_SAFETY_DEFLT` / `BRD_SAFETY_MASK`)

Controls the behavior of the hardware Safety Switch (often on the GPS module).

- **Default:** Determines if the safety switch is active at boot.
- **Mask:** Allows specific channels (e.g., landing gear, gimbal) to move even when the safety switch is active (safe).

2. IMU Heater (`BRD_HEAT_...`)

Many high-end boards (Cube, Durandal) have internal heating resistors to keep the IMU at a constant temperature (usually 45-60°C) for consistent calibration.

- **Tuning:** `BRD_HEAT_P/I` controls the PID loop for the heater.

3. IO Processor (`BRD_IO_ENABLE`)

Some boards (Pixhawk 1/Cube) use a separate co-processor (IOMCU) for PWM output safety.

- `BRD_IO_ENABLE`: Must be 1 to use the MAIN/AUX output split on IOMCU-equipped boards.

4. SD Card (`BRD_SD_SLOWDOWN`)

If you have logging issues or bad SD cards, this parameter can slow down the SPI clock to the card to improve reliability (at the cost of speed).

Parameter Breakdown

- `BRD_TYPE`: Autodetected board ID. Read-only.
- `BRD_SERIAL_NUM`: Unique ID of the board.
- `BRD_BOOT_DELAY`: Delays bootup to allow peripherals (sensors) to power up and stabilize.
- `BRD_PWM_VOLT_SEL`: Selects 3.3V or 5V for servo rail output (if supported by hardware hardware).

Integration Guide

Disabling the Safety Switch

If you don't have a switch installed and the board refuses to arm ("Hardware Safety Switch"):

1. Set `BRD_SAFETY_DEFLT = 0`.
2. Reboot.

SD Card Issues

If you see "No SD Card" or logging gaps:

1. Try `BRD_SD_SLOWDOWN = 1` or higher.

Developer Notes

- **Library:** `libraries/AP_HAL_ChibiOS` (mostly).
- **Scope:** These are often set by `hwdef.dat` at compile time but can be overridden.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





BRD_ALT_CONFIG (Alternative HW config)

DEFAULT: 0	RANGE: 0 10

Alternative HW config (BRD_ALT_CONFIG)

Description

Selects pre-defined alternative pin assignments or hardware options for specific boards. Refer to your board's documentation for valid values.

BRD_BOOT_DELAY (Boot delay)

DEFAULT: 0	RANGE: 0 10000	UNITS: ms

Boot delay (BRD_BOOT_DELAY)

Description

Delays the main autopilot initialization process on startup. This is helpful if certain external sensors or peripherals (like some GPS modules) need more time to power up before ArduPilot attempts to communicate with them.

BRD_HEAT_I (Board Heater I gain)

DEFAULT: 0.1	RANGE: 0 1

Board Heater I gain (BRD_HEAT_I)

Description

Compensates for long-term temperature steady-state errors in the IMU heater.

BRD_HEAT_IMAX (Board Heater IMAX)

DEFAULT: 100	RANGE: 0 100

Board Heater IMAX (BRD_HEAT_IMAX)

Description

Limits the maximum authority of the heater's integral term to prevent "wind-up" and overheating.

BRD_HEAT_LOWMGN (Board heater temp lower margin)

DEFAULT: 0	RANGE: 0 20	UNITS: degC

Board heater temp lower margin (BRD_HEAT_LOWMGN)

Description

Safety margin for heater-based arming checks. The aircraft will refuse to arm until the IMU has reached at least (BRD_HEAT_TARG - BRD_HEAT_LOWMGN).

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





BRD_HEAT_P (Board Heater P gain)

DEFAULT: 50	RANGE: 1 500

Board Heater P gain (BRD_HEAT_P)

Description

Determines how aggressively the heater reacts to temperature deviations from the target.

BRD_HEAT_TARG (Board heater temperature target)

DEFAULT: -1	RANGE: -1 80	UNITS: degC

Board heater temperature target (BRD_HEAT_TARG)

Description

Target temperature for the integrated IMU heater. Keeping the IMU at a constant temperature reduces thermal drift and improves sensor accuracy, especially in cold environments.

BRD_IO_DSHOT (Load DShot FW on IO)

DEFAULT: 0	RANGE: 0 1

Load DShot FW on IO (BRD_IO_DSHOT)

Description

Enables DShot digital ESC protocol support on the primary I/O outputs (channels 1-8) by loading specialized firmware onto the IOMCU.

BRD_IO_ENABLE (Enable IO co-processor)

DEFAULT: 1	RANGE: 0 2

Enable IO co-processor (BRD_IO_ENABLE)

Description

Controls the integrated I/O co-processor (IOMCU) found on many flight controllers (e.g., Pixhawk). The IOMCU handles low-level PWM output and safety switch logic.

Tuning & Behavior

- **Default Value:** 1 (Enabled)
- **Values:** 0:Disabled, 1:Enabled, 2:EnableNoFWUpdate
- **Requires reboot to take effect.**

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





BRD_OPTIONS (Board options)

DEFAULT: 0

Board options (BRD_OPTIONS)

Description

Bitmask for low-level hardware configuration options and developer features.

Tuning & Behavior

- **Default Value:** 0 (Standard)
- **Bitmask:**
  - Bit 0: Enable hardware watchdog
  - Bit 1: Disable MAVftp
  - Bit 3: Enable Debug Pins
  - Bit 7: Skip board validation

BRD_PWM_VOLT_SEL (Set PWM Out Voltage)

DEFAULT: 0	RANGE: 0 1	UNITS: V

Set PWM Out Voltage (BRD_PWM_VOLT_SEL)

Description

Selects the logic voltage level (3.3V or 5V) for the PWM output pins. Using 5V can improve signal integrity and reduce the impact of electrical noise on long servo wires.

BRD_RADIO_ABLVL (Internal Radio Auto-Bind RSSI Level)

DEFAULT: 0	RANGE: 0 31

Internal Radio Auto-Bind RSSI Level (BRD_RADIO_ABLVL)

Description

`BRD_RADIO_ABLVL` is a security filter for the BRD_RADIO_ABTIME feature.

It defines how close a transmitter must be before the drone will allow itself to be bound to it. By requiring a high signal strength (high RSSI), it ensures that your drone only binds to the transmitter you are holding in your hand, rather than someone else's transmitter in a nearby field.

Tuning & Behavior

- **Default:** 0 (Any strength accepted - Least secure).
- **Recommendation:** Set this to a value that requires the transmitter to be within **1-2 meters** of the drone during binding.
- **Usage:** Only used when the radio is actively in Auto-Bind mode.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





BRD_RADIO_ABTIME (Internal Radio Auto-Bind Timeout)

DEFAULT: 0	RANGE: 0 120	UNITS: s

Internal Radio Auto-Bind Timeout (BRD_RADIO_ABTIME)

Description

`BRD_RADIO_ABTIME` determines how long the integrated receiver waits for its "known" transmitter before it gives up and enters **Auto-Bind** mode.

This is a convenience feature for consumer drones. If you lose or replace your transmitter, you can power on the drone and wait for this timeout to expire. The drone will then start looking for a new transmitter to bind with.

- **0 (Default):** Auto-bind is disabled. You must trigger binding manually (usually via a button or MAVLink command).
- **1-120:** Wait time in seconds.

Tuning & Behavior

- **Default:** 0.
- **Usage:** Set to **10** or **20** if you want the drone to automatically enter bind mode if you power it on before your transmitter.

BRD_RADIO_BZOFS (Internal Radio Buzzer Offset)

DEFAULT: 25	RANGE: 0 40

Internal Radio Buzzer Offset (BRD_RADIO_BZOFS)

Description

`BRD_RADIO_BZOFS` allows for a fine-tuning adjustment of the pitch of the buzzer inside the handheld transmitter (when using integrated SPI radios like SkyViper).

Due to manufacturing tolerances in the tiny piezo buzzers used in these controllers, some might sound higher or lower than intended. This parameter allows the user to shift the frequency up or down so that the status beeps and melodies sound correct.

Tuning & Behavior

- **Default:** 25.
- **Adjustment:** Change this value if your transmitter's beeps sound "flat" or "off-key." Higher values increase the frequency (higher pitch).

BRD_RADIO_DEBUG (Internal Radio Debug Level)

DEFAULT: 0	RANGE: 0 4

Internal Radio Debug Level (BRD_RADIO_DEBUG)

Description

`BRD_RADIO_DEBUG` is a troubleshooting tool for flight controllers with integrated SPI-based radio receivers (like SkyViper).

It controls how much information about the wireless link (packet loss, signal timing, binding status) is sent to the flight controller's internal console or MAVLink GCS.

- **0: Disabled (Default).**
- **1-4:** Increasing levels of diagnostic data.

Tuning & Behavior

- **Warning:** Higher debug levels can flood the communication ports and may cause slight performance hits. Only enable this if you are actively diagnosing a "No RC Link" issue with an integrated radio.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY

BRD_RADIO_DISCRC (Internal Radio Disable RX CRC)

DEFAULT: 0	RANGE: 0 1

Internal Radio Disable RX CRC (BRD_RADIO_DISCRC)

Description

`BRD_RADIO_DISCRC` is a low-level developer tool for debugging integrated SPI radios.

CRC is a mathematical check that ensures a packet wasn't corrupted during its flight through the air. Normally, if a packet fails this check, it is thrown away. By disabling this check, the autopilot will attempt to process *every* signal it receives, even if it is garbage.

- **0: Enabled (Safe/Default).** Corrupt packets are ignored.
- **1: Disabled (DANGEROUS).** Corrupt packets are accepted.

Tuning & Behavior

- **Warning: NEVER enable this for flight.** Accepting corrupt packets can lead to "Phantom Stick Movements" where the drone suddenly receives a command to full throttle or hard roll due to a bit-flip in the air.
- **Usage:** Only used on the bench to see if a radio link is "almost" working but failing due to slight noise issues.

BRD_RADIO_FCCTST (Internal Radio FCC Test Mode)

DEFAULT: 0	RANGE: 0 100

Internal Radio FCC Test Mode (BRD_RADIO_FCCTST)

Description

`BRD_RADIO_FCCTST` is used for regulatory compliance testing (FCC/CE).

When set to a non-zero value, the integrated radio will lock onto that specific frequency channel and transmit a continuous signal (or specific test pattern). This is required by labs to measure things like "Occupied Bandwidth" and "Max Power."

Tuning & Behavior

- **0: Disabled (Default).** Normal flight operation.
- **1-100:** Sets the radio to transmit on the corresponding channel.
- **Warning: RC input is disabled while this mode is active.** You cannot fly while this parameter is non-zero.

BRD_RADIO_PPSCH (Internal Radio PPS Channel)

DEFAULT: 0	RANGE: 0 16

Internal Radio PPS Channel (BRD_RADIO_PPSCH)

Description

`BRD_RADIO_PPSCH` allows you to monitor the "Frame Rate" of your radio link in real-time.

It maps the number of packets received per second (PPS) to a virtual RC channel. You can then view this value on your OSD or GCS to see if the link is slowing down (indicating interference or range limits) or staying at its full operational speed.

- **0:** Disabled.
- **1-16:** The RC channel number used to report PPS.

Tuning & Behavior

- **Recommendation:** Set to an unused channel (e.g. 15) to monitor link health during long-range flights.
- **Context:** Used for integrated SPI radios where the autopilot has direct access to low-level packet statistics.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





BRD_RADIO_PROT (Internal Radio Protocol)

DEFAULT: 0	RANGE: 0 2

Internal Radio Protocol (BRD_RADIO_PROT)

Description

`BRD_RADIO_PROT` defines the communication standard used between the drone's internal receiver and your handheld transmitter. This is used on flight controllers with integrated SPI radios (like the SkyViper).

- **0: Auto (Recommended).** The autopilot will attempt to detect if the transmitter is using DSM2 or DSMX.
- **1: DSM2.** Legacy Spektrum protocol.
- **2: DSMX.** Current Spektrum protocol (more robust against interference).

Tuning & Behavior

- **Default:** 0.
- **Recommendation:** Leave at **0 (Auto)** unless you have a transmitter that is known to only support one specific protocol.

BRD_RADIO_SIGCH (Internal Radio RSSI Channel)

DEFAULT: 0	RANGE: 0 16

Internal Radio RSSI Channel (BRD_RADIO_SIGCH)

Description

`BRD_RADIO_SIGCH` allows you to monitor the health of your wireless link by injecting the RSSI (Received Signal Strength Indicator) into a virtual RC channel.

This allows standard OSDs and Ground Control Stations to display the signal strength of the integrated SPI radio just like a traditional receiver.

- **0:** Disabled.
- **1-16:** The RC channel number where RSSI data will be injected (usually set to channel 8 or 16).

Tuning & Behavior

- **Default:** 0.
- **Setup:** If you set this to 16, ensure your OSD or GCS is configured to look at Channel 16 for RSSI information.

BRD_RADIO_STKMD (Internal Radio Stick Mode)

DEFAULT: 2	RANGE: 1 2

Internal Radio Stick Mode (BRD_RADIO_STKMD)

Description

`BRD_RADIO_STKMD` defines which sticks on your transmitter control which flight axes.

- **Mode 1:** Throttle and Roll on the right stick. Pitch and Yaw on the left stick.
- **Mode 2 (Default):** Throttle and Yaw on the left stick. Pitch and Roll on the right stick. (Standard for most of the world).

Tuning & Behavior

- **Default:** 2.
- **Usage:** Only change this if you are a pilot trained in Mode 1 flight. This parameter is sent to the handheld transmitter to reconfigure its internal mixing.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





BRD_RADIO_TELEM (Internal Radio Telemetry Enable)

DEFAULT: 0	RANGE: 0 1

Internal Radio Telemetry Enable (BRD_RADIO_TELEM)

Description

`BRD_RADIO_TELEM` enables the downlink of flight data from the drone's integrated radio receiver back to the pilot's handset.

This is primarily used for the **Cypress (DSM)** radio driver on specific flight controllers (like the SkyViper) to send battery status and other simple telemetry data to a Spektrum-compatible transmitter.

- **0: Disabled.**
- **1: Enabled.**

Tuning & Behavior

- **Default:** 0.
- **Note:** This parameter is specific to the internal SPI radio and does not affect external telemetry radios connected to UART ports.

BRD_RADIO_TESTCH (Internal Radio Factory Test Channel)

DEFAULT: 0	RANGE: 0 100

Internal Radio Factory Test Channel (BRD_RADIO_TESTCH)

Description

`BRD_RADIO_TESTCH` is a production and assembly tool for manufacturers.

It allows a drone's integrated radio to communicate on a fixed, pre-determined frequency channel without going through the standard "Bind" procedure. This allows factory workers to quickly verify that the radio hardware is functioning correctly before shipping.

- **0: Disabled (Default).** Normal operation with standard binding.
- **1-100:** Locks the radio to that specific test channel.

Tuning & Behavior

- **Default:** 0.
- **Warning: Do not change this.** If this is set to a non-zero value, you will not be able to bind your normal transmitter to the drone.

BRD_RADIO_TPPSCH (Internal Radio Telemetry PPS Channel)

DEFAULT: 0	RANGE: 0 16

Internal Radio Telemetry PPS Channel (BRD_RADIO_TPPSCH)

Description

`BRD_RADIO_TPPSCH` provides the "Uplink" counterpart to BRD_RADIO_PPSCH.

It reports how many telemetry packets the handheld transmitter is successfully receiving from the drone every second. This value is transmitted back to the drone and mapped to a virtual RC channel. Monitoring both PPS and TPPS allows you to identify if a signal problem is one-way (e.g., a noisy VTX drowning out telemetry) or two-way (range limit).

Tuning & Behavior

- **Default:** 0.
- **Setup:** Map this to an RC channel and monitor it on your OSD to verify that your telemetry link is stable during flight.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





BRD_RADIO_TSIGCH (Internal Radio Telemetry RSSI Channel)

DEFAULT: 0	RANGE: 0 16

Internal Radio Telemetry RSSI Channel (BRD_RADIO_TSIGCH)

Description

`BRD_RADIO_TSIGCH` reports how "Loudly" the handheld transmitter is hearing the drone.

This is the telemetry signal strength (RSSI) measured at the ground station/transmitter end. Like BRD_RADIO_SIGCH (which measures strength at the drone), this parameter maps the ground-side measurement to a virtual RC channel.

Tuning & Behavior

- **Default:** 0.
- **Significance:** If your drone's SIGCH is high but TSIGCH is low, it means the drone is receiving commands perfectly, but the ground station is struggling to hear the drone's telemetry return signal (often caused by a noisy video transmitter on the drone).

BRD_RADIO_TXMAX (Internal Radio Max TX Power)

DEFAULT: 8	RANGE: 1 8

Internal Radio Max TX Power (BRD_RADIO_TXMAX)

Description

`BRD_RADIO_TXMAX` sets the maximum power limit for the **Handheld Transmitter** (remote control) when using a two-way integrated radio system (like SkyViper).

The autopilot sends this value back to the transmitter via telemetry, commanding it to limit its output power to the specified level. This is useful for adhering to regional power regulations or managing battery life on the controller.

Tuning & Behavior

- **Default:** 8 (Maximum).
- **Range:** 1 (Lowest) to 8 (Highest).
- **Usage:** For maximum control range outdoors, leave at **8**. If you are only flying in a small indoor arena and want to reduce interference with other electronics, you can lower this value.

BRD_RADIO_TXPOW (Internal Radio Transmit Power)

DEFAULT: 8	RANGE: 1 8

Internal Radio Transmit Power (BRD_RADIO_TXPOW)

Description

`BRD_RADIO_TXPOW` controls the strength of the signal sent from the drone's internal radio back to the transmitter (Telemetry).

This is used on flight controllers with built-in SPI receivers (like the SkyViper or some small AIO boards). Increasing the power improves range but consumes more battery and generates more heat on the radio chip.

- **1:** Minimum power (Short range, safe for bench testing).
- **8:** Maximum power (Long range).

Tuning & Behavior

- **Default:** 8.
- **Recommendation:** Leave at **8** for outdoor flight. Reduce to **1** or **2** if you are only flying indoors or working on the bench to reduce electrical noise.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





BRD_RADIO_TYPE (Internal Radio Type)

DEFAULT: 0	RANGE: 0 3

Internal Radio Type (BRD_RADIO_TYPE)

Description

`BRD_RADIO_TYPE` enables the integrated radio receiver found on some flight controllers (like the SkyViper or specialized AIO racing boards). These receivers are connected directly to the CPU via an SPI bus, rather than a standard UART.

- **0: None.** Internal radio is disabled.
- **1: CYRF6936.** DSM/Spektrum compatible.
- **2: CC2500.** FrSky/Futaba compatible.
- **3: BK2425.**

Tuning & Behavior

- **Reboot Required:** Yes.
- **Setup:** Once enabled, you must bind your transmitter. Use the board-specific binding procedure (usually a button or a MAVLink command).

BRD_RTC_TYPES (Allowed RTC types)

DEFAULT: 1

Allowed RTC types (BRD_RTC_TYPES)

Description

Specifies which sources (GPS, Internal, System) are allowed to set the vehicle's Real Time Clock.

BRD_RTC_TZ_MIN (Timezone offset)

DEFAULT: 0	UNITS: min

Timezone offset (BRD_RTC_TZ_MIN)

Description

Offsets the system time from UTC for local time reporting in logs.

BRD_SAFETYOPTION (Options for safety button behavior)

DEFAULT: 3

Options for safety button behavior (BRD_SAFETYOPTION)

Description

Configures the specific interaction rules for the hardware safety button, such as whether it can be used to re-engage safety while armed.

Tuning & Behavior

- **Default Value:** 3 (Button active for Safety On/Off)
- **Bitmask:**
  - 0: ActiveForSafetyDisable
  - 1: ActiveForSafetyEnable
  - 2: ActiveWhenArmed
  - 3: Force safety on when the aircraft disarms

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY

BRD_SAFETY_DEFLT (Sets default state of the safety switch)

DEFAULT: 1	RANGE: 0 1

Sets default state of the safety switch (BRD_SAFETY_DEFLT)

Description

Determines if the hardware safety switch starts in the "Safe" (blashing) or "Armed" (solid) state upon boot.

Tuning & Behavior

- **Default Value:** 1 (Starts in Safe state)
- **Values:** 0:Disabled (Unsafe at boot), 1:Enabled (Safe at boot)
- **Safety switch behavior can still be toggled manually after boot.**

BRD_SAFETY_MASK (Outputs which ignore the safety switch state)

DEFAULT: 0

Outputs which ignore the safety switch state (BRD_SAFETY_MASK)

Description

Allows specific servo/motor outputs to function even if the hardware safety switch is in the "Safe" position. Commonly used for non-propulsion servos like camera gimbals or retractable landing gear.

Tuning & Behavior

- **Default Value:** 0
- **Bitmask corresponds to output channels 1 through 14.**

BRD_SBUS_OUT (SBUS output rate)

DEFAULT: 0	RANGE: 0 7	UNITS: Hz

SBUS output rate (BRD_SBUS_OUT)

Description

Configures the refresh rate of the SBUS output port.

Tuning & Behavior

- **Default Value:** 0 (Disabled)
- **Values:** 1:50Hz, 2:75Hz, 3:100Hz, 4:150Hz, 5:200Hz, 6:250Hz, 7:300Hz

BRD_SD_FENCE (SDCard Fence size)

DEFAULT: 0	RANGE: 0 64	UNITS: kB

SDCard Fence size (BRD_SD_FENCE)

Description

Allocates dedicated storage space on the SD card for complex geofence definitions (fence.stg).

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





BRD_SD_MISSION (SDCard Mission size)

DEFAULT: 0	RANGE: 0 64	UNITS: kB

SDCard Mission size (BRD_SD_MISSION)

Description

Allocates dedicated storage space on the SD card for mission command storage (mission.stg), allowing for much larger missions than internal EEPROM.

BRD_SD_SLOWDOWN (microSD slowdown)

DEFAULT: 0	RANGE: 0 32

microSD slowdown (BRD_SD_SLOWDOWN)

Description

Reduces the SPI bus speed for the microSD card. This can resolve "No SD Card" or logging errors on certain combinations of flight boards and card types.

BRD_SER1_RTSCTS (Serial 1 flow control)

DEFAULT: 2	RANGE: 0 3

Serial 1 flow control (BRD_SER1_RTSCTS)

Description

Enables hardware flow control (RTS/CTS) for the first telemetry port. This prevents data loss during high-bandwidth telemetry streaming.

Tuning & Behavior

- **Default Value:** 2 (Auto)
- **Values:** 0:Disabled, 1:Enabled, 2:Auto, 3:RS-485

BRD_SER2_RTSCTS (Serial 2 flow control)

DEFAULT: 2	RANGE: 0 3

Serial 2 flow control (BRD_SER2_RTSCTS)

Description

Enables hardware flow control for the second telemetry port.

BRD_SER3_RTSCTS (Serial 3 flow control)

DEFAULT: 2	RANGE: 0 3

Serial 3 flow control (BRD_SER3_RTSCTS)

Description

Enables hardware flow control for the third serial port.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





BRD_SER4_RTSCTS (Serial 4 flow control)

DEFAULT: 2	RANGE: 0 3

Serial 4 flow control (BRD_SER4_RTSCTS)

Description

Enables hardware flow control for the fourth serial port.

BRD_SER5_RTSCTS (Serial 5 flow control)

DEFAULT: 2	RANGE: 0 3

Serial 5 flow control (BRD_SER5_RTSCTS)

Description

Enables hardware flow control for the fifth serial port.

BRD_SER6_RTSCTS (Serial 6 flow control)

DEFAULT: 2	RANGE: 0 3

Serial 6 flow control (BRD_SER6_RTSCTS)

Description

Enables hardware flow control for the sixth serial port.

BRD_SER7_RTSCTS (Serial 7 flow control)

DEFAULT: 2	RANGE: 0 3

Serial 7 flow control (BRD_SER7_RTSCTS)

Description

Enables hardware flow control for the seventh serial port.

BRD_SER8_RTSCTS (Serial 8 flow control)

DEFAULT: 2	RANGE: 0 3

Serial 8 flow control (BRD_SER8_RTSCTS)

Description

Enables hardware flow control for the eighth serial port.

BRD_SERIAL_NUM (User-defined serial number)

DEFAULT: 0	RANGE: -8388608 8388607

User-defined serial number (BRD_SERIAL_NUM)

Description

Allows users to assign a unique ID number to their aircraft. This is useful for fleet management and identifying specific airframes in log files.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





BRD_TYPE (Board type)

DEFAULT: 0

Board type (BRD_TYPE)

Description

Specifies the hardware platform type. While ArduPilot usually auto-detects the board, this parameter can be used to force a specific driver set.

Tuning & Behavior

- **Default Value:** 0 (Auto-detect)
- **Values:** 1:PX4V1, 2:Pixhawk, 3:Cube/Pixhawk2, 4:Pixracer, etc.
- **Requires reboot to take effect.**

BRD_VBUS_MIN (Autopilot board voltage requirement)

DEFAULT: 4.3	RANGE: 4.0 5.5	UNITS: V

Autopilot board voltage requirement (BRD_VBUS_MIN)

Description

Pre-arm safety threshold for the flight controller's internal supply voltage. Arming is blocked if the input voltage is too low, indicating a potentially unreliable power source.

BRD_VSERVO_MIN (Servo voltage requirement)

DEFAULT: 0	RANGE: 3.3 12.0	UNITS: V

Servo voltage requirement (BRD_VSERVO_MIN)

Description

Pre-arm safety threshold for the servo power rail. Arming is blocked if the voltage powering the servos is below this value.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **BRD_RADIO** parameter group configures the **Internal Radio** found on certain flight controllers (e.g., SkyViper or boards with integrated SiK/ExpressLRS hardware).

Unlike external radios connected via UART, these are directly managed by the board's HAL (Hardware Abstraction Layer).

Key Concepts

1. Transmit Power (`BRD_RADIO_TXPOW`)

Sets the output power of the integrated radio module.

- **Safety:** Ensure compliance with local RF regulations (e.g., 20dBm / 100mW).

2. Stick Mode (`BRD_RADIO_STKMD`)

Defines the mapping of the transmitter sticks (Mode 1, Mode 2, etc.) for integrated controllers.

Developer Notes

- **Library:** `libraries/AP_Radio`.
- **Context:** Primarily used for RTF (Ready-To-Fly) consumer-grade hardware running ArduPilot.

BRD_RADIO_ABLVL (Auto-bind level)

DEFAULT: 0	RANGE: 0 31

Auto-bind level (BRD_RADIO_ABLVL)

Description

Security threshold for auto-binding. Higher values require the transmitter to be very close to the receiver during the auto-bind process, preventing accidental binding to other transmitters nearby.

BRD_RADIO_ABTIME (Auto-bind time)

DEFAULT: 0	RANGE: 0 120	UNITS: s

Auto-bind time (BRD_RADIO_ABTIME)

Description

Configures the timeout period of inactivity after which the receiver will automatically enter "binding" mode to search for a new transmitter.

- **A value of 0 disables auto-bind.**

BRD_RADIO_BZOFS (Transmitter buzzer adjustment)

DEFAULT: 25	RANGE: 0 40

Transmitter buzzer adjustment (BRD_RADIO_BZOFS)

Description

Fine-tunes the audio frequency (pitch) of the transmitter's internal buzzer.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





BRD_RADIO_DEBUG (Radio debug level)

DEFAULT: 0	RANGE: 0 4

Radio debug level (BRD_RADIO_DEBUG)

Description

Sets the verbosity of radio-specific debug information sent to the serial console.

BRD_RADIO_DISCRC (Disable receive CRC)

DEFAULT: 0	RANGE: 0 1

Disable receive CRC (BRD_RADIO_DISCRC)

Description

Disables the Cyclical Redundancy Check (CRC) for incoming radio packets.

- **WARNING: Should only be used for debugging. Disabling CRC allows corrupted data to be processed as valid commands.**

BRD_RADIO_FCCTST (FCC test mode)

DEFAULT: 0	RANGE: 0 6

FCC test mode (BRD_RADIO_FCCTST)

Description

Puts the internal radio into a specialized continuous-transmit state required for regulatory certification.

- **WARNING: The radio will not process RC inputs while in this mode.**

BRD_RADIO_PPSCH (Packet rate channel)

DEFAULT: 0	RANGE: 0 16

Packet rate channel (BRD_RADIO_PPSCH)

Description

Maps the Packet-Per-Second (PPS) rate to a virtual RC channel.

BRD_RADIO_PROT (Radio Protocol)

DEFAULT: 0

Radio Protocol (BRD_RADIO_PROT)

Description

Selects the specific transmission protocol for the direct-attached radio.

Tuning & Behavior

- **Default Value:** 0 (Auto)
- **Values:** 0:Auto, 1:DSM2, 2:DSMX

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





BRD_RADIO_SIGCH (RSSI signal strength channel)

DEFAULT: 0	RANGE: 0 16

RSSI signal strength channel (BRD_RADIO_SIGCH)

Description

Maps the RSSI (Signal Strength) value to a virtual RC channel, allowing it to be monitored by the GCS or used in other logic.

BRD_RADIO_STKMD (Stick input mode)

DEFAULT: 2	RANGE: 1 2

Stick input mode (BRD_RADIO_STKMD)

Description

Configures the transmitter stick mapping. Mode 2 (throttle on left) is standard in most regions.

Tuning & Behavior

- **Default Value:** 2 (Mode 2)
- **Values:** 1:Mode1, 2:Mode2

BRD_RADIO_TELEM (Enable telemetry)

DEFAULT: 0	RANGE: 0 1

Enable telemetry (BRD_RADIO_TELEM)

Description

Enables bidirectional telemetry communication over the internal radio link.

BRD_RADIO_TESTCH (Factory test channel)

DEFAULT: 0	RANGE: 0 8

Factory test channel (BRD_RADIO_TESTCH)

Description

Overrides the normal frequency-hopping behavior to lock the radio onto a specific channel for production testing.

BRD_RADIO_TPPSCH (Telemetry PPS channel)

DEFAULT: 0	RANGE: 0 16

Telemetry PPS channel (BRD_RADIO_TPPSCH)

Description

Maps the telemetry Packet-Per-Second (PPS) rate measured by the transmitter to a virtual RC channel.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





BRD_RADIO_TSIGCH (Telemetry RSSI value channel)

DEFAULT: 0	RANGE: 0 16

Telemetry RSSI value channel (BRD_RADIO_TSIGCH)

Description

Maps the telemetry signal strength (RSSI) measured by the transmitter back to a virtual RC channel.

BRD_RADIO_TXMAX (Transmitter maximum transmit power)

DEFAULT: 8	RANGE: 1 8

Transmitter maximum transmit power (BRD_RADIO_TXMAX)

Description

Sets the upper limit for the radio's transmission power.

BRD_RADIO_TXPOW (Telemetry Transmit power)

DEFAULT: 8	RANGE: 1 8

Telemetry Transmit power (BRD_RADIO_TXPOW)

Description

Sets the transmission power level for telemetry data sent back to the transmitter.

BRD_RADIO_TYPE (Radio type)

DEFAULT: 0

Radio type (BRD_RADIO_TYPE)

Description

Enables support for built-in or direct-attached radio receivers (e.g., on SkyViper vehicles).

Tuning & Behavior

- **Default Value:** 0 (None)
- **Values:** 0:None, 1:CYRF6936, 2:CC2500, 3:BK2425
- **Requires reboot to take effect.**

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **BTAG** parameter group supports **Smart Battery Tagging** (NFC/RFID or SMBus). This system tracks the health and usage history of individual battery packs across multiple flights.

Unlike standard `BATT` parameters which configure the *sensor*, `BTAG` stores persistent data about the *battery pack itself* if the hardware supports retrieving a unique ID from the battery.

Key Concepts

1. Cycle Counting

The autopilot tracks charge/discharge cycles to estimate battery aging.

- `BTAG_CYCLE_MIN`: Minimum discharge (mAh or %) required to count as a "cycle."

2. Usage History

- `BTAG_FIRST_USE`: Timestamp of first flight.
- `BTAG_LAST_USE`: Timestamp of most recent flight.

Parameter Breakdown

- `BTAG_CAPACITY`: The factory capacity of the detected pack.

Developer Notes

- **Context:** Requires specific hardware support (e.g., Tattu/GensAce Smart Batteries or custom NFC readers).

BTAG_CAPACITY (Battery Tag Capacity (Invalid/Legacy))

DEFAULT: 3300	UNITS: mAh

Battery Tag Capacity (BTAG_CAPACITY)

Description

This parameter is not found as BTAG_CAPACITY in the main codebase. It appears to be a miscategorized battery parameter.

BTAG_CYCLE_MIN (Battery Tag Cycle Min (Invalid/Legacy))

DEFAULT: 10

Battery Tag Cycle Min (BTAG_CYCLE_MIN)

Description

This parameter is not found as BTAG_CYCLE_MIN in the main codebase.

BTAG_FIRST_USE (Battery Tag First Use (Invalid/Legacy))

DEFAULT: 0

Battery Tag First Use (BTAG_FIRST_USE)

Description

This parameter is not found as BTAG_FIRST_USE in the main codebase.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





BTAG_LAST_USE (Battery Tag Last Use (Invalid/Legacy))

DEFAULT: 0

Battery Tag Last Use (BTAG_LAST_USE)

Description

This parameter is not found as BTAG_LAST_USE in the main codebase.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





BUTTON INPUT CONFIGURATION (BTN)

Overview

The **BTN** parameter group configures up to 4 physical push-buttons connected to the autopilot's GPIO pins. These buttons can be used to trigger flight mode changes, relay toggles, camera actions, or safety functions without needing an RC transmitter.

Note: This is for *physical* buttons on the drone, not buttons on your RC controller (see `RCx_OPTION`) or Joystick (see `JS_BUTTON`).

Key Concepts

1. Function Mapping (`BTN_FUNCn`)

Assigns an action to the button press.

- **Disabled:** Do nothing.
- **Mode Change:** Switch to Loiter, RTL, Auto, etc.
- **Camera:** Trigger shutter.
- **Relay:** Toggle a relay pin.
- **Arm/Disarm:** Toggle arming state (dangerous).

2. Shift Functions (`BTN_SFUNCn`)

Allows a button to perform a *different* action if a designated "Shift" button is held down simultaneously.

Parameter Breakdown

- `BTN_ENABLE`: Master switch.
- `BTN_PINn`: GPIO/Servo pin number connected to the button.
- `BTN_FUNCn`: Primary action.
- `BTN_OPTIONSn`: Debounce logic and active high/low settings.

Integration Guide

1. **Hardware:** Connect a momentary switch between a Signal Pin and Ground.
2. **Enable:** Set `BTN_ENABLE = 1`.
3. **Map Pin:** Set `BTN_PIN1` to the servo rail pin number (e.g., 50 for AUX1).
4. **Set Function:** Set `BTN_FUNC1` to the desired action.
5. **Test:** Press the button and check the GCS "Messages" tab for confirmation.

Developer Notes

- **Library:** `libraries/AP_Button`
- **Polling:** Buttons are polled at 10Hz-50Hz.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





BTN1_FUNCTION (Joystick Button Function (Instance 1))

DEFAULT: 0	RANGE: 0 111

Joystick Button Function (BTN1_FUNCTION)

Description

`BTN1_FUNCTION` defines the "Primary Action" for the first button on your USB joystick or game controller.

This is extremely common in **ArduSub** and **ArduBlimp** applications. By assigning functions to specific buttons, you can fly the vehicle entirely from a joystick without needing a traditional RC transmitter.

Common Values

- **0: None.**
- **1: shift.** (Enables the BTN1_SFUNCTION for all other buttons).
- **2: arm_toggle.**
- **3: arm.**
- **4: disarm.**
- **5-13: mode_X.** (Sets a specific flight mode).
- **21-27: mount/camera.**
- **31-36: lights.**
- **42-43: gain_inc/dec.**

Tuning & Behavior

- **Default:** 0.
- **Multiple Buttons:** This parameter is replicated for every button on your controller (BTN2, BTN3, etc.).

BTN1_SFUNCTION (Joystick Button Shift-Function (Instance 1))

DEFAULT: 0	RANGE: 0 111

Joystick Button Shift-Function (BTN1_SFUNCTION)

Description

`BTN1_SFUNCTION` allows you to double the number of commands available on your joystick.

If any button on your controller is assigned to **Function 1 (shift)**, then pressing this button (Button 1) while that shift button is held down will trigger the action defined here instead of the primary `BTN1_FUNCTION`.

BTN_DELAY (Button Delay (Invalid/Legacy))

DEFAULT: 300	RANGE: 0 3000	UNITS: ms

Button Delay (BTN_DELAY)

Description

This parameter is not found as BTN_DELAY in the main codebase. It appears to be a miscategorized OSD parameter.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





BTN_ENABLE (Enable button reporting)

DEFAULT: 0	RANGE: 0 1

Enable button reporting (BTN_ENABLE)

Description

Global switch to enable the button interface module. This allows physical buttons connected to the flight controller's GPIO pins to trigger auxiliary functions.

Tuning & Behavior

- **Default Value:** 0 (Disabled)
- **Requires reboot to make other BTN_ parameters visible in the GCS.**

BTN_FUNC1 (Button Pin 1 RC Channel function)

DEFAULT: 0

Button Pin 1 RC Channel function (BTN_FUNC1)

Description

Selects the action (Auxiliary Function) triggered when the button connected to PIN1 is pressed. Uses the same function IDs as RCx_OPTION.

BTN_FUNC2 (Button Pin 2 RC Channel function)

DEFAULT: 0

Button Pin 2 RC Channel function (BTN_FUNC2)

Description

Selects the action triggered when the button connected to PIN2 is pressed.

BTN_FUNC3 (Button Pin 3 RC Channel function)

DEFAULT: 0

Button Pin 3 RC Channel function (BTN_FUNC3)

Description

Selects the action triggered when the button connected to PIN3 is pressed.

BTN_FUNC4 (Button Pin 4 RC Channel function)

DEFAULT: 0

Button Pin 4 RC Channel function (BTN_FUNC4)

Description

Selects the action triggered when the button connected to PIN4 is pressed.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





BTN_FUNCTION (Function for button)

DEFAULT: 0	RANGE: 0 100

Function for button (BTN_FUNCTION)

Description

This parameter assigns a **Primary Function** to a specific button on a connected USB Joystick or Gamepad (communicating via MAVLink). This allows for direct control of vehicle modes, camera triggers, relays, and servos from the controller.

The Mathematics

Enum Map.

The Engineer's View

Defined in `libraries/AP_JSButton/AP_JSButton.cpp`.

- **0:** Disabled.
- **1:** Mode Change (Manual/Stabilize).
- **...** (See Wiki for full list).

Tuning & Behavior

- **Default Value:** 0 (Disabled)
- **Setup:** Identify the button ID from your GCS/Driver, then set the corresponding `BTN_FUNCTION` parameter.

BTN_OPTIONS1 (Button Pin 1 Options)

DEFAULT: 0

Button Pin 1 Options (BTN_OPTIONS1)

Description

Advanced configuration for the first button, allowing for inverted logic or PWM signal detection instead of simple high/low voltage.

Tuning & Behavior

- **Default Value:** 0
- **Bitmask:**
  - Bit 0: PWM Input
  - Bit 1: InvertInput

BTN_OPTIONS2 (Button Pin 2 Options)

DEFAULT: 0

Button Pin 2 Options (BTN_OPTIONS2)

Description

Advanced configuration for the second button.

BTN_OPTIONS3 (Button Pin 3 Options)

DEFAULT: 0

Button Pin 3 Options (BTN_OPTIONS3)

Description

Advanced configuration for the third button.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





BTN_OPTIONS4 (Button Pin 4 Options)

DEFAULT: 0

Button Pin 4 Options (BTN_OPTIONS4)

Description

Advanced configuration for the fourth button.

BTN_PIN1 (First button Pin)

DEFAULT: -1

First button Pin (BTN_PIN1)

Description

Specifies the hardware pin connected to the first physical button.

Tuning & Behavior

- **Default Value:** -1 (Disabled)
- **Common Values:** 50-55 for AUXOUT 1-6 pins.

BTN_PIN2 (Second button Pin)

DEFAULT: -1

Second button Pin (BTN_PIN2)

Description

Specifies the hardware pin connected to the second physical button.

BTN_PIN3 (Third button Pin)

DEFAULT: -1

Third button Pin (BTN_PIN3)

Description

Specifies the hardware pin connected to the third physical button.

BTN_PIN4 (Fourth button Pin)

DEFAULT: -1

Fourth button Pin (BTN_PIN4)

Description

Specifies the hardware pin connected to the fourth physical button.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





BTN_REPORT_SEND (Report send time)

DEFAULT: 10	RANGE: 0 3600	UNITS: s

Report send time (BTN_REPORT_SEND)

Description

Determines for how long the flight controller will continue to broadcast a button state change message over MAVLink after the event occurs.

BTN_SFUNCTION (Function for button when the shift mode is toggled on)

DEFAULT: 0	RANGE: 0 100

Function for button when the shift mode is toggled on (BTN_SFUNCTION)

Description

This parameter assigns a **Secondary Function** to a joystick button. This function is triggered instead of the primary `BTN_FUNCTION` when the user is simultaneously holding down the designated "Shift" button (see `BTN_OPTIONS`). This effectively doubles the number of commands available on a limited gamepad.

The Mathematics

Logic:

$$\text{Action} = (\text{Shift Active})?\text{BTN_SFUNCTION} : \text{BTN_FUNCTION}$$

The Engineer's View

Defined in `libraries/AP_JSButton/AP_JSButton.cpp`.

Tuning & Behavior

- **Default Value:** 0 (Disabled)
- **Example:** Map Button 1 to "Mode Loiter" (primary) and "Mode RTL" (secondary/shifted).

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **BUZZER** parameter group controls the behavior of the active piezoelectric buzzer connected to the flight controller. The buzzer provides audible feedback for system status (Arming, GPS Lock, Failsafe) and acts as a "Lost Model Alarm."

Key Concepts

1. Volume Control

- `BUZZER_VOLUME`: Sets the PWM duty cycle for the buzzer pin.
  - **100%:** Full volume.
  - **20%:** Quiet (good for bench testing).
  - **0%:** Silent.

Integration Guide

- **Too Loud?** If the startup tone is annoying during development, set `BUZZER_VOLUME = 10`.
- **Tone Control:** Specific tones (Startup, Arming, Warning) are controlled by the `NTF_BUZZ_` (Notify) parameters, not here.

Developer Notes

- **Library:** `AP_Notify` (ToneAlarm driver).
- **Hardware:** Requires an active buzzer (DC driven), not a passive speaker.

BUZZER_VOLUME (Buzzer volume)

DEFAULT: 100	RANGE: 0 100	UNITS: %

Buzzer volume (BUZZER_VOLUME)

Description

The **Buzzer Volume** parameter adjusts the loudness of the onboard piezoelectric buzzer. This is particularly useful for reducing noise during bench testing or indoor operations ("stealth mode"), while allowing for full volume during field operations to ensure audible status alerts and lost-model location.

The Mathematics

The volume is controlled by varying the duty cycle or hardware gain of the buzzer driver, proportional to the percentage set.

$$\text{Output Power} \propto \text{BUZZER_VOLUME}\%$$

The Engineer's View

This parameter is part of the `AP_Periph` (DroneCAN Peripheral) firmware, defined in `Tools/AP_Periph/Parameters.cpp`. It is read by the `AP_Notify` library. The value is clamped between 0 and 100. On supported hardware, this adjusts the PWM duty cycle driving the buzzer; on simpler hardware (on/off only), values $< 50\%$ might silence it or have no effect, while values $\geq 50\%$ enable it.

Tuning & Behavior

- **Default Value:** 100%
- **Range:** 0 - 100
- **0:** Silent.
- **20-50:** Good for indoor bench testing.
- **100:** Maximum volume for outdoor use.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





CAMERA TRIGGER CONFIGURATION (CAM)

Overview

The **CAM** parameter group configures the **Camera Trigger** system for aerial mapping and photography. It allows the autopilot to automatically trigger the camera shutter at precise distance intervals (`CAM_TRIGG_DIST`) or via RC commands.

ArduPilot supports up to 2 simultaneous cameras (`CAM1` and `CAM2`).

Key Concepts

1. Trigger Type (`CAM_TYPE`)

How does the autopilot talk to the camera?

- **0:** Disabled.
- **1 (Servo):** PWM pulse on a servo rail pin (e.g., IR trigger, Sony Multiport).
- **2 (Relay):** Hard high/low voltage toggle (e.g., CHDK).
- **4 (Mount):** Sends "Shoot" command to the connected Gimbal/Mount.
- **5 (MAVLink):** Sends `CAMERA_TRIGGER` MAVLink message to a smart camera.

2. Distance Triggering (`CAM_TRIGG_DIST`)

The core feature for mapping. The autopilot tracks the distance traveled ("Ground Distance") and triggers the camera every X meters to ensure consistent overlap.

- `CAM_AUTO_ONLY`: If enabled, distance triggering only works in Auto mode (safety).

3. Feedback (`CAM_FEEDBAK_PIN`)

For high-precision RTK mapping (PPK), the autopilot needs to know exactly when the shutter opened, not just when the command was sent.

- Connect the camera's "Hotshoe" flash output to a digital input pin.
- The autopilot logs the precise GPS time of the flash in the `CAM` log message.

Parameter Breakdown

- `CAM_DURATION`: How long to hold the button down (Relay/Servo).
- `CAM_SERVO_ON` / `OFF`: PWM values for "Push" and "Release".
- `CAM_MIN_INTERVAL`: Minimum time between shots (prevents buffer overflow).

Integration Guide

Setting up a Servo Trigger

1. **Pin:** Map a servo output to `CameraTrigger` (Function 10).
2. **Type:** Set `CAM_TYPE = 1`.
3. **Values:** Set `CAM_SERVO_ON` to the PWM that fires the shutter (e.g., 1900) and `OFF` to neutral (e.g., 1100).
4. **Test:** Use "Trigger Camera NOW" in the GCS.

Developer Notes

- **Library:** `libraries/AP_Camera`
- **Logging:** Generates `TRIG` (commanded) and `CAM` (feedback) log messages.

CAM1_DURATION (Camera shutter duration held open)

DEFAULT: 0.1	RANGE: 0 5	UNITS: s

Camera shutter duration held open

Note: This parameter functions identically to CAM_DURATION.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





CAM1_FEEDBAK_PIN (Camera feedback pin)

DEFAULT: -1	RANGE: null

Camera feedback pin

Note: This parameter functions identically to CAM_FEEDBAK_PIN.

CAM1_FEEDBAK_POL (Camera feedback pin polarity)

DEFAULT: 1	RANGE: null

Camera feedback pin polarity

Note: This parameter functions identically to CAM_FEEDBAK_POL.

CAM1_HFOV (Camera horizontal field of view)

DEFAULT: 0	RANGE: 0 360	UNITS: deg

Camera horizontal field of view

Note: This parameter functions identically to CAM_HFOV.

CAM1_INTRVAL_MIN (Camera minimum time interval between photos)

DEFAULT: 0	RANGE: 0 10	UNITS: s

Camera minimum time interval between photos

Note: This parameter functions identically to CAM_INTRVAL_MIN.

CAM1_MNT_INST (Camera Mount instance)

DEFAULT: 0	RANGE: null

Camera Mount instance

Note: This parameter functions identically to CAM_MNT_INST.

CAM1_OPTIONS (Camera options)

DEFAULT: 0	RANGE: null

Camera options

Note: This parameter functions identically to CAM_OPTIONS.

CAM1_RELAY_ON (Camera relay ON value)

DEFAULT: 1	RANGE: null

Camera relay ON value

Note: This parameter functions identically to CAM_RELAY_ON.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





CAM1_SERVO_OFF (Camera servo OFF PWM value)

DEFAULT: 1100	RANGE: 1000 2000	UNITS: PWM

Camera servo OFF PWM value

Note: This parameter functions identically to CAM_SERVO_OFF.

CAM1_SERVO_ON (Camera servo ON PWM value)

DEFAULT: 1300	RANGE: 1000 2000	UNITS: PWM

Camera servo ON PWM value

Note: This parameter functions identically to CAM_SERVO_ON.

CAM1_TRIGG_DIST (Camera trigger distance)

DEFAULT: 0	RANGE: 0 1000	UNITS: m

Camera trigger distance

Note: This parameter functions identically to CAM_TRIGG_DIST.

CAM1_TYPE (Camera shutter (trigger) type)

DEFAULT: 0	RANGE: null

Camera shutter (trigger) type

Note: This parameter functions identically to CAM_TYPE.

CAM1_VFOV (Camera vertical field of view)

DEFAULT: 0	RANGE: 0 180	UNITS: deg

Camera vertical field of view

Note: This parameter functions identically to CAM_VFOV.

CAM2_DURATION (Camera shutter duration held open)

DEFAULT: 0.1	RANGE: 0 5	UNITS: s

Camera shutter duration held open

Note: This parameter functions identically to CAM_DURATION.

CAM2_FEEDBAK_PIN (Camera feedback pin)

DEFAULT: -1	RANGE: null

Camera feedback pin

Note: This parameter functions identically to CAM_FEEDBAK_PIN.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





CAM2_FEEDBAK_POL (Camera feedback pin polarity)

DEFAULT: 1	RANGE: null

Camera feedback pin polarity

Note: This parameter functions identically to CAM_FEEDBAK_POL.

CAM2_HFOV (Camera horizontal field of view)

DEFAULT: 0	RANGE: 0 360	UNITS: deg

Camera horizontal field of view

Note: This parameter functions identically to CAM_HFOV.

CAM2_INTRVAL_MIN (Camera minimum time interval between photos)

DEFAULT: 0	RANGE: 0 10	UNITS: s

Camera minimum time interval between photos

Note: This parameter functions identically to CAM_INTRVAL_MIN.

CAM2_MNT_INST (Camera Mount instance)

DEFAULT: 0	RANGE: null

Camera Mount instance

Note: This parameter functions identically to CAM_MNT_INST.

CAM2_OPTIONS (Camera options)

DEFAULT: 0	RANGE: null

Camera options

Note: This parameter functions identically to CAM_OPTIONS.

CAM2_RELAY_ON (Camera relay ON value)

DEFAULT: 1	RANGE: null

Camera relay ON value

Note: This parameter functions identically to CAM_RELAY_ON.

CAM2_SERVO_OFF (Camera servo OFF PWM value)

DEFAULT: 1100	RANGE: 1000 2000	UNITS: PWM

Camera servo OFF PWM value

Note: This parameter functions identically to CAM_SERVO_OFF.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





CAM2_SERVO_ON (Camera servo ON PWM value)

DEFAULT: 1300	RANGE: 1000 2000	UNITS: PWM

Camera servo ON PWM value

Note: This parameter functions identically to CAM_SERVO_ON.

CAM2_TRIGG_DIST (Camera trigger distance)

DEFAULT: 0	RANGE: 0 1000	UNITS: m

Camera trigger distance

Note: This parameter functions identically to CAM_TRIGG_DIST.

CAM2_TYPE (Camera shutter (trigger) type)

DEFAULT: 0	RANGE: null

Camera shutter (trigger) type

Note: This parameter functions identically to CAM_TYPE.

CAM2_VFOV (Camera vertical field of view)

DEFAULT: 0	RANGE: 0 180	UNITS: deg

Camera vertical field of view

Note: This parameter functions identically to CAM_VFOV.

CAM_AUTO_ONLY (Distance-triggering in AUTO mode only)

DEFAULT: 0	RANGE: 0 1

Distance-triggering in AUTO mode only (CAM_AUTO_ONLY)

Description

Restricts the CAM_TRIGG_DIST functionality to autonomous modes.

CAM_DURATION (Camera shutter duration)

DEFAULT: 0.1	RANGE: 0 5	UNITS: s

Camera shutter duration (CAM_DURATION)

Description

Duration of the trigger pulse sent to the camera. For most digital cameras, a short pulse (0.1s to 0.5s) is sufficient to trigger a capture.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





CAM_FEEDBAK_PIN (Camera feedback pin)

DEFAULT: -1

Camera feedback pin (CAM_FEEDBAK_PIN)

Description

Specifies the hardware pin connected to the camera's hot shoe or flash feedback port. This allows ArduPilot to log the *exact* moment the shutter opened, rather than just the moment the command was sent.

CAM_FEEDBAK_POL (Camera feedback pin polarity)

DEFAULT: 1	RANGE: 0 1

Camera feedback pin polarity (CAM_FEEDBAK_POL)

Description

Defines whether the feedback signal from the camera is active-high or active-low.

CAM_HFOV (Camera horizontal field of view)

DEFAULT: 0	RANGE: 0 360	UNITS: deg

Camera horizontal field of view (CAM_HFOV)

Description

Used for geo-tagging and FOV visualization in the GCS.

CAM_INTRVAL_MIN (Camera minimum time interval)

DEFAULT: 0	RANGE: 0 10	UNITS: s

Camera minimum time interval (CAM_INTRVAL_MIN)

Description

A rate-limiting safety feature. Photos will not be triggered faster than this interval, even if other trigger conditions (like CAM_TRIGG_DIST) are met. This allows the camera hardware time to save images.

CAM_MAX_ROLL (Maximum photo roll angle)

DEFAULT: 0	RANGE: 0 180	UNITS: deg

Maximum photo roll angle (CAM_MAX_ROLL)

Description

A quality-control feature that prevents photos from being taken during high-bank maneuvers, ensuring images are relatively level with the ground.

- **0 disables the check.**

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





CAM_MNT_INST (Camera Mount instance)

DEFAULT: 0	RANGE: 0 1

Camera Mount instance (CAM_MNT_INST)

Description

Links the camera instance to a specific gimbal (mount) instance for coordinated control.

CAM_OPTIONS (Camera options)

DEFAULT: 0

Camera options (CAM_OPTIONS)

Description

Configuration bitmask for various camera behaviors.

Tuning & Behavior

- **Default Value:** 0
- **Bitmask:**
  - Bit 0: Recording Starts at arming and stops at disarming

CAM_POS (Camera Position Offset (Invalid/Legacy))

DEFAULT: 0.0	RANGE: -5 5	UNITS: m

Camera Position Offset (CAM_POS)

Description

This parameter is not found as CAM_POS in the main camera library. It appears to be a miscategorized Precision Landing parameter.

CAM_RC_BTN_DELAY (RunCam Press Hold Time)

DEFAULT: 300	RANGE: 100 1000	UNITS: ms

RunCam Press Hold Time (CAM_RC_BTN_DELAY)

Description

`CAM_RC_BTN_DELAY` defines the duration of each "Click" sent to the camera.

When the autopilot simulates a button press (e.g. to start a recording), it must "hold" that signal for a minimum amount of time for the camera's internal processor to register it. If this value is too low, the camera may ignore commands.

Tuning & Behavior

- **Default:** 300 ms.
- **Significance:** Most RunCam models work best with a 300ms pulse. If your camera is intermittently ignoring start/stop commands, try increasing this to **500ms**.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





CAM_RC_BT_DELAY (RunCam Button Delay)

DEFAULT: 300	RANGE: 0 1000	UNITS: ms

RunCam Button Delay (CAM_RC_BT_DELAY)

Description

`CAM_RC_BT_DELAY` determines the navigation speed of the RunCam OSD menu.

When you use your transmitter sticks to move through the camera's settings menu, the autopilot sends "Virtual Button" pulses. This delay ensures that the pulses aren't so fast that the camera skips menu items or becomes unresponsive.

Tuning & Behavior

- **Default:** 300 ms.
- **Recommendation:** Leave at **300**. If the menu feels sluggish, you can try reducing to **200**.

CAM_RC_CONTROL (RunCam Control Options)

DEFAULT: 1	RANGE: 0 15

RunCam Control Options (CAM_RC_CONTROL)

Description

`CAM_RC_CONTROL` defines the "Magic Gestures" used to control your RunCam actions from the transmitter.

It is a bitmask to enable specific control methods:

- **Bit 0 (1): Roll Right.** Enter the camera menu by holding the Roll stick to the right while disarmed.
- **Bit 1 (2): Pitch Down.**
- **Bit 2 (4): Auto-Record.** Starts recording automatically as soon as the drone is armed.

Tuning & Behavior

- **Default:** 1 (Roll Right).
- **Recommendation:** Enable **Bit 2 (4)** if you want to ensure you always capture your flight footage without having to remember to press a button.

CAM_RC_FEATURES (RunCam Feature Bitmask)

DEFAULT: 0	RANGE: 0 15

RunCam Feature Bitmask (CAM_RC_FEATURES)

Description

`CAM_RC_FEATURES` allows you to customize what ArduPilot can do with your RunCam.

- **Bit 0 (1):** Enable Start/Stop Recording.
- **Bit 1 (2):** Enable OSD Menu control.
- **Bit 2 (4):** Enable Camera Power Control (if supported by hardware).

CAM_RC_MDE_DELAY (RunCam Mode Transition Delay)

DEFAULT: 500	RANGE: 0 2000	UNITS: ms

RunCam Mode Transition Delay (CAM_RC_MDE_DELAY)

Description

`CAM_RC_MDE_DELAY` handles the timing of mode switches (e.g., from taking a photo to starting a video).

Action cameras often take a moment to initialize the storage buffer or change sensor resolutions. This parameter ensures the autopilot waits for the camera to be ready before sending the next command.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





CAM_RC_TYPE (RunCam Device Type)

DEFAULT: 0	RANGE: 0 5

RunCam Device Type (CAM_RC_TYPE)

Description

`CAM_RC_TYPE` enables serial control for **RunCam** action cameras (like the Split, Hybrid, or Thumb).

By connecting the camera's UART pins to the flight controller and setting this parameter, you can start/stop recording and navigate the camera's internal OSD menu using your transmitter sticks.

- **0: Disabled.**
- **1: RunCam Split.**
- **2: RunCam Hybrid.**
- **5: RunCam Thumb.**

Tuning & Behavior

- **Default:** 0.
- **Requirement:** Requires a serial port to be configured for RunCam protocol (`SERIALx_PROTOCOL` = 26).
- **Benefit:** Allows for hands-free recording control, ensuring you never forget to "hit record" before takeoff.

CAM_RELAY_ON (Camera relay ON value)

DEFAULT: 1	RANGE: 0 1

Camera relay ON value (CAM_RELAY_ON)

Description

Configures the logic level for relay-based camera triggering.

CAM_SERVO_OFF (Camera servo OFF PWM value)

DEFAULT: 1100	RANGE: 1000 2000	UNITS: PWM

Camera servo OFF PWM value (CAM_SERVO_OFF)

Description

The default (idle) PWM value for the camera trigger servo.

CAM_SERVO_ON (Camera servo ON PWM value)

DEFAULT: 1300	RANGE: 1000 2000	UNITS: PWM

Camera servo ON PWM value (CAM_SERVO_ON)

Description

The PWM value sent to the camera trigger servo when a photo is requested.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





CAM_TRIGG_DIST (Camera trigger distance)

DEFAULT: 0	RANGE: 0 1000	UNITS: m

Camera trigger distance (CAM_TRIGG_DIST)

Description

Automates aerial photography by triggering the camera every time the aircraft travels a specified distance.

Tuning & Behavior

- **Default Value:** 0 (Disabled)
- **Useful for mapping missions to ensure consistent overlap between photos.**

CAM_TYPE (Camera shutter (trigger) type)

DEFAULT: 0

Camera shutter (trigger) type (CAM_TYPE)

Description

Defines the communication method used to trigger the camera.

Tuning & Behavior

- **Default Value:** 0 (None)
- **Values:** 0:None, 1:Servo, 2:Relay, 3:GoPro (Solo), 4:Mount (Siyi/Topotek/Viewpro/Xacti), 5:MAVLink, 6:MAVLinkCamV2, 7:Scripting

CAM_VFOV (Camera vertical field of view)

DEFAULT: 0	RANGE: 0 180	UNITS: deg

Camera vertical field of view (CAM_VFOV)

Description

Used for geo-tagging and FOV visualization in the GCS.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **CAN** parameter group configures the **Controller Area Network (CAN)** interfaces on the autopilot. CAN is a robust, differential signaling bus used for connecting critical peripherals (GPS, Compass, ESCs, Servos) with high reliability and noise immunity.

ArduPilot supports multiple CAN drivers (`CAN_Dn_...`), each mapped to a physical port (`CAN_Px_...`).

Key Concepts

1. Port vs. Driver

- **Port (`CAN_Px_...`):** Configures the physical layer (Bitrate, Electrical).
  - `CAN_P1_DRIVER` : Assigns a logical driver (1 or 2) to physical port 1.
  - `CAN_P1_BITRATE` : Speed (typically 1000000 for 1Mbit).
- **Driver (`CAN_Dx_...`):** Configures the protocol layer (DroneCAN, PiccoloCAN, KDE, etc.).
  - `CAN_D1_PROTOCOL` : Selects the language spoken on this driver (1 = DroneCAN).

2. DroneCAN (formerly UAVCAN)

The most common protocol.

- `CAN_Dn_UC_NODE` : The Node ID of the autopilot itself.
- `CAN_Dn_UC_ESC_BM` : Bitmask to detect DroneCAN ESCs.

3. CAN FD (Flexible Data-Rate)

Newer H7 boards support CAN FD for higher speeds (up to 5Mbit).

- `CAN_P1_FDBITRATE` : Data phase bitrate.
- `CAN_FDMODE` : Enables FD frame format.

Parameter Breakdown

Physical Layer

- `CAN_P1_DRIVER` : 0=Disabled, 1=First Driver, 2=Second Driver.
- `CAN_P1_BITRATE` : 1000000 (1Mbit) is standard.

Protocol Layer (Driver 1)

- `CAN_D1_PROTOCOL` :
  - 1: DroneCAN
  - 4: PiccoloCAN
  - 7: KDECAN
  - 10: ToshibaCAN
- `CAN_D1_UC_OPTION` : Options for DroneCAN (e.g., enable DNA server).

Integration Guide

Setting up a DroneCAN GPS (Here3)

1. **Driver:** Set `CAN_P1_DRIVER = 1` .
2. **Protocol:** Set `CAN_D1_PROTOCOL = 1` (DroneCAN).
3. **GPS Type:** Set `GPS_TYPE = 9` (DroneCAN).
4. **Compass:** Set `COMPASS_TYPEMASK` to allow DroneCAN (usually default).
5. **Reboot:** The GPS should appear.

Troubleshooting

- **No Devices:** Check termination. A 120-ohm resistor is required at each end of the CAN bus.
- **Node ID Conflicts:** Use the "DroneCAN/UAVCAN" inspector in Mission Planner to see active nodes.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Developer Notes

- **Library:** `libraries/AP_CANManager`
- **Tunnels:** Support for tunneling Serial-over-CAN (`CAN_D1_UC_SER_EN`).

CAN_D1_PC_ECU_ID (PiccoloCAN ECU Node ID)

DEFAULT: 0	RANGE: 0 255

PiccoloCAN ECU Node ID (CAN_D1_PC_ECU_ID)

Description

`CAN_D1_PC_ECU_ID` identifies the destination for engine control commands over PiccoloCAN.

- **0:** Disabled.
- **1-254:** Specific ECU Node ID.
- **255:** Broadcast to all ECUs.

Tuning & Behavior

- **Default Value:** 0.
- **Recommendation:** Match the ID configured in your Piccolo ECU.

CAN_D1_PC_ECU_RT (PiccoloCAN ECU Output Rate (Driver 1))

DEFAULT: 50	RANGE: 1 500	UNITS: Hz

PiccoloCAN ECU Output Rate (CAN_D1_PC_ECU_RT)

Description

`CAN_D1_PC_ECU_RT` sets the update frequency for an internal combustion engine's Electronic Control Unit (ECU) connected via PiccoloCAN.

CAN_D1_PC_ESC_BM (PiccoloCAN ESC Bitmask)

DEFAULT: 0	RANGE: 0 4294967295

PiccoloCAN ESC Bitmask (CAN_D1_PC_ESC_BM)

Description

`CAN_D1_PC_ESC_BM` identifies which motor channels use the PiccoloCAN protocol.

PiccoloCAN is a specialized CAN protocol used primarily in industrial and long-endurance airframes. This bitmask tells ArduPilot which of the internal motor outputs should be translated into PiccoloCAN commands on this driver.

Tuning & Behavior

- **Default Value:** 0.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY

CAN_D1_PC_ESC_RT (PiccoloCAN ESC Output Rate (Driver 1))

DEFAULT: 50	RANGE: 1 500	UNITS: Hz

PiccoloCAN ESC Output Rate (CAN_D1_PC_ESC_RT)

Description

`CAN_D1_PC_ESC_RT` sets the update speed for motors connected via the PiccoloCAN protocol.

- **Higher Value (400-500Hz):** Provides faster motor response and better flight stability, similar to high-speed DShot.
- **Lower Value (50Hz):** Standard update rate.

Tuning & Behavior

- **Recommendation:** Set to **400** for high-performance multirotors. Ensure your PiccoloCAN ESCs support the requested frequency.

CAN_D1_PC_SRV_BM (PiccoloCAN Servo Bitmask (Driver 1))

DEFAULT: 65535	RANGE: 0 65535

PiccoloCAN Servo Bitmask (CAN_D1_PC_SRV_BM)

Description

`CAN_D1_PC_SRV_BM` selects which servos on the drone are controlled via the PiccoloCAN network.

- **Bit 0 (1):** Servo 1
- **Bit 1 (2):** Servo 2
- ...

Tuning & Behavior

- **Default Value:** 65535 (All 16 channels enabled).

CAN_D1_PC_SRV_RT (PiccoloCAN Servo Output Rate (Driver 1))

DEFAULT: 50	RANGE: 1 500	UNITS: Hz

PiccoloCAN Servo Output Rate (CAN_D1_PC_SRV_RT)

Description

`CAN_D1_PC_SRV_RT` defines the update frequency for control surfaces (ailerons, elevators) using PiccoloCAN servos.

CAN_D1_PROTOCOL (Primary protocol for virtual driver 1)

DEFAULT: 1

Primary protocol for virtual driver 1 (CAN_D1_PROTOCOL)

Description

Defines the software protocol (e.g., DroneCAN, PiccoloCAN) used by the first virtual CAN driver.

Tuning & Behavior

- **Default Value:** 1 (DroneCAN)
- **Values:** 0:Disabled, 1:DroneCAN, 4:PiccoloCAN, 6:EFI_NWPMU, 8:KDECAN, 10:Scripting

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





CAN_D1_PROTOCOL2 (CAN Driver 1 Protocol 2)

DEFAULT: 0	RANGE: 0 12

CAN Driver 1 Protocol 2 (CAN_D1_PROTOCOL2)

Description

`CAN_D1_PROTOCOL2` allows "Dual Mode" operation on a single CAN bus.

Typically, you can only run one protocol per bus (e.g., DroneCAN). However, some protocols use different message ID lengths (29-bit vs 11-bit) and can coexist. This parameter allows you to enable a secondary 11-bit protocol (like some BMS or ESC telemetry formats) on the same wires as your main DroneCAN network.

- **0: Disabled.**
- **10: EFL_ESCORP.** 11-bit ESC telemetry.

CAN_D1_UC_ESC_BM (DroneCAN ESC Bitmask)

DEFAULT: 0	RANGE: 0 4294967295

DroneCAN ESC Bitmask (CAN_D1_UC_ESC_BM)

Description

`CAN_D1_UC_ESC_BM` maps specific ArduPilot servo output channels to DroneCAN ESC commands.

If you have 4 DroneCAN ESCs controlling motors 1-4, you must set this bitmask to include channels 1, 2, 3, and 4. This tells the flight controller: "Don't send PWM for these motors; send a digital CAN message instead."

- **Bit 0 (1):** Motor/Servo 1
- **Bit 1 (2):** Motor/Servo 2
- **Bit 2 (4):** Motor/Servo 3
- **Bit 3 (8):** Motor/Servo 4

The Engineer's View

Defined in `AP_DroneCAN.cpp`.
When a bit is set, the standard PWM output for that channel is suppressed (or acts as a backup, depending on configuration), and `uavcan.equipment.esc.RawCommand` messages are broadcast on the CAN bus.

Tuning & Behavior

- **Default Value:** 0
- **QuadCopter (Motors 1-4):** Set to **15** (1+2+4+8).
- **HexaCopter (Motors 1-6):** Set to **63**.

CAN_D1_UC_ESC_OF (DroneCAN ESC Offset)

DEFAULT: 0	RANGE: 0 18

DroneCAN ESC Offset (CAN_D1_UC_ESC_OF)

Description

`CAN_D1_UC_ESC_OF` shifts the motor numbering on the CAN bus.

If you have 8 motors, but your CAN ESCs are programmed as IDs 9 through 16, you can use this offset to align the autopilot's internal motor 1 with CAN ID 9.

Tuning & Behavior

- **Default Value:** 0.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





CAN_D1_UC_ESC_RV (DroneCAN Reversible ESC Bitmask)

DEFAULT: 0	RANGE: 0 4294967295

DroneCAN Reversible ESC Bitmask (CAN_D1_UC_ESC_RV)

Description

CAN_D1_UC_ESC_RV identifies which CAN-based ESCs support reversing their motor direction in real-time.

- **Standard ESC:** Only accepts 0..100% speed.
- **Reversible ESC:** Accepts -100%..+100% speed.

Tuning & Behavior

- **Default Value:** 0.
- **Recommendation:** Set bits for all motors that need to provide reverse thrust (e.g., for 3D quads or rovers).

CAN_D1_UC_NODE (DroneCAN Node ID (Driver 1))

DEFAULT: 0	RANGE: 0 125

DroneCAN Node ID (CAN_D1_UC_NODE)

Description

CAN_D1_UC_NODE sets the autopilot's own address on the DroneCAN network.

Every device on the bus (GPS, ESC, Battery) needs a unique ID.

- **0 (Default):** Dynamic allocation. The autopilot will claim the first available ID (usually 10).
- **1-125:** Fixed ID.

Tuning & Behavior

- **Recommendation:** Leave at **0** for most setups. If you have a complex network with many devices and want to enforce a strict ID map, set this to a fixed value (e.g. 10).

CAN_D1_UC_NTF_RT (DroneCAN Notify Transmit Rate)

DEFAULT: 20	RANGE: 1 200	UNITS: Hz

DroneCAN Notify Transmit Rate (CAN_D1_UC_NTF_RT)

Description

CAN_D1_UC_NTF_RT controls how often the flight controller updates CAN-based status lights (LEDs) and buzzers.

If you are using a CAN LED (like the NeoPixel modules on some GPS units), this setting determines the smoothness of color transitions and how quickly the lights react to mode changes or arming events.

Tuning & Behavior

- **Default Value:** 20 Hz.
- **Recommendation:** Leave at **20 Hz**. This is fast enough for smooth visual feedback without wasting bus bandwidth.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





CAN_D1_UC_OPTION (DroneCAN Options)

DEFAULT: 0	RANGE: 0 1024

DroneCAN Options (CAN_D1_UC_OPTION)

Description

`CAN_D1_UC_OPTION` provides advanced configuration flags for the DroneCAN stack.

- **1 (Bit 0):** Clear DNA Database. Force re-allocation of Node IDs on next boot.
- **4 (Bit 2):** Enable CAN FD (Flexible Data-rate) if supported by hardware.
- **16 (Bit 4):** Send Servo output as raw PWM (instead of normalized -1..+1).

Tuning & Behavior

- **Default Value:** 0
- **Recommendation:**
  - **CAN FD:** Enable this if you have CAN FD peripherals (requires H7 or G4 processor).
  - **DNA Reset:** Set to **1** and reboot if you have duplicate Node ID errors. It will clear itself back to 0.

CAN_D1_UC_POOL (DroneCAN Memory Pool Size)

DEFAULT: 8192	RANGE: 1024 32768	UNITS: bytes

DroneCAN Memory Pool Size (CAN_D1_UC_POOL)

Description

`CAN_D1_UC_POOL` sets the RAM budget for the DroneCAN communications stack.

The CAN bus is a "Message Based" network. As more devices (GPS, Compass, ESCs) send data, the flight controller needs a larger buffer to store and process those messages. If this pool is too small, messages will be dropped, leading to "Sensor Unhealthy" errors or sluggish control.

The Engineer's View

Defined in `AP_DroneCAN.cpp`.
This memory is allocated from the system heap during driver initialization. It is used by the `libcanard` library to manage the transfer buffers and reassemble multi-frame transfers.

Tuning & Behavior

- **Default Value:** 8192 (8KB) on most boards; 16384 (16KB) on H7 boards.
- **High Load:** If you have many CAN ESCs and sensors (high bus utilization), increase this to **16384** or **32768**.
- **Low Memory Boards:** On 1MB Flash boards (like F4), be careful increasing this as it takes away from the main system memory.
- **Reboot Required:** Yes.

CAN_D1_UC_RLY_RT (DroneCAN Relay Transmit Rate)

DEFAULT: 0	RANGE: 0 200	UNITS: Hz

DroneCAN Relay Transmit Rate (CAN_D1_UC_RLY_RT)

Description

`CAN_D1_UC_RLY_RT` sets the heartbeat rate for CAN-based relays.

- **0:** Send only on change (standard).
- **Non-Zero:** Periodically re-broadcast the relay state at this rate. This is safer for hardware that needs a "Keep Alive" signal.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





CAN_D1_UC_S1_BD (DroneCAN Serial 1 Baud Rate)

DEFAULT: 57	RANGE: 1 2000

DroneCAN Serial 1 Baud Rate (CAN_D1_UC_S1_BD)

Description

`CAN_D1_UC_S1_BD` sets the speed of the "Virtual UART" bridge.

This allows you to tunnel serial data (like GPS or telemetry) through the CAN bus. The value is in kilobaud (e.g., 57 = 57600 baud).

Tuning & Behavior

- **Default Value:** 57 (57600).
- **Recommendation:** Set this to match the physical baud rate of the device plugged into your remote CAN node (e.g. 115 for a high-speed GPS).

CAN_D1_UC_S1_IDX (DroneCAN Serial 1 Bridge Index)

DEFAULT: -1	RANGE: -1 10

DroneCAN Serial 1 Bridge Index (CAN_D1_UC_S1_IDX)

Description

`CAN_D1_UC_S1_IDX` links a specific autopilot serial manager instance to the CAN bridge.

If you want the device on the other end of the CAN bus to appear as "Serial 1" on your flight controller, set this to 1.

Tuning & Behavior

- **Default Value:** -1 (Disabled).

CAN_D1_UC_S1_NOD (DroneCAN Serial 1 Node ID)

DEFAULT: 0	RANGE: 0 127

DroneCAN Serial 1 Node ID (CAN_D1_UC_S1_NOD)

Description

`CAN_D1_UC_S1_NOD` creates a virtual serial bridge over the CAN bus.

This allows you to connect a Serial GPS to a "Node" (like an AP_Periph adapter) and have ArduPilot talk to it as if it were plugged directly into a UART on the flight controller. This parameter sets the ID of that remote Node.

Tuning & Behavior

- **Default Value:** 0 (Disabled).
- **Recommendation:** Set to the Node ID of your DroneCAN-to-Serial adapter.

CAN_D1_UC_S1_PRO (DroneCAN Serial 1 Protocol)

DEFAULT: -1	RANGE: -1 60

DroneCAN Serial 1 Protocol (CAN_D1_UC_S1_PRO)

Description

`CAN_D1_UC_S1_PRO` defines what language the serial bridge speaks. It uses the standard `SERIAL_PROTOCOL` values (e.g., 5 for GPS, 2 for MAVLink).

Tuning & Behavior

- **Default Value:** -1 (Same as Serial Manager).

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





CAN_D1_UC_S2_BD (DroneCAN Serial default baud rate)

DEFAULT: 57600	RANGE: null	UNITS: null

DroneCAN Serial default baud rate

Note: This parameter functions identically to CAN_D1_UC_S1_BD.

CAN_D1_UC_S2_IDX (Serial port number on remote CAN node)

DEFAULT: -1	RANGE: null	UNITS: null

Serial port number on remote CAN node

Note: This parameter functions identically to CAN_D1_UC_S1_IDX.

CAN_D1_UC_S2_NOD (Serial CAN remote node number)

DEFAULT: 0	RANGE: null	UNITS: null

Serial CAN remote node number

Note: This parameter functions identically to CAN_D1_UC_S1_NOD.

CAN_D1_UC_S2_PRO (Serial protocol of DroneCAN serial port)

DEFAULT: -1	RANGE: null	UNITS: null

Serial protocol of DroneCAN serial port

Note: This parameter functions identically to CAN_D1_UC_S1_PRO.

CAN_D1_UC_S3_BD (Serial baud rate on remote CAN node)

DEFAULT: 57600	RANGE: null	UNITS: null

Serial baud rate on remote CAN node

Note: This parameter functions identically to CAN_D1_UC_S1_BD.

CAN_D1_UC_S3_IDX (Serial port number on remote CAN node)

DEFAULT: 0	RANGE: null	UNITS: null

Serial port number on remote CAN node

Note: This parameter functions identically to CAN_D1_UC_S1_IDX.

CAN_D1_UC_S3_NOD (Serial CAN remote node number)

DEFAULT: 0	RANGE: null	UNITS: null

Serial CAN remote node number

Note: This parameter functions identically to CAN_D1_UC_S1_NOD.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





CAN_D1_UC_S3_PRO (Serial protocol of DroneCAN serial port)

DEFAULT: -1	RANGE: null	UNITS: null

Serial protocol of DroneCAN serial port

Note: This parameter functions identically to CAN_D1_UC_S1_PRO.

CAN_D1_UC_SER_EN (DroneCAN Serial Bridge Enable)

DEFAULT: 0	RANGE: 0 1

DroneCAN Serial Bridge Enable (CAN_D1_UC_SER_EN)

Description

`CAN_D1_UC_SER_EN` activates the "Virtual Serial Port" system on the first CAN bus.

When enabled, you can map physical serial devices (like a GPS or telemetry radio) that are connected to remote DroneCAN nodes (like an AP_Periph adapter) as if they were plugged directly into the flight controller.

Tuning & Behavior

- **0:** Disabled.
- **1:** Enabled.
- **Reboot Required:** Yes.

CAN_D1_UC_SRV_BM (DroneCAN Servo Bitmask)

DEFAULT: 0	RANGE: 0 4294967295

DroneCAN Servo Bitmask (CAN_D1_UC_SRV_BM)

Description

`CAN_D1_UC_SRV_BM` maps specific output channels to DroneCAN Servo commands.

If you are using CAN-based servos (like the Hitag series), you use this bitmask to tell ArduPilot which logical outputs (Aileron, Elevator, etc.) should be routed to the CAN bus instead of the PWM rail.

The Engineer's View

Sends `uavcan.equipment.actuator.ArrayCommand`.
Can be configured to send raw PWM values or normalized -1 to +1 commands depending on `CAN_D1_UC_OPTION`.

Tuning & Behavior

- **Default Value:** 0
- **Recommendation:** Only set bits for channels physically connected to CAN servos.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





CAN_D1_UC_SRV_RT (DroneCAN Servo Transmit Rate)

DEFAULT: 50	RANGE: 1 200	UNITS: Hz

DroneCAN Servo Transmit Rate (CAN_D1_UC_SRV_RT)

Description

`CAN_D1_UC_SRV_RT` sets the "Update Frequency" for CAN-based servos.

If you are using DroneCAN servos (or an AP_Periph node acting as a servo adapter), this parameter determines how many times per second the flight controller sends new position commands.

Tuning & Behavior

- **Default Value:** 50 Hz.
- **High Performance:** Set to **100 Hz** or **200 Hz** for modern digital CAN servos to improve control response and stabilization.
- **Bandwidth:** Higher rates use more CAN bus bandwidth. If your bus is congested, keep this at 50Hz.

CAN_D2_PC_ECU_ID (ECU Node ID)

DEFAULT: PICCOLO_CAN_ECU_ID_DEFAULT	RANGE: 0 255	UNITS: null

ECU Node ID

Note: This parameter functions identically to CAN_D1_PC_ECU_ID.

CAN_D2_PC_ECU_RT (ECU command output rate)

DEFAULT: PICCOLO_MSG_RATE_HZ_DEFAULT	RANGE: 1 500	UNITS: Hz

ECU command output rate

Note: This parameter functions identically to CAN_D1_PC_ECU_RT.

CAN_D2_PC_ESC_BM (Output channels to be transmitted as ESC over DroneCAN)

DEFAULT: 0	RANGE: null	UNITS: null

Output channels to be transmitted as ESC over DroneCAN

Note: This parameter functions identically to CAN_D1_PC_ESC_BM.

CAN_D2_PC_ESC_RT (ESC output rate)

DEFAULT: PICCOLO_MSG_RATE_HZ_DEFAULT	RANGE: 1 500	UNITS: Hz

ESC output rate

Note: This parameter functions identically to CAN_D1_PC_ESC_RT.

CAN_D2_PC_SRV_BM (Output channels to be transmitted as servo over DroneCAN)

DEFAULT: 0	RANGE: null	UNITS: null

Output channels to be transmitted as servo over DroneCAN

Note: This parameter functions identically to CAN_D1_PC_SRV_BM.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





CAN_D2_PC_SRV_RT (Servo output rate)

DEFAULT: 50	RANGE: 1 200	UNITS: Hz

Servo output rate

Note: This parameter functions identically to CAN_D1_PC_SRV_RT.

CAN_D2_PROTOCOL (Enable use of specific protocol over virtual driver)

DEFAULT: float(AP_CAN::Protocol::DroneCAN	RANGE: null	UNITS: null

Enable use of specific protocol over virtual driver

Note: This parameter configures instance 2. It functions identically to CAN_D1_PROTOCOL.

CAN_D2_PROTOCOL2 (Secondary protocol with 11 bit CAN addressing)

DEFAULT: float(AP_CAN::Protocol::None	RANGE: null	UNITS: null

Secondary protocol with 11 bit CAN addressing

Note: This parameter functions identically to CAN_D1_PROTOCOL2.

CAN_D2_UC_ESC_BM (Output channels to be transmitted as ESC over DroneCAN)

DEFAULT: 0	RANGE: null	UNITS: null

Output channels to be transmitted as ESC over DroneCAN

Note: This parameter functions identically to CAN_D1_UC_ESC_BM.

CAN_D2_UC_ESC_OF (ESC Output channels offset)

DEFAULT: 0	RANGE: 0 18	UNITS: null

ESC Output channels offset

Note: This parameter functions identically to CAN_D1_UC_ESC_OF.

CAN_D2_UC_ESC_RV (Bitmask for output channels for reversible ESCs over DroneCAN.)

DEFAULT: 0	RANGE: null	UNITS: null

Bitmask for output channels for reversible ESCs over DroneCAN.

Note: This parameter functions identically to CAN_D1_UC_ESC_RV.

CAN_D2_UC_NODE (Own node ID)

DEFAULT: AP_DRONECAN_DEFAULT_NODE	RANGE: 1 125	UNITS: null

Own node ID

Note: This parameter functions identically to CAN_D1_UC_NODE.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





CAN_D2_UC_NTF_RT (Notify State rate)

DEFAULT: 20	RANGE: 1 200	UNITS: Hz

Notify State rate

Note: This parameter functions identically to CAN_D1_UC_NTF_RT.

CAN_D2_UC_OPTION (DroneCAN options)

DEFAULT: 0	RANGE: null	UNITS: null

DroneCAN options

Note: This parameter functions identically to CAN_D1_UC_OPTION.

CAN_D2_UC_POOL (CAN pool size)

DEFAULT: DRONECAN_NODE_POOL_SIZE	RANGE: 1024 16384	UNITS: null

CAN pool size

Note: This parameter functions identically to CAN_D1_UC_POOL.

CAN_D2_UC_RLY_RT (DroneCAN relay output rate)

DEFAULT: 0	RANGE: 0 200	UNITS: Hz

DroneCAN relay output rate

Note: This parameter functions identically to CAN_D1_UC_RLY_RT.

CAN_D2_UC_S1_BD (DroneCAN Serial default baud rate)

DEFAULT: 57600	RANGE: null	UNITS: null

DroneCAN Serial default baud rate

Note: This parameter functions identically to CAN_D1_UC_S1_BD.

CAN_D2_UC_S1_IDX (DroneCAN Serial1 index)

DEFAULT: -1	RANGE: 0 100	UNITS: null

DroneCAN Serial1 index

Note: This parameter functions identically to CAN_D1_UC_S1_IDX.

CAN_D2_UC_S1_NOD (Serial CAN remote node number)

DEFAULT: 0	RANGE: 0 127	UNITS: null

Serial CAN remote node number

Note: This parameter functions identically to CAN_D1_UC_S1_NOD.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY

CAN_D2_UC_S1_PRO (Serial protocol of DroneCAN serial port)

DEFAULT: -1	RANGE: null	UNITS: null

Serial protocol of DroneCAN serial port

Note: This parameter functions identically to CAN_D1_UC_S1_PRO.

CAN_D2_UC_S2_BD (DroneCAN Serial default baud rate)

DEFAULT: 57600	RANGE: null	UNITS: null

DroneCAN Serial default baud rate

Note: This parameter functions identically to CAN_D2_UC_S1_BD.

CAN_D2_UC_S2_IDX (Serial port number on remote CAN node)

DEFAULT: -1	RANGE: null	UNITS: null

Serial port number on remote CAN node

Note: This parameter functions identically to CAN_D2_UC_S1_IDX.

CAN_D2_UC_S2_NOD (Serial CAN remote node number)

DEFAULT: 0	RANGE: null	UNITS: null

Serial CAN remote node number

Note: This parameter functions identically to CAN_D2_UC_S1_NOD.

CAN_D2_UC_S2_PRO (Serial protocol of DroneCAN serial port)

DEFAULT: -1	RANGE: null	UNITS: null

Serial protocol of DroneCAN serial port

Note: This parameter functions identically to CAN_D2_UC_S1_PRO.

CAN_D2_UC_S3_BD (Serial baud rate on remote CAN node)

DEFAULT: 57600	RANGE: null	UNITS: null

Serial baud rate on remote CAN node

Note: This parameter functions identically to CAN_D2_UC_S1_BD.

CAN_D2_UC_S3_IDX (Serial port number on remote CAN node)

DEFAULT: 0	RANGE: null	UNITS: null

Serial port number on remote CAN node

Note: This parameter functions identically to CAN_D2_UC_S1_IDX.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





CAN_D2_UC_S3_NOD (Serial CAN remote node number)

DEFAULT: 0	RANGE: null	UNITS: null

Serial CAN remote node number

Note: This parameter functions identically to CAN_D2_UC_S1_NOD.

CAN_D2_UC_S3_PRO (Serial protocol of DroneCAN serial port)

DEFAULT: -1	RANGE: null	UNITS: null

Serial protocol of DroneCAN serial port

Note: This parameter functions identically to CAN_D2_UC_S1_PRO.

CAN_D2_UC_SER_EN (DroneCAN Serial enable)

DEFAULT: 0	RANGE: null	UNITS: null

DroneCAN Serial enable

Note: This parameter functions identically to CAN_D1_UC_SER_EN.

CAN_D2_UC_SRV_BM (Output channels to be transmitted as servo over DroneCAN)

DEFAULT: 0	RANGE: null	UNITS: null

Output channels to be transmitted as servo over DroneCAN

Note: This parameter functions identically to CAN_D1_UC_SRV_BM.

CAN_D2_UC_SRV_RT (Servo output rate)

DEFAULT: 50	RANGE: 1 200	UNITS: Hz

Servo output rate

Note: This parameter functions identically to CAN_D1_UC_SRV_RT.

CAN_D3_PC_ECU_ID (ECU Node ID)

DEFAULT: PICCOLO_CAN_ECU_ID_DEFAULT	RANGE: 0 255	UNITS: null

ECU Node ID

Note: This parameter functions identically to CAN_D1_PC_ECU_ID.

CAN_D3_PC_ECU_RT (ECU command output rate)

DEFAULT: PICCOLO_MSG_RATE_HZ_DEFAULT	RANGE: 1 500	UNITS: Hz

ECU command output rate

Note: This parameter functions identically to CAN_D1_PC_ECU_RT.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





CAN_D3_PC_ESC_BM (Output channels to be transmitted as ESC over DroneCAN)

DEFAULT: 0	RANGE: null	UNITS: null

Output channels to be transmitted as ESC over DroneCAN

Note: This parameter functions identically to CAN_D1_PC_ESC_BM.

CAN_D3_PC_ESC_RT (ESC output rate)

DEFAULT: PICCOLO_MSG_RATE_HZ_DEFAULT	RANGE: 1 500	UNITS: Hz

ESC output rate

Note: This parameter functions identically to CAN_D1_PC_ESC_RT.

CAN_D3_PC_SRV_BM (Output channels to be transmitted as servo over DroneCAN)

DEFAULT: 0	RANGE: null	UNITS: null

Output channels to be transmitted as servo over DroneCAN

Note: This parameter functions identically to CAN_D1_PC_SRV_BM.

CAN_D3_PC_SRV_RT (Servo output rate)

DEFAULT: 50	RANGE: 1 200	UNITS: Hz

Servo output rate

Note: This parameter functions identically to CAN_D1_PC_SRV_RT.

CAN_D3_PROTOCOL (Enable use of specific protocol over virtual driver)

DEFAULT: float(AP_CAN::Protocol::DroneCAN	RANGE: null	UNITS: null

Enable use of specific protocol over virtual driver

Note: This parameter functions identically to CAN_D1_PROTOCOL.

CAN_D3_PROTOCOL2 (Secondary protocol with 11 bit CAN addressing)

DEFAULT: float(AP_CAN::Protocol::None	RANGE: null	UNITS: null

Secondary protocol with 11 bit CAN addressing

Note: This parameter functions identically to CAN_D1_PROTOCOL2.

CAN_D3_UC_ESC_BM (Output channels to be transmitted as ESC over DroneCAN)

DEFAULT: 0	RANGE: null	UNITS: null

Output channels to be transmitted as ESC over DroneCAN

Note: This parameter functions identically to CAN_D1_UC_ESC_BM.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





CAN_D3_UC_ESC_OF (ESC Output channels offset)

DEFAULT: 0	RANGE: 0 18	UNITS: null

ESC Output channels offset

Note: This parameter functions identically to CAN_D1_UC_ESC_OF.

CAN_D3_UC_ESC_RV (Bitmask for output channels for reversible ESCs over DroneCAN.)

DEFAULT: 0	RANGE: null	UNITS: null

Bitmask for output channels for reversible ESCs over DroneCAN.

Note: This parameter functions identically to CAN_D1_UC_ESC_RV.

CAN_D3_UC_NODE (Own node ID)

DEFAULT: AP_DRONECAN_DEFAULT_NODE	RANGE: 1 125	UNITS: null

Own node ID

Note: This parameter functions identically to CAN_D1_UC_NODE.

CAN_D3_UC_NTF_RT (Notify State rate)

DEFAULT: 20	RANGE: 1 200	UNITS: Hz

Notify State rate

Note: This parameter functions identically to CAN_D1_UC_NTF_RT.

CAN_D3_UC_OPTION (DroneCAN options)

DEFAULT: 0	RANGE: null	UNITS: null

DroneCAN options

Note: This parameter functions identically to CAN_D1_UC_OPTION.

CAN_D3_UC_POOL (CAN pool size)

DEFAULT: DRONECAN_NODE_POOL_SIZE	RANGE: 1024 16384	UNITS: null

CAN pool size

Note: This parameter functions identically to CAN_D1_UC_POOL.

CAN_D3_UC_RLY_RT (DroneCAN relay output rate)

DEFAULT: 0	RANGE: 0 200	UNITS: Hz

DroneCAN relay output rate

Note: This parameter functions identically to CAN_D1_UC_RLY_RT.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





CAN_D3_UC_S1_BD (DroneCAN Serial default baud rate)

DEFAULT: 57600	RANGE: null	UNITS: null

DroneCAN Serial default baud rate

Note: This parameter functions identically to CAN_D1_UC_S1_BD.

CAN_D3_UC_S1_IDX (DroneCAN Serial1 index)

DEFAULT: -1	RANGE: 0 100	UNITS: null

DroneCAN Serial1 index

Note: This parameter functions identically to CAN_D1_UC_S1_IDX.

CAN_D3_UC_S1_NOD (Serial CAN remote node number)

DEFAULT: 0	RANGE: 0 127	UNITS: null

Serial CAN remote node number

Note: This parameter functions identically to CAN_D1_UC_S1_NOD.

CAN_D3_UC_S1_PRO (Serial protocol of DroneCAN serial port)

DEFAULT: -1	RANGE: null	UNITS: null

Serial protocol of DroneCAN serial port

Note: This parameter functions identically to CAN_D1_UC_S1_PRO.

CAN_D3_UC_S2_BD (DroneCAN Serial default baud rate)

DEFAULT: 57600	RANGE: null	UNITS: null

DroneCAN Serial default baud rate

Note: This parameter functions identically to CAN_D3_UC_S1_BD.

CAN_D3_UC_S2_IDX (Serial port number on remote CAN node)

DEFAULT: -1	RANGE: null	UNITS: null

Serial port number on remote CAN node

Note: This parameter functions identically to CAN_D3_UC_S1_IDX.

CAN_D3_UC_S2_NOD (Serial CAN remote node number)

DEFAULT: 0	RANGE: null	UNITS: null

Serial CAN remote node number

Note: This parameter functions identically to CAN_D3_UC_S1_NOD.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





CAN_D3_UC_S2_PRO (Serial protocol of DroneCAN serial port)

DEFAULT: -1	RANGE: null	UNITS: null

Serial protocol of DroneCAN serial port

Note: This parameter functions identically to CAN_D3_UC_S1_PRO.

CAN_D3_UC_S3_BD (Serial baud rate on remote CAN node)

DEFAULT: 57600	RANGE: null	UNITS: null

Serial baud rate on remote CAN node

Note: This parameter functions identically to CAN_D3_UC_S1_BD.

CAN_D3_UC_S3_IDX (Serial port number on remote CAN node)

DEFAULT: 0	RANGE: null	UNITS: null

Serial port number on remote CAN node

Note: This parameter functions identically to CAN_D3_UC_S1_IDX.

CAN_D3_UC_S3_NOD (Serial CAN remote node number)

DEFAULT: 0	RANGE: null	UNITS: null

Serial CAN remote node number

Note: This parameter functions identically to CAN_D3_UC_S1_NOD.

CAN_D3_UC_S3_PRO (Serial protocol of DroneCAN serial port)

DEFAULT: -1	RANGE: null	UNITS: null

Serial protocol of DroneCAN serial port

Note: This parameter functions identically to CAN_D3_UC_S1_PRO.

CAN_D3_UC_SER_EN (DroneCAN Serial enable)

DEFAULT: 0	RANGE: null	UNITS: null

DroneCAN Serial enable

Note: This parameter functions identically to CAN_D1_UC_SER_EN.

CAN_D3_UC_SRV_BM (Output channels to be transmitted as servo over DroneCAN)

DEFAULT: 0	RANGE: null	UNITS: null

Output channels to be transmitted as servo over DroneCAN

Note: This parameter functions identically to CAN_D1_UC_SRV_BM.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





CAN_D3_UC_SRV_RT (Servo output rate)

DEFAULT: 50	RANGE: 1 200	UNITS: Hz

Servo output rate

Note: This parameter functions identically to CAN_D1_UC_SRV_RT.

CAN_FDMODE (Enable CAN FD Mode)

DEFAULT: 0	RANGE: 0 1

Enable CAN FD Mode (CAN_FDMODE)

Description

`CAN_FDMODE` upgrades the CAN bus protocol to **CAN FD** (Flexible Data-rate).

Standard CAN is limited to 8 bytes per frame and typically 1 Mbps. CAN FD supports up to 64 bytes per frame and much higher data rates (up to 5-8 Mbps) for the data payload phase. This reduces bus congestion and latency.

- **0: Disabled (Classic CAN).**
- **1: Enabled (CAN FD).**

Tuning & Behavior

- **Default:** 0.
- **Hardware Requirement:** All devices on the bus MUST support CAN FD. If a single legacy CAN device is connected, communication will fail.
- **AP_Periph:** This parameter is commonly found on AP_Periph nodes (like GPS or Airspeed sensors) to switch them into high-speed mode.

CAN_LOGLEVEL (CAN Loglevel)

DEFAULT: 0	RANGE: 0 4

CAN Loglevel (CAN_LOGLEVEL)

Description

Controls the verbosity of CAN-related diagnostic messages sent to the internal log. Useful for troubleshooting device initialization issues.

Tuning & Behavior

- **Default Value:** 0 (None)
- **Values:** 0:Log None, 1:Log Error, 2:Log Warning, 3:Log Info, 4:Log Everything

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





CAN_MIRROR_PORTS (CAN Port Mirroring)

DEFAULT: 0	RANGE: 0 3

CAN Port Mirroring (CAN_MIRROR_PORTS)

Description

`CAN_MIRROR_PORTS` turns the flight controller (or AP_Periph node) into a CAN Bridge.

Any message received on one port is instantly copied and sent out the other port. This is useful for creating a "Pass-Through" node or daisy-chaining buses that are electrically separate.

- **Bitmask:**
  - Bit 0: Port 1
  - Bit 1: Port 2

Tuning & Behavior

- **Default:** 0 (Disabled).
- **Warning:** Both ports must be set to the same bitrate.
- **Danger:** Do not create a loop (connecting Port 1 to Port 2 externally) while this is enabled, or you will create a "Broadcast Storm" that floods the bus and crashes the network.

CAN_NODE (DroneCAN node ID used by this node on all networks)

DEFAULT: 0	RANGE: 0 127

DroneCAN node ID used by this node on all networks (CAN_NODE)

Description

This parameter sets the Node ID for the DroneCAN peripheral. The Node ID is the unique address of the device on the CAN bus.

The Mathematics

Address Arbitration.

The Engineer's View

Defined in `Tools/AP_Periph/Parameters.cpp`.

- **0:** Dynamic Node Allocation (DNA). The device listens for the flight controller (Node Allocator) to assign it a free ID.
- **1-125:** Static ID. The device claims this ID immediately.

Tuning & Behavior

- **Default Value:** 0 (Dynamic)
- **Recommendation:** Use 0 for most setups. Use static IDs if you need fixed addressing for specific scripting or redundancy requirements.

CAN_P1_BITRATE (Bitrate of CAN interface 1)

DEFAULT: 1000000	RANGE: 10000 1000000

Bitrate of CAN interface 1 (CAN_P1_BITRATE)

Description

Defines the communication speed for the first physical CAN bus. All devices on the bus must use the same bitrate to communicate.

Tuning & Behavior

- **Default Value:** 1000000 (1 Mbps)
- **Range:** 10,000 to 1,000,000
- **1 Mbps is standard for DroneCAN.**

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





CAN_P1_DRIVER (Index of virtual driver for CAN interface 1)

DEFAULT: 1	RANGE: 0 3

Index of virtual driver for CAN interface 1 (CAN_P1_DRIVER)

Description

Enables the first CAN port and links it to one of the virtual CAN driver instances (D1, D2, or D3).

Tuning & Behavior

- **Default Value:** 1
- **Values:** 0:Disabled, 1:First driver (D1), 2:Second driver (D2), 3:Third driver (D3)
- **Requires reboot to take effect.**

CAN_P1_FDBITRATE (CAN Port 1 FD Bitrate)

DEFAULT: 5	RANGE: 1 8	UNITS: Mbps

CAN Port 1 FD Bitrate (CAN_P1_FDBITRATE)

Description

`CAN_P1_FDBITRATE` enables "High Speed" mode for modern CAN FD devices.

Standard CAN is limited to 1Mbps. CAN FD (Flexible Data-rate) allows the data payload portion of the packet to be sent much faster.

- **5 (5 Mbps):** Standard for many modern DroneCAN FD peripherals.
- **2 (2 Mbps):** Conservative setting.

Tuning & Behavior

- **Requirement:** Your hardware (Flight Controller and Peripheral) must support CAN FD.

CAN_P1_OPTIONS (CAN Port 1 Options)

DEFAULT: 0	RANGE: 0 31

CAN Port 1 Options (CAN_P1_OPTIONS)

Description

`CAN_P1_OPTIONS` provides hardware-level tweaks for the CAN interface.

- **Bit 0 (1): Log All Frames.** Records every single CAN packet to the DataFlash log. (WARNING: Generates huge logs).
- **Bit 1 (2): Require Timestamps.** Rejects frames that don't have hardware timestamps.

Tuning & Behavior

- **Default Value:** 0.

CAN_P2_BITRATE (Bitrate of CAN interface)

DEFAULT: 1000000	RANGE: 10000 1000000	UNITS: null

Bitrate of CAN interface

Note: This parameter functions identically to CAN_P1_BITRATE.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





CAN_P2_DRIVER (Index of virtual driver to be used with physical CAN interface)

DEFAULT: HAL_CAN_DRIVER_DEFAULT	RANGE: null	UNITS: null

Index of virtual driver to be used with physical CAN interface

Note: This parameter functions identically to CAN_P1_DRIVER.

CAN_P2_FDBITRATE (Bitrate of CANFD interface)

DEFAULT: HAL_CANFD_SUPPORTED	RANGE: null	UNITS: null

Bitrate of CANFD interface

Note: This parameter functions identically to CAN_P1_FDBITRATE.

CAN_P2_OPTIONS (quadplane options)

DEFAULT: 0	RANGE: null	UNITS: null

quadplane options

Note: This parameter functions identically to CAN_P1_OPTIONS.

CAN_P3_BITRATE (Bitrate of CAN interface)

DEFAULT: 1000000	RANGE: 10000 1000000	UNITS: null

Bitrate of CAN interface

Note: This parameter functions identically to CAN_P1_BITRATE.

CAN_P3_DRIVER (Index of virtual driver to be used with physical CAN interface)

DEFAULT: HAL_CAN_DRIVER_DEFAULT	RANGE: null	UNITS: null

Index of virtual driver to be used with physical CAN interface

Note: This parameter functions identically to CAN_P1_DRIVER.

CAN_P3_FDBITRATE (Bitrate of CANFD interface)

DEFAULT: HAL_CANFD_SUPPORTED	RANGE: null	UNITS: null

Bitrate of CANFD interface

Note: This parameter functions identically to CAN_P1_FDBITRATE.

CAN_P3_OPTIONS (quadplane options)

DEFAULT: 0	RANGE: null	UNITS: null

quadplane options

Note: This parameter functions identically to CAN_P1_OPTIONS.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





CAN_SLCAN_CPORT (SLCAN CAN port)

DEFAULT: 0	RANGE: 0 3

SLCAN CAN port (CAN_SLCAN_CPORT)

Description

Defines which physical CAN bus is mapped to the SLCAN bridge for direct hardware access (e.g., using UAVCAN GUI Tool).

CAN_SLCAN_SDELAY (SLCAN Start Delay)

DEFAULT: 1	RANGE: 0 127	UNITS: s

SLCAN Start Delay (CAN_SLCAN_SDELAY)

Description

`CAN_SLCAN_SDELAY` introduces a pause after the SLCAN virtual interface is initialized but before it begins forwarding CAN frames.

This is primarily a developer or diagnostic feature used when bridging a computer to the DroneCAN bus via the flight controller's USB port (SLCAN mode).

Tuning & Behavior

- **Default:** 1 second.
- **Usage:** Increase this if your GCS or SLCAN tool struggles to connect immediately after the port opens.

CAN_SLCAN_SERNUM (SLCAN Serial Port)

DEFAULT: -1

SLCAN Serial Port (CAN_SLCAN_SERNUM)

Description

Specifies the USB or hardware serial port instance that serves as the bridge to the CAN bus selected in CAN_SLCAN_CPORT.

CAN_SLCAN_TIMOUT (SLCAN Timeout)

DEFAULT: 0	UNITS: s

SLCAN Timeout (CAN_SLCAN_TIMOUT)

Description

The inactivity period after which the SLCAN bridge is automatically closed.

CAN_TERMINATE (Force Terminate (Invalid/Legacy))

DEFAULT: 0

Force Terminate (CAN_TERMINATE)

Description

This parameter is not found as CAN_TERMINATE in the main codebase. It appears to be a miscategorized Advanced Failsafe parameter.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **CC** parameter group configures the **Custom Controller** library. This feature provides a sandbox for developers and researchers to implement custom attitude control algorithms (e.g., INDI, L1 adaptive, or novel PID variants) without replacing the core `AC_AttitudeControl` library.

The custom controller runs in parallel with the standard controller, and its output is mixed or swapped in based on the configuration.

Key Concepts

1. Axis Masking (`CC_AXIS_MASK`)

You don't have to replace the entire controller. You can take over just one axis.

- **Bit 0:** Roll.
- **Bit 1:** Pitch.
- **Bit 2:** Yaw.
- **Example:** Set to 4 to use the custom controller for Yaw only, while keeping standard PID for Roll/Pitch.

2. Backend Selection (`CC_TYPE`)

Selects which custom backend logic to execute.

- **0:** Disabled.
- **1:** Empty (Pass-through/Template).
- **2:** PID (Example implementation).
- **3:** INDI (Incremental Nonlinear Dynamic Inversion).

3. Generic Parameters (`CC_PARAMx`)

To avoid hardcoding gains, the library exposes generic floating-point parameters (`CC_PARAM1` ... `CC_PARAM3`) that are readable by the custom backend code.

Parameter Breakdown

- `CC_TYPE`: Active backend.
- `CC_AXIS_MASK`: Active axes.
- `CC_PARAM1`: User-defined tuning variable 1.

Integration Guide

- **For Users:** Unless you are running a specific firmware build that documents what `CC_PARAM1` does, leave this disabled (`CC_TYPE = 0`).
- **For Developers:** Use `libraries/AC_CustomControl` as a template. The `update()` method is called at the main loop rate (400Hz+).

Developer Notes

- **Library:** `libraries/AC_CustomControl`
- **Safety:** If your custom controller outputs `NaN` or crashes, the vehicle will likely crash. Test in SITL first.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





CC_AXIS_MASK (Custom Controller bitmask)

DEFAULT: 0

Custom Controller bitmask (CC_AXIS_MASK)

Description

Defines the specific axes that are handed over to the custom controller. When an axis bit is set, the standard ArduPilot attitude controller for that axis is bypassed.

Tuning & Behavior

- **Default Value:** 0
- **Bitmask:**
  - Bit 0: Roll
  - Bit 1: Pitch
  - Bit 2: Yaw

CC_PARAM1 (Custom Controller Param 1)

DEFAULT: 0.0f

Custom Controller Param 1 (CC_PARAM1)

Description

This is a generic placeholder parameter provided by the **Custom Control** library. It allows developers to pass tunable values to their custom C++ controller algorithms without needing to modify the main parameter table or recompile the entire parameter definitions.

The Mathematics

User-defined by the custom controller implementation.

The Engineer's View

Defined in `libraries/AC_CustomControl/AC_CustomControl_Empty.cpp`.

- **Usage:** If you are running standard firmware, this parameter does nothing. If you have loaded a custom build with a specific `AC_CustomControl` backend, refer to that backend's documentation for the meaning of this value (e.g., gain, limit, or coefficient).

Tuning & Behavior

- **Default Value:** 0.0
- **Note:** Only useful for developers or users of custom control branches.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





CC_PARAM2 (Custom Controller Param 2)

`DEFAULT: 0.0f`

Custom Controller Param 2 (CC_PARAM2)

Description

Generic parameter placeholder for **Custom Control** backends. See `CC_PARAM1` for details.

The Mathematics

User-defined.

The Engineer's View

Defined in `libraries/AC_CustomControl/AC_CustomControl_Empty.cpp`.

Tuning & Behavior

- **Default Value:** 0.0

CC_PARAM3 (Custom Controller Param 3)

`DEFAULT: 0.0f`

Custom Controller Param 3 (CC_PARAM3)

Description

Generic parameter placeholder for **Custom Control** backends. See `CC_PARAM1` for details.

The Mathematics

User-defined.

The Engineer's View

Defined in `libraries/AC_CustomControl/AC_CustomControl_Empty.cpp`.

Tuning & Behavior

- **Default Value:** 0.0

CC_TYPE (Custom control type)

`DEFAULT: 0`

Custom control type (CC_TYPE)

Description

Enables the custom attitude control module and selects the backend implementation (e.g., a simple template or a PID-based controller).

Tuning & Behavior

- **Default Value:** 0 (None)
- **Values:** 0:None, 1:Empty (Template), 2:PID
- **Requires reboot to initialize the selected controller.**

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **CCNT** parameter group provides additional generic placeholders for **Custom Control** backends.

For full documentation, please see the CC group.

PARAM1 (Custom Control Parameter 1)

DEFAULT: 0.0	RANGE: -1000 1000

Custom Control Parameter 1 (PARAM1)

Description

`PARAM1` is a generic "Hook" for developers using the `AC_CustomControl` library.

If you are writing your own flight control laws (e.g., for a university research project or a specialized vehicle), this parameter provides a convenient way to pass values from the GCS into your C++ code without having to modify the global parameter table.

Tuning & Behavior

- **Default Value:** 0.
- **Usage:** Only active if `CCNT_TYPE` is set to a custom backend.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **CELL** parameter group configures the **4G/LTE Cellular Telemetry** subsystem. This allows the autopilot to communicate over cellular networks using supported modems (e.g., Skywire, SIMCOM) connected via Serial or USB (PPP).

Note: The parameter `CELL_COUNT` listed in some databases often belongs to the **OSD** library (used for displaying average cell voltage) rather than the Cellular driver itself. This overlap can be confusing.

Key Concepts

1. Cellular Modems

ArduPilot can manage basic modem initialization (AT commands) to establish an IP connection. Once connected, telemetry is routed over UDP/TCP to a Ground Station IP.

2. OSD Helper (`CELL_COUNT`)

(If present) This parameter helps the OSD calculate the "Average Cell Voltage" by dividing the total pack voltage by the number of cells.

- **Auto (-1):** Guesses the cell count based on total voltage at boot (assuming 3.7V-4.2V/cell).
- **Manual:** Forces a specific count (e.g., 6S, 12S). Useful if you plug in a partially discharged pack that might be mistaken for a full pack of lower cell count.

Parameter Breakdown

- `CELL_ENABLE` : Master switch for cellular driver.
- `CELL_TYPE` : Driver selection.
- `CELL_PIN` : PIN code for the SIM card.
- `CELL_APN` : Access Point Name string.

(Note: `CELL_COUNT` is functionally part of the OSD group logic).

Integration Guide

1. **Hardware:** Connect modem to a UART.
2. **Serial:** Set `SERIALx_PROTOCOL = 1` (MAVLink) or specific modem protocol.
3. **Config:** Set `CELL_ENABLE` , `CELL_TYPE` , and `CELL_APN` .

Developer Notes

- **Library:** `libraries/AP_Cellular` (Driver) and `libraries/AP_OSD` (Display).

CELL_COUNT (OSD Battery Cell Count)

DEFAULT: -1	RANGE: -1 12

OSD Battery Cell Count (CELL_COUNT)

Description

Configures the battery cell count used by the On-Screen Display to calculate per-cell voltage.

- **-1 (default) enables auto-detection based on total voltage at boot.**

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **CHUTE** parameter group configures the emergency **Parachute Release** system. This system acts as a last-resort recovery method to save the vehicle and minimize ground impact energy in the event of a catastrophic failure (e.g., motor loss, mid-air collision, or loss of attitude control).

Key Concepts

1. Automatic Triggering

The parachute can be deployed automatically if the autopilot detects:

- **Crash:** Vehicle is tumbling or hitting limits.
- **Critical Sink Rate:** Falling faster than `CHUTE_CRT_SINK`.
- **Motor Loss:** Requires thrust loss detection to be active.

2. Motor Interlock

Crucial Safety Feature: When the parachute deploys, the autopilot **IMMEDIATELY kills all motors**. This prevents the props from cutting the parachute lines or entangling the chute.

3. Release Mechanism

- **Servo:** Moves a servo to a specific PWM position to release a latch/pin.
- **Relay:** Toggles a voltage pin (e.g., for a pyrotechnic charge or solenoid).

Parameter Breakdown

- `CHUTE_ENABLED`: Master switch.
- `CHUTE_TYPE`: 10=Servo, 11=Relay.
- `CHUTE_SERVO_ON` / `OFF`: PWM positions for deployed/stowed states.
- `CHUTE_ALT_MIN`: Minimum altitude to allow auto-deployment (prevents accidental firing on the ground).
- `CHUTE_DELAY_MS`: Delay between killing motors and firing the chute (allows props to spin down).

Integration Guide

Testing (Ground)

1. **Remove Props.**
2. Set `CHUTE_ALT_MIN = 0` (temporarily).
3. Arm the vehicle.
4. Switch to a mode or flip a switch assigned to `Parachute Release` (RC Option 22).
5. Verify the motors stop and the servo moves.
6. **Reset `CHUTE_ALT_MIN` before flight.**

Developer Notes

- **Library:** `libraries/AP_Parachute`
- **Integration:** Tightly coupled with `AP_Motors` to ensure immediate shutdown.

CHUTE_ALT_MIN (Parachute min altitude)

DEFAULT: 10	RANGE: 0 32000	UNITS: m

Parachute min altitude (CHUTE_ALT_MIN)

Description

Safety floor for parachute release. To prevent injury to people on the ground or damage to the aircraft from a low-altitude deployment, the parachute will only trigger if the vehicle is above this altitude.

- **0 disables the altitude check.**

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





CHUTE_CRT_SINK (Critical sink speed rate)

DEFAULT: 0	RANGE: 0 15	UNITS: m/s

Critical sink speed rate (CHUTE_CRT_SINK)

Description

Automatic trigger threshold based on descent rate. If the vehicle is falling faster than this value for more than one second, the parachute will automatically deploy.

- **0 disables automatic sink-rate triggering.**

CHUTE_DELAY_MS (Parachute release delay)

DEFAULT: 500	RANGE: 0 5000	UNITS: ms

Parachute release delay (CHUTE_DELAY_MS)

Description

Pause duration between the command to stop the motors (to prevent entanglement) and the actual release of the parachute.

CHUTE_ENABLED (Parachute release enabled or disabled)

DEFAULT: 0	RANGE: 0 1

Parachute release enabled or disabled (CHUTE_ENABLED)

Description

Master switch for the emergency parachute system.

Tuning & Behavior

- **Default Value:** 0 (Disabled)
- **Values:** 0:Disabled, 1:Enabled
- **When enabled, the autopilot will monitor for crash conditions and can trigger the parachute automatically or manually.**

CHUTE_OPTIONS (Parachute options)

DEFAULT: 0

Parachute options (CHUTE_OPTIONS)

Description

Bitmask for advanced parachute behaviors.

Tuning & Behavior

- **Default Value:** 0
- **Bitmask:**
  - Bit 0: Hold open forever after release
  - Bit 1: Skip disarm before parachute release

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY

CHUTE_SERVO_OFF (Servo OFF PWM value)

DEFAULT: 1100	RANGE: 1000 2000	UNITS: PWM

Servo OFF PWM value (CHUTE_SERVO_OFF)

Description

The idle (closed) PWM position for the parachute servo.

CHUTE_SERVO_ON (Parachute Servo ON PWM value)

DEFAULT: 1300	RANGE: 1000 2000	UNITS: PWM

Parachute Servo ON PWM value (CHUTE_SERVO_ON)

Description

The active (release) PWM position for the parachute servo.

CHUTE_TYPE (Parachute release mechanism type)

DEFAULT: 0

Parachute release mechanism type (CHUTE_TYPE)

Description

Defines the hardware interface used to trigger the parachute.

Tuning & Behavior

- **Default Value:** 0 (Relay)
- **Values:** 0:Relay, 10:Servo
- **For Servo type, you must also assign a SERVOx_FUNCTION to 'ParachuteRelease'.**

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **CIRCLE** parameter group configures the behavior of the **Circle** flight mode. In this mode, the vehicle flies a perfect circle around a center point (typically the location where the mode was engaged) while keeping the nose pointed towards the center (ROI).

Key Concepts

1. Radius and Direction

- `CIRCLE_RADIUS`: Defines the size of the circle (in centimeters or meters depending on firmware version, typically cm).
- `CIRCLE_RATE`: Defines the angular speed of the orbit (deg/s).
- `CIRCLE_DIR`: 0 = Clockwise, 1 = Counter-Clockwise. *(Note: Often controlled by the sign of `CIRCLE_RATE` in newer versions).*

2. Point of Interest

By default, the vehicle faces the center.

- `CIRCLE_OPTIONS`: Can be used to change facing behavior (e.g., face direction of flight).

Parameter Breakdown

- `CIRCLE_RADIUS`: Radius of the orbit.
- `CIRCLE_RATE`: Angular velocity (deg/s). High rate + small radius = fast flight.

Integration Guide

- **Engagement:** Fly to the desired center point, switch to `CIRCLE`. The vehicle will fly out to `CIRCLE_RADIUS` and begin orbiting.
- **Adjustment:** In Copter, you can often adjust the radius dynamically using the Pitch/Roll stick while in Circle mode.

Developer Notes

- **Library:** `ArduCopter/mode_circle.cpp`, `ArduPlane/mode_circle.cpp`.
- **Difference:** Plane often uses `LOITER_RADIUS` for standard loiters, but `CIRCLE` mode is specific.

CIRCLE_DIR (Circle Direction (Rover))

DEFAULT: 0	RANGE: 0 1

Circle Direction (CIRCLE_DIR)

Description

Selects clockwise or counter-clockwise movement for Rover vehicles in Circle mode.

Tuning & Behavior

- **Default Value:** 0 (Clockwise)
- **Values:** 0:Clockwise, 1:Counter-Clockwise

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





CIRCLE_OPTIONS (Circle options)

DEFAULT: 1

Circle options (CIRCLE_OPTIONS)

Description

Configuration bitmask for tailoring the vehicle's orientation and starting logic during Circle mode.

Tuning & Behavior

- **Default Value:** 1
- **Bitmask:**
  - Bit 0: Manual Control (Use pitch/roll to adjust radius/rate in flight)
  - Bit 1: Face Direction of Travel
  - Bit 2: Start at Center (Begin from the circle's center rather than flying to the perimeter)
  - Bit 3: ROI at Center (Keep gimbal pointed at the circle center)

CIRCLE_RADIUS (Circle Radius)

DEFAULT: 1000	RANGE: 0 200000	UNITS: cm

Circle Radius (CIRCLE_RADIUS)

Description

Specifies the horizontal distance from the center point that the vehicle will maintain while in the "Circle" flight mode.

Tuning & Behavior

- **Default Value:** 1000 cm (10 m)
- **Range:** 0 to 200,000 cm
- **A value of 0 results in the vehicle performing a "Panorama" (rotating in place).**

CIRCLE_RATE (Circle rate)

DEFAULT: 20	RANGE: -90 90	UNITS: deg/s

Circle rate (CIRCLE_RATE)

Description

Defines the angular speed and direction of the vehicle's movement around the circle.

Tuning & Behavior

- **Default Value:** 20 deg/s
- **Range:** -90 to 90 deg/s
- **Positive values result in clockwise rotation; negative values result in counter-clockwise rotation.**

CIRCLE_SPEED (Circle Speed (Rover))

DEFAULT: 0	RANGE: 0 10	UNITS: m/s

Circle Speed (CIRCLE_SPEED)

Description

Defines the linear velocity of the Rover while it is circling.

- **0 uses the global waypoint speed (WP_SPEED).**

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **CMD** parameter group (specifically `CMD_TOTAL`) is a read-only reporting mechanism for the onboard mission storage.

Key Concepts

1. Mission Storage

ArduPilot stores mission items (Waypoints, Do-Jumps, ROI, etc.) in EEPROM or FRAM. This parameter simply reports how many items are currently valid and loaded.

Parameter Breakdown

- `CMD_TOTAL`: The total count of mission items stored on the vehicle.
  - **0:** No mission loaded (or only Home).
  - **>0:** Valid mission present.

Integration Guide

- **Usage:** Ground Control Stations check this value to verify that a "Write Waypoints" operation was successful.
- **Do Not Edit:** Changing this manually will corrupt the mission index.

Developer Notes

- **Library:** `libraries/AP_Mission`

CMD_TOTAL (Number of loaded mission items)

DEFAULT: 0	RANGE: 0 65535

Number of loaded mission items (CMD_TOTAL)

Description

This is a **Read-Only** system parameter that reports the total number of waypoint or mission commands currently uploaded to the vehicle. It is used by the Ground Control Station (GCS) to verify mission upload success and by the autopilot to know the bounds of the mission index.

The Mathematics

N/A. Simple integer counter.

The Engineer's View

Defined in `AntennaTracker/Parameters.cpp`. It reflects the value of `mission.num_commands()`.

- **0:** No mission loaded (or only Home).
- **>0:** Valid mission present.

Tuning & Behavior

- **Read Only:** Do not attempt to change this parameter manually. It is updated automatically when you upload a mission.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **COMMON** parameter group is a container for parameters that don't fit neatly into a specific library subclass or are shared generically across vehicle types (though often defined in vehicle-specific code).

In this specific documentation set, it largely contains the **Flight Mode** switch mapping parameters for Rover (and historically other vehicles before they moved to `FLTMODE_`).

Key Concepts

1. Mode Switch Mapping (`MODEx`)

ArduPilot allows you to map up to 6 flight modes to a single RC channel (usually Channel 5 or 8) using PWM ranges.

- `MODE1`: PWM 0-1230.
- `MODE2`: PWM 1231-1360.
- `MODE3`: PWM 1361-1490.
- `MODE4`: PWM 1491-1620.
- `MODE5`: PWM 1621-1749.
- `MODE6`: PWM 1750+.

Parameter Breakdown

- `MODE1`..`MODE6`: Selects the active mode for that switch position (e.g., MANUAL, STEERING, AUTO).
- `DUMMY`: A test parameter with no effect.

Integration Guide

- **Setup:** Use the "Flight Modes" tab in Mission Planner. It automatically sets these parameters based on the dropdown menu selections.

Developer Notes

- **Note:** Copter uses `FLTMODE_x`. Plane uses `FLTMODE_x`. Rover uses `MODE_x`. This inconsistency is historical.

DUMMY (Dummy Parameter)

`DEFAULT: 0`

Dummy Parameter (DUMMY)

Description

This is a test parameter used by the ArduPilot Replay tool and other internal test suites. It has no effect on vehicle flight behavior.

The Mathematics

None.

The Engineer's View

Defined in `Tools/Replay/Replay.cpp`.

Tuning & Behavior

- **Ignore:** Do not modify.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





MODE3 (Mode3)

DEFAULT: 0	RANGE: 0 255

Mode3 (MODE3)

Description

This parameter assigns the flight/drive mode active when the RC Mode Switch (channel 5 or 8) is in its middle position (PWM ~1425us).

The Mathematics

PWM Mapping Logic.

The Engineer's View

Defined in `Rover/Parameters.cpp` (and other vehicle folders). Maps integer ID to mode.

Tuning & Behavior

- **Default Value:** Vehicle dependent.
- **Select:** Choose from the list of available flight modes (e.g., MANUAL, ACRO, STEERING, HOLD, AUTO).

MODE4 (Mode4)

DEFAULT: 0	RANGE: 0 255

Mode4 (MODE4)

Description

This parameter assigns the flight/drive mode active when the RC Mode Switch is in position 4 (PWM ~1555us). See `MODE3`.

The Mathematics

PWM Mapping.

The Engineer's View

Defined in `Rover/Parameters.cpp`.

Tuning & Behavior

- **Default Value:** Vehicle dependent.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





MODE5 (Mode5)

DEFAULT: 0	RANGE: 0 255

Mode5 (MODE5)

Description

This parameter assigns the flight/drive mode active when the RC Mode Switch is in position 5 (PWM ~1685us). See `MODE3`.

The Mathematics

PWM Mapping.

The Engineer's View

Defined in `Rover/Parameters.cpp`.

Tuning & Behavior

- **Default Value:** Vehicle dependent.

MODE6 (Mode6)

DEFAULT: 0	RANGE: 0 255

Mode6 (MODE6)

Description

This parameter assigns the flight/drive mode active when the RC Mode Switch is in position 6 (High, PWM > 1750us). See `MODE3`.

The Mathematics

PWM Mapping.

The Engineer's View

Defined in `Rover/Parameters.cpp`.

Tuning & Behavior

- **Default Value:** Vehicle dependent.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





COMPASS CONFIGURATION (COMPASS)

Overview

The **COMPASS** parameter group configures the **Magnetometers** (Electronic Compasses). Reliable heading information is critical for GPS flight modes (Loiter, Auto, RTL) to determine the vehicle's yaw relative to the world.

ArduPilot supports up to 3 compasses. `COMPASS` (1), `COMPASS2`, and `COMPASS3`.

Key Concepts

1. Orientation (`COMPASS_ORIENT`)

Most Critical Setting. If using an external GPS/Compass module, the sensor is often mounted upside down or rotated relative to the flight controller.

- **External:** Set `COMPASS_EXTERNAL = 1`.
- **Rotation:** Match the mounting (e.g., `ROTATION_ROLL_180`).

2. Calibration (Offsets & Scaling)

- **Offsets (`COMPASS_OFS_X/Y/Z`):** Compensates for "Hard Iron" distortion (permanent magnets, screws, currents) on the frame.
- **Scaling (`COMPASS_SCALE`):** Compensates for "Soft Iron" distortion (ferrous metals stretching the field) and sensor axis sensitivity differences.

3. Declination (`COMPASS_DEC`)

The difference between Magnetic North and True North.

- **Auto (`COMPASS_AUTODEC`):** Recommended. Calculates declination based on GPS location table.

4. Motor Compensation (`COMPASS_MOT_...`)

High current in power wires creates a magnetic field that shifts the compass as throttle increases.

- **CompassMot:** A calibration routine that measures this interference vs. throttle/current and subtracts it in flight.

Parameter Breakdown

- `COMPASS_ENABLE`: Master switch.
- `COMPASS_LEARN`: In-flight learning of offsets (Compass Learn).
- `COMPASS_USE`: Trusted for navigation? (Uncheck for internal compasses with high interference).
- `COMPASS_DEV_ID`: Hardware ID.

Integration Guide

Typical Setup

1. **Hardware:** Mount GPS/Compass mast away from power wires/ESCs.
2. **Calibrate:** Run "Onboard Mag Calibration" ("The Dance"). Rotate vehicle on all axes until progress bars fill.
3. **Verify:** Check if North is North.
4. **Prioritize:** Disable internal compasses (`COMPASS_USE2 = 0`, etc.) if they show high interference ("Compass Variance" errors).

Developer Notes

- **Library:** `libraries/AP_Compass`
- **MagFit:** Advanced 3D ellipsoid fitting is used for calibration.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





COMPASS_AUTODEC (Auto Declination)

DEFAULT: 1	RANGE: 0 1

Auto Declination (COMPASS_AUTODEC)

Description

Allows ArduPilot to automatically look up the correct magnetic declination using an internal world magnetic model and your current GPS coordinates.

COMPASS_AUTO_ROT (Automatically check orientation)

DEFAULT: 2	RANGE: 0 3

Automatically check orientation (COMPASS_AUTO_ROT)

Description

A calibration helper that determines if the user's COMPASS_ORIENT setting matches the physical data collected during calibration.

Tuning & Behavior

- **Default Value:** 2 (CheckAndFix)
- **Values:** 0:Disabled, 1:CheckOnly, 2:CheckAndFix

COMPASS_CAL_FIT (Compass calibration fitness)

DEFAULT: 16	RANGE: 4 32

Compass calibration fitness (COMPASS_CAL_FIT)

Description

Determines the "pass/fail" strictness of the compass calibration algorithm.

Tuning & Behavior

- **Default Value:** 16
- **Lower values (e.g., 4) are stricter and harder to pass, but result in better heading accuracy.**
- **Higher values (e.g., 32) are more relaxed and useful if you are in a location with high local magnetic interference.**

COMPASS_CUS_PIT (Custom orientation pitch offset)

DEFAULT: 0	RANGE: -180 180	UNITS: deg

Custom orientation pitch offset (COMPASS_CUS_PIT)

Description

Precise pitch-axis compensation for custom compass mounting.

COMPASS_CUS_ROLL (Custom orientation roll offset)

DEFAULT: 0	RANGE: -180 180	UNITS: deg

Custom orientation roll offset (COMPASS_CUS_ROLL)

Description

Allows for precise, degree-by-degree compensation of the compass mounting angle in the roll axis.

- **Only active if COMPASS_ORIENT is set to 100 (Custom).**

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





COMPASS_CUS_YAW (Custom orientation yaw offset)

DEFAULT: 0	RANGE: -180 180	UNITS: deg

Custom orientation yaw offset (COMPASS_CUS_YAW)

Description

Precise yaw-axis (heading) compensation for custom compass mounting.

COMPASS_DEC (Compass declination)

DEFAULT: 0	RANGE: -3.142 3.142	UNITS: rad

Compass declination (COMPASS_DEC)

Description

Manual offset to correct for the difference between Magnetic North and True North at your current location.

Tuning & Behavior

- **Default Value:** 0 rad
- **Range:** -3.142 to 3.142 rad
- **Usually set automatically if COMPASS_AUTODEC is enabled.**

COMPASS_DEV_ID (Compass device id)

DEFAULT: 0

Compass device id (COMPASS_DEV_ID)

Description

Uniquely identifies the physical magnetometer hardware (e.g., its bus type and address).

- **ReadOnly: Automatically detected by the flight controller.**

COMPASS_DEV_ID2 (Compass 2 device id)

DEFAULT: 0

Compass 2 device id (COMPASS_DEV_ID2)

Description

Automatically detected hardware ID for the secondary magnetometer.

COMPASS_DEV_ID3 (Compass 3 device id)

DEFAULT: 0

Compass 3 device id (COMPASS_DEV_ID3)

Description

Hardware ID for the tertiary magnetometer.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





COMPASS_DEV_ID4 (Compass 4 device id)

DEFAULT: 0

Compass 4 device id (COMPASS_DEV_ID4)

Description

Automatically detected hardware ID for the 4th magnetometer.

COMPASS_DEV_ID5 (Compass 5 device id)

DEFAULT: 0

Compass 5 device id (COMPASS_DEV_ID5)

Description

Automatically detected hardware ID for the 5th magnetometer.

COMPASS_DEV_ID6 (Compass 6 device id)

DEFAULT: 0

Compass 6 device id (COMPASS_DEV_ID6)

Description

Automatically detected hardware ID for the 6th magnetometer.

COMPASS_DEV_ID7 (Compass 7 device id)

DEFAULT: 0

Compass 7 device id (COMPASS_DEV_ID7)

Description

Automatically detected hardware ID for the 7th magnetometer.

COMPASS_DEV_ID8 (Compass 8 device id)

DEFAULT: 0

Compass 8 device id (COMPASS_DEV_ID8)

Description

Automatically detected hardware ID for the 8th magnetometer.

COMPASS_DIA (Compass soft-iron diagonal)

DEFAULT: 1.0

Compass soft-iron diagonal (COMPASS_DIA)

Description

Correction factors for "soft-iron" distortion, which occurs when nearby metal elements distort the magnetic field rather than creating their own.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





COMPASS_DIA2 (Compass 2 soft-iron diagonal)

DEFAULT: 1.0

Compass 2 soft-iron diagonal (COMPASS_DIA2)

Description

Soft-iron correction for the secondary magnetometer.

COMPASS_DIA3 (Compass 3 soft-iron diagonal)

DEFAULT: 1.0

Compass 3 soft-iron diagonal (COMPASS_DIA3)

Description

Soft-iron correction for the tertiary magnetometer.

COMPASS_DISBLMSK (Compass disable driver type mask)

DEFAULT: 0

Compass disable driver type mask (COMPASS_DISBLMSK)

Description

Prevents ArduPilot from probing for specific magnetometer hardware types. Useful for speeding up boot times or avoiding hardware conflicts.

Tuning & Behavior

- **Default Value:** 0 (Probe everything)
- **Bitmask Examples:**
  - Bit 0: HMC5883
  - Bit 11: DroneCAN
  - Bit 16: RM3100

COMPASS_ENABLE (Enable Compass)

DEFAULT: 1	RANGE: 0 1

Enable Compass (COMPASS_ENABLE)

Description

Master power switch for the magnetometer library.

Tuning & Behavior

- **Default Value:** 1 (Enabled)
- **Values:** 0:Disabled, 1:Enabled
- **Must be enabled to collect magnetometer data or perform a compass calibration.**

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





COMPASS_EXTERN2 (Compass 2 is attached via an external cable)

DEFAULT: 0	RANGE: 0 2

Compass 2 is attached via an external cable (COMPASS_EXTERN2)

Description

Tells ArduPilot if the secondary compass is an external unit.

COMPASS_EXTERN3 (Compass 3 is attached via an external cable)

DEFAULT: 0	RANGE: 0 2

Compass 3 is attached via an external cable (COMPASS_EXTERN3)

Description

Tells ArduPilot if the tertiary compass is an external unit.

COMPASS_EXTERNAL (Compass is attached via an external cable)

DEFAULT: 0	RANGE: 0 2

Compass is attached via an external cable (COMPASS_EXTERNAL)

Description

Tells ArduPilot whether the compass is part of the flight controller (Internal) or located on a separate module (External, like a GPS puck).

Tuning & Behavior

- **Default Value:** 0 (Internal)
- **Values:** 0:Internal, 1:External, 2:ForcedExternal
- **When set to 1 or 2, COMPASS_ORIENT is used instead of the board's global AHRS_ORIENTATION.**

COMPASS_FLTR_RNG (Range in which sample is accepted)

DEFAULT: 0	RANGE: 0 100	UNITS: %

Range in which sample is accepted (COMPASS_FLTR_RNG)

Description

A noise-rejection filter for magnetometer data. Rejects samples that differ significantly from the recent average.

- **0 disables the filter.**

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





COMPASS_LEARN (Learn compass offsets automatically)

DEFAULT: 0	RANGE: 0 3

Learn compass offsets automatically (COMPASS_LEARN)

Description

Enables the "Compass Learning" feature where the flight controller refines the sensor offsets in-flight by analyzing the magnetometer data as the vehicle rotates.

Tuning & Behavior

- **Default Value:** 0 (Disabled)
- **Values:** 0:Disabled, 1:Internal-Learning, 2:EKF-Learning, 3:InFlight-Learning
- **Useful for refining a coarse initial calibration, but should generally be disabled once offsets have stabilized.**

COMPASS_MOT (Motor interference compensation (X, Y, Z))

DEFAULT: 0	RANGE: -1000 1000	UNITS: mGauss/A

Motor interference compensation (COMPASS_MOT)

Description

Scaling factors for dynamic magnetic interference. These values determine how much the compass heading should be adjusted based on the current power level.

COMPASS_MOT2 (Compass 2 Motor interference compensation)

DEFAULT: 0	RANGE: -1000 1000	UNITS: mGauss/A

Compass 2 Motor interference compensation (COMPASS_MOT2)

Description

Dynamic magnetic interference compensation for the secondary magnetometer.

COMPASS_MOT3 (Compass 3 Motor interference compensation)

DEFAULT: 0	RANGE: -1000 1000	UNITS: mGauss/A

Compass 3 Motor interference compensation (COMPASS_MOT3)

Description

Dynamic magnetic interference compensation for the tertiary magnetometer.

COMPASS_MOTCT (Motor interference compensation type)

DEFAULT: 0	RANGE: 0 2

Motor interference compensation type (COMPASS_MOTCT)

Description

Enables the "CompassMot" feature, which compensates for magnetic interference generated by high-current battery wires and motors.

Tuning & Behavior

- **Default Value:** 0 (Disabled)
- **Values:** 0:Disabled, 1:Use Throttle (Simpler), 2:Use Current (Requires accurate battery monitor)

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





COMPASS_ODI (Compass soft-iron off-diagonal)

DEFAULT: 0.0

Compass soft-iron off-diagonal (COMPASS_ODI)

Description

Advanced soft-iron calibration values that account for cross-axis magnetic field coupling.

COMPASS_ODI2 (Compass 2 soft-iron off-diagonal)

DEFAULT: 0.0

Compass 2 soft-iron off-diagonal (COMPASS_ODI2)

Description

Advanced soft-iron correction for the secondary magnetometer.

COMPASS_ODI3 (Compass 3 soft-iron off-diagonal)

DEFAULT: 0.0

Compass 3 soft-iron off-diagonal (COMPASS_ODI3)

Description

Advanced soft-iron correction for the tertiary magnetometer.

COMPASS_OFFS_MAX (Compass maximum offset)

DEFAULT: 1800	RANGE: 500 3000

Compass maximum offset (COMPASS_OFFS_MAX)

Description

A safety threshold for magnetometer calibration. If the resulting offsets are larger than this value, the calibration will be rejected as unhealthy.

Tuning & Behavior

- **Default Value:** 1800 (Default)
- **Range:** 500 to 3000
- **High offsets indicate the compass is too close to significant metal masses or high-current wires.**

COMPASS_OFS (Compass offsets (X, Y, Z))

DEFAULT: 0	RANGE: -400 400	UNITS: mGauss

Compass offsets (COMPASS_OFS)

Description

The fundamental hard-iron calibration values for the primary magnetometer. These represent static magnetic fields within the aircraft itself.

Tuning & Behavior

- **Default Value:** 0 0 0
- **Usually set automatically during the "Compass Calibration" process.**
- **Note: In the codebase, this is stored as separate X, Y, Z parameters but often presented as a vector.**

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





COMPASS_OFS2 (Compass 2 offsets)

DEFAULT: 0	RANGE: -400 400	UNITS: mGauss

Compass 2 offsets (COMPASS_OFS2)

Description

Hard-iron calibration values for the secondary magnetometer.

COMPASS_OFS3 (Compass 3 offsets)

DEFAULT: 0	RANGE: -400 400	UNITS: mGauss

Compass 3 offsets (COMPASS_OFS3)

Description

Hard-iron calibration values for the tertiary magnetometer.

COMPASS_OPTIONS (Compass options)

DEFAULT: 0

Compass options (COMPASS_OPTIONS)

Description

Advanced configuration for the compass library.

Tuning & Behavior

- **Default Value:** 0
- **Bitmask:**
  - Bit 0: CalRequireGPS (Force GPS lock for calibration)
  - Bit 1: Allow DroneCAN replacement

COMPASS_ORIENT (Compass orientation)

DEFAULT: 0

Compass orientation (COMPASS_ORIENT)

Description

Defines the software rotation for the primary compass. This is critical if the compass (often part of the GPS module) is mounted in any direction other than perfectly forward and upright.

Tuning & Behavior

- **Default Value:** 0 (None)
- **Common Values:** 0:None, 8:Roll180 (very common for GPS/Compass modules), 100:Custom

COMPASS_ORIENT2 (Compass 2 orientation)

DEFAULT: 0

Compass 2 orientation (COMPASS_ORIENT2)

Description

Software rotation for the secondary compass.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





COMPASS_ORIENT3 (Compass 3 orientation)

DEFAULT: 0

Compass 3 orientation (COMPASS_ORIENT3)

Description

Software rotation for the tertiary compass.

COMPASS_PMOT1 (Compass Per-Motor Compensation (Motor 1))

DEFAULT: 0	RANGE: -10.0 10.0

Compass Per-Motor Compensation (COMPASS_PMOT1)

Description

`COMPASS_PMOT1` is an advanced calibration setting used to cancel out magnetic interference caused by high-current wires and motors.

In some small or tightly packed drones, the battery wires or motors are so close to the compass that they "pull" the needle whenever the throttle is applied. This causes the drone's heading to drift during a punch-out. This parameter stores the learned compensation factor for the first motor in the system.

Tuning & Behavior

- **Default:** 0.
- **Calibration:** This value is calculated automatically by running the **Compass Motor Calibration** (MagCali) routine in your Ground Control Station while safely tethering the drone and varying the throttle.
- **Recommendation:** Do not edit this manually. Use the GCS calibration tool to find the correct values for your specific frame.

COMPASS_PMOT2 (Compass per-motor2 Z)

DEFAULT: 0	RANGE: null

Compass per-motor2 Z

Note: This parameter functions identically to COMPASS_PMOT1.

COMPASS_PMOT3 (Compass per-motor3 Z)

DEFAULT: 0	RANGE: null

Compass per-motor3 Z

Note: This parameter functions identically to COMPASS_PMOT1.

COMPASS_PMOT4 (Compass per-motor4 Z)

DEFAULT: 0	RANGE: null

Compass per-motor4 Z

Note: This parameter functions identically to COMPASS_PMOT1.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





COMPASS_PMOT_1 (Per-motor interference - Motor 1)

DEFAULT: 0

Per-motor interference - Motor 1 (COMPASS_PMOT_1)

Description

Individual magnetic interference coefficient for the first motor.

- **Note: Subsequent parameters (PMOT_2, PMOT_3, etc.) exist for each motor.**

COMPASS_PMOT_2 (Compass per-motor2 Z)

DEFAULT: 0	RANGE: null

Compass per-motor2 Z

Note: This parameter functions identically to COMPASS_PMOT_1.

COMPASS_PMOT_3 (Compass per-motor3 Z)

DEFAULT: 0	RANGE: null

Compass per-motor3 Z

Note: This parameter functions identically to COMPASS_PMOT_1.

COMPASS_PMOT_4 (Compass per-motor4 Z)

DEFAULT: 0	RANGE: null

Compass per-motor4 Z

Note: This parameter functions identically to COMPASS_PMOT_1.

COMPASS_PMOT_EN (Per-motor interference enable)

DEFAULT: 0	RANGE: 0 1

Per-motor interference enable (COMPASS_PMOT_EN)

Description

Enables an advanced version of CompassMot that tracks and compensates for the magnetic interference of each individual motor, rather than the aircraft's total power draw.

COMPASS_PMOT_EXP (Per-motor interference exponent)

DEFAULT: 1.0

Per-motor interference exponent (COMPASS_PMOT_EXP)

Description

Mathematical exponent used to model how magnetic interference scales with individual motor power in advanced compensation modes.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





COMPASS_PRIO1_ID (Compass 1 Priority ID)

DEFAULT: 0

Compass 1 Priority ID (COMPASS_PRIO1_ID)

Description

Forcefully designates a specific magnetometer (by its Device ID) as the primary sensor for heading estimation.

COMPASS_PRIO2_ID (Compass 2 Priority ID)

DEFAULT: 0

Compass 2 Priority ID (COMPASS_PRIO2_ID)

Description

Designates a specific magnetometer as the secondary priority sensor.

COMPASS_PRIO3_ID (Compass 3 Priority ID)

DEFAULT: 0

Compass 3 Priority ID (COMPASS_PRIO3_ID)

Description

Designates a specific magnetometer as the tertiary priority sensor.

COMPASS_SCALE (Compass scale factor)

DEFAULT: 0	RANGE: 0 1.3

Compass scale factor (COMPASS_SCALE)

Description

A sensitivity adjustment for the magnetometer. If set to 0, no scaling is applied.

COMPASS_SCALE2 (Compass 2 scale factor)

DEFAULT: 0	RANGE: 0 1.3

Compass 2 scale factor (COMPASS_SCALE2)

Description

Sensitivity adjustment for the secondary magnetometer.

COMPASS_SCALE3 (Compass 3 scale factor)

DEFAULT: 0	RANGE: 0 1.3

Compass 3 scale factor (COMPASS_SCALE3)

Description

Sensitivity adjustment for the tertiary magnetometer.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





COMPASS_USE (Use compass for yaw)

DEFAULT: 1	RANGE: 0 1

Use compass for yaw (COMPASS_USE)

Description

Determines if the primary compass data is actually used by the EKF (Extended Kalman Filter) to calculate the vehicle's heading (yaw).

Tuning & Behavior

- **Default Value:** 1 (Enabled)
- **If disabled, the vehicle must rely on GPS-based yaw (GHY) or other heading sources.**

COMPASS_USE2 (Compass 2 used for yaw)

DEFAULT: 1	RANGE: 0 1

Compass 2 used for yaw (COMPASS_USE2)

Description

Determines if the secondary compass data is used by the EKF.

COMPASS_USE3 (Compass 3 used for yaw)

DEFAULT: 1	RANGE: 0 1

Compass 3 used for yaw (COMPASS_USE3)

Description

Determines if the tertiary compass data is used by the EKF.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **CRASH** parameter group configures the automatic **Crash Detection** logic. This safety feature is designed to rapidly identify when the vehicle has impacted the ground or an obstacle and is no longer under control.

Upon detection, the system immediately **Disarms the Motors** to:

1. Prevent fire (stalled motors overheat ESCs).
2. Prevent injury to bystanders.
3. Minimize damage to the airframe and props.

Key Concepts

1. Detection Logic

Different vehicles use different logic:

- **Plane:** Looks for sudden X-axis deceleration (impact) while flying (`CRASH_ACC_THRESH`).
- **Copter:** Looks for a sustained error between Target Angle and Actual Angle (loss of control) combined with low throttle usage.

2. Check Logic (`CRASH_DETECT`)

(Note: Often `CRASH_ACC_THRESH` or `FS_CRASH_CHECK`).

- If enabled, the code runs in the fast loop to monitor IMU spikes.

Parameter Breakdown

- `CRASH_ACC_THRESH`: (Plane) Impact force threshold (m/s/s). 0 to disable.
- `CRASH_DETECT`: (Some firmwares) Bitmask options.

Integration Guide

- **False Positives:** If you hand-launch or use a bungee, ensure `CRASH_ACC_THRESH` is high enough not to trigger on release.
- **Rough Landings:** If the plane disarms the moment it touches down (but bounces), you might need to relax the threshold or improve the landing flare logic.

Developer Notes

- **Library:** Vehicle specific (`ArduPlane/crash_check.cpp`).
- **Safety:** This is a critical safety feature for large/heavy drones.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





CRASH_ACC_THRESH (Crash Deceleration Threshold)

DEFAULT: 0	RANGE: 0 127	UNITS: m/s^2

Crash Deceleration Threshold (CRASH_ACC_THRESH)

Description

This parameter sets the sensitivity of the crash detection logic in ArduPlane. If the aircraft experiences a sudden deceleration (impact) along its longitudinal axis (X-axis) exceeding this value, the autopilot assumes a crash has occurred and instantly disarms the motor to prevent fire, injury, or further damage.

The Mathematics

$$\text{Crash Triggered} \iff a_x < -\text{CRASH_ACC_THRESH}$$

(Where negative a_x represents sudden braking/impact).

The Engineer's View

Defined in `ArduPlane/Parameters.cpp`. Checked in the main loop or inertial nav task.

- **0:** Crash detection disabled.
- **Note:** False positives can occur during rough landings or bungee launches if set too low.

Tuning & Behavior

- **Default Value:** 0 (Disabled)
- **Typical:** 25 m/s^2 (~2.5G) for foamies.
- **Adjustment:** Increase if you get disarmed during hard landings. Decrease if it fails to disarm on impact.

CRASH_ANGLE (Crash Angle)

DEFAULT: 0	RANGE: 0 90	UNITS: deg

Crash Angle (CRASH_ANGLE)

Description

Defines the tip-over threshold for Rover vehicles. If the vehicle's roll or pitch angle exceeds this value, it will be considered crashed and the motors will be disabled.

- **A value of 0 disables the tip-over check.**

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





CRUISE MODE CONFIGURATION (CRUISE)

Overview

The **CRUISE** parameter group (often appearing as `TRIM_ARSPD_CM`, `CRUISE_SPEED`, or `CRUISE_THROTTLE`) defines the baseline "Standard" flight/drive conditions for the vehicle.

These parameters tell the autopilot "This is what normal looks like." They are foundational for the navigation controllers (TECS / Speed Controller).

Key Concepts

1. Target Speed (`CRUISE_SPEED`)

The default target speed in meters/second for Auto, Cruise, and RTL modes.

- **Plane:** Indicated Airspeed.
- **Rover:** Ground Speed.

2. Base Throttle (`CRUISE_THROTTLE`)

The approximate throttle percentage required to maintain `CRUISE_SPEED` in level flight/driving.

- **Importance:** This provides a "FeedForward" baseline for the throttle controller. The PID loops only work to correct errors *around* this base value. If set correctly, the controller has to do very little work.

Parameter Breakdown

- `CRUISE_SPEED`: Target speed (m/s).
- `CRUISE_THROTTLE`: Target throttle (%).
- `CRUISE_ALT_FLOOR`: (Plane) Minimum altitude floor.

Integration Guide

Tuning

1. **Fly/Drive Manual:** Establish a steady speed that feels efficient.
2. **Observe:** Note the speed and throttle percentage from the OSD or logs.
3. **Set:** Update `CRUISE_SPEED` and `CRUISE_THROTTLE` with these observed values.

Developer Notes

- **Aliases:** In older Plane parameters, `TRIM_ARSPD_CM` is the canonical name for `CRUISE_SPEED`.
- **Dependencies:** `TECS` uses `CRUISE_SPEED` as the center of the speed envelope.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





CRUISE_ALT_FLOOR (Minimum altitude for FBWB and CRUISE mode)

DEFAULT: 0	RANGE: 0 100	UNITS: m

Minimum altitude for FBWB and CRUISE mode (CRUISE_ALT_FLOOR)

Description

This parameter creates a virtual "floor" or minimum altitude for the FBWB and CRUISE flight modes. If the pilot tries to descend below this altitude (measured above Home), the autopilot will ignore the pitch-down command and level off. This is a safety feature to prevent low-level crashes during long-range cruising.

The Mathematics

$$\text{Pitch Target} = \text{Constrain}(\text{Pilot Input}) \text{ s.t. } \text{Alt} \geq \text{CRUISE_ALT_FLOOR}$$

The Engineer's View

Defined in `ArduPlane/Parameters.cpp`.

- **0:** Disabled.
- **>0:** Minimum altitude in meters.

Tuning & Behavior

- **Default Value:** 0 (Disabled)
- **Safety:** Setting this to e.g., 50m ensures you never accidentally fly into terrain while cruising.

CRUISE_SPEED (Target Cruise Speed)

DEFAULT: 2.0	RANGE: 0 100.0	UNITS: m/s

Target Cruise Speed (CRUISE_SPEED)

Description

`CRUISE_SPEED` is the "Default Speed Limit" for your Rover or Boat.

When you start an **Auto** mission or trigger a **Return to Launch (RTL)**, the autopilot will attempt to maintain this speed while navigating between waypoints. It acts as the primary velocity target for the cruise control system.

Tuning & Behavior

- **Default:** 2.0 m/s (approx 7.2 km/h).
- **Recommendation:** Set this to a speed where your vehicle can still turn reliably and safely.
- **Physics:** This works in combination with CRUISE_THROTTLE to establish the initial power level for the propulsion system.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY

CRUISE_THROTTLE (Target Cruise Throttle)

DEFAULT: 50	RANGE: 0 100	UNITS: %

Target Cruise Throttle (CRUISE_THROTTLE)

Description

`CRUISE_THROTTLE` provides the "Feed-Forward" value for the speed controller.

It tells the autopilot: "To go at CRUISE_SPEED, you typically need about X% throttle." This gives the controller a starting point. If the vehicle is moving too slowly, the autopilot will increase throttle above this value; if moving too fast, it will decrease it.

Tuning & Behavior

- **Default:** 50%.
- **Calibration:** Drive your vehicle in Manual or Acro mode at your desired cruise speed. Note the throttle percentage shown on your GCS. Enter that value here.
- **Effect:** A correctly set cruise throttle leads to much smoother acceleration and more stable speed tracking during autonomous missions.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **CUST** parameter group allows for defining arbitrary 3D rotations (Euler Angles) for sensor mounting.

While standard ArduPilot parameters (like `AHRS_ORIENTATION` or `COMPASS_ORIENT`) provide a list of preset rotations (e.g., Yaw90, Pitch180), these presets are typically limited to 45-degree or 90-degree increments. The **CUST** group enables precise alignment for sensors mounted at odd angles (e.g., a GPS mast tilted 15 degrees back).

Key Concepts

1. Enable Custom Rotation (`CUST_ROT_ENABLE`)

This master switch enables the custom rotation engine.

2. Defining the Rotation (`CUST_ROTx_...`)

You can define up to 2 unique custom rotations. Each is defined by a Roll, Pitch, and Yaw angle.

- **Rotation 1:** Used when `AHRS_ORIENTATION` (or other orientation param) is set to **100 (Custom 1)**.
- **Rotation 2:** Used when the parameter is set to **101 (Custom 2)**.

Parameter Breakdown

- `CUST_ROT1_ROLL` / `PITCH` / `YAW`: Euler angles for Custom Rotation 1.
- `CUST_ROT2_ROLL` / `PITCH` / `YAW`: Euler angles for Custom Rotation 2.

Integration Guide

Example: Tilted GPS

If you mount your GPS/Compass module on a mast that is tilted 30 degrees backward:

1. Set `CUST_ROT_ENABLE = 1`.
2. Set `CUST_ROT1_PITCH = 30`.
3. Set `CUST_ROT1_ROLL` and `CUST_ROT1_YAW` to 0 (assuming it faces forward).
4. Set `COMPASS_ORIENT = 100` (Custom 1).
5. Reboot and verify the heading works correctly when you tilt the vehicle.

Developer Notes

- **Library:** `libraries/AP_CustomRotations`
- **Math:** Uses standard Euler sequence (Yaw → Pitch → Roll) to construct a rotation matrix.

CUST_ROT1_PITCH (Custom Rotation 1 - Pitch)

DEFAULT: 0	RANGE: -180 180	UNITS: deg

Custom Rotation 1 - Pitch (CUST_ROT1_PITCH)

Description

Defines the pitch angle for the first custom software rotation.

- **Ordering: Euler 321 (Yaw, then Pitch, then Roll).**

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





CUST_ROT1_ROLL (Custom Rotation 1 - Roll)

DEFAULT: 0	RANGE: -180 180	UNITS: deg

Custom Rotation 1 - Roll (CUST_ROT1_ROLL)

Description

Defines the roll angle for the first custom software rotation.

- **Ordering: Euler 321 (Yaw, then Pitch, then Roll).**

CUST_ROT1_YAW (Custom Rotation 1 - Yaw)

DEFAULT: 0	RANGE: -180 180	UNITS: deg

Custom Rotation 1 - Yaw (CUST_ROT1_YAW)

Description

Defines the yaw angle for the first custom software rotation.

- **Ordering: Euler 321 (Yaw, then Pitch, then Roll).**

CUST_ROT2_PITCH (Custom Rotation 2 - Pitch)

DEFAULT: 0	RANGE: -180 180	UNITS: deg

Custom Rotation 2 - Pitch (CUST_ROT2_PITCH)

Description

Defines the pitch angle for the second custom software rotation.

CUST_ROT2_ROLL (Custom Rotation 2 - Roll)

DEFAULT: 0	RANGE: -180 180	UNITS: deg

Custom Rotation 2 - Roll (CUST_ROT2_ROLL)

Description

Defines the roll angle for the second custom software rotation.

CUST_ROT2_YAW (Custom Rotation 2 - Yaw)

DEFAULT: 0	RANGE: -180 180	UNITS: deg

Custom Rotation 2 - Yaw (CUST_ROT2_YAW)

Description

Defines the yaw angle for the second custom software rotation.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





CUST_ROT_ENABLE (Enable Custom rotations)

DEFAULT: 0	RANGE: 0 1

Enable Custom rotations (CUST_ROT_ENABLE)

Description

Master switch for the custom rotations library. When enabled, it allows sensors (like compasses or IMUs) to use a specific user-defined rotation (Custom 1 or Custom 2) instead of the standard 45/90 degree increments.

Tuning & Behavior

- **Default Value:** 0 (Disabled)
- **Requires reboot to take effect.**

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **DARM** parameter group (specifically `DARM_RATEMAX`) manages the behavior of the **AP_Logger** system when the vehicle is **Disarmed**.

Normally, high-frequency logging is only required during flight. However, it is often useful to log data while on the ground (for debugging GPS health, sensor noise, or configuration issues). This parameter prevents the log file from growing massively while the drone sits idle.

Key Concepts

1. Rate Limiting

Instead of logging IMU data at 400Hz or GPS data at 10Hz, the logger can be throttled down to a lower frequency (e.g., 1Hz or 5Hz) while disarmed.

Parameter Breakdown

- `DARM_RATEMAX`: The maximum frequency (Hz) for logging messages while disarmed.
  - **0:** No limit (Logs at full speed if `LOG_DISARMED` is enabled).
  - **>0:** Limits logging to this rate.

Integration Guide

- **Debug:** Set `DARM_RATEMAX = 0` and `LOG_DISARMED = 1` if you are diagnosing sensor issues on the bench.
- **Normal:** Keep at default or set to a low value (e.g., 5Hz) to save SD card space if you leave `LOG_DISARMED` on.

Developer Notes

- **Library:** `libraries/AP_Logger`
- **Logic:** Acts as a gatekeeper in the `Write_*` methods of the logger.

DARM_RATEMAX (Disarmed Logging Max Rate)

DEFAULT: 0	UNITS: Hz

Disarmed Logging Max Rate (DARM_RATEMAX)

Description

Limits the frequency of data logging while the vehicle is disarmed. This helps conserve SD card space when the aircraft is powered on but not flying.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **DDS** parameter group configures the **Micro-ROS (DDS)** client embedded within ArduPilot. This system allows the autopilot to communicate natively with **ROS 2** (Robot Operating System) nodes running on a companion computer (e.g., Raspberry Pi, Jetson).

This replaces the traditional MAVLink-to-ROS bridge (mavros) with a direct, high-bandwidth, low-latency link using the XRCE-DDS protocol.

Key Concepts

1. Connection

The Micro-ROS client connects to a **Micro-ROS Agent** running on the companion computer via Serial or UDP.

2. Domain ID (`DDS_DOMAIN_ID`)

In DDS, the "Domain" is like a virtual network segment. Robots on Domain 0 cannot see messages from robots on Domain 1. This allows multiple robots to operate on the same WiFi network without crosstalk.

3. Reliability (`DDS_MAX_RETRY` / `DDS_TIMEOUT_MS`)

Configures how the client behaves if the Agent is not reachable (e.g., companion computer booting up).

Parameter Breakdown

- `DDS_ENABLE`: Master switch.
- `DDS_DOMAIN_ID`: ROS 2 Domain ID (default 0).
- `DDS_UDP_PORT`: Port for UDP transport (default 2019 or 8888 depending on transport).
- `DDS_TIMEOUT_MS`: Ping timeout.

Integration Guide

Setting up ROS 2

1. **Companion:** Install ROS 2 (Humble or later) and the `micro_ros_agent`.
2. **ArduPilot:** Set `DDS_ENABLE = 1`.
3. **Transport:** Connect Serial or Ethernet.
4. **Launch:** Start the agent: `ros2 run micro_ros_agent micro_ros_agent serial --dev /dev/ttyACM0 ...`
5. **Verify:** Run `ros2 topic list` to see ArduPilot topics (IMU, Battery, etc.).

Developer Notes

- **Library:** `libraries/AP_DDS`
- **Protocol:** XRCE-DDS (eProsima).

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





DDS_DOMAIN_ID (DDS Domain ID (ROS2))

DEFAULT: 0	RANGE: 0 232

DDS Domain ID (DDS_DOMAIN_ID)

Description

`DDS_DOMAIN_ID` is used for high-level robotics integration with **ROS2** (Robot Operating System).

In a ROS2 network, different groups of robots and computers can be isolated by using a "Domain ID." For the autopilot to talk to your companion computer (e.g. Jetson, Raspberry Pi) running ROS2 nodes, this parameter must match the `ROS_DOMAIN_ID` environment variable set on your computer.

Tuning & Behavior

- **Default:** 0.
- **Usage:** Ensure this matches your ROS2 network configuration. If they don't match, the autopilot and the companion computer will not "see" each other.
- **Reboot Required:** Yes.

DDS_ENABLE (DDS enable)

DEFAULT: 1	RANGE: 0 1

DDS enable (DDS_ENABLE)

Description

Master switch for the DDS (Data Distribution Service) client, which enables high-speed, native communication with ROS2 (Robot Operating System 2) environments.

Tuning & Behavior

- **Default Value:** 1 (Enabled if compiled in)
- **Requires reboot to take effect.**

DDS_MAX_RETRY (DDS ping max attempts)

DEFAULT: 10	RANGE: 0 100

DDS ping max attempts (DDS_MAX_RETRY)

Description

This parameter controls the connection robustness between ArduPilot and the ROS 2 (Robot Operating System) DDS Agent. It limits how many times the client will try to establish a session before giving up.

The Mathematics

Counter logic.

The Engineer's View

Defined in `libraries/AP_DDS/AP_DDS_Client.cpp`.

- **0:** Unlimited retries (Best for persistent autonomy).
- **>0:** Give up after N attempts.

Tuning & Behavior

- **Default Value:** 10
- **Recommendation:** Set to 0 for production robots where the companion computer might boot slower than the autopilot.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





DDS_TIMEOUT_MS (DDS Connection Timeout)

DEFAULT: 1000	**RANGE:** 100 5000	**UNITS:** ms

DDS Connection Timeout (DDS_TIMEOUT_MS)

Description

`DDS_TIMEOUT_MS` defines how patient the autopilot is when trying to establish a link with the ROS2 network.

DDS_UDP_PORT (DDS UDP port)

DEFAULT: 2019	**RANGE:** 1 65535

DDS UDP port (DDS_UDP_PORT)

Description

Specifies the network port used for UDP-based DDS communication.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **DE** parameter group (specifically `DE_PIN`) is a helper configuration for the **Torqeedo** electric boat motor driver.

Torqeedo motors often communicate via **RS485**. To use a standard UART-to-RS485 adapter, the autopilot must manually control the **Driver Enable (DE)** pin to switch the transceiver between Transmit and Receive modes (Half-Duplex).

Key Concepts

1. RS485 Half-Duplex

- **Idle/Receive:** The DE pin is held Low. The autopilot listens for data.
- **Transmit:** The autopilot pulls the DE pin High, sends the data bytes, and then immediately pulls DE Low again.

Parameter Breakdown

- `DE_PIN`: The GPIO pin number connected to the RS485 DE/RE pin.
  - **-1:** Automatic/None (Use Hardware Flow Control/CTS if available).

Integration Guide

- **Wiring:** Connect the converter's DE and RE pins together to the Flight Controller's `DE_PIN`.
- **Torqeedo:** This is part of the `AP_Torqeedo` library setup.

Developer Notes

- **Library:** `libraries/AP_Torqeedo`
- **Context:** Only relevant if `TRQD_TYPE` is set.

DE_PIN (Torqeedo DE pin)

`DEFAULT: -1`

Torqeedo DE pin (DE_PIN)

Description

Specifies the hardware pin used for the Data Enable (DE) signal when communicating with a Torqeedo motor via an RS485 converter.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **DEBUG** parameter group (containing the single parameter `DEBUG`) controls the verbosity of the internal **Task Scheduler**.

ArduPilot runs a cooperative scheduler that manages hundreds of tasks (Attitude loop, GPS update, Logging, etc.). If a task takes too long, it can delay critical flight control loops. This parameter helps developers identify those bottlenecks.

Key Concepts

1. Scheduler Slips

A "Slip" occurs when the scheduler cannot run a task at its intended frequency because the CPU is busy doing something else.

2. Task Overruns

An "Overrun" occurs when a specific task takes more time to execute than it was allocated budget for (e.g., a logging task taking 500us when it should only take 100us).

Parameter Breakdown

- `DEBUG`:
  - **0:** Disabled.
  - **1 (Show Slips):** Report when the main loop runs late.
  - **2 (Show Overruns):** Report which task caused the delay.
  - **3 (Show Load):** Report CPU utilization.

Integration Guide

- **Performance Tuning:** If you see "Scheduler Slips" in the message log during flight, enable `DEBUG = 2` to find the culprit task.
- **Warning:** Enabling debug output consumes CPU itself. Turn off for normal flight.

Developer Notes

- **Library:** `libraries/AP_Scheduler`

DEBUG (Scheduler debug level)

DEFAULT: 0	RANGE: 0 3

Scheduler debug level (DEBUG)

Description

This parameter controls the debug output of the real-time task scheduler. It is used by developers and advanced integrators to diagnose performance issues, such as CPU overload, loop rate instability, or specific tasks taking too long to execute.

The Mathematics

N/A. Enum configuration.

The Engineer's View

Defined in `libraries/AP_Scheduler/AP_Scheduler.cpp`.

- **0 (Disabled):** Normal operation.
- **1 (Show Slips):** Logs a message if a task runs later than scheduled (Scheduler Slip).
- **2 (Show Overruns):** Logs a message if a task exceeds its allocated time budget.
- **3 (Show Load):** Displays CPU load statistics.

Tuning & Behavior

- **Default Value:** 0
- **Use Case:** Enable temporarily if you suspect the flight controller is overloaded or if you are developing new code and want to verify timing compliance.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **DECEL** parameter group (specifically `DECEL_MAX`) sets the physical braking limits for **ArduRover** (Ground Vehicles and Boats).

Unlike aircraft, ground vehicles need separate limits for acceleration (to prevent wheelies) and deceleration (to prevent skidding or front-flips).

Key Concepts

1. Braking Aggression

This parameter tells the navigation controller how hard it can apply the brakes when approaching a waypoint or stopping.

- **Too High:** The vehicle might skid, lock wheels, or tip forward.
- **Too Low:** The vehicle will overshoot waypoints or crash into obstacles because it can't stop in time.

Parameter Breakdown

- `DECEL_MAX`: Maximum deceleration in m/s^2.
  - **Example:** 1.0 m/s^2 means the vehicle can slow down by 1 m/s every second.

Integration Guide

1. **Test:** Drive at full speed in Manual mode and slam the brakes.
2. **Observe:** Does it skid? Does the nose dip dangerously?
3. **Tune:** Set `DECEL_MAX` slightly lower than the physical limit you observed.

Developer Notes

- **Library:** `ArduRover/AP_L1_Control` (Navigation).

DECEL_MAX (Throttle deceleration max)

DEFAULT: 0	RANGE: 0 10	UNITS: m/s^2

Throttle deceleration max (DECEL_MAX)

Description

Limits the maximum rate at which the Rover's speed can be reduced by the automatic throttle controller.

- **A value of 0 means no limit (or uses ATC_ACCEL_MAX).**

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **DEFLT** parameter group is specific to **DroneCAN / AP_Periph** firmware. It allows the user to reconfigure the "default" personality of a peripheral node without reflashing it.

Since the same hardware (e.g., a generic F4 board) can serve as a GPS node, an ESC node, or a Servo controller, this parameter tells the firmware which subsystems to initialize at boot.

Key Concepts

1. Personality Selection

A single `AP_Periph` binary might support multiple roles.

- **GPS Node:** Initializes UARTs for GPS.
- **ESC Node:** Initializes PWM outputs for motor control.
- **Rangefinder Node:** Initializes I2C/Serial for Lidar.

Parameter Breakdown

- `DEFLT_MODE`: Bitmask or Enum selecting the active drivers.

Integration Guide

- **Usage:** Usually set by the factory. If you are repurposing a generic CAN node, change this to match your connected sensors.

Developer Notes

- **Library:** `Tools/AP_Periph`

DEFLT_MODE (Mount Default Mode)

```
DEFAULT: 3
```

Mount Default Mode (DEFLT_MODE)

Description

Specifies the initial operating state of the gimbal (mount) when the flight controller powers up.

Tuning & Behavior

- **Default Value:** 3 (RC Targeting)
- **Values:** 0:Retracted, 1:Neutral, 2:MavLink Targeting, 3:RC Targeting, 4:GPS Point

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **DELAY** parameter group appears to be a legacy or miscategorized entry in the documentation structure.

In modern ArduPilot, delays are almost always associated with the specific system they affect, to correct for sensor lag or actuation timing.

Common Delays found elsewhere

- `VISO_DELAY_MS`: Visual Odometry latency.
- `CHUTE_DELAY_MS`: Parachute release delay.
- `GPS_DELAY_MS`: GPS measurement lag.
- `FLOW_DELAY_MS`: Optical Flow sensor lag.

Developer Notes

- **Status:** Likely deprecated or an extraction artifact.

DELAY_MS (Sensor Delay (Invalid/Legacy))

DEFAULT: 0	UNITS: ms

Sensor Delay (DELAY_MS)

Description

This parameter is not found as a top-level DELAY_MS in the main codebase. It likely refers to a specific sensor delay in a subgroup (e.g., Visual Odometry or Parachute).

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **DEV** parameter group (specifically `DEV_OPTIONS`) contains flags for enabling experimental or developer-only features in **ArduCopter**.

These features are often unstable, untested, or intended for deep debugging of internal state.

Key Concepts

1. Developer Flags

Bitmask options that toggle specific internal behaviors.

- **Bit 0:** Enable ADS-B simulation (SITL).
- **Bit 1:** Enable debug logging for certain subsystems.
- (Options vary significantly between firmware versions).

Parameter Breakdown

- `DEV_OPTIONS`: Bitmask.

Integration Guide

- **Warning:** Do not enable these unless you are an ArduPilot developer or strictly following instructions from one. They can alter flight behavior unexpectedly.

Developer Notes

- **Library:** `ArduCopter/Parameters.cpp`

DEV_OPTIONS (Development options)

`DEFAULT: 0`

Development options (DEV_OPTIONS)

Description

A bitmask for enabling experimental or development-only features. These options are typically used by ArduPilot developers and may change without notice.

Tuning & Behavior

- **Default Value:** 0
- **Bitmask Examples (Copter):**
  - Bit 0: ADSBMavlinkProcessing
  - Bit 1: DevOptionVFR_HUDRelativeAlt

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY

Overview

The **DID** parameter group configures the **Remote ID** (OpenDroneID) subsystem. This is required for compliance with FAA (USA), EASA (Europe), and other CAA regulations regarding UAV identification.

Remote ID systems broadcast the drone's position, altitude, velocity, and serial number over Bluetooth/Wi-Fi to local receivers (like smartphones).

Key Concepts

1. ODID Hardware

ArduPilot supports communicating with external Remote ID modules (like CubeID, Bluemark, Holybro) via MAVLink (`DID_MAVPORT`) or CAN (`DID_CANDRIVER`).

2. Data Source

The autopilot feeds the ODID module with live GPS and telemetry data. The module then handles the BLE/WiFi broadcasting.

Parameter Breakdown

- `DID_ENABLE`: Master switch.
- `DID_MAVPORT`: Serial port connected to the ODID module (if using Serial).
- `DID_CANDRIVER`: CAN driver connected to the ODID module (if using CAN).
- `DID_OPTIONS`: Configuration flags.

Integration Guide

1. **Hardware:** Connect the Remote ID module to a Telemetry port or CAN bus.
2. **Enable:** Set `DID_ENABLE = 1`.
3. **Port:** If Serial, set `DID_MAVPORT` to the serial port index (e.g., 2 for Serial2). Ensure `SERIAL2_PROTOCOL` is set to MAVLink 2.
4. **Verify:** Check GCS for "OpenDroneID" status messages.

Developer Notes

- **Library:** `libraries/AP_OpenDroneID`
- **Protocol:** Uses the OpenDroneID MAVLink extension.

DID_BARO_ACC (Barometer vertical accuracy)

DEFAULT: -1.0	UNITS: m

Barometer vertical accuracy (DID_BARO_ACC)

Description

Specifies the vertical position error (accuracy) of the barometer for Remote ID reporting.

- **-1.0 (default) disables reporting this metric.**

DID_CANDRIVER (DroneCAN driver number)

DEFAULT: 0	RANGE: 0 2

DroneCAN driver number (DID_CANDRIVER)

Description

Selects which CAN driver instance should be used to transmit Remote ID data to a DroneCAN-capable transponder.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





DID_ENABLE (Enable ODID subsystem)

DEFAULT: 0	RANGE: 0 1

Enable ODID subsystem (DID_ENABLE)

Description

Master switch for OpenDroneID (Remote ID) functionality. Enables compliance with broadcast ID regulations.

Tuning & Behavior

- **Default Value:** 0 (Disabled)
- **Requires reboot to take effect.**

DID_MAVPORT (MAVLink serial port)

DEFAULT: -1	RANGE: -1 6

MAVLink serial port (DID_MAVPORT)

Description

Specifies the physical serial port connected to the Remote ID broadcast hardware (transponder).

- **-1 disables serial output (e.g., if using DroneCAN).**

DID_OPTIONS (OpenDroneID options)

DEFAULT: 0

OpenDroneID options (DID_OPTIONS)

Description

Configuration bitmask for Remote ID enforcement and identity locking.

Tuning & Behavior

- **Default Value:** 0
- **Bitmask:**
  - Bit 0: EnforceArming (Refuse to arm if Remote ID is not healthy)
  - Bit 1: AllowNonGPSPosition
  - Bit 2: LockUASIDOnFirstBasicIDRx

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **DIR** parameter group contains legacy or driver-specific settings for **Analog Wind Vanes**.

Note: In modern ArduPilot versions, these are often aliased to or superseded by the **WNDVN** (Wind Vane) parameter group. However, the `DIR_` prefix persists for specific analog direction drivers (like the Modern Devices Wind Sensor).

Key Concepts

1. Analog Direction Sensing

Many simple wind vanes output a voltage (0-3.3V or 0-5V) proportional to the wind angle (0-360 degrees).

- `DIR_PIN`: The ADC pin reading this voltage.
- `DIR_V_MIN` / `DIR_V_MAX`: The voltage range corresponding to 0-360 degrees.

2. Offset (`DIR_OFS`)

Mechanical installation error correction. If the vane points North but the autopilot thinks it points East, use this offset to correct it.

Parameter Breakdown

- `DIR_PIN`: ADC input pin.
- `DIR_FILT`: Filter frequency (Hz) to smooth the noisy analog signal.
- `DIR_DZ`: Deadzone (degrees) around North crossover (0/360 transition).

Integration Guide

- **See:** WNDVN for main Wind Vane configuration. Use `DIR_` parameters only if your selected `WNDVN_TYPE` requires them.

Developer Notes

- **Library:** `libraries/AP_WindVane`

DIR_DELAY (Direction Change Delay (Invalid/Legacy))

DEFAULT: 1.0	RANGE: 0 5	UNITS: s

Direction Change Delay (DIR_DELAY)

Description

This parameter is not found as a common wind vane parameter. It appears to be a miscategorized Torqeedo motor parameter.

DIR_DZ (Wind vane deadzone)

DEFAULT: 0	RANGE: 0 360	UNITS: deg

Wind vane deadzone (DIR_DZ)

Description

A small range of movement around the zero-point that is ignored by the software to prevent rapid, jittery direction changes when the wind is perfectly head-on.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





DIR_FILT (apparent Wind vane direction filter)

DEFAULT: 0.5	RANGE: 0 10	UNITS: Hz

apparent Wind vane direction filter (DIR_FILT)

Description

Low-pass filter frequency for smoothing the apparent wind direction data. Helps to remove noise caused by turbulence or physical sensor vibration.

DIR_OFS (Wind vane headwind offset)

DEFAULT: 0.0	RANGE: 0 360	UNITS: deg

Wind vane headwind offset (DIR_OFS)

Description

Software calibration used to align the zero-point of the wind vane with the longitudinal axis (nose) of the aircraft.

DIR_PIN (Wind vane analog voltage pin)

DEFAULT: 15

Wind vane analog voltage pin (DIR_PIN)

Description

Specifies the analog-to-digital converter (ADC) pin connected to the wind vane sensor.

DIR_V_MAX (Wind vane voltage maximum)

DEFAULT: 3.3	RANGE: 0 5.0	UNITS: V

Wind vane voltage maximum (DIR_V_MAX)

Description

Calibration point for the maximum voltage output of the analog wind vane sensor, corresponding to its highest range.

DIR_V_MIN (Wind vane voltage minimum)

DEFAULT: 0.0	RANGE: 0 5.0	UNITS: V

Wind vane voltage minimum (DIR_V_MIN)

Description

Calibration point for the minimum voltage output of the analog wind vane sensor, corresponding to its lowest range.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **DIS** parameter group (specifically `DIS_MASK`) is a utility feature primarily found in **ArduBlimp** (and sometimes Sub/Rover) to selectively disable control axes.

Key Concepts

1. Axis Isolation

When tuning or testing a complex vehicle like a Blimp, it is often useful to disable certain axes (e.g., disable Yaw control to focus on Pitch tuning). This mask allows enabling/disabling axes without unplugging motors.

Parameter Breakdown

- `DIS_MASK`: Bitmask of axes to disable.
  - **Bit 0:** Disable Sway (Y).
  - **Bit 1:** Disable Surge (X).
  - **Bit 2:** Disable Heave (Z).
  - **Bit 3:** Disable Yaw.

Integration Guide

- **Default:** 0 (All enabled).
- **Usage:** Set to 15 (1+2+4+8) to disable all output for safe bench testing of other systems.

Developer Notes

- **Library:** `ArduBlimp` (Vehicle specific).

DIS_MASK (Disable output mask)

DEFAULT: 0	RANGE: 0 15	UNITS: Bitmask

Disable output mask (DIS_MASK)

Description

This parameter allows specific control axes (Surge, Sway, Heave, Yaw) to be completely disabled when the Blimp is in **Velocity** or **Loiter** modes. This is typically used for testing, tuning individual axes, or for airframe configurations that lack actuation in certain directions.

The Mathematics

The parameter is a bitmask. If a bit is set (1), the corresponding axis control output is forced to zero (or disabled logic).

- **Bit 0:** Disable Y-axis (Sway / Right)
- **Bit 1:** Disable X-axis (Surge / Forward)
- **Bit 2:** Disable Z-axis (Heave / Down)
- **Bit 3:** Disable Yaw

The Engineer's View

Defined in `Blimp/Parameters.cpp`. Checked in `Blimp/Loiter.cpp` (and likely `ModeVelocity.cpp`).
Note the unusual ordering found in the source code: Bit 0 is checked against `err_xyz.y` and Bit 1 against `err_xyz.x`.

Tuning & Behavior

- **Default Value:** 0 (All axes enabled)
- **1 (2^0):** Disable Sway.
- **2 (2^1):** Disable Surge.
- **4 (2^2):** Disable Heave.
- **8 (2^3):** Disable Yaw.
- **Example:** Set to 15 (1+2+4+8) to disable all control in these modes.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





AUTO DISARM CONFIGURATION (DISARM)

Overview

The **DISARM** parameter group (specifically `DISARM_DELAY`) configures the automatic disarming safety feature.

If the vehicle is Armed but the motors are at idle (throttle zero) and the vehicle is not moving (landed), the autopilot will automatically disarm itself after a set timeout. This prevents injury if the pilot approaches the vehicle forgetting it is still live.

Key Concepts

1. Landed State Detection

The timer only starts counting when the autopilot is confident it is **Landed**. This requires consistent low throttle and low accelerometer variance.

2. Delay Timer

- `DISARM_DELAY`: Time in seconds.
  - **0:** Disabled (Will never auto-disarm).
  - **10 (Default):** Disarms after 10 seconds of idle on the ground.

Parameter Breakdown

- `DISARM_DELAY`: Timeout (seconds).

Integration Guide

- **Safety:** Keep this enabled (10s or 15s) for all vehicles.
- **Exception:** For long-endurance loiter on the ground (e.g., waiting for GPS), you might disable it, but use the Safety Switch instead.

Developer Notes

- **Vehicle:** All (Copter, Plane, Rover).
- **Logic:** Managed by the `AP_Motors` or vehicle main loop.

DISARM_DELAY (Disarm delay)

DEFAULT: 10	RANGE: 0 127	UNITS: s

Disarm delay (DISARM_DELAY)

Description

Defines the period of inactivity (typically at zero throttle and landed) after which the flight controller will automatically disarm the motors for safety.

Tuning & Behavior

- **Default Value:** 10 s (Typical)
- **Range:** 0 to 127 s
- **Setting to 0 disables the auto-disarm feature.**

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **DIST** parameter group (specifically `DIST_MAX`) acts as a simple **Geofence** radius limit for some vehicle types (primarily Rover).

Key Concepts

1. Distance from Home

Limits how far the vehicle can travel from the Home position (Arming location).

- **Action:** If the limit is breached, the vehicle typically triggers RTL (Return to Launch) or stops.

Parameter Breakdown

- `DIST_MAX`: Maximum allowed distance from Home (meters).
  - **0:** Disabled.

Integration Guide

- **Simple Safety:** Useful for keeping a rover within RC range without setting up complex polygon fences.

Developer Notes

- **Vehicle: ArduRover**.
- **Note:** Plane and Copter typically use the full `FENCE_` library instead.

DIST_MAX (Follow distance maximum)

DEFAULT: 100	RANGE: 1 1000	UNITS: m

Follow distance maximum (DIST_MAX)

Description

Safety limit for Follow mode. If the target moves further away than this distance, the aircraft will stop following and typically enter Loiter or RTL to prevent a runaway scenario.

Tuning & Behavior

- **Default Value:** 100 m
- **Range:** 1 to 1000 m
- **Ensures the aircraft stays within a reasonable radio or visual range of the target.**

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY

Overview

The **DISTANCE** parameter group is specific to the **Antenna Tracker** firmware. It prevents the tracker from trying to track a vehicle that is too close.

Key Concepts

1. Cone of Silence

When a vehicle is very close (or directly overhead), the angular rate required to track it becomes infinite (singularity). Attempting to track here causes violent servo movements.

- `DISTANCE_MIN` defines this "safe zone" radius.

Parameter Breakdown

- `DISTANCE_MIN`: Minimum distance in meters.
  - If `Target Distance` < `DISTANCE_MIN`: Stop tracking (Hold position).

Integration Guide

- **Launch:** Set this to 5-10 meters so the tracker doesn't freak out while you are arming the drone next to it.

Developer Notes

- **Vehicle: AntennaTracker**.

DISTANCE_MIN (Distance minimum to target)

DEFAULT: 0	RANGE: 0 100	UNITS: m

Distance minimum to target (DISTANCE_MIN)

Description

This parameter defines a "Cone of Silence" radius around the Antenna Tracker. If the target vehicle comes closer than this distance, the tracker stops updating its pan/tilt angles and holds position. This prevents the tracker from whipping around violently or grinding its gears trying to track a target passing directly overhead or nearby.

The Mathematics

$$\text{If Distance} < \text{DISTANCE_MIN} \implies \text{Hold Position}$$

The Engineer's View

Defined in `AntennaTracker/Parameters.cpp`.

Tuning & Behavior

- **Default Value:** 0 (Disabled)
- **Recommendation:** Set to 5-10 meters to protect the mechanism during launch/landing near the tracker.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY

Overview

The **DOCK** parameter group configures the **Autonomous Docking** behavior for **ArduRover**. This mode allows a rover to precisely approach and connect with a docking station (usually for charging).

Key Concepts

1. Approach Logic

The rover approaches a target (the dock) at a slow speed.

- **Direction:** The rover expects to approach from a specific vector.
- **Correction:** It uses heading correction to ensure it enters straight.

Parameter Breakdown

- `DOCK_SPEED`: Approach speed (m/s). Needs to be slow enough to stop accurately but fast enough to maintain steering.
- `DOCK_DIR`: The direction (heading) the dock faces.
- `DOCK_STOP_DIST`: Distance from the target to cut motors.

Integration Guide

- **Target:** The dock location is usually provided by a mission waypoint or a localized beacon.
- **Usage:** Switch to `DOCK` mode when near the station.

Developer Notes

- **Library:** `Rover/mode_dock.cpp`.

DOCK_DIR (Docking Approach Direction)

DEFAULT: -1	RANGE: 0 360	UNITS: deg

Docking Approach Direction (DOCK_DIR)

Description

`DOCK_DIR` specifies the "Entry Vector" for autonomous docking.

Many docking stations (like a boat ramp or a specialized charging garage) can only be entered from one specific direction. If the rover approached from the wrong side, it would hit a wall or miss the latch. This parameter defines the required compass heading for the final approach leg.

- **-1:** Unset / No specific direction required.
- **0-360:** Compass heading (e.g. 90 = East).

Tuning & Behavior

- **Default:** -1.
- **Usage:** Align your docking station and use a compass to find the required approach heading. The drone will automatically maneuver to line itself up with this vector before starting the final approach.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





DOCK_HDG_CORR_EN (Dock Heading Correction Enable)

DEFAULT: 0	RANGE: 0 1

Dock Heading Correction Enable (DOCK_HDG_CORR_EN)

Description

`DOCK_HDG_CORR_EN` is used for precision docking where the vehicle must enter the dock from a specific direction (e.g. into a narrow charging slot).

If disabled, the vehicle flies directly from its current position to the dock's location. If enabled, the vehicle will first navigate to a "Capture Point" and then follow a straight vector (the "Docking Lane") to ensure it arrives perfectly squared up to the target.

- **1: Enabled.** Vehicle attempts a straight-in head-on approach.
- **0: Disabled.** Vehicle flies direct-to-target.

Tuning & Behavior

- **Default:** 0.
- **Requirement:** Requires the dock to provide its own heading information via MAVLink or a beacon.

DOCK_HDG_CORR_WT (Dock Heading Correction Weight)

DEFAULT: 0.75	RANGE: 0 1.0

Dock Heading Correction Weight (DOCK_HDG_CORR_WT)

Description

`DOCK_HDG_CORR_WT` fine-tunes the geometry of the docking path.

DOCK_SPEED (Dock Mode Speed Limit)

DEFAULT: 0	RANGE: 0 100	UNITS: m/s

Dock Mode Speed Limit (DOCK_SPEED)

Description

`DOCK_SPEED` sets the "Approach Velocity" for autonomous docking.

In **Dock** mode (used for Rovers and Boats), the vehicle navigates to a precise coordinate or a MAVLink beacon to dock for charging or recovery. Because this requires high precision and safety, this parameter caps the speed to ensure the vehicle doesn't "over-shoot" the dock or cause a collision.

Tuning & Behavior

- **Default:** 0 (Uses the vehicle's standard cruise speed).
- **Recommendation:** Set to a very low value (e.g. **0.5 m/s** or **1.0 m/s**) to ensure a safe and controlled approach.
- **Significance:** Slower speeds allow the position control loops more time to react to external disturbances (like currents or uneven ground) as the drone enters the final docking phase.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





DOCK_STOP_DIST (Docking Target Stop Distance)

DEFAULT: 0.3	RANGE: 0 2.0	UNITS: m

Docking Target Stop Distance (DOCK_STOP_DIST)

Description

`DOCK_STOP_DIST` defines the "Final Proximity" for a successful dock.

This parameter tells the autopilot exactly when to call the mission "Complete" and shut down the motors. Because real-world vehicles have momentum, this sets the point at which the braking logic triggers to ensure the vehicle stops exactly at the target.

Tuning & Behavior

- **Default:** 0.3 meters (30 cm).
- **Calibration:** If your rover consistently hits the docking station too hard, increase this value. If it stops too far away and fails to engage the charging pins, decrease it.
- **Accuracy:** This requires a highly accurate position estimate (e.g. RTK GPS or a specialized docking beacon).

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **DOMAIN** parameter group (specifically `DOMAIN_ID`) allows configuring the **ROS 2 Domain ID** for the DDS interface.

Note: This is often an alias or legacy name for `DDS_DOMAIN_ID`.

Key Concepts

1. ROS 2 Domains

In ROS 2, the "Domain ID" isolates groups of robots.

- **Domain 0:** Default.
- **Domain 1-232:** Alternate domains.

Parameter Breakdown

- `DOMAIN_ID`: Integer ID.

Integration Guide

- **Match:** This value **must** match the `ROS_DOMAIN_ID` environment variable set on your companion computer (Raspberry Pi/Jetson). If they differ, the robot will not receive messages.

Developer Notes

- **Library:** `libraries/AP_DDS`.

DOMAIN_ID (DDS Domain ID)

DEFAULT: 0	RANGE: 0 232

DDS Domain ID (DOMAIN_ID)

Description

Matches the ArduPilot DDS client to a specific ROS2 logical network (domain). Only nodes with the same Domain ID can communicate with each other.

Tuning & Behavior

- **Default Value:** 0
- **Range:** 0 to 232
- **Must match the ROS_DOMAIN_ID environment variable on your companion computer.**

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **DRV** parameter group (specifically `DRV_OPTIONS` or `GPS_DRV_OPTIONS`) provides a mechanism to enable experimental features or workarounds for specific hardware drivers (primarily GPS).

Key Concepts

1. Driver-Specific Flags

The meaning of the bits depends heavily on the specific GPS driver being used (u-Blox, SBP, NMEA).

- **Examples:** Enable/Disable specific NMEA messages, change baud rate negotiation, or enable experimental constellations (Galileo/BeiDou) if the driver doesn't support a dedicated parameter for it.

Parameter Breakdown

- `DRV_OPTIONS`: Bitmask.

Integration Guide

- **Documentation:** Refer to the full parameter list for your specific firmware version to see what the bits do. It changes frequently.

Developer Notes

- **Library:** `libraries/AP_GPS`
- **Scope:** Used inside `AP_GPS_uBlox.cpp`, etc.

`DRV_OPTIONS` (GPS Driver Options)

`DEFAULT: 0`

GPS Driver Options (DRV_OPTIONS)

Description

Bitmask for configuring driver-specific GPS behaviors, such as enabling specific NMEA sentences or handling hardware-level timing.

Tuning & Behavior

- **Default Value:** 0
- **Bitmask Examples:**
  - Bit 0: Use GPS_DRV_OPTIONS bit 0 (varies by backend)

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

Key Concepts

DSARM_SCR (OSD Disarm Screen)

DEFAULT: 0	RANGE: 0 4

OSD Disarm Screen (DSARM_SCR)

Description

Specifies which OSD layout should be automatically displayed when the vehicle is disarmed.

- **A value of 0 keeps the current OSD screen active.**

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **DSHOT** parameter group (often appearing as `MOT_PWM_TYPE` options or `SERVO_BLH_` in older setups) configures the **DShot** digital motor protocol.

DShot is a digital protocol for ESCs that is faster, more robust against noise, and requires no calibration (unlike PWM/OneShot).

Key Concepts

1. Protocol Speed

DShot comes in varying speeds:

- **DShot150:** 150 kbit/s. Slowest, most robust against noise/long wires.
- **DShot300:** 300 kbit/s. Standard.
- **DShot600:** 600 kbit/s. Fast.
- **DShot1200:** 1200 kbit/s. Very fast, requires high-quality wiring and F4/F7/H7 processors.

2. Bidirectional DShot

Allows the ESC to send RPM telemetry back down the signal wire. This is critical for the **Harmonic Notch Filter**.

Parameter Breakdown

- `DSHOT_RATE`: 0=Disabled, 150, 300, 600, 1200.
- `DSHOT_ESC`: Bitmask to select which outputs use DShot.

Integration Guide

1. **Set Type:** Set `MOT_PWM_TYPE = 4` (DShot150) or `6` (DShot600).
2. **Enable:** Configure the `SERVOx_FUNCTION` to Motors.
3. **No Cal:** Do not perform ESC calibration. The endpoints are fixed (1000-2000 digital).

Developer Notes

- **Library:** `libraries/AP_HAL_ChibiOS` (RCOutput)
- **Requirement:** Hardware timer support (DMA). Not all pins on all boards support DShot. Check `hwdef.dat` or the "Alt Config" documentation.

DSHOT_ESC (Servo DShot ESC type)

DEFAULT: 0	RANGE: 0 4

Servo DShot ESC type (DSHOT_ESC)

Description

Specifies the firmware generation of the connected DShot ESCs, which determines the availability of features like bi-directional telemetry.

Tuning & Behavior

- **Default Value:** 0 (None)
- **Values:** 0:None, 1:BLHeli32/Kiss/AM32, 2:BLHeli_S/BlueJay, 3:BLHeli32/AM32/Kiss+EDT, 4:BLHeli_S/BlueJay+EDT

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





DSHOT_RATE (Servo DShot output rate)

DEFAULT: 0	RANGE: 0 4

Servo DShot output rate (DSHOT_RATE)

Description

Defines the communication frequency between the flight controller and DShot-capable ESCs.

Tuning & Behavior

- **Default Value:** 0 (1Khz fixed for low loop rates)
- **Values:** 1:loop-rate, 2:double loop-rate, 3:triple loop-rate, 4:quadruple loop rate
- **Higher rates improve responsiveness but increase CPU load.**

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





DIFFERENTIAL SPOILER CONFIGURATION (DSPOILER)

Overview

The **DSPOILER** parameter group configures **Differential Spoilers**, also known as **Split Ailerons**, **Decelerons**, or **Drag Rudders**.

This control surface configuration is common on flying wings (like the B-2 Spirit) where there is no vertical stabilizer. Yaw control is achieved by splitting the aileron on one wing (opening it like a clam shell) to create drag on that side.

Key Concepts

1. Drag for Yaw

- **Yaw Left:** Open Left Spoiler (Left Upper Surface Up, Left Lower Surface Down).
- **Yaw Right:** Open Right Spoiler.

2. Crow Braking

Both spoilers can be opened simultaneously to act as airbrakes, increasing drag for steep descents without increasing airspeed.

- `DSPOILER_CROW_W1` / `W2`: Weighting for how much inner/outer surfaces move during braking.

3. Aileron Match (`DSPOILER_AILMTCH`)

Ensures that the roll authority is maintained even when spoilers are deployed.

Parameter Breakdown

- `DSPOILER_OPTS`: Options bitmask.
- `DSPOILER_CROW_W1`: Weighting for inner control surface.
- `DSPOILER_CROW_W2`: Weighting for outer control surface.

Integration Guide

1. **Outputs:** Map 4 servo outputs: Left Outer, Left Inner, Right Inner, Right Outer.
2. **Functions:** Use `24` (Left Elevon) and `77` (Right Elevon) or specific spoiler functions depending on the mix.
3. **Tune:** Adjust rudder mixing to get sufficient yaw authority without inducing adverse roll.

Developer Notes

- **Library:** `ArduPlane/servos.cpp`
- **Complexity:** Requires careful mixing to handle Roll (Aileron), Pitch (Elevator), and Yaw (Drag) simultaneously on the same surfaces.

DSPOILER_AILMTCH (Crow flap aileron matching)

DEFAULT: 100	RANGE: 0 100	UNITS: %

Crow flap aileron matching (DSPOILER_AILMTCH)

Description

Determines how much the spoilers continue to act as ailerons while the Crow braking is active. 100% means they maintain full roll authority.

DSPOILER_CROW_W1 (Crow outer flap weight)

DEFAULT: 0	RANGE: 0 100	UNITS: %

Crow outer flap weight (DSPOILER_CROW_W1)

Description

Defines the scaling factor for the "Crow" brake effect (flaps down, spoilers up) on the outer pair of differential spoilers.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





DSPOILER_CROW_W2 (Crow inner flap weight)

DEFAULT: 0	RANGE: 0 100	UNITS: %

Crow inner flap weight (DSPOILER_CROW_W2)

Description

Defines the scaling factor for the "Crow" brake effect on the inner pair of differential spoilers.

DSPOILER_OPTS (Crow flap options)

DEFAULT: 3

Crow flap options (DSPOILER_OPTS)

Description

Bitmask for advanced configuration of differential spoilers and crow flaps.

Tuning & Behavior

- **Default Value:** 3
- **Bitmask:**
  - Bit 0: Enable outer spoilers
  - Bit 1: Enable inner spoilers
  - Bit 2: Progressive crow (scales with airspeed)
  - Bit 3: Use pitch trim for crow

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY

Overview

Key Concepts

DSPOILR_RUD_RATE (Differential spoilers rudder rate)

DEFAULT: 100	RANGE: -100 100	UNITS: %

Differential spoilers rudder rate (DSPOILR_RUD_RATE)

Description

Defines the authority of the rudder stick over the differential spoilers. When set to 100%, full rudder input results in full spoiler deflection on one side to aid in yaw.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **EAHRS** parameter group configures the **External Attitude Heading Reference System** driver. This allows ArduPilot to utilize high-end, industrial-grade AHRS/INS units (like VectorNav, MicroStrain, or Inertial Labs) as the primary source of attitude and position data, bypassing the internal EKF.

Key Concepts

1. External Fusion

Instead of the autopilot taking raw IMU/GPS data and running its own EKF, it trusts the "Solution" provided by the external box. The external unit does the heavy lifting of sensor fusion.

2. Sensor Selection (`EAHRS_SENSORS`)

You can choose which data streams to consume:

- **Attitude:** Roll, Pitch, Yaw.
- **Position:** Lat, Lon, Alt.
- **Velocity:** NED velocity.
- **Raw Data:** Accelerometers, Gyros (for logging).

Parameter Breakdown

- `EAHRS_TYPE`: Driver selection.
  - 0: Disabled.
  - 1: VectorNav (VN-100/200/300).
  - 2: MicroStrain (3DM-GX5).
  - 3: Inertial Labs.
- `EAHRS_RATE`: Data update rate (Hz).
- `EAHRS_SENSORS`: Bitmask of active data channels.

Integration Guide

1. **Hardware:** Connect the unit to a High-Speed Serial Port (Telem 1/2).
2. **Serial:** Set `SERIALx_PROTOCOL = 36` (AHRS) and baud rate (usually 921600 or higher).
3. **Enable:** Set `EAHRS_TYPE` to your device.
4. **Reboot.**
5. **Check:** Look for `AHRS_EXTERNAL` status in the GCS.

Developer Notes

- **Library:** `libraries/AP_ExternalAHRS`
- **Latency:** Critical factor. The driver handles timestamp correction to align external data with the autopilot's clock.

EAHRS_LOG_RATE (AHRS logging rate)

DEFAULT: 10	RANGE: 0 400	UNITS: Hz

AHRS logging rate (EAHRS_LOG_RATE)

Description

Controls how frequently the data from the external AHRS is written to the aircraft's dataflash logs (EAHR and EAHV log messages).

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





EAHRS_OPTIONS (External AHRS options)

DEFAULT: 0

External AHRS options (EAHRS_OPTIONS)

Description

Bitmask for advanced configuration of the external AHRS interface.

Tuning & Behavior

- **Default Value:** 0
- **Bitmask:**
  - Bit 0: Vector Nav use uncompensated values (direct sensor data)

EAHRS_RATE (AHRS data rate)

DEFAULT: 50	RANGE: 50 400	UNITS: Hz

AHRS data rate (EAHRS_RATE)

Description

Configures the frequency at which the flight controller requests orientation and navigation data from the external AHRS module.

- **Minimum value is 50 Hz.**

EAHRS_SENSORS (External AHRS sensors)

DEFAULT: 15	RANGE: 0 15	UNITS: Bitmask

External AHRS sensors (EAHRS_SENSORS)

Description

This parameter determines which sensor data streams from the connected External AHRS (e.g., VectorNav, MicroStrain) should be consumed by the autopilot. This allows for a hybrid setup where, for example, the external GPS and IMU are used, but the internal barometer is preferred.

The Mathematics

Bitmask logic:

- **Bit 0:** Use GPS data.
- **Bit 1:** Use IMU data (Accel/Gyro).
- **Bit 2:** Use Barometer data.
- **Bit 3:** Use Compass/Magnetometer data.

The Engineer's View

Defined in `libraries/AP_ExternalAHRS/AP_ExternalAHRS.cpp`.

- **Default (15 / 0xF):** Use all available sensors from the external unit.
- **Usage:** If the external unit has a poor quality barometer or is mounted in a pressurized box, you might disable Bit 2.

Tuning & Behavior

- **Default Value:** 15 (All)
- **Set to:** The sum of the bits for the desired sensors.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





EAHRS_TYPE (AHRS type)

DEFAULT: 0

AHRS type (EAHRS_TYPE)

Description

Enables the External AHRS library and selects the driver for the specific third-party inertial navigation system (INS) or AHRS hardware.

Tuning & Behavior

- **Default Value:** 0 (None)
- **Values:** 0:None, 1:VectorNav, 2:MicroStrain5, 5:InertialLabs, 7:MicroStrain7
- **Requires reboot after changing to initialize the external device communication.**

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **EFI** parameter group configures the interface for monitoring internal combustion engines equipped with **Electronic Fuel Injection** (EFI) systems or engine control units (ECUs).

ArduPilot does not *control* the injection timing (the ECU does that); instead, it *monitors* the engine status to provide:

1. **RPM:** For throttle governing and vibration filtering.
2. **Fuel Flow:** For accurate remaining range estimation.
3. **Health:** Cylinder temperatures, manifold pressure, and ECU errors.

Key Concepts

1. Driver Support

Supports major UAV engine manufacturers:

- **NWPMU:** Northwest Power Management Unit.
- **Currawong:** CAN-based ECUs.
- **Hirth:** Serial interface.
- **Loweheiser:** Serial interface.
- **MegaSquirt:** Via CAN or Serial.

2. Fuel Density (`EFI_FUEL_DENS`)

Gasoline weight varies by blend and temperature. Setting this correctly ensures accurate weight/balance calculations if the GCS tracks fuel weight.

Parameter Breakdown

- `EFI_TYPE`: Driver selection.
- `EFI_PORT`: Serial port selection (if using UART-based ECU).
- `EFI_BAUDRATE`: Serial speed.

Integration Guide

1. **Connection:** Wire the ECU telemetry output to an Autopilot UART or CAN bus.
2. **Configure:** Set `EFI_TYPE` and port settings.
3. **Verify:** Check the "RPM" and "Fuel" values in the Mission Planner "Status" tab.

Developer Notes

- **Library:** `libraries/AP_EFI`
- **Logging:** Generates `EFI` log messages with detailed engine stats.

`EFI_BAUDRATE` (EFI Port Baud Rate)

DEFAULT: 0	RANGE: 1 1000000

EFI Port Baud Rate (EFI_BAUDRATE)

Description

`EFI_BAUDRATE` sets the speed for the serial connection between the flight controller and the engine's ECU.

- **Default:** 0 (Use driver default).
- **Usage:** Set this to match the output rate of your ECU (e.g. 57600 or 115200).
- **Reboot Required:** Yes.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





EFI_COEF1 (EFI Calibration Coefficient 1)

DEFAULT: 0	RANGE: 0 1

EFI Calibration Coefficient 1 (EFI_COEF1)

Description

Scaling factor (slope) for fuel flow calculation when using the MegaSquirt (MS) EFI protocol.

EFI_COEF2 (EFI Calibration Coefficient 2)

DEFAULT: 0	RANGE: 0 10

EFI Calibration Coefficient 2 (EFI_COEF2)

Description

Offset calibration value for fuel flow when using the MegaSquirt protocol.

EFI_COEF3 (Throttle linearisation - Third Order)

DEFAULT: 0	RANGE: -1 1

Throttle linearisation - Third Order

Note: This parameter functions identically to EFI_COEF1.

EFI_EN (EFI Throttle Linearization Enable)

DEFAULT: 0	RANGE: 0 1

EFI Throttle Linearization Enable (EFI_EN)

Description

`EFI_EN` enables advanced throttle mapping for combustion engines.

Combustion engines are inherently non-linear; the power produced at 50% throttle position is rarely 50% of the total available power. This can make the autopilot's altitude and speed control loops (TECS) perform poorly. Enabling this feature allows ArduPilot to use a measured "Power Curve" to translate its requested power into the correct throttle percentage for the ECU.

- **0: Disabled (Default).** Uses a simple linear throttle-to-RPM mapping.
- **1: Enabled.** Activates the linearization model.

EFI_FUEL_DENS (ECU Fuel Density)

DEFAULT: 0	RANGE: 0 10000	UNITS: kg/m^3

ECU Fuel Density (EFI_FUEL_DENS)

Description

The density of the fuel being used, required for accurate mass-based fuel consumption calculations.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





EFI_OFS (EFI Throttle Linearization Offset)

DEFAULT: 0	RANGE: 0 1.0

EFI Throttle Linearization Offset (EFI_OFS)

Description

`EFI_OFS` defines the starting point of the power curve calibration.

EFI_PORT (EFI Serial Port)

DEFAULT: -1	RANGE: 0 10

EFI Serial Port (EFI_PORT)

Description

Defines the hardware serial port instance used by the AP_Periph node to communicate with the EFI ECU.

EFI_THRLIN_OFS (throttle linearization offset)

DEFAULT: 0	RANGE: 0 100

throttle linearization offset (EFI_THRLIN_OFS)

Description

Constant offset added to the linearized throttle output.

EFI_THR_COEF1 (Throttle linearisation - First Order)

DEFAULT: 1	RANGE: -1 1

Throttle linearisation - First Order (EFI_THR_COEF1)

Description

Primary linear scaling coefficient for the EFI throttle linearization polynomial.

EFI_THR_COEF2 (Throttle linearisation - Second Order)

DEFAULT: 0	RANGE: -1 1

Throttle linearisation - Second Order (EFI_THR_COEF2)

Description

Quadratic term for the EFI throttle linearization polynomial.

EFI_THR_COEF3 (Throttle linearisation - Third Order)

DEFAULT: 0	RANGE: -1 1

Throttle linearisation - Third Order (EFI_THR_COEF3)

Description

Cubic term for the EFI throttle linearization polynomial.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





EFI_TYPE (EFI communication type)

DEFAULT: 0

EFI communication type (EFI_TYPE)

Description

Enables the Electronic Fuel Injection (EFI) subsystem and selects the communication protocol used to talk to the engine's ECU.

Tuning & Behavior

- **Default Value:** 0 (None)
- **Values:** 1:Serial-MS (MegaSquirt), 2:NWPMU, 3:Serial-Lutan, 5:DroneCAN, 6:Currawong-ECU, 7:Scripting, 8:Hirth, 9:MAVLink
- **Requires reboot after changing to initialize the driver.**

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **EK2** parameter group configures the **Legacy EKF2** state estimator. For many years, this was the default navigation filter for ArduPilot. It fuses data from IMUs, GPS, Barometers, and Compasses to determine Position, Velocity, and Attitude.

Note: As of ArduPilot 4.1+, EKF3 (`EK3`) is the default. EKF2 is kept for legacy compatibility on F4 boards where EKF3 might be too heavy, or for users who trust its proven track record.

Key Concepts

1. Source Switching (`EK2_GPS_TYPE`, `EK2_ALT_SOURCE`)

EKF2 allows selecting primary sources, but it is less flexible than EKF3.

- **GPS Type:** 0=GPS 3D Vel, 1=GPS 2D Vel, etc.
- **Alt Source:** 0=Baro, 1=Rangefinder, 2=GPS, 3=Beacon.

2. Innovation Gates (`_I_GATE`)

The filter rejects sensor data that deviates too far from the current estimate.

- `EK2_POS_I_GATE`: How many standard deviations a GPS position update can "jump" before being rejected as a glitch.

3. Noise Parameters (`_M_NSE`, `_P_NSE`)

Tuning parameters representing the expected noise of the sensors.

- **Increase** if your sensors are noisy/vibrating (trusts the sensor less).
- **Decrease** if you have high-end sensors (trusts the sensor more).

Parameter Breakdown

- `EK2_ENABLE`: Master switch.
- `EK2_IMU_MASK`: Selects which IMUs to use (1=IMU1, 2=IMU2, 3=Both).
- `EK2_MAG_CAL`: EKF2 can learn magnetometer offsets in flight.

Integration Guide

- **Selection:** To use EKF2, set `AHRS_EKF_TYPE = 2` and `EK2_ENABLE = 1`.
- **Tuning:** Generally, default parameters work for 99% of vehicles. Modifying noise parameters is only for advanced users with specific vibration issues.

Developer Notes

- **Library:** `libraries/AP_NavEKF2`
- **States:** 22-state filter.

EK2_ABIAS_P_NSE (Accelerometer bias stability)

DEFAULT: 5.0E-03	RANGE: 0.00001 0.005	UNITS: m/s^3

Accelerometer bias stability (EK2_ABIAS_P_NSE)

Description

Controls how quickly the EKF adapts to changes in accelerometer bias.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





EK2_ACC_P_NSE (Accelerometer noise)

DEFAULT: 0.6	RANGE: 0.01 1.0	UNITS: m/s^2

Accelerometer noise (EK2_ACC_P_NSE)

Description

Specifies the expected high-frequency noise level in the accelerometers.

EK2_ALT_M_NSE (Altitude measurement noise)

DEFAULT: 3.0	RANGE: 0.1 10.0	UNITS: m

Altitude measurement noise (EK2_ALT_M_NSE)

Description

Specifies the expected noise level in the altitude data (typically Baro). Increasing this value reduces the weighting of altitude measurements, making the filter respond more slowly to altitude changes.

EK2_ALT_SOURCE (Primary altitude sensor source)

DEFAULT: 0	RANGE: 0 3

Primary altitude sensor source (EK2_ALT_SOURCE)

Description

Selects the primary hardware source for altitude estimation within the EKF2 algorithm.

Tuning & Behavior

- **Default Value:** 0 (Baro)
- **Values:** 0:Use Baro, 1:Use Range Finder, 2:Use GPS, 3:Use Range Beacon

EK2_BCN_DELAY (Range beacon measurement delay)

DEFAULT: 50	RANGE: 0 127	UNITS: ms

Range beacon measurement delay (EK2_BCN_DELAY)

Description

Compensates for processing delay in range-beacon based positioning systems.

EK2_BCN_I_GTE (Range beacon measurement gate size)

DEFAULT: 500	RANGE: 100 1000

Range beacon measurement gate size (EK2_BCN_I_GTE)

Description

Strictness of the range beacon consistency check.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





EK2_BCN_M_NSE (Range beacon measurement noise)

DEFAULT: 1.0	RANGE: 0.1 10.0	UNITS: m

Range beacon measurement noise (EK2_BCN_M_NSE)

Description

Specifies the expected noise level in the range beacon data.

EK2_CHECK_SCALE (GPS accuracy check scaler)

DEFAULT: 100	RANGE: 50 200	UNITS: %

GPS accuracy check scaler (EK2_CHECK_SCALE)

Description

Adjusts the global "strictness" of the pre-flight GPS accuracy checks.

EK2_EAS_I_GATE (Airspeed measurement gate size)

DEFAULT: 400	RANGE: 100 1000

Airspeed measurement gate size (EK2_EAS_I_GATE)

Description

Strictness of the airspeed sensor consistency check.

EK2_EAS_M_NSE (Equivalent airspeed measurement noise)

DEFAULT: 1.4	RANGE: 0.5 5.0	UNITS: m/s

Equivalent airspeed measurement noise (EK2_EAS_M_NSE)

Description

Specifies the expected noise level in the airspeed sensor data.

EK2_ENABLE (Enable EKF2)

DEFAULT: 0	RANGE: 0 1

Enable EKF2 (EK2_ENABLE)

Description

Global switch to enable the second-generation Extended Kalman Filter. Enabling EKF2 makes the math run in the background, but it is only used for flight control if AHRS_EKF_TYPE is set to 2.

Tuning & Behavior

- **Default Value:** 0 (Disabled)
- **Requires reboot after changing.**

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





EK2_FLOW_DELAY (Optical Flow measurement delay)

DEFAULT: 10	RANGE: 0 127	UNITS: ms

Optical Flow measurement delay (EK2_FLOW_DELAY)

Description

Compensates for the processing delay in the optical flow sensor.

EK2_FLOW_I_GATE (Optical Flow measurement gate size)

DEFAULT: 300	RANGE: 100 1000

Optical Flow measurement gate size (EK2_FLOW_I_GATE)

Description

Strictness of the optical flow consistency check.

EK2_FLOW_M_NSE (Optical flow measurement noise)

DEFAULT: 0.25	RANGE: 0.05 1.0	UNITS: rad/s

Optical flow measurement noise (EK2_FLOW_M_NSE)

Description

Specifies the expected noise level in the optical flow data.

EK2_FLOW_USE (Optical flow use bitmask)

DEFAULT: 1	RANGE: 0 2

Optical flow use bitmask (EK2_FLOW_USE)

Description

Determines how optical flow sensor data is utilized.

- **1: Navigation (Velocity estimation)**
- **2: Terrain (Height estimation)**

EK2_GBIAS_P_NSE (Rate gyro bias stability)

DEFAULT: 1.0E-04	RANGE: 0.00001 0.001	UNITS: rad/s^2

Rate gyro bias stability (EK2_GBIAS_P_NSE)

Description

Controls how quickly the EKF adapts to changes in gyro bias (drift). Higher values allow for faster adaptation but result in noisier bias estimates.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





EK2_GLITCH_RAD (GPS glitch radius gate size)

DEFAULT: 25	RANGE: 10 100	UNITS: m

GPS glitch radius gate size (EK2_GLITCH_RAD)

Description

Defines the maximum allowed distance "jump" in GPS position before the EKF flags it as a glitch.

EK2_GPS_CHECK (Arm Checks to Perform (bitmask))

DEFAULT: ARMING_CHECK_ALL	RANGE: null	UNITS: null

Arm Checks to Perform (bitmask)

Note: This parameter functions identically to EK3_GPS_CHECK for EKF2.

EK2_GPS_TYPE (GPS mode control)

DEFAULT: 0	RANGE: 0 3

GPS mode control (EK2_GPS_TYPE)

Description

Determines how the EKF incorporates GPS data (velocity and position) into its estimation.

Tuning & Behavior

- **Default Value:** 0 (GPS 3D Vel and 2D Pos)
- **Values:** 0:GPS 3D Vel and 2D Pos, 1:GPS 2D vel and 2D pos, 2:GPS 2D pos, 3:No GPS
- **Option 3 is useful for indoor flight with optical flow.**

EK2_GSCL_P_NSE (EKF2 Gyro Scale Factor Process Noise)

DEFAULT: 0.00001	RANGE: 0.000001 0.001	UNITS: Hz

EKF2 Gyro Scale Factor Process Noise (EK2_GSCL_P_NSE)

Description

`EK2_GSCL_P_NSE` determines how fast the EKF2 learns the scale factor errors of the gyroscopes.

Gyroscopes are not perfectly linear; they might report 101 degrees/sec when the drone actually rotated 100 degrees. This parameter allows the EKF to identify and correct that "Scale Error" over time.

Tuning & Behavior

- **Default Value:** 0.00001 Hz.
- **Higher Value:** Faster learning, but heading might become unstable during aggressive maneuvers.
- **Lower Value:** Slower, more stable learning.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





EK2_GSF_RST_MAX (Maximum number of resets to the EKF-GSF yaw estimate allowed)

DEFAULT: 2	RANGE: 1 10

Maximum resets to the EKF-GSF yaw estimate allowed (EK2_GSF_RST_MAX)

Description

Limits the number of times the EKF can "jump" its heading based on the GSF estimate to prevent repeated resets in difficult magnetic environments.

EK2_GSF_RUN_MASK (Bitmask of which EKF-GSF yaw estimators run)

DEFAULT: 3

Bitmask of which EKF-GSF yaw estimators run (EK2_GSF_RUN_MASK)

Description

Enables the Gaussian Sum Filter (GSF) yaw estimator for specific EKF cores. GSF provides a yaw estimate that doesn't rely on the magnetometer.

EK2_GSF_USE_MASK (Bitmask of which EKF-GSF yaw estimators are used)

DEFAULT: 3

Bitmask of which EKF-GSF yaw estimators are used (EK2_GSF_USE_MASK)

Description

Determines which EKF cores are permitted to use the GSF yaw estimate for emergency recovery or as a primary source.

EK2_GYRO_P_NSE (Rate gyro noise)

DEFAULT: 3.0E-02	RANGE: 0.0001 0.1	UNITS: rad/s

Rate gyro noise (EK2_GYRO_P_NSE)

Description

Specifies the expected high-frequency noise level in the gyroscopes. Higher values make the EKF trust the gyros less and rely more on other sensors (like compass and GPS).

EK2_HGT_DELAY (Height measurement delay)

DEFAULT: 60	RANGE: 0 250	UNITS: ms

Height measurement delay (EK2_HGT_DELAY)

Description

Compensates for the processing and transmission time delay inherent in the altitude sensor (usually the barometer).

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





EK2_HGT_I_GATE (Height measurement gate size)

DEFAULT: 500	RANGE: 100 1000

Height measurement gate size (EK2_HGT_I_GATE)

Description

Determines the strictness of the altitude check.

EK2_HRT_FILT (Height rate filter crossover frequency)

DEFAULT: 2.0	RANGE: 0.1 30.0	UNITS: Hz

Height rate filter crossover frequency (EK2_HRT_FILT)

Description

Tuning for the vertical velocity (climb/sink rate) estimation.

EK2_IMU_MASK (Bitmask of active IMUs)

DEFAULT: 3

Bitmask of active IMUs (EK2_IMU_MASK)

Description

Defines which physical IMUs (Accelerometers/Gyros) will have an associated EKF2 core instance.

EK2_MAGB_P_NSE (Body magnetic field process noise)

DEFAULT: 0.001	RANGE: 0.00001 0.01	UNITS: Gauss/s

Body magnetic field process noise (EK2_MAGB_P_NSE)

Description

Specifies how quickly the EKF adapts its estimate of the aircraft's internal magnetic interference (bias).

EK2_MAGE_P_NSE (Earth magnetic field process noise)

DEFAULT: 0.001	RANGE: 0.00001 0.01	UNITS: Gauss/s

Earth magnetic field process noise (EK2_MAGE_P_NSE)

Description

Specifies how quickly the EKF adapts its estimate of the local Earth magnetic field.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY

EK2_MAG_CAL (Magnetometer default fusion mode)

DEFAULT: 3	RANGE: 0 4

Magnetometer default fusion mode (EK2_MAG_CAL)

Description

Configures the EKF's magnetometer processing logic.

Tuning & Behavior

- **Default Value:** 3 (After first climb yaw reset) for Copter, 0 (When flying) for Plane
- **Values:** 0:When flying, 1:When manoeuvring, 2:Never, 3:After first climb yaw reset, 4:Always
- **Settings 0, 1, and 3 are standard for most aircraft.**

EK2_MAG_EF_LIM (EarthField error limit)

DEFAULT: 50	RANGE: 0 500	UNITS: mGauss

EarthField error limit (EK2_MAG_EF_LIM)

Description

Sanity check between the EKF's learned magnetic field and the mathematical World Magnetic Model (WMM).

EK2_MAG_I_GATE (Magnetometer measurement gate size)

DEFAULT: 300	RANGE: 100 1000

Magnetometer measurement gate size (EK2_MAG_I_GATE)

Description

Determines the strictness of the magnetometer check.

EK2_MAG_MASK (Bitmask of cores using heading fusion)

DEFAULT: 0

Bitmask of cores using heading fusion (EK2_MAG_MASK)

Description

Allows specific EKF cores to be forced into a simpler (and sometimes more robust) magnetic heading fusion mode.

EK2_MAG_M_NSE (Magnetometer measurement noise)

DEFAULT: 0.05	RANGE: 0.01 0.5	UNITS: Gauss

Magnetometer measurement noise (EK2_MAG_M_NSE)

Description

Specifies the expected noise level in the magnetometer data. Increasing it reduces the weighting of the compass in heading estimation.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





EK2_MAX_FLOW (Maximum valid optical flow rate)

DEFAULT: 2.5	RANGE: 1.0 4.0	UNITS: rad/s

Maximum valid optical flow rate (EK2_MAX_FLOW)

Description

Upper bound for valid optical flow readings. Readings above this rate are ignored as being potentially corrupted by motion blur or sensor limits.

EK2_NOAID_M_NSE (Non-GPS position uncertainty)

DEFAULT: 10.0	RANGE: 0.5 50.0	UNITS: m

Non-GPS position uncertainty (EK2_NOAID_M_NSE)

Description

Specifies how much "drift" is allowed when the aircraft is flying without a position source (dead-reckoning).

EK2_OGN_HGT_MASK (EKF reference height correction bitmask)

DEFAULT: 0

EKF reference height correction bitmask (EK2_OGN_HGT_MASK)

Description

Configuration for correcting barometric drift using GPS altitude data.

EK2_OPTIONS (Optional EKF behaviour)

DEFAULT: 0

Optional EKF behaviour (EK2_OPTIONS)

Description

Configuration bitmask for advanced EKF behaviors.

- **Bit 0: DisableExternalNavigation**

EK2_POSNE_M_NSE (GPS horizontal position noise)

DEFAULT: 1.0	RANGE: 0.1 10.0	UNITS: m

GPS horizontal position noise (EK2_POSNE_M_NSE)

Description

Specifies the expected noise level in the GPS horizontal position data.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





EK2_POS_I_GATE (GPS position measurement gate size)

DEFAULT: 500	RANGE: 100 1000

GPS position measurement gate size (EK2_POS_I_GATE)

Description

Determines the strictness of the GPS position check.

EK2_RNG_I_GATE (Range finder measurement gate size)

DEFAULT: 500	RANGE: 100 1000

Range finder measurement gate size (EK2_RNG_I_GATE)

Description

Strictness of the rangefinder consistency check.

EK2_RNG_M_NSE (Range finder measurement noise)

DEFAULT: 0.5	RANGE: 0.1 10.0	UNITS: m

Range finder measurement noise (EK2_RNG_M_NSE)

Description

Specifies the expected noise level in the rangefinder data.

EK2_RNG_USE_HGT (Range finder switch height percentage)

DEFAULT: -1	RANGE: -1 70	UNITS: %

Range finder switch height percentage (EK2_RNG_USE_HGT)

Description

Threshold for automatically switching to the rangefinder as the primary altitude source when flying close to the ground.

- **-1 disables this feature.**

EK2_RNG_USE_SPD (Range finder max ground speed)

DEFAULT: 2.0	RANGE: 2.0 6.0	UNITS: m/s

Range finder max ground speed (EK2_RNG_USE_SPD)

Description

Safety limit that disables rangefinder-based altitude estimation at high speeds, where tilt and terrain variations can introduce significant errors.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





EK2_TAU_OUTPUT (Output complementary filter time constant)

DEFAULT: 25	RANGE: 10 50	UNITS: cs

Output complementary filter time constant (EK2_TAU_OUTPUT)

Description

Determines the responsiveness of the final estimated output.

EK2_TERR_GRAD (Maximum terrain gradient)

DEFAULT: 0.1	RANGE: 0 0.2

Maximum terrain gradient (EK2_TERR_GRAD)

Description

Assumption about the maximum steepness of the ground, used for terrain height estimation.

EK2_VELD_M_NSE (GPS vertical velocity noise)

DEFAULT: 0.5	RANGE: 0.05 5.0	UNITS: m/s

GPS vertical velocity noise (EK2_VELD_M_NSE)

Description

Specifies the expected noise level in the GPS vertical velocity (climb/sink) data.

EK2_VELNE_M_NSE (GPS horizontal velocity noise)

DEFAULT: 0.5	RANGE: 0.05 5.0	UNITS: m/s

GPS horizontal velocity noise (EK2_VELNE_M_NSE)

Description

Specifies the expected noise level in the GPS horizontal velocity data. Increasing this value reduces the weighting of GPS velocity measurements in the EKF.

Tuning & Behavior

- **Default Value:** 0.5 m/s (Plane/Rover), 0.3 m/s (Copter)
- **Range:** 0.05 to 5.0 m/s

EK2_VEL_I_GATE (GPS velocity innovation gate size)

DEFAULT: 500	RANGE: 100 1000

GPS velocity innovation gate size (EK2_VEL_I_GATE)

Description

Determines the "strictness" of the GPS velocity check. Decreasing it makes the EKF more likely to reject GPS data that seems slightly inconsistent with other sensors. Increasing it makes the EKF more tolerant but risks accepting bad GPS data.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





EK2_WIND_PSCALE (Height rate to wind process noise scaler)

DEFAULT: 0.5	RANGE: 0.0 1.0

Height rate to wind process noise scaler (EK2_WIND_PSCALE)

Description

Accounts for the fact that wind speed and direction typically change more rapidly with altitude.

EK2_WIND_P_NSE (Wind velocity process noise)

DEFAULT: 0.1	RANGE: 0.01 1.0	UNITS: m/s^2

Wind velocity process noise (EK2_WIND_P_NSE)

Description

Controls how quickly the EKF adapts to changing wind conditions.

EK2_YAW_I_GATE (Yaw measurement gate size)

DEFAULT: 300	RANGE: 100 1000

Yaw measurement gate size (EK2_YAW_I_GATE)

Description

Determines the strictness of the heading (yaw) check.

EK2_YAW_M_NSE (Yaw measurement noise)

DEFAULT: 0.5	RANGE: 0.05 1.0	UNITS: rad

Yaw measurement noise (EK2_YAW_M_NSE)

Description

Specifies the expected noise level in the yaw (heading) data.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





EXTENDED KALMAN FILTER 3 (EK3)

Overview

The **EK3** parameter group configures the **Modern EKF3** state estimator. This is the default and recommended navigation filter for all modern ArduPilot vehicles.

EKF3 differs from EKF2 by offering **Source Switching** ("Lane Switching"). It can seamlessly transition between different navigation sources in flight (e.g., GPS outdoors → Optical Flow indoors → Beacon) based on pilot switches or signal health.

Key Concepts

1. Source Switching (`EK3_SRCn_...`)

You can define up to 3 sets of sensor configurations ("Sources").

- `EK3_SRC1_POSXY`: Horizontal Position Source (GPS, Flow, Beacon, None).
- `EK3_SRC1_VELXY`: Horizontal Velocity Source.
- `EK3_SRC1_POSZ`: Vertical Position Source (Baro, Rangefinder, GPS).
- `EK3_SRC1_YAW`: Heading Source (Compass, GPS Moving Baseline, External).

The pilot can switch between SRC1, SRC2, and SRC3 using an RC channel (`RCx_OPTION = 90`).

2. Multi-Core Affinity (`EK3_AFFINITY`)

On H7 processors, EKF3 can run multiple "Lanes" (Instances) on different IMUs simultaneously. `EK3_AFFINITY` controls which IMU is assigned to which EKF lane to maximize redundancy.

3. Visual Odometry / GPS Denied

EKF3 is the engine that makes non-GPS flight possible.

- **Optical Flow:** `EK3_SRCx_POSXY = 0`, `VELXY = 5` (Optical Flow).
- **Visual Odometry:** `EK3_SRCx_POSXY = 6` (External Nav).

Parameter Breakdown

- `EK3_ENABLE`: Master switch.
- `EK3_PRIMARY`: Which lane is currently driving the vehicle (Read-only status mostly).
- `EK3_MAG_CAL`: In-flight mag calibration options.
- `EK3_FLOW_DELAY`: Time lag for optical flow sensor.

Integration Guide

Setting up GPS + Optical Flow Backup

1. **Source 1 (GPS):** `EK3_SRC1_POSXY = 3` (GPS), `VELXY = 3` (GPS), `POSZ = 1` (Baro).
2. **Source 2 (Flow):** `EK3_SRC2_POSXY = 0` (None), `VELXY = 5` (OptFlow), `POSZ = 2` (Rangefinder).
3. **Switching:** Assign RC6 to `EKF Pos Source`.
    - Low PWM = Source 1 (GPS).
    - Mid PWM = Source 2 (Flow).

Developer Notes

- **Library:** `libraries/AP_NavEKF3`
- **States:** 24-state filter.
- **Complexity:** EKF3 is significantly more complex to tune than EKF2 but offers far greater capability.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





EK3_ABIAS_P_NSE (EKF3 Accelerometer Bias Process Noise)

DEFAULT: 0.002	RANGE: 0.0001 0.01	UNITS: m/s^3

EKF3 Accelerometer Bias Process Noise (EK3_ABIAS_P_NSE)

Description

`EK3_ABIAS_P_NSE` tells the EKF how fast it should expect the accelerometer's "zero point" (bias) to drift over time.

Accelerometers drift due to temperature changes and internal stress. The EKF constantly estimates this bias to subtract it.

- **Higher Value:** The EKF expects the bias to change rapidly. It will adapt quickly to temperature shifts but might be less stable.
- **Lower Value:** The EKF expects a very stable sensor. It will lock onto a bias value and resist changing it.

The Engineer's View

Sets the process noise covariance (Q) for the accelerometer bias states (Z axis and X/Y axes).
This essentially sets the "bandwidth" of the bias estimation loop.

Tuning & Behavior

- **Default Value:** 0.002
- **Recommendation:** Leave at default unless you are seeing "Z-Accel Bias" warnings on a high-quality IMU, in which case you might lower it slightly.

EK3_ACC_BIAS_LIM (EKF3 Accelerometer Bias Limit)

DEFAULT: 1.0	RANGE: 0.1 2.5	UNITS: m/s^2

EKF3 Accelerometer Bias Limit (EK3_ACC_BIAS_LIM)

Description

`EK3_ACC_BIAS_LIM` is a safety "Sanity Check" for your IMU.

Accelerometers are not perfect; they often have a small constant error (bias). The EKF tries to "Learn" and subtract this error during flight. However, if the IMU is failing or vibrating excessively, the learned bias might become huge and incorrect. This parameter limits how much bias the EKF is allowed to compensate for. If the limit is reached, the EKF will stop trusting that IMU.

The Mathematics

The EKF state vector includes a bias term b_acc. This parameter enforces a hard constraint:

$$|b_acc| \leq \text{EK3_ACC_BIAS_LIM}$$

If the filter's internal estimate exceeds this value, it indicates a hardware issue or a significant environmental disturbance.

Tuning & Behavior

- **Default Value:** 1.0 m/s^2.
- **High Vibration:** On very noisy gas-powered helicopters, you might need to increase this to **1.5** to prevent premature EKF lane switching.
- **Health Warning:** If your logs show the bias consistently hitting this limit, your flight controller likely has a physical mounting issue or a dying sensor.

EK3_ACC_P_NSE (Accelerometer noise)

DEFAULT: 0.35	RANGE: 0.01 1.0	UNITS: m/s^2

Accelerometer noise (EK3_ACC_P_NSE)

Description

Specifies the expected noise level in the accelerometers for EKF3.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





EK3_AFFINITY (EKF3 Sensor Affinity Options)

DEFAULT: 0

EKF3 Sensor Affinity Options (EK3_AFFINITY)

Description

Advanced configuration to "lock" specific sensors (e.g., GPS2 or Baro3) to specific EKF cores. This allows the system to evaluate independent combinations of hardware sensors.

Tuning & Behavior

- **Default Value:** 0
- **Bitmask:**
  - Bit 0: EnableGPSAffinity
  - Bit 1: EnableBaroAffinity
  - Bit 2: EnableCompassAffinity
  - Bit 3: EnableAirspeedAffinity

EK3_ALT_M_NSE (Altitude measurement noise)

DEFAULT: 2.0	RANGE: 0.1 100.0	UNITS: m

Altitude measurement noise (EK3_ALT_M_NSE)

Description

Specifies the expected noise level in the altitude data (Baro/GPS).

EK3_BCN_DELAY (EKF3 Beacon Delay)

DEFAULT: 50	RANGE: 0 500	UNITS: ms

EKF3 Beacon Delay (EK3_BCN_DELAY)

Description

`EK3_BCN_DELAY` compensates for the time it takes for a Range Beacon (like Pozyx) or Optical Flow sensor to process data and send it to the flight controller.

The EKF uses a "Time Horizon" to rewind history and fuse this delayed data with the correct IMU timestamp.

Tuning & Behavior

- **Default Value:** 50 ms.
- **Recommendation:** Check your beacon system's datasheet. Incorrect delay causes "toilet bowling" (spiraling) when stopping or changing direction.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





EK3_BCN_I_GTE (EKF3 Beacon Innovation Gate)

DEFAULT: 500	RANGE: 100 1000	UNITS: %

EKF3 Beacon Innovation Gate (EK3_BCN_I_GTE)

Description

`EK3_BCN_I_GTE` controls the strictness of the "Glitch Filter" for indoor positioning beacons (like Pozyx or Marvelmind).

When the EKF receives a distance measurement from a beacon, it compares it to where it *thinks* the drone is. If the difference (Innovation) is too large, the EKF assumes the measurement is a "multipath" reflection or error and rejects it.

- **Low Value (e.g. 300):** Strict. Rejects noisy data easily. Good for high-precision flight in clean environments.
- **High Value (e.g. 800):** Relaxed. Accepts jumpy data. Use if your beacons are noisy but you trust them more than the IMU.

The Mathematics

The gate check uses the Normalized Innovation Squared (NIS):

$$\frac{(z_meas - z_pred)^2}{S} < \left(\frac{\text{GATE}}{100}\right)^2$$

Where:

- z is the range measurement.
- S is the innovation covariance (expected uncertainty).
- **500** represents 5 Standard Deviations (σ).

The Engineer's View

Defined in `AP_NavEKF3`.
This parameter only applies when `EK3_SRC1_POSXY` is set to **Beacon (4)**.
Unlike GPS gating (which has separate velocity and position gates), beacon gating is applied to the raw range measurement from each anchor.

Tuning & Behavior

- **Default Value:** 500 (5-Sigma).
- **Recommendation:**
  - **Standard Indoor:** Keep at **500**.
  - **Metal Building (Reflections):** Decrease to **300** to reject multipath spikes.
  - **Fast Flight:** Increase to **800** if the drone moves faster than the beacons can update, causing false rejections due to lag.

EK3_BCN_M_NSE (EKF3 Beacon Measurement Noise)

DEFAULT: 1.0	RANGE: 0.1 10.0	UNITS: m

EKF3 Beacon Measurement Noise (EK3_BCN_M_NSE)

Description

`EK3_BCN_M_NSE` tells the EKF how much to "Trust" your indoor positioning beacons (like Pozyx or Marvelmind).

If your beacons have a lot of multi-path interference (echoes off walls), you should increase this value to prevent the drone from twitching. If they are very precise, you can decrease it to get a tighter position hold.

Tuning & Behavior

- **Default Value:** 1.0 m.
- **High Precision:** Set to **0.5** if your beacon system consistently reports accuracy better than 10cm.
- **Noisy Environment:** Set to **2.0** if you see the drone jumping around in tight indoor spaces.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY

EK3_BETA_MASK (Bitmask controlling sideslip angle fusion)

DEFAULT: 0

Bitmask controlling sideslip angle fusion (EK3_BETA_MASK)

Description

Enables the "synthetic sideslip" assumption, which assumes that fixed-wing aircraft generally fly with zero sideslip. This allows the EKF to better estimate heading and wind even without a compass or during sensor failures.

Tuning & Behavior

- **Default Value:** 0
- **Bitmask:**
  - Bit 0: Always (Force fusion even if a yaw sensor is available)
  - Bit 1: WhenNoYawSensor (Only use if magnetometer/GPS-Yaw fails)

EK3_CHECK_SCALE (EKF3 Consistency Check Scale)

DEFAULT: 100	RANGE: 50 200	UNITS: %

EKF3 Consistency Check Scale (EK3_CHECK_SCALE)

Description

`EK3_CHECK_SCALE` is the "Sensitivity Knob" for the EKF failsafe system.

It scales the thresholds used to determine if the IMUs, Compass, and GPS are healthy. If you have a very high-vibration airframe that constantly triggers false "EKF Variance" warnings, you can increase this to make the autopilot more "Relaxed."

Tuning & Behavior

- **Default Value:** 100 (100%).
- **Increased (e.g., 150):** Autopilot becomes more tolerant of noise and small sensor errors. Safer for noisy airframes but slower to detect actual sensor failures.
- **Decreased (e.g., 80):** Autopilot becomes "Paranoid." It will detect failures faster but is more likely to trigger false failsafes in gusty wind.

EK3_DRAG_BCOEF_X (Ballistic coefficient for X axis drag)

DEFAULT: 0.0	RANGE: 0.0 1000.0	UNITS: kg/m^2

Ballistic coefficient for X axis drag (EK3_DRAG_BCOEF_X)

Description

A physical parameter representing the aerodynamic drag of the aircraft's body in the forward/backward direction. When set, it allows the EKF to estimate wind velocity based on the tilt required to maintain position.

- **Set to > 0 to enable wind estimation for multirotors.**

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





EK3_DRAG_BCOEF_Y (EKF3 Lateral Drag Coefficient)

DEFAULT: 0.0	RANGE: 0.0 1000.0	UNITS: kg/m²

EKF3 Lateral Drag Coefficient (EK3_DRAG_BCOEF_Y)

Description

`EK3_DRAG_BCOEF_Y` is the "Side-Wind Profile" for your drone. It tells the flight controller how much the wind pushes on the *side* of the drone when it is flying sideways or in a crosswind.

Most drones have a different profile from the side than they do from the front. For example, a drone with a wide battery or long arms will catch more wind from the side. `EK3_DRAG_BCOEF_Y` allows the EKF3 to accurately calculate the lateral (sideways) component of the wind vector.

- **Set to 0 (Default):** Sideways wind estimation is disabled.
- **Set > 1.0:** Enables lateral wind estimation.

The Mathematics

The EKF uses the quadratic drag model for the lateral axis (y):

$$a_y = \frac{1}{\text{BCOEF_Y}} \cdot V_rel_y \cdot |V_rel_y|$$

Where:

- $V_rel_y = V_ground_y - V_wind_y$.
- BCOEF_Y is the value of `EK3_DRAG_BCOEF_Y`.

Note on Symmetry: On a perfectly square quadcopter, `BCOEF_Y` should be identical to `BCOEF_X`. On rectangular drones (longitudinal frames), `BCOEF_Y` is usually **lower** than `BCOEF_X` because the side of the drone has more surface area (more drag).

The Engineer's View

This parameter is the `_ballisticCoef_y` member in `NavEKF3`.

To calculate it manually:

1. Fly the drone sideways at a steady 5-10 m/s on a calm day.
2. Record the average roll angle (ϕ).
3. Calculate lateral acceleration from tilt: $a_tilt = g \cdot \tan(\phi)$.
4. $\text{BCOEF_Y} = \frac{V^2}{a_tilt}$.

Tuning & Behavior

- **Default Value:** 0.0 (Disabled)
- **Range:** 0.0 - 1000.0 kg/m²
- **Effect of Increasing:** The EKF assumes the side of the drone is "slippery." It will calculate a higher side-wind speed for a given roll angle.
- **Effect of Decreasing:** The EKF assumes the side of the drone is "blunt." It will calculate a lower side-wind speed for a given roll angle.

Use Case Recommendations

- **Dead-Cat / Long-Range Frames: Set to 75% of `BCOEF_X`.** These frames have very long side profiles and catch significant crosswinds.
- **Standard Quad: Set to match `BCOEF_X`.**
- **Drones with Large Side Panels (Advertising/Industrial): Decrease to 10.0.** Large surface areas create massive lateral drag.

Troubleshooting

- **Scenario:** During a crosswind, the drone's position drifts sideways, and the GCS doesn't show any lateral wind.
  - **Diagnosis:** `EK3_DRAG_BCOEF_Y` is set to 0.
  - **Fix:** Set `EK3_DRAG_BCOEF_Y` to match your `BCOEF_X` value.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY

EK3_DRAG_MCOEF (Momentum coefficient for propeller drag)

DEFAULT: 0.0	RANGE: 0.0 1.0	UNITS: 1/s

Momentum coefficient for propeller drag (EK3_DRAG_MCOEF)

Description

Accounts for the "rotor drag" effect where air velocity normal to the rotor disc is lost, creating a force. This is distinct from body drag and is essential for accurate wind estimation on multicopters.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





EK3_DRAG_M_NSE (EKF3 Drag Observation Noise)

DEFAULT: 0.5	RANGE: 0.1 2.0	UNITS: m/s^2

EKF3 Drag Observation Noise (EK3_DRAG_M_NSE)

Description

`EK3_DRAG_M_NSE` is the key to ArduPilot's **Internal Wind Vane**.

When a multirotor flies forward (or hovers in a headwind), it must lean. The drone's "face" (frontal area) catches the wind, creating drag. The EKF3 can reverse-calculate the wind speed by looking at how far the drone has to lean to stay stationary. This is called **Drag Fusion**.

`EK3_DRAG_M_NSE` defines how much the EKF trusts its "Calculated Drag" vs. other sensors.

- **Low Value:** The EKF trusts its drag model implicitly. It will calculate wind speed very quickly and aggressively.
- **High Value:** The EKF is cautious. It assumes the drag calculation is noisy (e.g., due to gusts or frame vibrations) and will take longer to update its wind estimate.

The Mathematics

The EKF uses a simplified drag equation to predict the acceleration (a_drag) caused by the relative wind (V_rel):

$$a_drag = \text{BCOEF} \cdot |V_rel|^2 + \text{MCOEF} \cdot V_rel$$

`EK3_DRAG_M_NSE` represents the standard deviation (σ) of this observation. In the filter, it is squared to form the measurement covariance R:

$$R_drag = (\text{EK3_DRAG_M_NSE})^2$$

By minimizing the error between this predicted drag and the actual IMU-measured acceleration, the EKF can "solve" for the unknown wind vector.

The Engineer's View

This parameter is the `_dragObsNoise` member in `NavEKF3`.

It is inactive unless you have provided your drone's specific drag coefficients (`EK3_DRAG_BCOEF_X/Y`). Once enabled, the EKF3 core fuses the drag data every 100ms. This provides a "synthetic airspeed" estimate, which is incredibly useful for:

1. Maintaining ground speed during mapping.
2. Switching to "Dead Reckoning" more accurately if GPS is lost.

```
// AP_NavEKF3_Control.cpp
// Wind estimation logic incorporates _dragObsNoise to weight the accelerometer-derived drag observations.
```

Tuning & Behavior

- **Default Value:** 0.5 m/s²
- **Range:** 0.1 - 2.0 m/s²
- **Effect of Increasing:** Smoother, slower wind estimates. Less susceptible to vibration noise.
- **Effect of Decreasing:** Rapid, real-time wind tracking. Better for drones operating in highly gusty environments.

Use Case Recommendations

- **Professional Mapping Drone: Keep at 0.5.** Provides a reliable average wind speed for georeferencing.
- **Precision Spraying / Ag-Drones: Decrease to 0.3.** These drones must react instantly to wind shifts to prevent chemical drift; a faster wind estimate improves the ground-track accuracy.
- **Racing / High Vibration frames: Increase to 1.0.** High vibrations can "confuse" the drag model; a higher noise value prevents the EKF from making erratic wind corrections.

Troubleshooting

- **Scenario:** GCS reports wildly varying wind speeds (e.g. jumping from 5 to 20 knots) while hovering in a steady breeze.
  - **Diagnosis:** `EK3_DRAG_M_NSE` is too low, or your `BCOEF` parameters are incorrect.
  - **Fix:** Increase `EK3_DRAG_M_NSE` to 0.8.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





EK3_EAS_I_GATE (EKF3 Airspeed Innovation Gate)

DEFAULT: 400	RANGE: 100 1000	UNITS: %

EKF3 Airspeed Innovation Gate (EK3_EAS_I_GATE)

Description

`EK3_EAS_I_GATE` prevents the EKF from trusting bad airspeed data.

Pitot tubes can get clogged with rain or bugs. If the airspeed sensor reports a value that disagrees wildly with the GPS and IMU (e.g., claiming 100m/s when the GPS says 0), this gate rejects the reading to prevent the plane from stalling or diving.

Tuning & Behavior

- **Default Value:** 400 (4 Sigma).
- **Recommendation:** Set to **300** for better protection against bad sensors.

EK3_EAS_M_NSE (EKF3 Airspeed Measurement Noise)

DEFAULT: 1.4	RANGE: 0.5 5.0	UNITS: m/s

EKF3 Airspeed Measurement Noise (EK3_EAS_M_NSE)

Description

`EK3_EAS_M_NSE` defines how "Clean" the airspeed signal is.

- **1.4 (Default):** Standard for analog pitot tubes.
- **Lower Value:** Trusts the airspeed sensor more. Use for high-quality digital sensors (MS4525).
- **Higher Value:** Trusts the IMU/GPS more. Use for noisy sensors or turbulent conditions.

Tuning & Behavior

- **Default Value:** 1.4 m/s.

EK3_ENABLE (Enable EKF3)

DEFAULT: 1	RANGE: 0 1

Enable EKF3 (EK3_ENABLE)

Description

Global switch to enable the third-generation Extended Kalman Filter. ArduPilot defaults to EKF3 for most modern hardware. Enabling it makes the math run; set AHRS_EKF_TYPE to 3 to use it for flight control.

Tuning & Behavior

- **Default Value:** 1 (Enabled)
- **Requires reboot after changing.**

EK3_ERR_THRESH (EKF3 Lane Relative Error Sensitivity Threshold)

DEFAULT: 0.2	RANGE: 0.05 1

EKF3 Lane Relative Error Sensitivity Threshold (EK3_ERR_THRESH)

Description

Tuning for the multi-lane EKF logic. If a standby EKF lane is performing better than the primary by more than this threshold, the flight controller may switch to the healthier lane.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





EK3_FLOW_DELAY (EKF3 Optical Flow Delay)

DEFAULT: 10	RANGE: 0 500	UNITS: ms

EKF3 Optical Flow Delay (EK3_FLOW_DELAY)

Description

`EK3_FLOW_DELAY` accounts for the processing lag in your Optical Flow camera (e.g., HereFlow, CX-OF).

Flow sensors calculate motion by comparing image frames, which takes time. If this delay isn't accounted for, the EKF will think the motion happened *now* instead of *10ms ago*, causing instability.

Tuning & Behavior

- **Default Value:** 10 ms.
- **Recommendation:**
  - **HereFlow:** 10-20 ms.
  - **PX4Flow:** 10 ms.

EK3_FLOW_I_GATE (EKF3 Flow Innovation Gate)

DEFAULT: 300	RANGE: 100 1000	UNITS: %

EKF3 Flow Innovation Gate (EK3_FLOW_I_GATE)

Description

`EK3_FLOW_I_GATE` prevents the drone from chasing shadows.

Optical flow sensors can get confused by moving objects or sudden light changes. If the flow sensor reports a movement that is physically impossible given the drone's IMU data, this gate rejects it.

Tuning & Behavior

- **Default Value:** 300 (3 Sigma).
- **Recommendation:** Set to **500** if you fly over high-contrast terrain (like grass/concrete boundaries) that triggers false rejections.

EK3_FLOW_MAX (EKF3 Optical Flow Max Rate)

DEFAULT: 2.5	RANGE: 0.5 5.0	UNITS: rad/s

EKF3 Optical Flow Max Rate (EK3_FLOW_MAX)

Description

`EK3_FLOW_MAX` filters out bad data from the flow sensor.

Optical flow sensors often glitch when the image moves too fast (motion blur). This parameter tells the EKF to ignore any readings where the flow rate exceeds this limit.

Tuning & Behavior

- **Default Value:** 2.5 rad/s.
- **Recommendation:** Match this to the physical limits of your flow sensor (e.g. CX-OF or HereFlow).

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





EK3_FLOW_M_NSE (EKF3 Optical Flow Measurement Noise)

DEFAULT: 0.15	RANGE: 0.05 0.5	UNITS: rad/s

EKF3 Optical Flow Measurement Noise (EK3_FLOW_M_NSE)

Description

`EK3_FLOW_M_NSE` tells the EKF how "Trustworthy" the flow sensor is.

- **Low Value:** Trust the flow sensor more. Tighter position hold, but risk of "Toilet Bowling" if the lens is dirty.
- **High Value:** Trust the IMU more. Smoother flight, but more drift.

Tuning & Behavior

- **Default Value:** 0.15 rad/s.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





EK3_FLOW_USE (EKF3 Optical Flow Usage)

DEFAULT: 0	RANGE: 0 2

EKF3 Optical Flow Usage (EK3_FLOW_USE)

Description

`EK3_FLOW_USE` tells the flight controller what to do with the "eyes" of the drone—the Optical Flow sensor.

Optical Flow sensors (like the PX4Flow or HereFlow) look at the ground and track how fast the pixels move. This data can be used to stabilize the drone's position, but only if the EKF knows how to interpret it. This parameter enables the fusion of that visual data into the drone's navigation math.

- **0 (None):** The drone ignores the flow sensor.
- **1 (Only for Terrain):** The drone uses the flow data specifically to help estimate the distance to the ground (useful for landing on varying textures).
- **2 (For Navigation):** The drone uses the flow data as its primary source of speed (velocity). This is the setting used for **GPS-Denied flight** (indoor flight).

The Mathematics

When set to **2 (Navigation)**, the EKF uses the flow rate (ω_flow) and height (h) to calculate ground velocity (V_g):

$$V_g = \text{Filter}(\omega_flow - \omega_gyro) \cdot h$$

Where:

- ω_gyro is the drone's own rotation (subtracted to ensure the drone doesn't think it is "moving" just because it tilted).
- h is the altitude from a Rangefinder.

Constraint: Optical Flow **requires** a functioning Rangefinder. If `EK3_SRC1_POSZ` is not set to 2 (Rangefinder) or if the rangefinder fails, Optical Flow navigation will stop working.

The Engineer's View

This parameter is the `_flowUse` member in `NavEKF3`.

It is a prerequisite for fusing flow data in `AP_NavEKF3_OptFlowFusion.cpp`:

```
// AP_NavEKF3_OptFlowFusion.cpp
const bool fuse_optflow = (frontend->_flowUse == FLOW_USE_NAV) && frontend->sources.useVelXYSource(AP_NavEKF_Source::SourceXY::OPTFL
```

Note that in modern ArduPilot (4.1+), you must also set the source selection in `EK3_SRCn_VELXY` to `OpticalFlow` for this parameter to have full effect.

Tuning & Behavior

- **Default Value:** 0 (Disabled)
- **Range:** 0 - 2
- **Effect of Enabling (2):** The drone will be able to hover in one spot indoors without GPS.
- **Effect of Disabling:** The drone will drift indoors and require manual correction by the pilot.

Use Case Recommendations

- **Indoor Warehouse Robotics: Set to 2.** Allows the drone to navigate autonomous paths between shelves using only floor texture.
- **Outdoor GPS Fallback: Set to 2.** If GPS is lost near a building, the drone can "fail over" to Optical Flow to stay stationary rather than drifting into the building.
- **Precision Landing: Set to 1.** Helps the EKF refine its local height estimate relative to the landing pad texture.

Troubleshooting

- **Scenario:** Drone starts "orbiting" (circles) as soon as I switch to Loiter indoors with Optical Flow.
  - **Diagnosis:** Gyro-compensation mismatch. The EKF isn't correctly subtracting the drone's tilt from the flow data.
  - **Fix:** Ensure `FLOW_ORIENT_YAW` is correct and perform the "Flow Calibration" flight.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





EK3_GBIAS_P_NSE (EKF3 Gyroscope Bias Process Noise)

DEFAULT: 0.0001	RANGE: 0.00001 0.001	UNITS: rad/s^2

EKF3 Gyroscope Bias Process Noise (EK3_GBIAS_P_NSE)

Description

`EK3_GBIAS_P_NSE` controls how aggressively the EKF estimates the gyroscope's drift.

Gyro drift is the slow rotation reported by the sensor even when the drone is still.

The Engineer's View

Sets the process noise for the 3-axis gyro bias states.

$$\dot{b}_g = \mathcal{N}(0, \text{EK3_GBIAS_P_NSE})$$

Tuning & Behavior

- **Default Value:** 1E-04 (0.0001)
- **Recommendation:** Leave at default. If using very cheap gyros that drift wildly with temperature, increase this.

EK3_GLITCH_RAD (GPS glitch radius gate size)

DEFAULT: 25	RANGE: 10 100	UNITS: m

GPS glitch radius gate size (EK3_GLITCH_RAD)

Description

Defines the maximum allowed discrepancy between the predicted and measured GPS position.

EK3_GND_EFF_DZ (Baro height ground effect dead zone)

DEFAULT: 4.0	RANGE: 0.0 10.0	UNITS: m

Baro height ground effect dead zone (EK3_GND_EFF_DZ)

Description

Compensates for the localized high-pressure zone created under a vehicle with lift rotors during takeoff and landing. This "ground effect" can cause barometric altitude to report negative values or spikes; the dead zone ignores these fluctuations.

EK3_GPS_CHECK (GPS preflight check)

DEFAULT: 31

GPS preflight check (EK3_GPS_CHECK)

Description

Bitmask selecting which specific GPS quality metrics (sat count, HDOP, velocity consistency) are checked before allowing the EKF to use GPS data or allowing the vehicle to arm.

Tuning & Behavior

- **Default Value:** 31 (Standard rigorous checks)
- **Bitmask Examples:**
  - Bit 0: sat count
  - Bit 1: HDOP
  - Bit 2: speed error

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY

EK3_GPS_VACC_MAX (GPS vertical accuracy threshold)

DEFAULT: 0.0	RANGE: 0.0 10.0	UNITS: m

GPS vertical accuracy threshold (EK3_GPS_VACC_MAX)

Description

Safety threshold for using GPS for altitude. If the GPS reported vertical accuracy is worse than this value, the EKF will automatically fall back to the barometer.

- **0 deactivates the threshold check.**

EK3_GSF_RST_MAX (EKF3 GSF Reset Count Max)

DEFAULT: 2	RANGE: 1 10

EKF3 GSF Reset Count Max (EK3_GSF_RST_MAX)

Description

`EK3_GSF_RST_MAX` is a safety limit for the Emergency Yaw Recovery system.

The Gaussian Sum Filter (GSF) is a backup system that tries to figure out the drone's heading using GPS movement if the compass fails. If the main EKF detects a "Compass Variance" (Heading Error), it can perform an emergency reset to align with the GSF's estimate. This parameter limits how many times it can do this per flight.

- **Why Limit It?** If the GSF keeps triggering resets, something is fundamentally wrong (e.g. GPS glitching). Infinite resets could cause the drone to spin uncontrollably. Limiting it forces the drone to eventually fail-safe (Land) rather than constantly jumping its heading.

The Engineer's View

Defined in `AP_NavEKF3` as `_gsfResetMaxCount`.
This counter is incremented every time `resetYawToGSF()` is successfully called. Once the limit is reached, the EKF will refuse further corrections from the GSF, likely leading to a "EKF Compass Variance" failsafe if the primary compass is still bad.

Tuning & Behavior

- **Default Value:** 2.
- **Recommendation:** Leave at **2**. If your drone needs to reset its heading more than twice in one flight, you should not be flying.
- **Debug:** Set to **10** only if you are testing the GSF algorithm itself and want to force multiple resets.

EK3_GSF_RUN_MASK (Bitmask of which EKF-GSF yaw estimators run)

DEFAULT: 3

Bitmask of which EKF-GSF yaw estimators run (EK3_GSF_RUN_MASK)

Description

Enables the Gaussian Sum Filter (GSF) yaw estimator for specific EKF3 cores. GSF provides a yaw estimate derived from IMU and GPS velocity data, serving as a backup if the magnetometer is compromised.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





EK3_GSF_USE_MASK (EKF3 GSF Use Mask)

DEFAULT: 3	RANGE: 0 7

EKF3 GSF Use Mask (EK3_GSF_USE_MASK)

Description

EK3_GSF_USE_MASK selects the inputs for the emergency yaw estimator.

- **Bit 0 (1):** Use GPS. (Standard).
- **Bit 1 (2):** Use Airspeed. (For Planes).

Tuning & Behavior

- **Default Value:** 3 (GPS + Airspeed).
- **Recommendation:** Leave at default.

EK3_GYRO_P_NSE (Rate gyro noise)

DEFAULT: 0.015	RANGE: 0.0001 0.1	UNITS: rad/s

Rate gyro noise (EK3_GYRO_P_NSE)

Description

Specifies the expected noise level in the gyroscopes for EKF3.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





EK3_HGT_DELAY (EKF3 Height Measurement Delay)

DEFAULT: 60	RANGE: 0 250	UNITS: ms

EKF3 Height Measurement Delay (EK3_HGT_DELAY)

Description

`EK3_HGT_DELAY` accounts for the "speed of sound" (and calculation) in your drone's height sensors.

Accelerometers are instantaneous, but Barometers and Rangefinders take time to process their data. By the time the Barometer tells the EKF "I am at 10 meters," that measurement is actually 60 milliseconds old—the drone has already moved. If the EKF fuses that "old" data with "new" IMU data, the math becomes inconsistent, leading to vertical oscillations.

`EK3_HGT_DELAY` allows the EKF to reach back into its "memory buffer" and compare the Baro data against exactly what the drone was doing 60ms ago. This alignment is critical for a rock-solid, jitter-free altitude hold.

- **Default (60ms):** Optimized for standard MS5611 barometers.
- **Too Low:** The drone may oscillate vertically at high frequency (vibrate).
- **Too High:** The drone will be slow to respond to altitude changes and may "bounce" when stopping a climb.

The Mathematics

The EKF maintains a history of its internal states. When a height measurement (z_alt) arrives at time t, the EKF doesn't compare it to the current state $\hat{x}(t)$. Instead, it looks up the state from $t - \text{DELAY}$:

$$y_alt = z_alt(t) - H \cdot \hat{x}(t - \text{EK3_HGT_DELAY})$$

Where y_alt is the Innovation (error) used to correct the filter. This ensures that the sensor data is always fused with the IMU state that produced it.

The Engineer's View

This parameter is the `_hgtDelay_ms` member in `NavEKF3`.

It is particularly important when using **External Rangefinders** over I2C or Serial. These sensors have their own internal filtering and communication latency. If your Lidar has a 100ms latency, but `EK3_HGT_DELAY` is only 60ms, the EKF will experience vertical "ringing" because its corrections are based on incorrect time-alignments.

Tuning & Behavior

- **Default Value:** 60 ms
- **Range:** 0 - 250 ms
- **Effect of Increasing:** The drone trusts older data. Better for sensors with heavy internal filtering (slow Lidars).
- **Effect of Decreasing:** The drone expects faster data. Better for high-speed SPI barometers.

Use Case Recommendations

- **Standard Pixhawk / Cube: Keep at 60.** The internal Baro drivers are well-matched to this value.
- **Cheap Chinese Lidars (VL53L1X, etc.): Increase to 100.** These sensors have significant communication lag over I2C.
- **High-Speed SPI Baro (BMP388/SPL06): Decrease to 20.** Faster communication allows for a tighter altitude control loop.

Troubleshooting

- **Scenario:** Drone bobs up and down rhythmically (about 2-3 times per second) specifically when using a new Rangefinder.
  - **Diagnosis:** Height sensor delay mismatch. The EKF is comparing current height to past movements incorrectly.
  - **Fix:** Increase `EK3_HGT_DELAY` by 20ms increments until the bobbing stops.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





EK3_HGT_I_GATE (EKF3 Height Innovation Gate)

DEFAULT: 500	RANGE: 100 1000	UNITS: %

EKF3 Height Innovation Gate (EK3_HGT_I_GATE)

Description

`EK3_HGT_I_GATE` is the "Bouncer" for your altitude data.

The EKF predicts where the drone should be based on its motors and accelerometers. When the barometer (or GPS) provides a new height reading, the EKF calculates the difference (the "Innovation"). If the reading is too different from the prediction (e.g., a massive sudden jump), the "Gate" rejects it to prevent the drone from rocketing up or down due to a sensor glitch.

The Mathematics

The innovation y and its covariance S are used to calculate the normalized innovation squared (NIS):

$$extNIS = y^T S^{-1} y$$

The measurement is rejected if:

$$extNIS > \left(\frac{\text{EK3_HGT_I_GATE}}{100}\right)^2$$

A value of 500 means 5 standard deviations.

Tuning & Behavior

- **Default Value:** 500 (5 Sigma).
- **Glitchy Baro:** If you see "EKF primary changed: 0" frequently when flying near ground effect or in wind, you might need to **Increase** this value to 700 or 800.
- **Safety:** Setting this too high makes the EKF more vulnerable to slow sensor drift.

EK3_HRT_FILT (EKF3 Height Rate Filter Frequency)

DEFAULT: 2.0	RANGE: 0.1 30.0	UNITS: Hz

EKF3 Height Rate Filter Frequency (EK3_HRT_FILT)

Description

`EK3_HRT_FILT` tunes the vertical velocity estimator.

It combines the accelerometer data (fast, noisy) with the barometer/GPS data (slow, smooth). This frequency determines the "crossover point."

Tuning & Behavior

- **Default Value:** 2.0 Hz.
- **Higher Value:** More responsive to rapid climbs, but noisier.
- **Lower Value:** Smoother velocity estimate, but slight lag.

EK3_IMU_MASK (Bitmask of active IMUs)

DEFAULT: 3

Bitmask of active IMUs (EK3_IMU_MASK)

Description

Defines which physical IMUs will have an associated EKF3 core instance. Running multiple cores provides redundancy; the flight controller will automatically switch to the healthiest core.

Tuning & Behavior

- **Default Value:** 3 (IMU1 and IMU2)
- **Bitmask Examples:** 1:IMU1 only, 3:IMU1 & IMU2, 7:IMU1, IMU2 & IMU3.
- **Requires reboot to take effect.**

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





EK3_LOG_LEVEL (Logging Level)

DEFAULT: 0	RANGE: 0 3

Logging Level (EK3_LOG_LEVEL)

Description

Controls the verbosity of EKF3 data written to the log.

- **0: Full logging (standard)**
- **3: Disables EKF3 streaming logs entirely (to save CPU/bandwidth)**

EK3_MAGB_P_NSE (Body magnetic field process noise)

DEFAULT: 0.001	RANGE: 0.00001 0.01	UNITS: Gauss/s

Body magnetic field process noise (EK3_MAGB_P_NSE)

Description

Specifies how quickly EKF3 adapts its estimate of internal magnetic bias.

EK3_MAGE_P_NSE (Earth magnetic field process noise)

DEFAULT: 0.001	RANGE: 0.00001 0.01	UNITS: Gauss/s

Earth magnetic field process noise (EK3_MAGE_P_NSE)

Description

Specifies how quickly EKF3 adapts its estimate of the local Earth magnetic field.

EK3_MAG_CAL (Magnetometer default fusion mode)

DEFAULT: 3	RANGE: 0 4

Magnetometer default fusion mode (EK3_MAG_CAL)

Description

Configures the EKF3 magnetometer processing logic. Identical in behavior to EK2_MAG_CAL.

Tuning & Behavior

- **Default Value:** 3 (After first climb yaw reset) for Copter, 0 (When flying) for Plane
- **Values:** 0:When flying, 1:When manoeuvring, 2:Never, 3:After first climb yaw reset, 4:Always

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





EK3_MAG_EF_LIM (EKF3 Magnetic Earth Field Limit)

DEFAULT: 50	RANGE: 0 500	UNITS: mGauss

EKF3 Magnetic Earth Field Limit (EK3_MAG_EF_LIM)

Description

`EK3_MAG_EF_LIM` acts as a sanity check for the compass.

The autopilot knows roughly what the earth's magnetic field *should* look like at your location (based on the WMM table). If the EKF tries to learn a magnetic offset larger than this limit, it assumes the compass is being jammed by a localized magnetic anomaly (like a power line) and rejects the update.

Tuning & Behavior

- **Default Value:** 50 mGauss.
- **Recommendation:** Increase to **100** if you fly in areas with strong magnetic variations (e.g. volcanic soil).

EK3_MAG_I_GATE (Compass Innovation Gate)

DEFAULT: 300	RANGE: 100 1000

Compass Innovation Gate (EK3_MAG_I_GATE)

Description

`EK3_MAG_I_GATE` is the "BS Filter" for the compass.

The EKF is constantly predicting what the magnetic field *should* be based on its current heading. When it gets a new measurement from the physical compass, it calculates the difference (the "Innovation"). If that difference is too large, the EKF assumes the compass has gone haywire (maybe it's near a magnet) and **rejects the data** to prevent a crash.

- **Low Value (e.g., 100):** Very strict. If the compass signal is even slightly noisy, the EKF will ignore it. This is safe but can lead to "Compass Unhealthy" warnings.
- **High Value (e.g., 500):** Very relaxed. The EKF will trust almost anything the compass says. Dangerous if the compass is actually being distorted by interference.

The Mathematics

The gate uses a **Statistical Distance** (Mahalanobis Distance) check.
The innovation $y = z - H\hat{x}$ is checked against the innovation covariance S:

$$y^T S^{-1} y \leq \text{GATE}^2$$

If the error squared exceeds the gate threshold, the measurement is discarded.

The Engineer's View

Used in the measurement update loop of `AP_NavEKF3`.
A value of **300** corresponds to a 3-sigma gate (99.7% confidence interval). Measurements outside this are statistically likely to be errors rather than valid movement.

Tuning & Behavior

- **Default Value:** 300
- **Recommendation:** Keep at **300**.
- **Troubleshooting:** If you get constant "EKF Compass Innovation" errors but you are sure your compass is well-calibrated and away from metal, you can increase this to **500** to stop the warnings. However, this is usually masking a physical hardware issue (vibration or interference).

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





EK3_MAG_MASK (EKF3 Magnetometer Mask)

DEFAULT: 0	RANGE: 0 255

EKF3 Magnetometer Mask (EK3_MAG_MASK)

Description

`EK3_MAG_MASK` selects which compasses the EKF is allowed to listen to.

By default (0), the EKF uses the first healthy compass it finds. This mask forces it to use specific sensors.

- **Bit 0:** Compass 1.
- **Bit 1:** Compass 2.
- **Bit 2:** Compass 3.

Tuning & Behavior

- **Default Value:** 0 (All).
- **Recommendation:** If you have an external GPS/Compass and an internal noisy compass, set this to **1** (Bit 0) to force the EKF to only use the external one.

EK3_MAG_M_NSE (Magnetometer measurement noise)

DEFAULT: 0.05	RANGE: 0.01 0.5	UNITS: Gauss

Magnetometer measurement noise (EK3_MAG_M_NSE)

Description

Specifies the expected noise level in the magnetometer data.

EK3_MAX_FLOW (Maximum valid optical flow rate)

DEFAULT: 2.5f	RANGE: 1.0 4.0	UNITS: rad/s

Maximum valid optical flow rate (EK3_MAX_FLOW)

Description

This parameter sets a hard limit on the optical flow rate (angular velocity of the visual texture) that the Extended Kalman Filter (EKF) will accept. If the flow sensor reports a value higher than this, it is assumed to be a glitch (e.g., lighting change, shadow, or sensor error) or a maneuver exceeding the sensor's capability, and the measurement is rejected to prevent position divergence.

The Mathematics

$$\text{Reject} \iff |\vec{\omega}_\text{flow}| > \text{EK3_MAX_FLOW}$$

The Engineer's View

Defined in `libraries/AP_NavEKF3/AP_NavEKF3.cpp`. Used in the optical flow fusion logic. High flow rates often correlate with poor signal quality or motion blur, making the data unreliable.

Tuning & Behavior

- **Default Value:** 2.5 rad/s (~143 deg/s)
- **Range:** 1.0 - 4.0 rad/s
- **Tuning:** Increase slightly for highly agile quadcopters flying close to the ground, but be aware that sensor noise often increases at high rates.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





EK3_NOAID_M_NSE (EKF3 No-Aid Measurement Noise)

DEFAULT: 10.0	RANGE: 1.0 50.0	UNITS: m/s

EKF3 No-Aid Measurement Noise (EK3_NOAID_M_NSE)

Description

`EK3_NOAID_M_NSE` prevents the EKF from exploding when GPS is lost.

When all position sensors fail, the EKF assumes the drone is stationary (or moving at constant velocity) with a very high uncertainty. This parameter sets that uncertainty.

Tuning & Behavior

- **Default Value:** 10.0 m/s.

EK3_OGNM_TEST_SF (EKF3 On-Ground Movement Test Scale)

DEFAULT: 1.0	RANGE: 1.0 10.0

EKF3 On-Ground Movement Test Scale (EK3_OGNM_TEST_SF)

Description

`EK3_OGNM_TEST_SF` helps the EKF decide if it is flying.

The EKF assumes the drone is stationary on the ground. If the gyro/accel indicates movement that exceeds a threshold, the EKF switches to "In Air" mode. This parameter scales that threshold.

Tuning & Behavior

- **Default Value:** 1.0.
- **Recommendation:** Increase if you are launching from a moving boat or vehicle.

EK3_OGN_HGT_MASK (EKF3 Origin Height Mask)

DEFAULT: 0	RANGE: 0 7

EKF3 Origin Height Mask (EK3_OGN_HGT_MASK)

Description

`EK3_OGN_HGT_MASK` keeps your "Mean Sea Level" (MSL) altitude accurate even when you are flying using a Barometer or Lidar.

If you fly using `EK3_SRC1_POSZ = 1` (Baro), the EKF tracks your height relative to the takeoff point. However, barometers drift with weather changes. This parameter tells the EKF to use the GPS height (if available) to slowly correct the "Origin" height in the background, so that your logged global altitude remains valid over long flights.

Tuning & Behavior

- **Bit 0:** Correct datum when using **Baro** height.
- **Bit 1:** Correct datum when using **Range Finder** height.
- **Bit 2:** Apply corrections to **Local Position** (instead of Origin). *Experimental.*

Recommendation

- **Default Value:** 0.
- **Surveying:** If accurate MSL geotags are critical, set Bit 0 (Value 1).

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





EK3_OPTIONS (EKF3 Options)

DEFAULT: 0	RANGE: 0 1024

EKF3 Options (EK3_OPTIONS)

Description

`EK3_OPTIONS` is a "Power User" menu for the estimator.

- **Bit 0:** Enable "Tilt Alignment" during movement.
- **Bit 1:** Disable GPS glitch protection.
- **Bit 2:** Enable high-accuracy gyro bias learning.

Tuning & Behavior

- **Default Value:** 0.
- **Recommendation:** Leave at **0** unless you have a specific reason to change it (e.g., debugging a compass issue).

EK3_POSNE_M_NSE (GPS horizontal position noise)

DEFAULT: 0.5	RANGE: 0.1 10.0	UNITS: m

GPS horizontal position noise (EK3_POSNE_M_NSE)

Description

Specifies the expected noise level in the GPS horizontal position data for EKF3.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





EK3_POS_I_GATE (EKF3 Position Innovation Gate)

DEFAULT: 500	RANGE: 100 1000

EKF3 Position Innovation Gate (EK3_POS_I_GATE)

Description

`EK3_POS_I_GATE` is the "glitch detector" for your drone's position. It determines how far the GPS data can disagree with the drone's internal estimate before the EKF3 decides the GPS is lying.

In a Kalman Filter, the difference between the *expected* measurement and the *actual* measurement is called the **Innovation**. If your drone thinks it's at Point A, but the GPS suddenly says it's at Point B (100 meters away), the Innovation is huge. The "Gate" is a mathematical boundary—if the error is inside the gate, it's considered "normal noise" and the EKF updates its position. If it's outside the gate, it's considered a "glitch" and is ignored.

- **Low Value (e.g. 100):** Very strict. The EKF will reject even small GPS jumps. This is safe but can lead to the EKF "timing out" and losing its position estimate if the GPS is consistently slightly off.
- **High Value (e.g. 1000):** Very relaxed. The EKF will accept large position jumps. This allows for faster recovery from errors but increases the risk of the drone "darting" toward a fake GPS coordinate.

The Mathematics

The EKF uses a statistical test (Chi-Squared) to determine if a measurement is valid. The gate (G) is applied to the **Normalized Innovation Squared** (NIS):

$$NIS = \vec{y}^T \mathbf{S}^{-1} \vec{y} \leq G^2$$

Where:

- $\vec{y}$ is the Innovation vector (Measurement - Prediction).
- $\mathbf{S}$ is the Innovation Covariance (Total expected uncertainty).
- G is the value derived from `EK3_POS_I_GATE`.

Scale: The value in the GCS (e.g., 500) represents a standard deviation multiplier. A value of 500 corresponds to a "5-sigma" gate, which is extremely statistically unlikely to occur by chance (~0.00006% probability).

The Engineer's View

This parameter is the `_gpsPosInnovGate` member in `NavEKF3`.

It is checked during the `AP_NavEKF3_PosVelFusion.cpp` loop. If a measurement fails the gate, it is logged as an **EKF Primary Check Failure**. If measurements fail consistently for more than a second, the EKF will trigger a "Reset" or an "EKF Lane Switch," assuming that its internal state (not the sensor) is the one that is wrong.

Tuning & Behavior

- **Default Value:** 500 (Standard)
- **Range:** 100 - 1000
- **Effect of Increasing:** Faster recovery from large position resets. Better for high-speed flight where position uncertainty is naturally high.
- **Effect of Decreasing:** Better protection against small GPS multipath jumps. Essential for high-precision autonomous missions near obstacles.

Use Case Recommendations

- **Open Field Flight: Keep Default (500).**
- **RTK Precision Survey: Decrease to 200.** Since RTK is extremely accurate, any innovation larger than 20cm is almost certainly a sensor error and should be rejected instantly.
- **Rapid Racing / High G-Force: Increase to 800.** High-speed maneuvers create high uncertainty; a wider gate prevents the EKF from "locking out" the GPS during sharp turns.

Troubleshooting

- **Scenario:** Drone refuses to enter Loiter, and the GCS shows "EKF Position Variance."
  - **Diagnosis:** The GPS data is consistently failing the innovation gate.
  - **Fix:** If the GPS signal is clear, increase `EK3_POS_I_GATE` to 750 to help the EKF re-sync.
- **Scenario:** Drone "darts" 5 meters to the side before coming back.
  - **Diagnosis:** `EK3_POS_I_GATE` is too high, allowing the EKF to follow a bad GPS sample.
  - **Fix:** Reduce to 300.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





EK3_PRIMARY (Primary core number)

DEFAULT: 0	RANGE: 0 2

Primary core number (EK3_PRIMARY)

Description

Selects the initial primary EKF core. Lane 0 corresponds to the first IMU enabled in EK3_IMU_MASK.

EK3_RNG_I_GATE (EKF3 Rangefinder Innovation Gate)

DEFAULT: 500	RANGE: 100 1000	UNITS: %

EKF3 Rangefinder Innovation Gate (EK3_RNG_I_GATE)

Description

`EK3_RNG_I_GATE` filters out Lidar "Glitches."

Lidars often give false readings when flying over grass, water, or bright sunlight. This gate ensures that a sudden jump from 2m to 50m is ignored if the IMU says the drone hasn't moved.

Tuning & Behavior

- **Default Value:** 500 (5 Sigma).

EK3_RNG_M_NSE (EKF3 Rangefinder Measurement Noise)

DEFAULT: 0.5	RANGE: 0.1 5.0	UNITS: m

EKF3 Rangefinder Measurement Noise (EK3_RNG_M_NSE)

Description

`EK3_RNG_M_NSE` sets the trust level for the Lidar.

- **0.1:** Highly trusted. Use for precision landing on flat surfaces.
- **0.5 (Default):** Standard trust. Good for general flight.
- **1.0+:** Low trust. Use for sonars or noisy environments.

Tuning & Behavior

- **Default Value:** 0.5 m.

EK3_RNG_USE_HGT (EKF3 Rangefinder Use Height)

DEFAULT: -1	RANGE: -1 70	UNITS: %

EKF3 Rangefinder Use Height (EK3_RNG_USE_HGT)

Description

`EK3_RNG_USE_HGT` allows the EKF to automatically switch to Lidar altitude when close to the ground.

If set to 70%, and your Lidar has a 10m range, the EKF will switch from Baro to Lidar when you descend below 7 meters. This provides terrain following for landing without needing to change `EK3_SRC1_POSZ`.

Tuning & Behavior

- **-1 (Default):** Disabled. Lidar is only used if `EK3_SRCx_POSZ` = 2 (Rangefinder).
- **0-70:** Enabled. Percentage of `RNGFND_MAX_CM`.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





EK3_RNG_USE_SPD (EKF3 Rangefinder Use Speed)

DEFAULT: 2.0	RANGE: 2.0 6.0	UNITS: m/s

EKF3 Rangefinder Use Speed (EK3_RNG_USE_SPD)

Description

`EK3_RNG_USE_SPD` prevents using Lidar altitude when flying fast.

Many Lidars struggle when the ground is rushing by (motion blur) or when the drone is pitched over (measuring diagonal distance instead of vertical). This parameter forces a switch back to Baro/GPS if the drone exceeds this speed.

Tuning & Behavior

- **Default Value:** 2.0 m/s.
- **Recommendation:** Keep low (2-6 m/s) to ensure Lidar is only used for takeoff/landing hover.

EK3_SRC1_POSXY (Position Horizontal Source (Primary))

DEFAULT: 3

Position Horizontal Source (Primary) (EK3_SRC1_POSXY)

Description

Defines where the EKF3 algorithm gets its primary horizontal (latitude/longitude) position data.

Tuning & Behavior

- **Default Value:** 3 (GPS)
- **Values:** 0:None, 3:GPS, 4:Beacon, 6:ExternalNav
- **Most common setting is GPS.**

EK3_SRC1_POSZ (Position Vertical Source (Primary))

DEFAULT: 1

Position Vertical Source (Primary) (EK3_SRC1_POSZ)

Description

Defines where the EKF3 algorithm gets its primary altitude data.

Tuning & Behavior

- **Default Value:** 1 (Baro)
- **Values:** 0:None, 1:Baro, 2:RangeFinder, 3:GPS, 4:Beacon, 6:ExternalNav
- **Baro is standard for most flights.**

EK3_SRC1_VELXY (Velocity Horizontal Source (Primary))

DEFAULT: 3

Velocity Horizontal Source (Primary) (EK3_SRC1_VELXY)

Description

Defines where the EKF3 algorithm gets its primary horizontal speed and direction data.

Tuning & Behavior

- **Default Value:** 3 (GPS)
- **Values:** 0:None, 3:GPS, 4:Beacon, 5:OpticalFlow, 6:ExternalNav, 7:WheelEncoder

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





EK3_SRC1_VELZ (Vertical Velocity Source (Set 1))

DEFAULT: 3	RANGE: 0 6

Vertical Velocity Source (EK3_SRC1_VELZ)

Description

`EK3_SRC1_VELZ` defines where the Extended Kalman Filter (EKF) gets its vertical velocity (climb/descent rate) information.

Correct vertical velocity is essential for altitude hold and vertical position control. Most GPS modules provide a high-quality vertical velocity measurement based on Doppler shift, which is more responsive than deriving velocity from pressure changes (Barometer).

- **0: None.**
- **3: GPS.** (Standard for most outdoor drones).
- **4: Beacon.**
- **6: ExternalNav.** (For VIO or OptiTrack systems).

Tuning & Behavior

- **Default:** 3 (GPS).
- **Indoor Flight:** If flying indoors without GPS, this is typically set to **0** or provided by **ExternalNav**.

EK3_SRC1_YAW (Yaw Source (Primary))

DEFAULT: 1

Yaw Source (Primary) (EK3_SRC1_YAW)

Description

Defines where the EKF3 algorithm gets its primary heading (yaw) data.

Tuning & Behavior

- **Default Value:** 1 (Compass)
- **Values:** 0:None, 1:Compass, 2:GPS, 3:GPS with Compass Fallback, 6:ExternalNav, 8:GSF
- **Option 2 (GPS) refers to "moving baseline" dual-antenna GPS yaw.**

EK3_SRC2_POSXY (Position Horizontal Source (Secondary))

DEFAULT: (int8_t	RANGE: null

Position Horizontal Source (Secondary)

Note: This parameter functions identically to EK3_SRC1_POSXY.

EK3_SRC2_POSZ (Position Vertical Source (Secondary))

DEFAULT: (int8_t	RANGE: null

Position Vertical Source (Secondary)

Note: This parameter functions identically to EK3_SRC1_POSZ.

EK3_SRC2_VELXY (Velocity Horizontal Source (Secondary))

DEFAULT: (int8_t	RANGE: null

Velocity Horizontal Source (Secondary)

Note: This parameter functions identically to EK3_SRC1_VELXY.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





EK3_SRC2_VELZ (Velocity Vertical Source (Secondary))

DEFAULT: (int8_t	RANGE: null

Velocity Vertical Source (Secondary)

Note: This parameter functions identically to EK3_SRC1_VELZ.

EK3_SRC2_YAW (Custom yaw)

DEFAULT: 0	RANGE: null	UNITS: deg

Custom yaw

Note: This parameter functions identically to EK3_SRC1_YAW.

EK3_SRC3_POSXY (Position Horizontal Source (Tertiary))

DEFAULT: (int8_t	RANGE: null

Position Horizontal Source (Tertiary)

Note: This parameter functions identically to EK3_SRC1_POSXY.

EK3_SRC3_POSZ (Position Vertical Source (Tertiary))

DEFAULT: (int8_t	RANGE: null

Position Vertical Source (Tertiary)

Note: This parameter functions identically to EK3_SRC1_POSZ.

EK3_SRC3_VELXY (Velocity Horizontal Source (Tertiary))

DEFAULT: (int8_t	RANGE: null

Velocity Horizontal Source (Tertiary)

Note: This parameter functions identically to EK3_SRC1_VELXY.

EK3_SRC3_VELZ (Velocity Vertical Source (Tertiary))

DEFAULT: (int8_t	RANGE: null

Velocity Vertical Source (Tertiary)

Note: This parameter functions identically to EK3_SRC1_VELZ.

EK3_SRC3_YAW (Custom yaw)

DEFAULT: 0	RANGE: null	UNITS: deg

Custom yaw

Note: This parameter functions identically to EK3_SRC1_YAW.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





EK3_SRC_OPTIONS (EKF Source Options)

DEFAULT: 1

EKF Source Options (EK3_SRC_OPTIONS)

Description

Advanced configuration bitmask for blending or switching between multiple sensor sources (e.g., GPS and Optical Flow).

Tuning & Behavior

- **Default Value:** 1 (FuseAllVelocities)
- **Bitmask:**
  - Bit 0: FuseAllVelocities (Allows simultaneous use of multiple velocity sources)
  - Bit 1: AlignExtNavPos (Aligns external nav position when using optical flow)

EK3_TAU_OUTPUT (EKF3 Output Time Constant)

DEFAULT: 25	RANGE: 10 50	UNITS: cs

EKF3 Output Time Constant (EK3_TAU_OUTPUT)

Description

`EK3_TAU_OUTPUT` is the final smoothing filter for the EKF.

The raw EKF state can jump discretely when a GPS update arrives (5Hz or 10Hz). The "Output Observer" smooths these jumps to provide a continuous 400Hz stream to the position controller.

Tuning & Behavior

- **Default Value:** 25 (0.25 seconds).
- **Lower Value:** Faster response, more jitter.
- **Higher Value:** Smoother, more lag.

EK3_TERR_GRAD (EKF3 Terrain Gradient)

DEFAULT: 0.1	RANGE: 0 0.5	UNITS: m/m

EKF3 Terrain Gradient (EK3_TERR_GRAD)

Description

`EK3_TERR_GRAD` tells the EKF how hilly the ground is expected to be.

This is used for Optical Flow and Terrain Following. If the Lidar distance changes rapidly, the EKF uses this parameter to decide if the drone moved up/down or if the ground moved (a hill).

Tuning & Behavior

- **Default Value:** 0.1 (10% slope).
- **Hilly Terrain:** Increase to **0.5** if flying over steep hills.

EK3_VELD_M_NSE (GPS vertical velocity noise)

DEFAULT: 0.5	RANGE: 0.05 5.0	UNITS: m/s

GPS vertical velocity noise (EK3_VELD_M_NSE)

Description

Specifies the expected noise level in the GPS vertical velocity data for EKF3.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





EK3_VELNE_M_NSE (GPS horizontal velocity noise)

DEFAULT: 0.5	RANGE: 0.05 5.0	UNITS: m/s

GPS horizontal velocity noise (EK3_VELNE_M_NSE)

Description

Specifies the expected noise level in the GPS horizontal velocity data for EKF3.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





EK3_VEL_I_GATE (EKF3 Velocity Innovation Gate)

DEFAULT: 300	RANGE: 100 1000

EKF3 Velocity Innovation Gate (EK3_VEL_I_GATE)

Description

`EK3_VEL_I_GATE` is the "speed trap" for your drone's sensors. It determines how much a sensor's reported speed can disagree with the drone's internal math before the data is rejected as a glitch.

If your drone thinks it's stationary, but the GPS suddenly reports a speed of 50 m/s (common in poor signal conditions), the **Innovation** (error) is massive. `EK3_VEL_I_GATE` sets the threshold for this error. If the reported speed is outside this gate, the EKF ignores it, assuming the GPS is having a momentary glitch.

- **Low Value (e.g. 100):** Very strict. Prevents the drone from jerking or twitching if the GPS speed jumps, but can cause the EKF to lose sync if the drone is actually maneuvering aggressively.
- **High Value (e.g. 800):** Very relaxed. The EKF will follow even large jumps in speed data.
- **Default (300):** A balanced 3-sigma gate that is robust for standard flight.

The Mathematics

The gate (G) is part of the Chi-Squared consistency check:

$$\vec{y}_v^T \mathbf{S}_v^{-1} \vec{y}_v \leq G^2$$

Where:

- $\vec{y}$ is the Velocity Innovation ($V_{sensor} - V_{predicted}$).
- $\mathbf{S}_v$ is the Innovation Covariance matrix for velocity.
- G is derived from `EK3_VEL_I_GATE`.

Scaling: The value (300) represents a percentage of the standard deviation. 300 means a "3-sigma" threshold, which covers 99.7% of all expected sensor variations. Anything beyond this is statistically "impossible" and likely a glitch.

The Engineer's View

This parameter is the `_gpsVelInnovGate` member in `NavEKF3`.

It is applied in `AP_NavEKF3_PosVelFusion.cpp`. Velocity gating is more critical than position gating for **Flight Damping**. If a bad velocity sample passes the gate, it instantly creates a massive acceleration demand in the PID loops, causing a violent "jerk." Keeping this gate tight (e.g. 300) is what makes ArduPilot feel smooth even in noisy GPS environments.

Tuning & Behavior

- **Default Value:** 300
- **Range:** 100 - 1000
- **Effect of Increasing:** The drone is less likely to lose "EKF Health" during extreme 3D maneuvers or high-speed racing.
- **Effect of Decreasing:** Smoother flight; more aggressive rejection of GPS speed spikes.

Use Case Recommendations

- **Standard Multirotor: Keep Default (300).**
- **High-Speed Racing Drone: Increase to 500 - 600.** High-G turns create large velocity innovations that can exceed 3-sigma; a wider gate prevents the EKF from "locking out" the sensors during the turn.
- **Precision Docking / Automation: Decrease to 150.** In slow, precision movements, you want to reject any speed noise immediately to prevent the drone from "twitching" near the docking target.

Troubleshooting

- **Scenario:** GCS reports "EKF Velocity Variance" every time I perform a sharp turn.
  - **Diagnosis:** The maneuvers are so aggressive that the EKF's prediction is lagging behind the sensor, causing the error to hit the gate limit.
  - **Fix:** Increase `EK3_VEL_I_GATE` to 500.
- **Scenario:** Drone "lunges" forward randomly for a fraction of a second.
  - **Diagnosis:** `EK3_VEL_I_GATE` is too high, letting a bad GPS velocity sample corrupt the filter.
  - **Fix:** Reduce to 200.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





EK3_VIS_VERR_MAX (EKF3 Visual Odometry Max Error)

DEFAULT: 0.9	RANGE: 0.5 5.0	UNITS: m/s

EKF3 Visual Odometry Max Error (EK3_VIS_VERR_MAX)

Description

`EK3_VIS_VERR_MAX` defines the trust floor for Visual Odometry.

When the VO system reports poor quality (e.g., dark room, featureless wall), the EKF uses this error value.

Tuning & Behavior

- **Default Value:** 0.9 m/s.

EK3_VIS_VERR_MIN (EKF3 Visual Odometry Min Error)

DEFAULT: 0.1	RANGE: 0.05 0.5	UNITS: m/s

EKF3 Visual Odometry Min Error (EK3_VIS_VERR_MIN)

Description

`EK3_VIS_VERR_MIN` sets the trust ceiling for Visual Odometry (VO).

VO systems (like Intel RealSense or Zed) report a "Quality" metric. When quality is 100%, the EKF uses this error value.

- **0.1:** Very confident.
- **0.5:** Less confident.

Tuning & Behavior

- **Default Value:** 0.1 m/s.

EK3_WENC_VERR (EKF3 Wheel Encoder Velocity Error)

DEFAULT: 0.1	RANGE: 0.05 1.0	UNITS: m/s

EKF3 Wheel Encoder Velocity Error (EK3_WENC_VERR)

Description

`EK3_WENC_VERR` tells the EKF how accurate the wheel encoders are.

If you are using wheel odometry for non-GPS navigation (e.g., a rover indoors), this parameter sets the trust level.

- **Low Value:** High trust. The EKF will rely heavily on the wheels to estimate speed.
- **High Value:** Low trust. The EKF will rely more on the IMU.

Tuning & Behavior

- **Default Value:** 0.1 m/s.
- **Slippery Surface:** Increase to **0.5** if the wheels slip often.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





EK3_WIND_PSCALE (EKF3 Wind Process Noise Scale)

DEFAULT: 0.5	RANGE: 0.0 1.0

EKF3 Wind Process Noise Scale (EK3_WIND_PSCALE)

Description

`EK3_WIND_PSCALE` determines how much the EKF allows the wind estimate to change *while the drone is turning*.

When the drone maneuvers aggressively, it is harder to separate inertial forces from wind forces. This parameter tells the EKF to increase the uncertainty (noise) of the wind state during these times, allowing the estimate to shift faster to match reality.

- **High Value (1.0):** Wind estimate adapts rapidly during turns.
- **Low Value (0.0):** Wind estimate is "frozen" during turns.

The Engineer's View

This parameter scales the `_windProcessNoise` dynamically based on the maneuver status.

$$Q_wind = Q_base \times (1.0 + \text{Maneuver} \times \text{EK3_WIND_PSCALE})$$

Tuning & Behavior

- **Default Value:** 0.5.
- **Gliders/Soaring:** Increase to **1.0**. You want the wind estimate to update instantly as you circle in a thermal.
- **Mapping Plane:** Decrease to **0.1**. You want a stable, averaged wind estimate for uniform track spacing.

EK3_WIND_P_NSE (Wind velocity process noise)

DEFAULT: 0.1	RANGE: 0.01 2.0	UNITS: m/s^2

Wind velocity process noise (EK3_WIND_P_NSE)

Description

Controls how quickly EKF3 adapts to changing wind conditions.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





EK3_YAW_I_GATE (EKF3 Yaw Innovation Gate)

DEFAULT: 300	RANGE: 100 1000	UNITS: %

EKF3 Yaw Innovation Gate (EK3_YAW_I_GATE)

Description

`EK3_YAW_I_GATE` prevents the drone from following a "bad" compass.

If you fly near a large metal structure (like a bridge or power pylon), the magnetic field will twist. The compass will report a heading that is wildly different from what the Gyroscope expects. This gate detects that "Innovation" (disagreement) and rejects the compass data, forcing the drone to rely on Gyro Dead-Reckoning until it flies past the interference.

- **Low Value (100):** Very strict. Reject compass at the slightest hint of trouble. Good for flying near steel.
- **High Value (500):** Relaxed. Trust the compass even if it jumps.

The Mathematics

The Normalized Innovation Squared (NIS) for yaw is compared against the gate:

$$\text{Yaw Error}^2 / \text{Variance} < (\text{GATE}/100)^2$$

The Engineer's View

This is the primary defense against "Toilet Bowling" caused by magnetic interference. If the gate rejects the compass for too long (> 10 seconds), the EKF will declare a "Compass Variance" failsafe and may switch to GSF (GPS Yaw) if available.

Tuning & Behavior

- **Default Value:** 300 (3-Sigma).
- **Recommendation:** Keep at **300**.
- **High Interference:** If you must fly indoors near metal beams, reducing this to **100** can help the drone ignore the magnetic distortions and fly purely on Gyro (drift) for short periods.

EK3_YAW_M_NSE (Yaw measurement noise)

DEFAULT: 0.5	RANGE: 0.05 1.0	UNITS: rad

Yaw measurement noise (EK3_YAW_M_NSE)

Description

Specifies the expected noise level in the yaw (heading) data.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **ENABLE** parameter group (specifically `ENABLE_MASK`) configures which physical **Inertial Measurement Units (IMUs)** are initialized and used by the autopilot.

In modern flight controllers with multiple redundant sensors (e.g., The Cube Orange with 3 IMUs), this mask allows the user to disable a specific sensor if it is found to be defective or performing poorly without needing to recompile the firmware.

Key Concepts

1. IMU Enable Mask

The autopilot attempts to detect all sensors on the internal SPI/I2C buses.

- `ENABLE_MASK`: A bitmask where each bit corresponds to an IMU index.
  - **Bit 0:** IMU 1 (Internal)
  - **Bit 1:** IMU 2 (Internal)
  - **Bit 2:** IMU 3 (Internal)
  - **Value 7 (1+2+4):** Enables all three standard IMUs.

Integration Guide

- **Troubleshooting:** If you see "Inconsistent Accelerometers" or "IMU Fail" and logs show one specific IMU has huge spikes or offsets, you can temporarily disable it using this mask to maintain flight capability on the remaining healthy sensors.

Developer Notes

- **Library:** `libraries/AP_InertialSensor`
- **Relationship:** This parameter is often aliased as `INS_ENABLE_MASK` in the full parameter list.

ENABLE_MASK (IMU Enable Mask)

`DEFAULT: 127`

IMU Enable Mask (ENABLE_MASK)

Description

Defines which physical IMUs (Inertial Measurement Units) the flight controller should attempt to initialize and use.

- **Default Value:** 127 (Enable all detected IMUs)
- **Bitmask corresponds to IMU indices 1 through 7.**

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **ESC** parameter group (primarily within **AP_Periph** firmware) configures the communication with Electronic Speed Controllers (ESCs) for the purpose of telemetry and calibration.

While the main autopilot handles motor *control* (via PWM or DShot), a DroneCAN node (or the autopilot itself) can listen to the ESC's serial telemetry stream to report RPM, Temperature, Voltage, and Current for each individual motor.

Key Concepts

1. ESC Telemetry

Many modern ESCs (BLHeli_32, APD, KISS) output a low-speed serial stream (typically 115200 baud) containing real-time health data.

- `ESC_TELEM_PORT`: Selects the UART used to listen to the ESCs.
- `ESC_TELEM_RATE`: Frequency (Hz) at which this data is forwarded to the main autopilot via CAN or MAVLink.

2. ESC Calibration (`ESC_CALIBRATION`)

This parameter (in some firmwares) can trigger a passthrough mode to allow a PC-based configurator (like BLHeliSuite) to talk through the autopilot to the ESCs for firmware updates and tuning.

Parameter Breakdown

- `ESC_TELEM_PORT`: Serial port index for ESC data.
- `ESC_PWM_TYPE`: (AP_Periph) Defines the output protocol to the ESC (PWM, OneShot, DShot).
- `ESC_RATE`: The update rate of the motor control signal (Hz).

Integration Guide

1. **Hardware:** Connect the "TLM" pin from all ESCs together (usually) and wire to the RX pin of the designated `ESC_TELEM_PORT`.
2. **Enable:** Set the port and rate.
3. **Verify:** Check the `ESC_TELEMETRY` or `ESC` messages in the DataFlash log.

Developer Notes

- **Library:** `Tools/AP_Periph` and `libraries/AP_ESC_Telem`.
- **Advanced:** Highly integrated with the **Harmonic Notch Filter** (`INS_HNTCH_`), which uses individual motor RPM to dynamically target vibration frequencies.

ESC_APD_SERIAL_1 (ESC APD Serial 1 (Invalid/Legacy))

DEFAULT: 0

ESC APD Serial 1 (ESC_APD_SERIAL_1)

Description

This parameter is not found as a standard ESC parameter. It appears to be a miscategorized per-motor compass compensation value.

ESC_APD_SERIAL_2 (ESC APD Serial 2 (Invalid/Legacy))

DEFAULT: 0

ESC APD Serial 2 (ESC_APD_SERIAL_2)

Description

This parameter is not found in the main codebase.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





ESC_CALIBRATION (ESC Calibration Trigger)

DEFAULT: 0	RANGE: 0 9

ESC Calibration Trigger (ESC_CALIBRATION)

Description

`ESC_CALIBRATION` is used to match the autopilot's output signals to the input range of your **Electronic Speed Controllers (ESCs)**.

If your motors don't all start at the same time, or if they don't reach full power, you may need to calibrate them. Setting this parameter to **3 (Automatic)** and rebooting will put the drone into calibration mode, where it passes your transmitter's throttle stick directly to the motors (with safety limits).

Common Values

- **0: Normal.** No calibration.
- **3: Automatic.** Triggers calibration on next boot.
- **9: Disabled.** Prevents accidental calibration.

Tuning & Behavior

- **Safety: REMOVE PROPELLERS** before performing any ESC calibration.
- **Note:** Digital protocols like **DShot** do not require (and cannot use) this calibration process. This is only for PWM/OneShot ESCs.

ESC_CMD_TIMO (ESC Command Timeout)

DEFAULT: 100	RANGE: 0 1000	UNITS: ms

ESC Command Timeout (ESC_CMD_TIMO)

Description

`ESC_CMD_TIMO` defines how long the autopilot will wait for a response when communicating with "Smart" ESCs (like DroneCAN or serial ESCs).

ESC_EXT_TLM_RATE (ESC Extended Telemetry update rate)

DEFAULT: 5	RANGE: 0 1000	UNITS: Hz

ESC Extended Telemetry update rate (ESC_EXT_TLM_RATE)

Description

Controls the frequency of high-detail telemetry messages (e.g., input duty cycle, internal status flags) transmitted from the AP_Periph node via CAN.

ESC_PWM_TYPE (Output PWM type)

DEFAULT: 0

Output PWM type (ESC_PWM_TYPE)

Description

Configures the motor output protocol for the AP_Periph node.

Tuning & Behavior

- **Default Value:** 0 (Normal PWM)
- **Values:** 1:Normal, 2:OneShot, 3:OneShot125, 4:Brushed, 5:DShot150, 6:DShot300, 7:DShot600, 8:DShot1200

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





ESC_RATE (ESC Update Rate)

DEFAULT: 400	RANGE: 50 400	UNITS: Hz

ESC Update Rate (ESC_RATE)

Description

Defines the output refresh rate for traditional PWM-based ESCs on an AP_Periph node.

ESC_TELEM_PORT (ESC Telemetry Serial Port)

DEFAULT: -1	RANGE: 0 10

ESC Telemetry Serial Port (ESC_TELEM_PORT)

Description

Specifies the physical serial port on the AP_Periph node used to read telemetry data from the ESCs.

ESC_TELEM_RATE (ESC Telemetry update rate)

DEFAULT: 50	RANGE: 0 1000	UNITS: Hz

ESC Telemetry update rate (ESC_TELEM_RATE)

Description

Controls the frequency of telemetry messages (voltage, current, RPM) transmitted from the AP_Periph node to the main flight controller via CAN.

ESC_TLM_MAV_OFS (ESC Telemetry mavlink offset)

DEFAULT: 0	RANGE: 0 31

ESC Telemetry mavlink offset (ESC_TLM_MAV_OFS)

Description

Remaps the ESC instance IDs for MAVLink reporting. This allows, for example, motors 5-8 to be displayed as ESCs 1-4 on a ground station for easier monitoring.

- **A value of 4 shifts everything down by 4.**

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **EXT** parameter group (specifically `EXT_BUS`) defines the hardware bus assignment for **External Sensors**.

Key Concepts

1. External Bus Selection

When using sensors that are not built-in to the flight controller (e.g., an external I2C Barometer or Compass), the autopilot needs to know which physical I2C bus to probe.

- **Pixhawk/Cube:** Usually Bus 1 (External 1) or Bus 2 (Internal).
- `EXT_BUS`: Manually overrides the autodetection if a sensor is plugged into a non-standard port.

Parameter Breakdown

- `EXT_BUS`: I2C Bus ID (0, 1, 2, 3).
  - **-1:** Autodetect (Standard behavior).

Developer Notes

- **Library:** `libraries/AP_Baro` (often aliased as `BARO_EXT_BUS`).

EXT_BUS (External baro bus)

`DEFAULT: -1`

External baro bus (EXT_BUS)

Description

Specifies a particular I2C bus to search for an external barometer.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **F** parameter group (specifically `F_START_HZ` and `F_STOP_HZ`) configures the frequency range for the **In-Flight Fast Fourier Transform (FFT)** analysis engine.

The FFT engine scans the raw accelerometer data to find the loudest noise spikes (primarily caused by motor and propeller vibration). It then uses this data to automatically tune "Notch Filters" to erase that noise before it hits the PID controllers.

Key Concepts

1. Analysis Window

- `F_START_HZ`: The lowest frequency to look for noise (e.g., 80Hz).
- `F_STOP_HZ`: The highest frequency to look for noise (e.g., 600Hz).
- **Why Limit?** Excluding very low frequencies prevents the FFT from accidentally "erasing" the pilot's actual control inputs or the airframe's natural resonant handling.

Parameter Breakdown

- `F_START_HZ`: Minimum analysis frequency.
- `F_STOP_HZ`: Maximum analysis frequency.

Integration Guide

- **Tuning:** Set these to bracket the expected RPM range of your motors. A 5-inch racer might use 100-800Hz. A large 15-inch lifter might use 40-200Hz.

Developer Notes

- **Library:** `libraries/AP_InertialSensor/AP_InertialSensor_HarmonicNotch.cpp`
- **Relationship:** These are often aliased as `FFT_MINHZ` and `FFT_MAXHZ` in newer firmware versions.

F_START_HZ (System identification Start Frequency)

DEFAULT: 0.5	RANGE: 0.01 100	UNITS: Hz

System identification Start Frequency (F_START_HZ)

Description

Defines the starting frequency of the automated "chirp" signal used in the System Identification flight mode to analyze aircraft dynamics.

F_STOP_HZ (System identification Stop Frequency)

DEFAULT: 40	RANGE: 0.01 100	UNITS: Hz

System identification Stop Frequency (F_STOP_HZ)

Description

Defines the ending frequency of the automated "chirp" signal used in the System Identification flight mode.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **FAST** parameter group (specifically `FAST_SAMPLE`) controls high-rate hardware oversampling for **IMU Sensors**.

Key Concepts

1. IMU Fast Sampling

Modern IMU chips (InvenSense, Bosch) can sample data at multi-kHz rates internally. By enabling fast sampling, the autopilot's CPU pulls data as fast as the SPI bus allows (e.g., 8kHz or 16kHz) and performs a low-pass filter in software before down-sampling to the main loop rate (e.g., 400Hz).

- **Benefit:** Dramatically improves noise rejection and reduces "aliasing" (where high-frequency vibration looks like low-frequency control problems).

Parameter Breakdown

- `FAST_SAMPLE`: Bitmask to enable fast sampling for specific IMUs.
  - **Bit 0:** IMU 1.
  - **Bit 1:** IMU 2.

Developer Notes

- **Library:** `libraries/AP_InertialSensor`
- **Hardware:** Requires high-performance processors (F7, H7) and sensors capable of high-speed SPI (ICM-20602, ICM-42688).
- **Alias:** `INS_FAST_SAMPLE`.

FAST_SAMPLE (IMU Fast Sampling Mask)

```
DEFAULT: 0
```

IMU Fast Sampling Mask (FAST_SAMPLE)

Description

Defines which physical IMUs should be sampled at their maximum possible rate (fast sampling). This can significantly improve vibration rejection and overall flight stability but requires more CPU power.

- **Default Value:** 0 (Standard sampling)
- **Bitmask corresponds to IMU indices 1 through 7.**

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **FBWB** parameter group configures the behavior of the **Fly-By-Wire B** (FBWB) flight mode in **ArduPlane**.

FBWB is a "Pilot Assist" mode where the autopilot maintains a constant altitude and airspeed when sticks are centered.

- **Roll Stick:** Commands a bank angle (stiffness/turn).
- **Pitch Stick:** Commands a **climb or descent rate**, not a pitch angle.
- **Throttle Stick:** Commands an **airspeed**, not a raw motor power.

Key Concepts

1. Climb Rate (`FBWB_CLIMB_RATE`)

Defines how fast (m/s) the aircraft will climb when the pilot pulls the pitch stick all the way back.

- **Behavior:** The autopilot uses the `TECS` (Total Energy Control System) to manage pitch and throttle together to achieve this rate.

2. Stick Logic (`FBWB_ELEV_REV`)

Allows reversing the sense of the elevator stick for altitude control.

- **0 (Normal):** Back on stick = Climb.
- **1 (Reversed):** Back on stick = Descend.

Parameter Breakdown

- `FBWB_CLIMB_RATE`: Maximum climb/descent rate (m/s).
- `FBWB_ELEV_REV`: Reverse pitch stick sense for altitude.

Integration Guide

- **Usage:** FBWB is ideal for smooth cruising. It is essentially "Manual with an autopilot safety net."
- **Tuning:** Ensure your `TECS_` parameters (Total Energy) are well-tuned before relying heavily on FBWB climb rates.

Developer Notes

- **Library:** `ArduPlane/mode_fbwb.cpp`.

FBWB_CLIMB_RATE (FBWB Climb Rate)

DEFAULT: 2	RANGE: 0.1 20	UNITS: m/s

FBWB Climb Rate (FBWB_CLIMB_RATE)

Description

Defines the maximum vertical speed achievable when the elevator stick is fully deflected in Fly-By-Wire B (FBWB) or CRUISE modes.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





FBWB_ELEV_REV (Fly By Wire elevator reverse)

DEFAULT: 0	RANGE: 0 1

Fly By Wire elevator reverse (FBWB_ELEV_REV)

Description

This parameter reverses the "Stick-to-Altitude" logic in assisted flight modes (FBWB, CRUISE).

- **Standard (0):** Pulling back on the stick commands a climb (increase altitude).
- **Reversed (1):** Pulling back on the stick commands a descent (decrease altitude).

The Mathematics

Multiplier on Pilot Input.

The Engineer's View

Defined in `ArduPlane/Parameters.cpp`. This is *not* the servo reversal (`SERVOx_REVERSED`); it is the logic reversal for the altitude controller's setpoint generation.

Tuning & Behavior

- **Default Value:** 0 (Standard)
- **Warning:** Changing this to 1 is non-standard for aviation and extremely confusing for most pilots. Only use if you have a very specific, non-standard control interface.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





CAMERA FEEDBACK CONFIGURATION (FEEDBAK)

Overview

The **FEEDBAK** parameter group (often appearing as `CAM_FEEDBAK_` in the full parameter list) configures the high-precision **Camera Shutter Feedback** system.

For professional aerial mapping (PPK/RTK), it is not enough to know when the autopilot *commanded* a photo; you must know exactly when the shutter *actually opened*. This system listens for a hardware signal from the camera (usually via the Hotshoe or Flash port) to record a precise GPS time-stamp.

Key Concepts

1. Feedback Pin (`FEEDBAK_PIN`)

The GPIO/Digital input pin on the flight controller connected to the camera's feedback line.

2. Polarity (`FEEDBAK_POL`)

Defines whether the autopilot looks for a Low-to-High (Rising) or High-to-Low (Falling) voltage edge.

- **0:** Active Low.
- **1:** Active High.

Parameter Breakdown

- `FEEDBAK_PIN`: GPIO pin number.
- `FEEDBAK_POL`: Trigger edge polarity.

Integration Guide

1. **Hardware:** Wire the camera's hotshoe adapter to a digital input.
2. **Config:** Set `CAM_FEEDBAK_PIN` to the input pin ID.
3. **Verify:** Check the DataFlash log for `CAM` messages. The time difference between `TRIG` (command) and `CAM` (feedback) represents the total system latency.

Developer Notes

- **Library:** `libraries/AP_Camera`.
- **Accuracy:** This signal is handled via interrupt or high-frequency polling to ensure microsecond-level accuracy for post-processed kinematics (PPK).

FEEDBAK_PIN (Camera feedback pin)

DEFAULT: -1

Camera feedback pin (FEEDBAK_PIN)

Description

Specifies the hardware pin connected to the camera's hot shoe for high-precision shutter logging.

FEEDBAK_POL (Camera feedback pin polarity)

DEFAULT: 1	RANGE: 0 1

Camera feedback pin polarity (FEEDBAK_POL)

Description

Defines whether the feedback signal from the camera is active-high or active-low.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





GEOFENCE CONFIGURATION (FENCE)

Overview

The **FENCE** parameter group configures the **Geofence** safety system. A Geofence is a virtual perimeter (3D box, cylinder, or complex polygon) that the vehicle is not allowed to leave (Stay-In) or enter (Stay-Out).

If the vehicle breaches a fence, the autopilot takes autonomous action (usually RTL or Land) to bring it back to safety.

Key Concepts

1. Fence Types (`FENCE_TYPE`)

ArduPilot supports multiple fence styles:

- **Altitude (`ALT_MAX`):** Prevents the vehicle from flying above a legal ceiling.
- **Circle (`RADIUS`):** A cylindrical "Stay-In" fence around the Home or takeoff point.
- **Polygon:** Custom shapes drawn in the GCS (Mission Planner/QGC).

2. Actions (`FENCE_ACTION`)

What happens when a breach occurs?

- **0:** Report Only (GCS warns, but no autonomous move).
- **1:** RTL (Returns to Home or nearest Rally point).
- **2:** Land (Immediate descent).

3. Failsafes and Margins

- `FENCE_MARGIN` : The distance from the fence boundary where the "Warning" trigger fires, allowing the autopilot to brake before actually crossing the line.
- `FENCE_ALT_MIN` : (Plane) Prevents the aircraft from diving below a safe terrain-clearance floor.

Parameter Breakdown

- `FENCE_ENABLE` : Master switch.
- `FENCE_TYPE` : Bitmask of active fence types (1=Alt, 2=Circle, 4=Polygon).
- `FENCE_ACTION` : Breach behavior.
- `FENCE_ALT_MAX` : The hard altitude ceiling (meters).
- `FENCE_RADIUS` : Maximum distance from Home (meters).

Integration Guide

Setting up a Basic Safety Fence

1. **Enable:** Set `FENCE_ENABLE = 1` .
2. **Ceiling:** Set `FENCE_ALT_MAX = 120` (FAA legal limit in meters).
3. **Boundary:** Set `FENCE_RADIUS = 300` (Visible line of sight).
4. **Action:** Set `FENCE_ACTION = 1` (RTL).
5. **Verify:** Check for "Fence: OK" in the GCS pre-arm checklist.

Developer Notes

- **Library:** `libraries/AC_Fence` .
- **Storage:** Polygon points are stored alongside mission waypoints but in a separate memory area.
- **Precision:** Fence checks run at the main loop rate (400Hz) for near-instant detection.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





FENCE_ACTION (Fence Action)

DEFAULT: 1	RANGE: 0 5

Fence Action (FENCE_ACTION)

Description

Defines the automatic flight behavior triggered by a fence breach.

Tuning & Behavior

- **Default Value:** 1 (RTL or Land)
- **Values:** 0:Report Only, 1:RTL or Land, 2:Always Land, 3:SmartRTL, 4:Brake or Land
- **Note: Plane uses 1:RTL, 6:Guided.**

FENCE_ALT_MAX (Fence Maximum Altitude)

DEFAULT: 100	RANGE: 10 1000	UNITS: m

Fence Maximum Altitude (FENCE_ALT_MAX)

Description

The "Ceiling" of the geofence. If the vehicle flies higher than this altitude (relative to Home), the fence action will be triggered.

FENCE_ALT_MIN (Fence Minimum Altitude)

DEFAULT: -10	RANGE: -100 100	UNITS: m

Fence Minimum Altitude (FENCE_ALT_MIN)

Description

The "Floor" of the geofence. Useful for preventing underwater vehicles from diving too deep or keeping aircraft above a certain height.

FENCE_AUTOENABLE (Fence Auto-Enable)

DEFAULT: 0	RANGE: 0 3

Fence Auto-Enable (FENCE_AUTOENABLE)

Description

Allows the flight controller to automatically engage the geofence system at specific flight stages, such as after reaching a safe altitude following takeoff.

Tuning & Behavior

- **Default Value:** 0 (Always Disabled)
- **Values:**
  - 1: AutoEnableOnTakeoff
  - 2: AutoEnableDisableFloorOnLanding
  - 3: AutoEnableOnlyWhenArmed

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





FENCE_ENABLE (Fence enable/disable)

DEFAULT: 0	RANGE: 0 1

Fence enable/disable (FENCE_ENABLE)

Description

Master power switch for the geofence system. When enabled, the vehicle will monitor its position against defined boundaries and take automatic action if they are breached.

Tuning & Behavior

- **Default Value:** 0 (Disabled)
- **Requires at least one fence type (e.g., Circle or Polygon) to be configured in FENCE_TYPE.**

FENCE_MARGIN (Fence Margin)

DEFAULT: 2.0	RANGE: 1 10	UNITS: m

Fence Margin (FENCE_MARGIN)

Description

A buffer zone inside the fence boundary. The flight controller will attempt to turn the vehicle back once it enters this margin, rather than waiting for the hard boundary of the fence itself.

FENCE_NTF_FREQ (Video Transmitter Frequency (Invalid/Legacy))

DEFAULT: 0	RANGE: 1000 6000	UNITS: MHz

Video Transmitter Frequency (FENCE_NTF_FREQ)

Description

This parameter is not found as FENCE_NTF_FREQ in the main codebase. It appears to be a miscategorized Video Transmitter parameter.

- **Note: Hardware frequency setting for VTX is typically handled via VTX_FREQ.**

FENCE_OPTIONS (Fence options)

DEFAULT: 0

Fence options (FENCE_OPTIONS)

Description

Configuration bitmask for tailoring fence logic, such as preventing flight mode changes until a breach is cleared.

Tuning & Behavior

- **Default Value:** 0
- **Bitmask:**
  - Bit 0: Disable mode change following action until breach cleared
  - Bit 1: Allow union of inclusion areas

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





FENCE_RADIUS (Circular Fence Radius)

DEFAULT: 300	RANGE: 30 10000	UNITS: m

Circular Fence Radius (FENCE_RADIUS)

Description

Defines the maximum horizontal distance from the Home point allowed before a fence breach occurs.

- **Requires Bit 1 to be set in FENCE_TYPE.**

FENCE_RET_ALT (Fence Return Altitude)

DEFAULT: 0	RANGE: 0 32767	UNITS: m

Fence Return Altitude (FENCE_RET_ALT)

Description

Specifies the safe altitude to maintain while returning from a geofence breach.

- **Units are in meters.**

FENCE_RET_RALLY (Fence Return to Rally)

DEFAULT: 0	RANGE: 0 1

Fence Return to Rally (FENCE_RET_RALLY)

Description

For ArduPlane, specifies whether a fence breach should trigger a return to the nearest rally point instead of the primary fence return point.

- **0: Fence Return Point**
- **1: Nearest Rally Point**

FENCE_TOTAL (Fence polygon point total)

DEFAULT: 0	RANGE: 1 20

Fence polygon point total (FENCE_TOTAL)

Description

Reports the total number of vertices (points) currently loaded for the polygon fence.

- **Do not update this value manually.**

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





FENCE_TYPE (Fence Type)

`DEFAULT: 7`

Fence Type (FENCE_TYPE)

Description

Defines which boundary types are active. Multiple types can be enabled simultaneously using the bitmask.

Tuning & Behavior

- **Default Value:** 7 (Max Alt, Circle, and Polygon)
- **Bitmask:**
  - Bit 0: Max Altitude
  - Bit 1: Circle Centered on Home
  - Bit 2: Polygon (loaded from SD card or GCS)
  - Bit 3: Min Altitude (Floor)

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **FFT** parameter group configures the **In-Flight Fast Fourier Transform** analysis engine. This is an advanced feature that allows the autopilot to listen to the vibration of the motors in real-time to identify precise noise frequencies.

This data is used to drive the **Harmonic Notch Filter** (`INS_HNTCH_`), which dynamically "tunes" its suppression center to match the motor RPM, effectively erasing vibration before it can destabilize the PID controllers.

Key Concepts

1. Dynamic Noise Tracking

Unlike a static low-pass filter (which creates delay), the FFT engine tracks the peak noise frequency as the pilot changes throttle. This allows for very high-performance tunes even on noisy airframes.

2. Frequency Range (`FFT_MINHZ` / `FFT_MAXHZ`)

Tells the engine where to look for noise.

- **Default:** Usually 80-600Hz.
- **Large Drones:** Look lower (40-150Hz).
- **Racer Drones:** Look higher (150-1000Hz).

3. Sampling Mode (`FFT_SAMPLE_MODE`)

Controls how the data is pulled from the IMUs.

- **0:** Process all IMU data (CPU intensive).
- **1:** Process a subset of data (recommended for F4/F7).

Parameter Breakdown

- `FFT_ENABLE`: Master switch.
- `FFT_WINDOW_SIZE`: Analysis buffer size (larger = more accurate but more delay).
- `FFT_NUM_FRAMES`: How many windows to average (reduces jitter).
- `FFT_SNR_REF`: Signal-to-Noise ratio threshold to consider a peak valid.

Integration Guide

Setting up Dynamic Filtering

1. **Requirement:** A fast processor (F7 or H7) is highly recommended.
2. **Enable:** Set `FFT_ENABLE = 1`.
3. **Monitor:** Perform a test flight. Check the `FTN1` and `FTN2` log messages.
4. **Filter:** Once the FFT is tracking motor noise accurately, set `INS_HNTCH_MODE = 4` (FFT) to use this data for the notch filter.

Developer Notes

- **Library:** `libraries/AP_InertialSensor/AP_InertialSensor_HarmonicNotch.cpp`.
- **Backend:** Uses the KISS FFT library or CMSIS-DSP on ARM processors.

FFT_ATT_REF (FFT attenuation for bandwidth calculation)

DEFAULT: 15	RANGE: 0 100	UNITS: dB

FFT attenuation for bandwidth calculation (FFT_ATT_REF)

Description

Specifies the power drop (in dB) from the center frequency used to define the boundaries of a noise peak, determining its bandwidth.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





FFT_BW_HOVER (FFT learned bandwidth at hover)

DEFAULT: 20	RANGE: 0 200	UNITS: Hz

FFT learned bandwidth at hover (FFT_BW_HOVER)

Description

Records the typical width of the motor vibration peak during stable hover, used to configure the "Q" factor of the dynamic notch filter.

FFT_ENABLE (Enable Gyro FFT analyser)

DEFAULT: 0	RANGE: 0 1

Enable Gyro FFT analyser (FFT_ENABLE)

Description

Master switch for the Gyro FFT module. When enabled, the autopilot performs continuous frequency analysis of gyro noise, allowing for dynamic filtering of motor vibrations via the harmonic notch filter.

Tuning & Behavior

- **Default Value:** 0 (Disabled)
- **Requires reboot to take effect.**
- **Highly recommended for modern multicopters to improve flight performance.**

FFT_FREQ_HOVER (FFT learned hover frequency)

DEFAULT: 80.0	RANGE: 0 250	UNITS: Hz

FFT learned hover frequency (FFT_FREQ_HOVER)

Description

This value is automatically populated by the flight controller as it identifies the primary motor noise frequency while hovering. It serves as a baseline for the harmonic notch filter.

FFT_HMNC_FIT (FFT harmonic fit threshold)

DEFAULT: 10	RANGE: 0 100	UNITS: %

FFT harmonic fit threshold (FFT_HMNC_FIT)

Description

Advanced setting used to group multiple detected frequencies as harmonics of a single fundamental motor frequency. This improves the reliability of the dynamic notch filter by ensuring it tracks the core vibration source.

Tuning & Behavior

- **Default Value:** 10 %
- **A value of 0 disables harmonic matching.**

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





FFT_HMNC_PEAK (FFT harmonic peak target)

DEFAULT: 0	RANGE: 0 5

FFT harmonic peak target (FFT_HMNC_PEAK)

Description

Specifies the strategy for selecting which vibration peak the FFT should track.

- **0: Auto (Recommended)**
- **1: Highest energy peak**
- **4: Roll-Axis only**
- **5: Pitch-Axis only**

FFT_MAXHZ (Maximum Frequency)

DEFAULT: 450	RANGE: 20 495	UNITS: Hz

Maximum Frequency (FFT_MAXHZ)

Description

Defines the highest frequency the FFT engine will monitor. Typically set based on the maximum expected motor RPM.

FFT_MINHZ (Minimum Frequency)

DEFAULT: 50	RANGE: 20 400	UNITS: Hz

Minimum Frequency (FFT_MINHZ)

Description

Defines the lowest frequency the FFT engine will monitor. This should be set slightly below the expected idle RPM of your motors (converted to Hz).

FFT_NUM_FRAMES (FFT output averaging)

DEFAULT: 0	RANGE: 0 8

FFT output averaging (FFT_NUM_FRAMES)

Description

Smoothing factor for the FFT output. Averaging multiple frames reduces "jitter" in the tracked frequency but adds a small amount of latency to the filter's response.

- **0: No averaging (Fastest)**

FFT_OPTIONS (FFT options)

DEFAULT: 0

FFT options (FFT_OPTIONS)

Description

Bitmask for advanced FFT behaviors.

- **Bit 0: Enable post-filter FFT (Analyze noise *after* software filters)**
- **Bit 1: Check motor noise (Use ESC RPM as a reference)**

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





FFT_SAMPLE_MODE (FFT Sample Mode)

DEFAULT: 0	RANGE: 0 4

FFT Sample Mode (FFT_SAMPLE_MODE)

Description

Defines the source and rate of gyro data input to the FFT engine.

- **0: Gyro rate (Standard)**
- **1: Fast loop rate**
- **Requires reboot to take effect.**

FFT_SNR_REF (FFT SNR reference threshold)

DEFAULT: 25.0	RANGE: 0.0 100.0	UNITS: dB

FFT SNR reference threshold (FFT_SNR_REF)

Description

Determines how prominent a vibration frequency must be above the background noise floor to be considered a "peak" and targeted by a filter.

FFT_THR_REF (FFT learned thrust reference)

DEFAULT: 0.35	RANGE: 0.01 0.9

FFT learned thrust reference (FFT_THR_REF)

Description

The throttle value (0.0 to 1.0) at which the aircraft was hovering when FFT_FREQ_HOVER was identified. This is used to scale the filter frequency as throttle changes.

FFT_WINDOW_OLAP (FFT window overlap)

DEFAULT: 0.5	RANGE: 0 0.9

FFT window overlap (FFT_WINDOW_OLAP)

Description

Controls how much data from the previous FFT window is reused in the next calculation. A higher overlap (e.g., 0.75) results in smoother tracking of noise peaks.

FFT_WINDOW_SIZE (FFT window size)

DEFAULT: 32	RANGE: 32 1024

FFT window size (FFT_WINDOW_SIZE)

Description

Determines the balance between frequency resolution and time resolution.

- **Larger windows (e.g., 128, 256) provide more precise frequency identification but have higher latency and CPU usage.**
- **Smaller windows (e.g., 32, 64) react faster to changing noise but are less precise.**

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





FLOWHOLD MODE CONFIGURATION (FHLD)

Overview

The **FHLD** parameter group configures the behavior of the **FlowHold** flight mode. This mode uses an **Optical Flow** sensor and a **Rangefinder** (Lidar) to hold the vehicle's position without the need for GPS.

It is ideal for indoor flight or flying in "canyons" where GPS is unreliable. FlowHold is different from standard Loiter because it doesn't require a global coordinate lock; it simply tries to keep the ground textures underneath it stationary.

Key Concepts

1. Velocity Control (`FHLD_XY_P`)

This defines how aggressively the vehicle tries to correct horizontal drift.

- `FHLD_XY_P`: Proportional gain. Increase for a "locked-in" feel. Decrease if the vehicle oscillates.

2. Quality Threshold (`FHLD_QUAL_MIN`)

Optical flow sensors require good lighting and distinct ground texture.

- `FHLD_QUAL_MIN`: The minimum "confidence" score from the sensor (0-255). If quality drops below this, the vehicle will revert to Manual (AltHold) control.

3. Braking (`FHLD_BRAKE_RATE`)

Limits how fast the vehicle slows down when the pilot releases the sticks.

Parameter Breakdown

- `FHLD_XY_P` / `I`: Horizontal position PID gains.
- `FHLD_FILT_HZ`: Cutoff frequency for the flow data filter (Hz).
- `FHLD_FLOW_MAX`: Maximum expected angular flow rate.

Integration Guide

1. **Hardware:** Mount an Optical Flow sensor (e.g., HereFlow) and a Rangefinder.
2. **Calibrate:** Ensure the flow sensor is calibrated correctly (rotation and scaling).
3. **Tuning:** Start with default gains. If the vehicle drifts in FlowHold, increase `FHLD_XY_P`.

Developer Notes

- **Library:** `ArduCopter/mode_flowhold.cpp`.
- **Difference:** Unlike EKF-based Loiter (which can also use Flow), FlowHold is a simpler, stand-alone mode that works even if the EKF is not fully aligned.

FHLD_BRAKE_RATE (FlowHold Braking rate)

DEFAULT: 8	RANGE: 1 30	UNITS: deg/s

FlowHold Braking rate (FHLD_BRAKE_RATE)

Description

Defines the intensity of the automatic braking maneuver in FlowHold mode. Higher values result in a more aggressive stop.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





FHLD_FILT_HZ (FlowHold Filter Frequency)

DEFAULT: 5	RANGE: 1 100	UNITS: Hz

FlowHold Filter Frequency (FHLD_FILT_HZ)

Description

Smooths the noisy raw data from the optical flow sensor before it is used by the position controller.

FHLD_FLOW_MAX (FlowHold Flow Rate Max)

DEFAULT: 0.6	RANGE: 0.1 2.5

FlowHold Flow Rate Max (FHLD_FLOW_MAX)

Description

A safety limit on the optical flow input. This prevents the controller from reacting too aggressively to high flow rates, which can occur when flying very close to the ground.

FHLD_QUAL_MIN (FlowHold Flow quality minimum)

DEFAULT: 10	RANGE: 0 255

FlowHold Flow quality minimum (FHLD_QUAL_MIN)

Description

Threshold for the optical flow sensor's quality metric. If the sensor reports a quality below this value (due to poor lighting or low texture), the vehicle will fallback to AltHold behavior.

FHLD_XY_FILT_HZ (FlowHold filter frequency)

DEFAULT: 5	RANGE: 0 100	UNITS: Hz

FlowHold filter frequency (FHLD_XY_FILT_HZ)

Description

Smoothing filter for the input to the FlowHold PI controller.

FHLD_XY_I (FlowHold I gain)

DEFAULT: 0.1	RANGE: 0.02 1.00

FlowHold I gain (FHLD_XY_I)

Description

Corrects for long-term steady-state drift in FlowHold mode, such as that caused by wind or slight sensor bias.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





FHLD_XY_IMAX (FlowHold Integrator Max)

DEFAULT: 1000	**RANGE:** 0 4500	**UNITS:** cdeg

FlowHold Integrator Max (FHLD_XY_IMAX)

Description

Limits the maximum authority of the I-term to prevent "wind-up" and excessive tilt.

FHLD_XY_P (FlowHold P gain)

DEFAULT: 0.2	**RANGE:** 0.1 6.0

FlowHold P gain (FHLD_XY_P)

Description

Determines the immediate corrective response to position drift in FlowHold mode.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **FIELD** parameter group (specifically `FIELD_ELV` or `BARO_FIELD_ELV`) allows the user to provide a known reference altitude for the takeoff location.

Key Concepts

1. Absolute Altitude Reference

By default, the barometer measures altitude relative to the pressure at the moment of arming (0m AGL). If the user knows the exact altitude above sea level (AMSL) of the field, they can enter it here to provide a more accurate absolute vertical reference for the EKF and Ground Control Station.

Parameter Breakdown

- `FIELD_ELV`: Known elevation of the takeoff point in meters above sea level.
  - **0:** System attempts to determine origin altitude from GPS.

Developer Notes

- **Library:** `libraries/AP_Baro`.
- **Persistence:** In many configurations, this parameter resets to 0 on reboot to prevent using an incorrect elevation at a different location.

FIELD_ELV (field elevation)

DEFAULT: 0	UNITS: m

field elevation (FIELD_ELV)

Description

Manual input for the airfield's elevation above sea level. This improves absolute altitude accuracy.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





LOG FILE MANAGEMENT (FILE)

Overview

The **FILE** parameter group (often appearing as `LOG_FILE_` in the full parameter list) configures the low-level management of data logs stored on the **SD Card**.

These parameters ensure that the logging system does not crash the CPU during high-speed writes and that the SD card does not run out of space during a mission.

Key Concepts

1. Write Buffering (`FILE_BUFSIZE`)

The autopilot writes data to a RAM buffer before flushing it to the SD card.

- **Larger Buffer:** Smoother CPU performance, but more data is lost if the vehicle crashes/powers off before the buffer flushes.

2. Space Management (`FILE_MB_FREE`)

Ensures that the SD card is not filled to 100%. The autopilot will stop logging or delete old logs if the free space drops below this threshold.

3. Log Rotation (`FILE_DSRMROT`)

Determines when a new log file is created.

- **Standard:** Create a new file every time the vehicle is Armed.

Parameter Breakdown

- `FILE_BUFSIZE`: Logging buffer size in kilobytes.
- `FILE_MB_FREE`: Minimum free space (MB) to keep on the SD card.
- `FILE_RATEMAX`: Limits the maximum frequency of writes to the file system.

Developer Notes

- **Library:** `libraries/AP_Logger/AP_Logger_File.cpp`.
- **Performance:** High-speed logging (e.g., 400Hz IMU) requires a Class 10 or faster SD card.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





FILE_BUFSIZE (Logging File buffer size)

DEFAULT: 200	RANGE: 4 256	UNITS: kB

Logging File buffer size (FILE_BUFSIZE)

Description

This parameter defines the size of the memory buffer (in kilobytes) allocated for the file-based logging system. This buffer acts as a "waiting room" for data being written to the microSD card.

If the SD card is slow or busy (e.g., during high vibration or when the card is nearly full), this buffer allows ArduPilot to continue collecting sensor data without losing ("dropping") any log messages. A larger buffer provides more protection against "gaps" in your logs.

The Mathematics

The buffer size is a direct allocation of RAM.

$$RAM_logging = FILE_BUFSIZE \times 1024 \text{ bytes}$$

If the incoming data rate (bytes per second) exceeds the SD card's write speed for a duration t, the buffer fills at a rate:

$$\Delta Buffer = \int_0^t (Rate_in - Rate_out), dt$$

If $\Delta Buffer > FILE_BUFSIZE$, data is lost.

The Engineer's View

In `AP_Logger.cpp`, this parameter is defined as `_FILE_BUFSIZE`.

- **High Performance Boards:** Default is often **200kB**.
- **Limited RAM Boards:** Default can be as low as **16kB**.
- **Replay Mode:** When `LOG_REPLAY` is enabled (for EKF diagnostic testing), the data volume increases significantly. In these cases, it is highly recommended to increase `FILE_BUFSIZE` (to 100-200kB) to ensure the high-frequency sensor data required for a successful replay is not dropped.

Tuning & Behavior

- **Default Value:** Varies by board (usually 200 on H7/F7, 16 on F4).
- **Range:** 4 to 256 kB.
- **When to Increase:** If you see "Log Drop" or "Missing Data" warnings in your Ground Control Station, or if you are using high-rate logging for research or EKF Replay.
- **When to Decrease:** Almost never, unless you are critically low on RAM for Lua scripts or other features.
- **Reboot Required:** Yes. RAM allocation happens at startup.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





FILE_DSRMROT (Stop logging to current file on disarm)

DEFAULT: 0	RANGE: 0 1	UNITS: Option

Stop logging to current file on disarm (FILE_DSRMROT)

Description

This parameter controls how ArduPilot handles log file rotation when the vehicle disarms. It allows you to decide if multiple flights on the same battery should be saved in one large file or split into separate, distinct files.

- **0 (Default):** Keeps the same log file open until the system is rebooted. This results in one log file containing all activities (even across several arm/disarm cycles).
- **1 (Enabled):** Closes the current log file immediately upon disarming. If `LOG_DISARMED` is enabled, it will then start a brand-new log file for the next session.

The Mathematics

This parameter functions as a boolean gate in the state machine logic:

$$\text{IF (Disarmed AND } FILE_DSRMROT == 1) \rightarrow \text{Close current handle}$$

The Engineer's View

In `AP_Logger.cpp`, this parameter maps to `_params.file_disarm_rot`.
When the vehicle state changes to disarmed, the logging backend checks this flag.

- If set, `AP_Logger_File::stop_logging()` is called.
- This is useful for professional operators who want a clean 1-to-1 mapping between a flight mission and a `.bin` log file, making data analysis much more organized.

Tuning & Behavior

- **0:** Preferred for casual flying or if you are troubleshooting ground issues and want a continuous record.
- **1:** Recommended for commercial mapping, inspections, or competition flights. It ensures that if one flight is successful and the next crashes, the logs are already segmented and easier to manage.
- **Dependencies:** Works most effectively when `LOG_DISARMED` is also set to 1 or 2, ensuring that the "pre-arm" period for the next flight is captured in its own new file.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY

FILE_MB_FREE (Logging Minimum Free Space)

DEFAULT: 500	RANGE: 10 5000	UNITS: MB

Logging Minimum Free Space (FILE_MB_FREE)

Description

This parameter manages the "Automatic Cleanup" of your microSD card. It sets a minimum threshold for free space (in megabytes). If the card fills up and the free space drops below this value, ArduPilot will automatically delete the **oldest** log files to make room for new data.

This ensures that you always have enough space for your current flight and prevents the logging system from crashing or hanging due to a full filesystem.

The Mathematics

The logging backend monitors the available space on the disk.

$$\text{IF } (\text{DiskFreeMB} < FILE_MB_FREE) \rightarrow \text{Delete oldest .BIN file}$$

This loop continues until the condition is no longer met or no more logs can be deleted.

The Engineer's View

In `AP_Logger_File.cpp`, the cleanup logic is part of the maintenance task.

- It uses `AP::FS().get_disk_free()` to check remaining capacity.
- It iterates through the existing log list (starting from `last_log_num` backwards) to identify targets for deletion.
- By maintaining a significant buffer (default 500MB), it avoids file fragmentation issues and ensures the filesystem has overhead for metadata updates and other temporary files.

Tuning & Behavior

- **Default Value:** 500 MB.
- **Recommendation:** Set this to a value larger than your longest expected flight log (e.g., if a 1-hour flight produces a 100MB log, 500MB is a very safe buffer).
- **Agile Development:** If you fly many short missions and don't want to manually clear your card, this parameter is your best friend.
- **Critical Data:** If you cannot afford to lose old logs, you should set this lower and monitor your card manually, or better yet, increase the SD card size.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY

FILE_RATEMAX (Maximum logging rate for file backend)

DEFAULT: 0	RANGE: 0 1000	UNITS: Hz

Maximum logging rate for file backend (FILE_RATEMAX)

Description

This parameter acts as a "Data Governor" for the logging system. It sets a global cap on how frequently (in Hz) streaming log messages can be written to the SD card.

By setting a limit, you can prevent the autopilot from overwhelming a slow microSD card with too much data, which can cause lag or CPU spikes. This is particularly useful on vehicles with many high-frequency sensors or complex Lua scripts that generate a lot of custom logging.

The Mathematics

The logging backend tracks the last time a message of a specific type was written.

$$\Delta T_min = \frac{1}{FILE_RATEMAX}$$

If the time elapsed since the last log write is less than ΔT_min, the current message is skipped.

The Engineer's View

In `AP_Logger_File.cpp`, the streaming write logic checks `_params.file_ratemax`.

- **0:** Unlimited rate. The system logs every message as soon as it arrives (subject to buffer space).
- **Non-Zero:** Applies a per-instance throttle.
- Note that this only affects "streaming" messages (periodic sensor updates); critical events (errors, mode changes, parameters) are always logged immediately to ensure they are never missed.

Tuning & Behavior

- **Default Value:** 0 (Rate limiting disabled).
- **Range:** 0 to 1000 Hz.
- **When to Use:** If you see "High CPU" or "IO Performance" warnings in your logs, or if your `.BIN` files are becoming unnecessarily large for long endurance flights.
- **Typical Value:** Setting this to **50Hz** or **100Hz** is often sufficient for standard flight analysis while significantly reducing the load on the SD card interface.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





FILE_TIMEOUT (Timeout before giving up on file writes)

DEFAULT: 5	RANGE: 1 30	UNITS: s

Timeout before giving up on file writes (FILE_TIMEOUT)

Description

This parameter sets a "Safety Watchdog" for SD card operations. If the autopilot attempts to write to the microSD card and the operation hangs or fails repeatedly for a period longer than this timeout, the autopilot will give up and close the log file.

This is a critical safety feature: it prevents a faulty or slow SD card from "locking up" the main flight control loop, which could cause the aircraft to crash. It is better to lose the log than to lose the aircraft.

The Mathematics

The logging backend tracks the last successful write timestamp ($t_success$).

$$\text{IF } (t_now - t_success > FILE_TIMEOUT) \text{ AND WriteFailureActive} \rightarrow \text{Abort Logging}$$

The default value is **5 seconds**.

The Engineer's View

In `AP_Logger_File.cpp`, this parameter maps to `_params.file_timeout`.

- It is used in the `AP_Logger_File::periodic` task.
- If a write failure persists beyond the threshold, the file handle is invalidated (`_fd = -1`).
- The system will then wait for a separate "Re-open" period (typically 5 seconds) before trying to initialize a new log file, ensuring the system doesn't enter a tight loop of failing I/O.

Tuning & Behavior

- **Default Value:** 5 seconds.
- **Recommendation:** Leave at **5s**.
- **Increasing:** May be necessary for extremely high-volume logging on slower industrial cards, but increases the risk of "CPU Hang" during write retries.
- **Decreasing:** Makes the system more aggressive at cutting off logging if the card is slow.
- **Symptom of Timeout:** If your logging stops mid-flight but the aircraft continues to fly normally, your SD card likely exceeded this timeout due to write latency.
    - **Fix:** Format the card, use a Class 10/U3 card, or increase `FILE_BUFSIZE`.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **FILT** parameter group configures frequency-domain filters used to clean raw sensor data (IMU, Baro, Airspeed) before it is used by the control loops.

All digital sensors contain noise. Filtering removes high-frequency vibrations and "jitter," but introduces a small amount of "phase lag" (delay). Finding the right balance is the key to a stable vehicle.

Key Concepts

1. Low-Pass Filters (`FILT_HZ`)

A low-pass filter allows slow movements (pilot commands) to pass while blocking fast movements (motor vibrations).

- `FILT_HZ`: The cutoff frequency.
  - **Higher Hz:** Less delay, but more noise.
  - **Lower Hz:** Cleaner signal, but the vehicle might feel "mushy" or oscillate due to the delay in control response.

2. Notch Filters (`FILT1_NOTCH_...`)

A notch filter targets a **specific** narrow frequency range (e.g., exactly 120Hz) to erase a known vibration without affecting other frequencies.

Parameter Breakdown

- `FILT_HZ`: General filter cutoff frequency.
- `FILT1_NOTCH_FREQ`: Center frequency of the notch.
- `FILT1_NOTCH_Q`: Sharpness (width) of the notch.

Integration Guide

- **Default:** Start with default values.
- **Vibrations:** If your drone has high vibrations (check logs), you may need to lower the `FILT_HZ` for the D-term or enable a Harmonic Notch.

Developer Notes

- **Library:** `libraries/AP_Filter`.
- **Context:** These are often used as low-level helpers within `AP_InertialSensor` and `AC_AttitudeControl`.

FILT1_NOTCH_ATT (Notch Filter Attenuation)

DEFAULT: 0.1	RANGE: 0 1.0

Notch Filter Attenuation (FILT1_NOTCH_ATT)

Description

`FILT1_NOTCH_ATT` defines how much of the noise is removed. A value of 0.1 reduces the signal at that frequency to 10% of its original strength.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





FILT1_NOTCH_FREQ (Notch Filter Center Frequency)

DEFAULT: 0	RANGE: 10 500	UNITS: Hz

Notch Filter Center Frequency (FILT1_NOTCH_FREQ)

Description

`FILT1_NOTCH_FREQ` defines the "Target" of a specialized electrical filter used to clean up noisy sensor data.

A **Notch Filter** removes one specific frequency while leaving others untouched. This is most commonly used to eliminate the vibration frequency of a drone's motors from the accelerometer or gyroscope data, allowing for much sharper and more stable flight control.

Tuning & Behavior

- **Default:** 0 (Disabled).
- **Usage:** Find your frame's peak vibration frequency using the FFT diagnostic tool or by analyzing logs. Enter that frequency here.
- **Effect:** Reduces mechanical "noise" that would otherwise confuse the PID controller.
- **Integration:** Must be used with FILT1_NOTCH_Q to define the width of the filter.

FILT1_NOTCH_Q (Notch Filter Quality Factor (Q))

DEFAULT: 1.0	RANGE: 0.1 5.0

Notch Filter Quality Factor (FILT1_NOTCH_Q)

Description

`FILT1_NOTCH_Q` determines how narrow the "Notch" is.

- **Higher Q (e.g. 2.0):** Narrow notch. Removes a very specific frequency without affecting nearby data.
- **Lower Q (e.g. 0.5):** Wide notch. Removes a broader range of frequencies around the center.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY

FILT_HZ (FlowHold Filter Frequency)

DEFAULT: 5	RANGE: 1 100	UNITS: Hz

FlowHold Filter Frequency (FILT_HZ)

Description

This parameter sets the cutoff frequency for the low-pass filter applied to Optical Flow sensor data when using **FlowHold** mode. Optical flow sensors (which measure the "movement" of the ground beneath the drone) can be noisy due to surface texture, lighting, or vibration.

A lower frequency makes the drone's position hold feel "softer" and more resistant to noise, but it adds latency (delay) to the control loop. A higher frequency makes the drone react more instantly to movement but can cause "jitter" if the sensor data is noisy.

The Mathematics

The filter is a standard first-order Low Pass Filter (LPF). The relationship between the cutoff frequency (f_c) and the filter's time constant (τ) is:

$$\tau = \frac{1}{2\pi f_c}$$

The smoothing coefficient (α) used in each update step is:

$$\alpha = \frac{dt}{dt + \tau}$$

Where dt is the loop time (typically 0.0025s for a 400Hz control loop). At the default of **5Hz**, the time constant is approximately **0.031s**, providing significant smoothing for raw flow data.

The Engineer's View

In `AC_PI_2D.cpp`, the parameter `_filt_hz` is used to calculate the filter coefficient.

1. The code enforces a minimum frequency of `0.01Hz` to prevent division-by-zero errors.
2. In the `update()` loop, the raw flow error is passed through this filter before the Proportional (P) and Integral (I) gains are applied.
3. This ensures that high-frequency noise from the sensor does not "kick" the motors, which would cause audible chirping and wasted battery energy.

Tuning & Behavior

- **Default Value:** 5 Hz.
- **Range:** 1 to 100 Hz.
- **Effect of Increasing:** The vehicle will respond more aggressively to small drifts. Use this if the drone feels "lazy" in FlowHold but you have a very clean, high-quality sensor (e.g., HereFlow).
- **Effect of Decreasing:** The vehicle will feel more stable but may drift further before correcting. Use this if you see the drone "shaking" or "twitching" in a hover.
- **Dependencies:** Primarily used when `FLT_MODE` is set to **FlowHold**.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY

Overview

The **FLAP** parameter group configures the deployment behavior for aircraft **Flaps**. Flaps increase lift and drag, allowing for slower landing and takeoff speeds.

ArduPlane manages the deployment of flaps based on pilot switch position and, optionally, automatic flight stages (e.g., auto-landing).

Key Concepts

1. Deployment Stages (`FLAP_n_PERCNT`)

Defines the percentage of servo travel for each switch position.

- **Stage 1:** Partial flaps (often used for takeoff or loiter).
- **Stage 2:** Full flaps (used for final approach and landing).

2. Deployment Speed (`FLAP_SLEWRATE`)

Prevents the flaps from "snapping" into position, which could cause a sudden ballooning effect (rapid increase in lift).

- **Slew Rate:** Limits the degrees per second the servo can move.

Parameter Breakdown

- `FLAP_1_PERCNT`: Percentage for first stage (0-100%).
- `FLAP_2_PERCNT`: Percentage for second stage.
- `FLAP_SLEWRATE`: Maximum movement speed (% per second).

Integration Guide

1. **Outputs:** Map a servo output to `Flap` (Function 3).
2. **Config:** Set the percentages based on your airframe manufacturer's recommendations.
3. **Mixing:** Flaps often require a corresponding **Elevator-to-Flap** mix (see `TKOFF_FLAP_ELEV`) to counter the pitching moment they create.

Developer Notes

- **Library:** `ArduPlane/servos.cpp`.
- **Context:** Only relevant for **ArduPlane**.

FLAP_1_PERCNT (Automatic Flap Percentage (Stage 1))

DEFAULT: 0	RANGE: 0 100	UNITS: %

Automatic Flap Percentage (Stage 1) (FLAP_1_PERCNT)

Description

`FLAP_1_PERCNT` automates the first stage of flap deployment for fixed-wing aircraft.

Flaps are used to increase lift at low airspeeds. ArduPilot can automatically lower the flaps based on the current airspeed. This parameter sets how far the flaps should move (from 0% retracted to 100% full flaps) when the plane slows down below the FLAP_1_SPEED.

Tuning & Behavior

- **0:** Disables automatic flaps.
- **Value (%):** The target deflection. For example, setting this to **30** will lower the flaps 30% of their total range when the plane is slow.
- **Logic Link:** Used primarily in CRUISE, FBWB, and AUTO modes.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





FLAP_1_SPEED (Flap 1 Speed)

DEFAULT: 0	RANGE: 0 100	UNITS: m/s

Flap 1 Speed (FLAP_1_SPEED)

Description

Defines the airspeed threshold for automatically deploying the first stage (FLAP_1_PERCNT) of wing flaps.

Tuning & Behavior

- **Default Value:** 0 (Disabled)
- **Should be set higher than or equal to FLAP_2_SPEED.**

FLAP_2_PERCNT (Automatic Flap Percentage (Stage 2))

DEFAULT: 0	RANGE: 0 100	UNITS: %

Automatic Flap Percentage (Stage 2) (FLAP_2_PERCNT)

Description

`FLAP_2_PERCNT` defines the "Full Flaps" or secondary deployment stage.

When the plane slows down even further (below FLAP_2_SPEED), the autopilot will increase the flap deflection to this value.

FLAP_2_SPEED (Flap 2 Speed)

DEFAULT: 0	RANGE: 0 100	UNITS: m/s

Flap 2 Speed (FLAP_2_SPEED)

Description

Defines the airspeed threshold for automatically deploying the second stage (FLAP_2_PERCNT) of wing flaps.

FLAP_SLEWRATE (Flap Slew Rate)

DEFAULT: 75	RANGE: 0 100	UNITS: %/s

Flap Slew Rate (FLAP_SLEWRATE)

Description

Limits how quickly the flaps move from one position to another. This prevents sudden changes in lift and pitch when the flaps are deployed or retracted.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **FLASH** parameter group provides tools for managing the on-board non-volatile memory (Flash) of the autopilot or peripheral.

Currently, this is primarily used within **AP_Periph** firmware to trigger critical bootloader updates.

Key Concepts

1. Bootloader Update (`FLASH_BOOTLOADER`)

The bootloader is the first bit of code that runs when the board powers up. Updating it is risky but sometimes necessary to support new firmware features or hardware fixes.

- **Trigger:** Setting `FLASH_BOOTLOADER = 1` tells the firmware to overwrite the bootloader sector with an updated version embedded in the application binary.

Parameter Breakdown

- `FLASH_BOOTLOADER`:
  - **0:** Idle.
  - **1:** Perform update on next boot.

Integration Guide

- **Safety: DO NOT** trigger a bootloader update unless specifically instructed by firmware release notes. A power failure during this process can "brick" the device, requiring an ST-Link or JTAG debugger to recover.

Developer Notes

- **Library:** `Tools/AP_Periph`.

`FLASH_BOOTLOADER` (Trigger bootloader update)

DEFAULT: 0	RANGE: 0 1

Trigger bootloader update (FLASH_BOOTLOADER)

Description

DANGER ZONE. This parameter is a command trigger. When set to 1, the DroneCAN peripheral will attempt to upgrade its own bootloader using the version embedded in its current firmware image. This is a critical operation; power loss during this process can brick the device.

The Mathematics

N/A. State Machine Trigger.

The Engineer's View

Defined in `Tools/AP_Periph/Parameters.cpp`.

- **Mechanism:** On set, the system validates the embedded bootloader image, erases the bootloader sector, and writes the new image.
- **Safety:** Do not touch unless instructed by release notes.

Tuning & Behavior

- **Default Value:** 0
- **1:** Perform update. Parameter usually resets to 0 automatically or requires a reboot.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **FLIGHT** parameter group (specifically `FLIGHT_OPTIONS`) is a high-level bitmask that enables or modifies miscellaneous flight behaviors across different vehicle types.

This is often a "catch-all" for options that don't belong to a specific sensor or flight mode.

Key Concepts

1. Flight Options Bitmask

The bits in this parameter toggle experimental or non-standard behaviors.

- **Examples (Variable by Vehicle):**
  - Force arming even if some checks are failing (Advanced/Dangerous).
  - Enable specific MAVLink status reporting.
  - Alter behavior of the GCS failsafe.

Parameter Breakdown

- `FLIGHT_OPTIONS`: Bitmask.

Integration Guide

- **Recommendation:** Leave at 0 (Default) unless you are looking for a specific feature documented in the ArduPilot Wiki (e.g., "Set bit 5 of FLIGHT_OPTIONS to enable X").

Developer Notes

- **Library:** `AP_Vehicle` / `libraries/AP_Common`.

FLIGHT_OPTIONS (Flight options)

`DEFAULT: 0`

Flight options (FLIGHT_OPTIONS)

Description

Configuration bitmask for various aircraft-specific flight behaviors in ArduPlane.

Tuning & Behavior

- **Default Value:** 0
- **Bitmask Examples:**
  - Bit 3: CENTER_THROTTLE_TRIM
  - Bit 6: ENABLE_UNIT_SENSORS
  - Bit 10: DISARM_LANDED_AUTO

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **FLOW** parameter group configures the **Optical Flow Sensor**. This sensor uses a tiny high-speed camera and an image processor to track the movement of "features" (textures) on the ground. By measuring this "visual flow," the autopilot can estimate the vehicle's horizontal velocity.

Optical flow is the primary method for position hold in GPS-denied environments (Indoors, under tree canopies).

Key Concepts

1. Scaling (`FLOW_FXSCALER` / `FLOW_FYSCALER`)

The raw output of the flow sensor must be scaled based on the lens focal length and mounting height to convert "pixels per second" into "meters per second."

2. Height Requirement

Optical flow requires an accurate measurement of height above ground to calculate velocity correctly.

- **Rangefinder:** A Lidar or Ultrasonic sensor is almost always required alongside an Optical Flow sensor.

3. Orientation (`FLOW_ORIENT_YAW`)

The autopilot needs to know which way the flow camera is rotated relative to the flight controller to ensure the "Flow X" aligns with "Vehicle X."

Parameter Breakdown

- `FLOW_TYPE`: Driver selection (0=Disabled, 1=PX4Flow, 2=DroneCAN, 5=CX-OF, etc.).
- `FLOW_ADDR`: I2C address of the sensor.
- `FLOW_POS_X/Y/Z`: Offset of the sensor relative to the vehicle's center of gravity (for lever-arm compensation).

Integration Guide

1. **Hardware:** Mount the sensor facing straight down. Ensure good lighting.
2. **Enable:** Set `FLOW_TYPE` and `EK3_SRC1_VELXY = 5` (Optical Flow).
3. **Calibrate:** Perform a "Handheld Calibration" (see Wiki) to ensure the flow and gyro data are perfectly synced.
4. **Test:** Hover in Loiter mode. The vehicle should stay stationary without GPS.

Developer Notes

- **Library:** `libraries/AP_OpticalFlow`.
- **Accuracy:** Accuracy degrades with height. Most sensors are effective up to 10-20 meters AGL.
- **Surface Texture:** Flow will fail over perfectly uniform surfaces like smooth water or solid-color tiles.

FLOW_ADDR (Address on the bus)

DEFAULT: 0	RANGE: 0 127

Address on the bus (FLOW_ADDR)

Description

I2C address selection for the optical flow sensor.

Tuning & Behavior

- **Default Value:** 0
- **Range:** 0 127

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





FLOW_FXSCALER (X axis optical flow scale factor correction)

DEFAULT: 0	RANGE: -800 +800

X axis optical flow scale factor correction (FLOW_FXSCALER)

Description

Scale factor correction for the X axis optical flow rate.

Tuning & Behavior

- **Default Value:** 0
- **Range:** -800 800

FLOW_FYSCALER (Y axis optical flow scale factor correction)

DEFAULT: 0	RANGE: -800 +800

Y axis optical flow scale factor correction (FLOW_FYSCALER)

Description

Scale factor correction for the Y axis optical flow rate.

Tuning & Behavior

- **Default Value:** 0
- **Range:** -800 800

FLOW_HGT_OVR (Height override of sensor above ground)

DEFAULT: 0	RANGE: 0 2	UNITS: m

Height override of sensor above ground (FLOW_HGT_OVR)

Description

Fixed height override for rover vehicles.

Tuning & Behavior

- **Default Value:** 0 m
- **Range:** 0 2

FLOW_MAX (FlowHold Flow Rate Max)

DEFAULT: 0.6	RANGE: 0.1 2.5

FlowHold Flow Rate Max (FLOW_MAX)

Description

Limits the maximum optical flow rate considered valid during FlowHold mode.

Tuning & Behavior

- **Default Value:** 0.6
- **Range:** 0.1 2.5

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





FLOW_OPTIONS (Optical Flow Options (Invalid/Legacy))

DEFAULT: 0

Optical Flow Options (FLOW_OPTIONS)

Description

This parameter is not found in the current ArduPilot optical flow library. It may be a mis-scaffolded entry or from a specific vendor fork.

FLOW_ORIENT_YAW (Flow sensor yaw alignment)

DEFAULT: 0	RANGE: -17999 +18000	UNITS: cdeg

Flow sensor yaw alignment (FLOW_ORIENT_YAW)

Description

Yaw alignment of the flow sensor relative to the vehicle.

Tuning & Behavior

- **Default Value:** 0 cdeg
- **Range:** -17999 18000

FLOW_POS (Position offset)

DEFAULT: 0	RANGE: -5 5	UNITS: m

Position offset (FLOW_POS)

Description

XYZ position of the optical flow sensor focal point in body frame. Note: In the codebase this is split into _POS_X, _POS_Y, _POS_Z but exposed as a vector parameter.

Tuning & Behavior

- **Default Value:** 0 0 0 m
- **Range:** -5 5

FLOW_TYPE (Optical flow sensor type)

DEFAULT: 0

Optical flow sensor type (FLOW_TYPE)

Description

Optical flow sensor type.

Tuning & Behavior

- **Default Value:** 0 (None)
- **Values:** 0:None, 1:PX4Flow, 2:Pixart, 3:Bebop, 4:CXOF, 5:MAVLink, 6:DroneCAN, 7:MSP, 8:UPFLOW

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





FLIGHT MODE CONFIGURATION (FLTMODE)

Overview

The **FLTMODE** parameter group assigns specific autopilot behaviors (Flight Modes) to the physical switch positions on your RC transmitter.

ArduPilot supports up to 6 distinct flight modes accessible via a single RC channel (usually Channel 5). Each position corresponds to a specific PWM range.

Key Concepts

1. Switch Mapping (`FLTMODE1` ... `FLTMODE6`)

- **Position 1 (Low PWM):** Controlled by `FLTMODE1`.
- **Position 6 (High PWM):** Controlled by `FLTMODE6`.
- **Common Modes:**
  - **Manual/Stabilize:** Pilot has full control.
  - **Loiter/PosHold:** GPS-based position hold.
  - **Auto:** Follows a mission.
  - **RTL:** Returns to home.

2. Mode Channel (`FLTMODE_CH`)

Defines which RC channel is used for mode selection.

- **Default:** 5 (Standard).
- **Change:** Some pilots prefer channel 8 or others.

Parameter Breakdown

- `FLTMODE1`..`6`: Integer mode ID (Vehicle dependent).
- `FLTMODE_CH`: RC channel number.
- `FLTMODE_GCSBLOCK`: Prevents the Ground Station from changing modes while the RC transmitter is in a specific state.

Integration Guide

1. **Configure Transmitter:** Use "Mixes" to map a 3-position and 2-position switch to a single channel to achieve 6 positions.
2. **Assign Modes:** In the GCS "Flight Modes" screen, select your desired modes for each position.
3. **Verify:** Toggle the switches and ensure the active mode changes in the HUD.

Developer Notes

- **Library:** `AP_Vehicle` / `libraries/AP_Common`.
- **IDs:** Mode IDs are NOT the same across vehicles (e.g., Copter Mode 0 is Stabilize, but Plane Mode 0 is Manual).

FLTMODE1 (Flight Mode 1)

DEFAULT: 0

Flight Mode 1 (FLTMODE1)

Description

Defines the flight mode engaged when the FLTMODE_CH is at its lowest position.

FLTMODE2 (Flight Mode 2)

DEFAULT: (uint8_t	RANGE: null

Flight Mode 2

Note: This parameter functions identically to FLTMODE1.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





FLTMODE3 (Flight Mode 3)

DEFAULT: (uint8_t	RANGE: null

Flight Mode 3

Note: This parameter functions identically to FLTMODE1.

FLTMODE4 (Flight Mode 4)

DEFAULT: (uint8_t	RANGE: null

Flight Mode 4

Note: This parameter functions identically to FLTMODE1.

FLTMODE5 (Flight Mode 5)

DEFAULT: (uint8_t	RANGE: null

Flight Mode 5

Note: This parameter functions identically to FLTMODE1.

FLTMODE6 (Flight Mode 6)

DEFAULT: (uint8_t	RANGE: null

Flight Mode 6

Note: This parameter functions identically to FLTMODE1.

FLTMODE_CH (Flight mode channel)

DEFAULT: 8	RANGE: 1 16

Flight mode channel (FLTMODE_CH)

Description

Specifies which RC channel on your transmitter is used to switch between the 6 primary flight modes.

- **Default Value:** 8 (Plane), 5 (Copter)

FLTMODE_GCSBLOCK (Flight mode GCS block)

DEFAULT: 0

Flight mode GCS block (FLTMODE_GCSBLOCK)

Description

A safety feature that prevents specific flight modes from being engaged via a ground station command. This ensures that certain critical modes can only be entered using the physical RC transmitter sticks.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **FLTR** parameter group (specifically `FLTR_RNG` or `BARO_FLTR_RNG`) defines the "Sanity Check" range for the barometer filter.

Key Concepts

1. Filter Range (`FLTR_RNG`)

This parameter sets the maximum allowable change in altitude reading between successive samples that the filter will accept.

- **Purpose:** Rejects extreme, non-physical pressure spikes (e.g., from an electrical glitch or a door slamming in an indoor environment) before they are passed to the altitude controller.

Parameter Breakdown

- `FLTR_RNG`: Range in meters.

Developer Notes

- **Library:** `libraries/AP_Baro`.
- **Relationship:** Part of the robustness layer of the `AP_Baro` library.

FLTR_RNG (Range in which sample is accepted)

DEFAULT: 0	RANGE: 0 100	UNITS: %

Range in which sample is accepted (FLTR_RNG)

Description

A noise filter that rejects barometer samples that deviate too far from the rolling average.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **FMT** parameter group (specifically `FMT_VER` or `FORMAT_VERSION`) is an internal system parameter for **AP_Periph** (DroneCAN) hardware.

Key Concepts

1. Storage Versioning

As firmware evolves, the way parameters are stored in Flash memory might change.

- `FMT_VER`: An integer that tells the firmware which "layout" of parameters is currently on the chip.
- **Mismatch Behavior:** If you flash new firmware that expects Format 2, but the chip has Format 1, the firmware will detect the mismatch and perform a "Factory Reset" (EEPROM wipe) to ensure data integrity.

Parameter Breakdown

- `FMT_VER`: Internal format version number.

Integration Guide

- **Warning: Do not edit manually.** Changing this value will likely wipe all your settings on the next reboot.

Developer Notes

- **Library:** `Tools/AP_Periph`.

FMT_VER (ICE Parameter Format Version)

```
DEFAULT: 0
```

ICE Parameter Format Version (FMT_VER)

Description

An internal maintenance parameter used by ArduPilot to handle updates and migrations of the internal combustion engine (ICE) settings.

- **Hidden: Not typically intended for user adjustment.**

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





FOLLOW MODE CONFIGURATION (FOLL)

Overview

The **FOLL** parameter group configures the **Follow Mode**. This allows the vehicle to autonomously follow a "Lead" target (another MAVLink-enabled vehicle or a Ground Control Station).

The follower receives the Lead's position via telemetry (MAVLink `GLOBAL_POSITION_INT` or `FOLLOW_TARGET` messages) and calculates the necessary maneuvers to maintain a constant offset.

Key Concepts

1. Relative Offsets (`FOLL_OFS_...`)

Defines the distance (meters) the follower should keep from the leader.

- **X (Forward):** Positive = ahead of leader, Negative = behind.
- **Y (Right):** Positive = to the right, Negative = to the left.
- **Z (Down):** Positive = below the leader, Negative = above.

2. Offsets Type (`FOLL_OFS_TYPE`)

How are the offsets interpreted?

- **0 (North-East-Down):** Follower maintains fixed cardinal directions (e.g., always 5m North).
- **1 (Relative to Heading):** Follower maintains position relative to the Leader's nose (e.g., always 5m behind the tail).

3. Target Identification (`FOLL_SYSID`)

Specifies the MAVLink System ID of the target to follow.

- **0:** Follow the first GCS that sends a follow command.

Parameter Breakdown

- `FOLL_ENABLE`: Master switch.
- `FOLL_DIST_MAX`: Maximum distance (meters) to the leader before giving up (failsafe).
- `FOLL_POS_P`: Proportional gain for position correction.
- `FOLL_ACCEL_D`: Damping for acceleration (smoothes movements).

Integration Guide

1. **Requirement:** A stable telemetry link between the two vehicles or vehicle-to-GCS.
2. **Enable:** Set `FOLL_ENABLE = 1`.
3. **Mode:** Switch the follower vehicle to `FOLLOW` flight mode.
4. **Initiate:** Use the GCS to send a "Follow Me" command.

Developer Notes

- **Library:** `libraries/AP_Follow`.
- **Precision:** Follow performance depends heavily on the update rate of the telemetry link. 10Hz+ is recommended for close formation flight.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





FOLL_ACCEL_D (Follow Vertical Acceleration Limit)

DEFAULT: 0	RANGE: 0 50	UNITS: m/s^2

Follow Vertical Acceleration Limit (FOLL_ACCEL_D)

Description

This parameter sets the maximum vertical acceleration the drone is allowed to use while tracking a target in Follow mode.

By limiting the acceleration, you ensure that the drone transitions smoothly between climbing and descending, preventing sudden throttle spikes that could waste battery or destabilize a camera gimbal.

The Mathematics

The input shaper constrains the second derivative of position (Z):

$$|\frac{d^2Z}{dt^2}| \leq FOLL_ACCEL_D$$

This acts as a "ceiling" on the throttle demand logic. Even if the target suddenly drops 10 meters, the drone will only accelerate downwards at this specified rate until it reaches its maximum descent speed.

The Engineer's View

In `AP_Follow`, this parameter is part of the kinematic shaping configuration passed to the position controller.

- **0:** Disabled. The system uses the vehicle's standard vertical acceleration limits (e.g., `PILOT_ACCEL_Z`).
- **Non-Zero:** Overrides the default limits specifically for Follow mode. This allows you to have a very aggressive "Sport Mode" for manual flying but a gentle, cinematic behavior for Follow mode.

Tuning & Behavior

- **Default Value:** 0 (Disabled).
- **Range:** 0 to 50 m/s^2.
- **Smooth Filming:** Set to **1 or 2** m/s^2.
- **Aggressive Tracking:** Set to **5+** m/s^2 or leave at 0.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY

FOLL_ALT_TYPE (Follow Altitude Type)

DEFAULT: 1	RANGE: 0 1	UNITS: Option

Follow Altitude Type (FOLL_ALT_TYPE)

Description

This parameter defines the "Vertical Reference Frame" used when calculating the target's position. It tells the drone whether it should try to match the target's absolute height above sea level or its relative height above the launch point.

This is critical for terrain following. If the target climbs a hill, you need to know if the drone should also climb (Relative) or stay at a fixed barometric altitude (Absolute).

The Mathematics

The parameter selects the altitude source mode:

- **0: Absolute (AMSL).** The drone targets the raw GPS Mean Sea Level altitude reported by the lead vehicle.

$$Alt_target = Alt_Lead_AMSL + Offset_Z$$

- **1: Relative (Above Home).** The drone targets an altitude relative to its own home position.

$$Alt_target = (Alt_Lead_AMSL - Alt_Home) + Offset_Z$$

The Engineer's View

In `AP_Follow::handle_global_position_int_message()` (`libraries/AP_Follow/AP_Follow.cpp`):

1. The code receives a `GLOBAL_POSITION_INT` MAVLink message.
2. If `_alt_type == 1` (Relative), it converts the packet's `relative_alt` field into an `ABOVE_HOME` frame location.
3. If `_alt_type == 0` (Absolute), it uses the packet's raw `alt` (AMSL) field directly.

Tuning & Behavior

- **Default Value:** 1 (Relative).
- **Rover/Boat:** Usually set to **1**. If the boat is at "0m" relative to the water, the drone will fly "0m + Offset" above the water.
- **Aircraft:** Set to **0** if you want the follower to stay at a specific barometric altitude regardless of where the leader goes (e.g., for formation flight at a safe ATC-cleared altitude).

FOLL_DIST_MAX (Follow Max Distance)

DEFAULT: 100	RANGE: 1 1000	UNITS: m

Follow Max Distance (FOLL_DIST_MAX)

Description

`FOLL_DIST_MAX` is the "Leash" length.

If the target vehicle speeds away and the distance between your drone and the target exceeds this value, the autopilot will stop chasing and switch to a safe mode (usually Loiter). This prevents the drone from blindly flying over the horizon if it loses its connection to the target.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





FOLL_ENABLE (Follow Mode Enable)

DEFAULT: 0	RANGE: 0 1

Follow Mode Enable (FOLL_ENABLE)

Description

`FOLL_ENABLE` activates the dedicated control logic for chasing a target.

When enabled, the drone can receive position data from another vehicle (Lead drone) or a GCS-enabled device (like a phone in your pocket) and automatically fly to maintain a specific distance and orientation relative to that target.

- **0: Disabled.**
- **1: Enabled.**

Tuning & Behavior

- **Reboot Required:** Yes.
- **Requirement:** You must also set the Target's MAVLink System ID in FOLL_SYSID.

FOLL_JERK_D (Follow Vertical Jerk Limit)

DEFAULT: 0	RANGE: 0 50	UNITS: m/s^3

Follow Vertical Jerk Limit (FOLL_JERK_D)

Description

This parameter limits the "Jerk" (rate of change of acceleration) for vertical movements in Follow mode. It is part of the kinematic input shaper, which smooths out the target's movements before they are passed to the flight controller.

A lower jerk limit makes the drone's altitude changes feel smoother and more "organic," reducing mechanical stress and camera shake. A higher limit allows the drone to react more violently to sudden altitude changes by the target.

The Mathematics

The input shaper limits the third derivative of position (Z):

$$|\frac{d^3Z}{dt^3}| \leq FOLL_JERK_D$$

This effectively puts a "ramp" on the acceleration command. Instead of instantly demanding 2G climb, the system ramps up to that acceleration over a period defined by this limit.

The Engineer's View

In `AP_Follow`, this parameter is passed to the `AC_InputManager` or similar kinematic shaping libraries used by the vehicle's position controller.

- **0:** Disabled. The system uses the vehicle's default maximum jerk limits (e.g., `PILOT_JERK`).
- **Non-Zero:** Overrides the default jerk limit specifically for Follow mode operations.

Tuning & Behavior

- **Default Value:** 0 (Disabled/Use Vehicle Defaults).
- **Filming:** Set to a low value (e.g., 5 m/s^2/s) for ultra-smooth cinematic following.
- **Racing/Tracking:** Set higher or leave at 0 for maximum responsiveness.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





FOLL_OFS (Follow offsets (Vector))

DEFAULT: 0	RANGE: -100 100	UNITS: m

Follow offsets (FOLL_OFS)

Description

This vector parameter defines the positional offset of the follower vehicle relative to the lead vehicle. It allows the follower to maintain a specific formation (e.g., flying 5 meters behind and 2 meters above the leader).

The Mathematics

The offsets are applied in the NED (North-East-Down) or Forward-Right-Down frame, depending on `FOLL_OFS_TYPE`.
$$ \vec{P}*{target} = |*vec{P}*{lead} + \begin{bmatrix} \text{FOLL_OFS_X} \ \text{FOLL_OFS_Y} \ \text{FOLL_OFS_Z} \end{bmatrix} $$

The Engineer's View

Defined in `libraries/AP_Follow/AP_Follow.cpp` as a `Vector3f` parameter `_offset`. This creates three sub-parameters in the GCS: `FOLL_OFS_X`, `FOLL_OFS_Y`, and `FOLL_OFS_Z`.

- **X:** Forward/North offset.
- **Y:** Right/East offset.
- **Z:** Down offset (Positive = Below, Negative = Above).

Tuning & Behavior

- **FOLL_OFS_X:** +ve is ahead, -ve is behind.
- **FOLL_OFS_Y:** +ve is right, -ve is left.
- **FOLL_OFS_Z:** +ve is below, -ve is above.

FOLL_OFS_TYPE (Follow Offset Type)

DEFAULT: 0	RANGE: 0 1

Follow Offset Type (FOLL_OFS_TYPE)

Description

`FOLL_OFS_TYPE` determines how the drone calculates where to sit in relation to the target.

- **0: North-East-Down.** The drone stays at a fixed compass offset (e.g., 5m North and 5m East). Even if the target turns, the drone stays on the North side.
- **1: Relative to Target.** The drone stays at a fixed offset relative to the target's nose (e.g., 5m behind and 5m to the right). If the target turns, the drone "Orbits" around to stay in the same spot relative to the target's view. (Best for filming vehicles).

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





FOLL_OPTIONS (Follow options)

DEFAULT: 0	RANGE: null	UNITS: Bitmask

Follow options (FOLL_OPTIONS)

Description

This parameter allows the user to enable specialized behaviors for Follow mode via a bitmask. Currently, it primarily controls how camera gimbals interact with the target tracking system.

The Mathematics

The parameter is a bitmask (B):

- **Bit 0 (Value 1): Mount Follows lead vehicle on mode enter.**
  When this bit is set and the vehicle enters Follow mode, the autopilot will automatically command any connected gimbal (Mount) to point at the Lead Vehicle's calculated position.

The Engineer's View

In `AP_Follow` (and consumed by vehicle-specific code like `mode_follow.cpp`), this parameter maps to `_options`.

- **Mount Tracking:** If Bit 0 is set, the Follow library or the main vehicle code will set the Gimbal's Region of Interest (ROI) to the target's coordinates. This creates a "Cinematic Follow" effect where the drone chases the target while keeping it centered in the camera frame, regardless of the drone's actual orientation.

Tuning & Behavior

- **Default Value:** 0 (Disabled).
- **Bit 0 (1):** Enable this for filming applications. Note that this overrides manual gimbal control while in Follow mode.

FOLL_POS_P (Follow position error P gain)

DEFAULT: 0.1	RANGE: 0.01 1.0	UNITS: Gain

Follow position error P gain (FOLL_POS_P)

Description

This parameter tunes the aggressiveness of the drone when chasing a moving target. It is the Proportional gain of the position controller used specifically in Follow mode.

If the drone falls behind the target, this gain determines how hard it accelerates to catch up.

- **High P:** The drone reacts instantly to target movement but may twitch or overshoot if the target stops suddenly.
- **Low P:** The drone reacts smoothly but may lag behind the target, effectively "rubber-banding" as the distance increases.

The Mathematics

The controller calculates the position error (E_pos) between the drone and the target (plus offsets). The desired velocity adjustment is proportional to this error:

$$V_correction = FOLL_POS_P \times E_pos$$

This velocity demand is then passed to the vehicle's standard velocity controller (which has its own PIDs).

The Engineer's View

In `AP_Follow.cpp`, this is defined as `_p_pos`.
The Follow library does not control the motors directly. Instead, it calculates a 3D target position and feeds it into the vehicle's position controller (e.g., `AC_PosControl` for Copters). This P-gain works "upstream" of the main flight controller PIDs, shaping the demand signal.

Tuning & Behavior

- **Default Value:** 0.1.
- **Range:** 0.01 to 1.0.
- **Recommendation:** Start with the default. If the drone lags too far behind a fast-moving target, increase by 0.05 increments.
- **Overshoot:** If the drone flies past the target when the target stops, decrease this value.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





FOLL_SYSID (Follow Target System ID)

DEFAULT: 0	RANGE: 0 255

Follow Target System ID (FOLL_SYSID)

Description

`FOLL_SYSID` identifies who your drone is supposed to be chasing.

Every device on a MAVLink network (Drones, Laptops, Phones) has a unique ID number. By setting this parameter, you tell your drone to ignore other traffic and only follow the position updates coming from the specific ID entered here.

- **Default:** 0 (Follow the first target detected).
- **Recommendation:** Set this explicitly to your phone or lead drone's ID for safety and reliability.

FOLL_TIMEOUT (Follow Mode Data Timeout)

DEFAULT: 3000	RANGE: 0 10000	UNITS: ms

Follow Mode Data Timeout (FOLL_TIMEOUT)

Description

This parameter is intended to set the failsafe timeout for Follow Mode. If the drone stops receiving position updates from the target vehicle for this duration, it will stop tracking and potentially switch flight modes or loiter in place.

CRITICAL NOTE: As of the current codebase analysis, this parameter does **not exist** in the `AP_Follow` parameter table (`var_info`). The timeout is currently hardcoded to **3000ms (3 seconds)** via the macro `AP_FOLLOW_TIMEOUT_MS`.

The Mathematics

The logic checks the age of the last valid packet:

$$T_now - T_last_update > 3000ms$$

If this condition is met, `have_target()` returns `false`, causing the vehicle to stop updating its target position.

The Engineer's View

In `AP_Follow.cpp`, line 29:

```
#define AP_FOLLOW_TIMEOUT_MS    3000    // position estimate timeout after 1 second (comment says 1s, code is 3s)
```

The parameter `FOLL_TIMEOUT` appears in some documentation scaffolds but is not exposed in the `var_info` struct for user configuration. It is effectively a compile-time constant.

Tuning & Behavior

- **Default Value:** 3000 (Hardcoded).
- **Behavior:** If the MAVLink stream from the lead vehicle is interrupted (e.g., telemetry loss), the follower will continue on its last known trajectory for 3 seconds before giving up.
- **Correction:** Do not attempt to change `FOLL_TIMEOUT` in your Ground Control Station; it will likely not be found or will have no effect.

FOLL_YAW_BEHAVE (Follow Yaw Behavior)

DEFAULT: 1	RANGE: 0 3

Follow Yaw Behavior (FOLL_YAW_BEHAVE)

Description

`FOLL_YAW_BEHAVE` defines what the drone points its "Face" (camera) at while it is chasing a target.

- **0: None.** The drone keeps its current heading and does not automatically rotate.
- **1: Face Target (Default).** The drone always turns to point its nose directly at the person or object it is following. Good for filming yourself.
- **2: Face Next Waypoint.**
- **3: Face Direction of Travel.** The drone points its nose where it is going, like a real airplane.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **FORMAT** parameter group (specifically `FORMAT_VERSION`) is an internal system identifier used by **AP_Periph** (DroneCAN) devices.

Key Concepts

1. Storage Integrity

As DroneCAN peripheral firmware is updated, the way parameters are laid out in the device's internal memory (Flash/EEPROM) may change.

- **Version Check:** At boot, the firmware checks the `FORMAT_VERSION` stored in memory.
- **Wipe:** If the stored version does not match the version expected by the firmware, the device will automatically perform a factory reset of all parameters to prevent data corruption.

Parameter Breakdown

- `FORMAT_VERSION`: An integer representing the memory layout version.

Developer Notes

- **Warning: Do not modify.** Changing this value will cause the device to lose all calibration and configuration data on the next reboot.

FORMAT_VERSION (Eeprom format version number)

`DEFAULT: 0`

Eeprom format version number (FORMAT_VERSION)

Description

This is an internal system parameter used by the DroneCAN peripheral firmware to track the structure of its parameter storage (EEPROM/Flash). It ensures that if the firmware is updated to a version with a different storage layout, the system can detect the mismatch and perform a factory reset to prevent data corruption.

The Mathematics

Integer version comparison.

The Engineer's View

Defined in `Tools/AP_Periph/Parameters.cpp`.

Tuning & Behavior

- **Read Only:** Do not edit. Modifying this will likely cause a factory reset of all peripheral parameters on the next boot.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





FRAME CONFIGURATION (FRAME)

Overview

The **FRAME** parameter group defines the physical structure and motor layout of the vehicle. This is one of the first and most critical steps in configuring a new drone.

The frame settings tell the autopilot:

1. **Class:** What *type* of vehicle is it? (Quad, Hexa, Octa, Heli, etc.)
2. **Type:** How are the motors *arranged*? (X, Plus, V, H, etc.)
3. **Config:** (Sub) Specific motor mixer matrices for underwater vehicles.

Key Concepts

1. Frame Class (`FRAME_CLASS`)

Sets the number of motors and general physics model.

- **1:** Quad.
- **2:** Hexa.
- **3:** Octa.
- **...**

2. Frame Type (`FRAME_TYPE`)

Sets the rotation of the motor arms.

- **1:** X-Frame (Standard).
- **0:** Plus-Frame (Motors on the nose/tail/sides).

3. Sub Frame Config (`FRAME_CONFIG`)

(Specific to ArduSub) Selects from predefined thruster layouts like "Vectored" or "Vectored-6DOF."

Parameter Breakdown

- `FRAME_CLASS`: Vehicle architecture.
- `FRAME_TYPE`: Geometric layout.
- `FRAME_CONFIG`: (Sub only) Thruster mixer.

Integration Guide

- **Setup:** Use the "Frame Setup" screen in your GCS. Changing these values usually requires a reboot to load the new mixer matrix.
- **Warning:** If you have an X-frame but set it to Plus-frame, the drone will flip immediately on takeoff.

Developer Notes

- **Library:** `libraries/AP_Motors`.
- **Mixers:** Each class/type combination loads a specific static matrix that defines how Roll/Pitch/Yaw/Throttle maps to each motor PWM.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





FRAME_CLASS (Frame Class)

DEFAULT: 0	RANGE: 0 15

Frame Class (FRAME_CLASS)

Description

Defines the overall architecture of the vehicle (e.g., Quad, Hexa, Octa, Heli, etc.). This setting determines how many motors are used and how they are fundamentally controlled.

Tuning & Behavior

- **Default Value:** 0 (Undefined)
- **Common Values (Copter):** 1:Quad, 2:Hexa, 3:Octa, 6:Heli
- **Common Values (Rover):** 1:Ground Vehicle, 2:Boat
- **Requires reboot to take effect.**

FRAME_CONFIG (Frame configuration)

DEFAULT: 1	RANGE: 0 7

Frame configuration (FRAME_CONFIG)

Description

This parameter defines the physical arrangement of thrusters on the ROV. It selects the "Mixer Matrix" that translates pilot commands (Forward, Strafe, Vertical, Yaw, Pitch, Roll) into individual motor PWM outputs.

The Mathematics

$$ \vec{\omega}*{|text{motor}} = |mathbf{M}*{\text{mixer}} \times \vec{u}_{\text{control}} $$

The Engineer's View

Defined in `ArduSub/Parameters.cpp`. Values map to `AP_Motors6DOF::sub_frame_t`:

- **0:** BlueROV1
- **1:** Vectored (BlueROV2 Default)
- **2:** Vectored 6DOF
- **3:** Vectored 6DOF 90deg
- **4:** SimpleROV-3
- **5:** SimpleROV-4
- **6:** SimpleROV-5
- **7:** Custom

Tuning & Behavior

- **Critical:** Must be set correctly before the first dive. An incorrect setting will cause the ROV to spin uncontrollably or move in wrong directions.
- **Reboot Required:** Changes usually require a reboot to take effect properly.

FRAME_TYPE (Frame Type)

DEFAULT: 1	RANGE: 0 5

Frame Type (FRAME_TYPE)

Description

Defines the specific layout of the motors relative to the nose of the aircraft (e.g., Plus vs. X configuration).

Tuning & Behavior

- **Default Value:** 1 (X)
- **Common Values:** 0:Plus, 1:X, 2:V, 3:H, 10:Y6B
- **Requires reboot to take effect.**

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **FREQ** parameter group (often appearing as `FFT_MAXHZ` or `MOT_PWM_FREQ`) manages various frequency-domain settings across the autopilot.

Key Concepts

1. Vibration Tracking (`FFT_...HZ`)

Used by the in-flight FFT engine to bracket the expected motor noise.

- `MAXHZ`: The upper limit of the frequency sweep.

2. Motor PWM Frequency (`MOT_PWM_FREQ`)

(Often grouped here in documentation) Sets the update rate for the ESC signal.

- **490Hz:** Standard for most PWM ESCs.
- **16kHz - 24kHz:** Used for brushed motors or ultra-high-speed ESCs to reduce audible whine.

Parameter Breakdown

- `FFT_MAXHZ`: Upper bound for vibration analysis.
- `MOT_PWM_FREQ`: Output update rate.

Developer Notes

- **Library:** `libraries/AP_InertialSensor` (for FFT) or `libraries/AP_Motors` (for PWM).

FREQ_HZ (Fins frequency)

DEFAULT: 3	RANGE: 1 10	UNITS: Hz

Fins frequency (FREQ_HZ)

Description

Defines the flapping frequency for the propulsion fins on a Blimp vehicle.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **FRSKY** parameter group configures the telemetry output for **FrSky** radio systems (SmartPort and D-Series). This allows you to see live battery voltage, GPS position, and flight mode directly on your FrSky transmitter screen.

Key Concepts

1. Telemetry Options (`FRSKY_OPTIONS`)

A bitmask for advanced features.

- **Bit 0:** Enable Passthrough (Requires OpenTX/EdgeTX and a Lua script like Yaapu). This is the modern, high-bandwidth method.
- **Bit 1:** Enable individual sensor IDs.

2. D-Series vs SmartPort

- **D-Series:** Legacy (Hub protocol).
- **SmartPort (S.Port):** Modern, bi-directional protocol used by X-series receivers (X8R, R-XSR).

Parameter Breakdown

- `FRSKY_D_RATE` : Update rate for D-Series telemetry.
- `FRSKY_OPTIONS` : Bitmask for Passthrough and other features.

Integration Guide

1. **Hardware:** Connect the S.Port pin of your receiver to a UART TX pin. (Note: FrSky S.Port is inverted; many boards have a dedicated "FrSky" port or require `SERIALx_OPTIONS` to enable inversion).
2. **Protocol:** Set `SERIALx_PROTOCOL = 10` (Passthrough) or `4` (SmartPort).
3. **Lua Script:** If using Passthrough, download and install the **Yaapu Telemetry Script** on your Taranis/Horus/TX16S.

Developer Notes

- **Library:** `libraries/AP_Frsky_Telem`.

FRSKY_DNLINK1_ID (First downlink sensor id)

DEFAULT: 20	RANGE: 7 26

First downlink sensor id (FRSKY_DNLINK1_ID)

Description

Provides an additional SPort sensor ID for expanded telemetry reporting capacity.

- **Default Value:** 20

FRSKY_DNLINK2_ID (Second downlink sensor id)

DEFAULT: 7	RANGE: 7 26

Second downlink sensor id (FRSKY_DNLINK2_ID)

Description

Provides a third SPort sensor ID for expanded telemetry reporting capacity.

- **Default Value:** 7

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





FRSKY_DNLINK_ID (Default downlink sensor id)

DEFAULT: 27	RANGE: 7 27

Default downlink sensor id (FRSKY_DNLINK_ID)

Description

Defines the logical sensor ID that ArduPilot uses to transmit data over the FrSky SmartPort (SPort) downlink.

- **Default Value:** 27

FRSKY_OPTIONS (FRSky Telemetry Options)

DEFAULT: 0

FRSky Telemetry Options (FRSKY_OPTIONS)

Description

Configuration bitmask for tailoring the FrSky telemetry output.

- **Bit 0: EnableAirspeedAndGroundspeed (Force both to be sent)**

FRSKY_UPLINK_ID (Uplink sensor id)

DEFAULT: 13	RANGE: 7 26

Uplink sensor id (FRSKY_UPLINK_ID)

Description

Defines the logical sensor ID ArduPilot listens to for incoming data from the transmitter over the bidirectional SPort link.

- **Default Value:** 13

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **FS** parameter group configures the autopilot's **Failsafe** systems. These are automatic emergency protocols that trigger when something goes wrong (e.g., you lose radio contact or the battery is critically low).

Failsafes are the primary defense against "Fly-Aways."

Key Concepts

1. Triggers

The autopilot monitors several conditions:

- **Radio (RC):** Triggered if the signal from your transmitter is lost.
- **Battery:** Triggered based on `BATT_LOW_VOLT` or `BATT_LOW_MAH`.
- **GCS:** Triggered if the telemetry link to the Ground Control Station is lost (useful for long-range autonomous missions).
- **EKF:** Triggered if the state estimator becomes unhealthy (e.g., GPS glitch).

2. Actions (`FS_..._ACTION`)

What the drone does when a failsafe triggers:

- **0 (None):** Does nothing (Dangerous!).
- **1 (RTL):** Returns to the launch point and lands.
- **2 (Land):** Lands immediately at its current position.
- **3 (SmartRTL):** Retraces its path home.

Parameter Breakdown

- `FS_THR_ENABLE`: Enable/Disable Radio Failsafe.
- `FS_THR_VALUE`: The PWM threshold below which the autopilot assumes the radio is disconnected (usually ~950us).
- `FS_BATT_ENABLE`: Action for low battery.
- `FS_GCS_ENABL`: Action for GCS link loss.

Integration Guide

Testing Radio Failsafe

1. **Remove Props.**
2. Arm the vehicle in a safe mode (e.g., Loiter).
3. **Turn off your RC Transmitter.**
4. Verify the HUD switches to "Failsafe" and the vehicle mode changes to "RTL" or "Land."

Developer Notes

- **Library:** `libraries/AP_Vehicle` / `AP_Failsafe`.
- **Priority:** If multiple failsafes trigger, the system prioritizes the most conservative action (e.g., Land over RTL).

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





FS_ACTION (Failsafe Action (Rover))

DEFAULT: 0	RANGE: 0 4

Failsafe Action (FS_ACTION)

Description

`FS_ACTION` tells your Rover what to do if it loses its connection to the pilot (Radio or GCS).

- **0: Nothing.** The Rover continues whatever it was doing (DANGEROUS).
- **1: RTL.** The Rover returns to its home location.
- **2: Hold.** The Rover stops and waits for the signal to return.
- **3: SmartRTL.** The Rover retraces its path back to home.
- **4: SmartRTL or RTL.**

Tuning & Behavior

- **Default:** 0.
- **Recommendation:** Set to **1 (RTL)** or **2 (Hold)** for safety.
- **Dependencies:** Requires a valid Home position if using RTL.

FS_CRASH_CHECK (Crash Check Enable)

DEFAULT: 1	RANGE: 0 1

Crash Check Enable (FS_CRASH_CHECK)

Description

`FS_CRASH_CHECK` is a major safety feature for multirotors.

It uses the IMU and motor data to detect if the drone has hit an obstacle or flipped on its side. If a crash is detected (e.g., the drone is at an extreme angle and the motors are working hard but not moving it), the autopilot will instantly disarm the motors to prevent further damage or injury from spinning propellers.

- **0: Disabled.** (Dangerous).
- **1: Enabled (Default).**

Tuning & Behavior

- **Recommendation:** Always leave this enabled. It is one of the most effective ways to prevent "Fly-aways" or ESC fires after a minor collision.

FS_DR_ENABLE (Dead Reckoning Failsafe Enable)

DEFAULT: 1	RANGE: 0 1

Dead Reckoning Failsafe Enable (FS_DR_ENABLE)

Description

`FS_DR_ENABLE` allows the drone to try and come home even if its GPS "Goes Blind."

Normally, if GPS is lost, the drone can only land vertically. With Dead Reckoning (DR), the autopilot uses the compass, IMU, and air data to estimate its position and "Guess" its way back to the Home point. It isn't perfect, but it's often enough to get the drone back into transmitter range.

- **0: Disabled.** The drone will Land instantly if GPS is lost during a mission.
- **1: Enabled (Default).** The drone will try to fly toward Home for a short time using DR.

Tuning & Behavior

- **Requirement:** Requires a well-calibrated compass and a good wind estimation (from a pitot tube or EKF learning).

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





FS_DR_TIMEOUT (Dead Reckoning Failsafe Timeout)

DEFAULT: 30	RANGE: 0 120	UNITS: s

Dead Reckoning Failsafe Timeout (FS_DR_TIMEOUT)

Description

`FS_DR_TIMEOUT` defines how long you trust your drone to "Guess" its way home.

Dead reckoning becomes more inaccurate the longer it runs. If the drone hasn't found its home or regained GPS within this many seconds, it will stop its horizontal movement and land vertically to prevent drifting miles away into unknown territory.

FS_EKF_ACTION (EKF Failsafe Action)

DEFAULT: 1	RANGE: 1 3

EKF Failsafe Action (FS_EKF_ACTION)

Description

`FS_EKF_ACTION` determines how the drone reacts if its "Internal Map" becomes unreliable (see FS_EKF_THRESH).

- **1: Land (Default).** The drone will immediately perform a controlled descent and land. This is the safest option if the GPS or Compass data is bad.
- **2: AltHold.** The drone will switch to Altitude Hold mode. This stops it from following a bad GPS position but allows the pilot to take manual control and fly by sight.
- **3: Land (Strict).** Forces a landing even if the drone is currently in a manual mode like Stabilize.

FS_EKF_FILT (EKF Failsafe Filter Frequency)

DEFAULT: 0.5	RANGE: 0.1 5.0	UNITS: Hz

EKF Failsafe Filter Frequency (FS_EKF_FILT)

Description

`FS_EKF_FILT` determines how "Stable" the failsafe trigger is.

The EKF variance is constantly jumping up and down. This parameter smooths those jumps out.

- **Higher Value:** Faster reaction. The drone will failsafe immediately if a sensor glitches.
- **Lower Value:** Slower, more robust reaction. The drone will ignore tiny "Spikes" in variance but will take longer to realize if a sensor has truly failed.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





FS_EKF_THRESH (EKF Failsafe Threshold)

DEFAULT: 0.8	RANGE: 0.6 1.0

EKF Failsafe Threshold (FS_EKF_THRESH)

Description

`FS_EKF_THRESH` is the "Trust Limit" for your drone's brain.

The EKF merges data from the GPS, Compass, and IMU. It is constantly checking if these sensors agree with each other. If they don't (e.g. because the compass is glitching or the GPS has high interference), the EKF "Variance" increases.

- **Higher Value (1.0):** More tolerant. The drone will keep flying even if the sensors are slightly inconsistent.
- **Lower Value (0.6):** More sensitive. The drone will trigger a failsafe early if it detects any sensor disagreement.

Tuning & Behavior

- **Default Value:** 0.8.
- **Recommendation:** Leave at **0.8**. If you are flying in an area with a lot of magnetic interference (near steel buildings), you might need to increase this slightly, but be aware that the drone's position will be less accurate.
- **Action:** When this threshold is exceeded, the drone will trigger the action set in FS_EKF_ACTION (usually Landing or switching to a non-GPS mode).

FS_GCS_ENABL (GCS failsafe enable)

DEFAULT: GCS_FAILSAFE_OFF	RANGE: null

GCS failsafe enable

Note: This parameter functions identically to FS_GCS_ENABLE.

FS_GCS_ENABLE (GCS Failsafe Enable)

DEFAULT: 0	RANGE: 0 5

GCS Failsafe Enable (FS_GCS_ENABLE)

Description

`FS_GCS_ENABLE` defines what the drone does if it loses its MAVLink telemetry connection.

This is primarily used for long-range missions where the drone is controlled via a laptop rather than a traditional RC remote.

- **0: Disabled (Default).** The drone continues its mission even if the telemetry link is lost.
- **1: Enabled (RTL).** The drone returns home immediately.
- **2: Enabled (Continue in Auto).** If the drone is currently on an autonomous mission, it continues. If not, it returns home.

Tuning & Behavior

- **Safety:** Ensure you have a valid Home position and no obstacles in the RTL path before enabling this.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





FS_GCS_TIMEOUT (GCS Failsafe Timeout)

DEFAULT: 5	RANGE: 2 120	UNITS: s

GCS Failsafe Timeout (FS_GCS_TIMEOUT)

Description

`FS_GCS_TIMEOUT` sets the "Grace Period" for your telemetry link.

If you are using a laptop or tablet to control your drone (MAVLink control), this parameter defines how many seconds the autopilot will wait after losing the connection before it decides the link is broken and starts its failsafe action (usually RTL).

Tuning & Behavior

- **Default Value:** 5 seconds.
- **Recommendation:** Leave at **5** for most operations. If you are flying in an area with known signal gaps, you can increase this slightly, but be aware that the drone will continue its last commanded action until the timer expires.

FS_LEAK_ENABLE (Leak Failsafe Enable)

DEFAULT: 0	RANGE: 0 3

Leak Failsafe Enable (FS_LEAK_ENABLE)

Description

`FS_LEAK_ENABLE` defines the emergency response for water-based vehicles (ROVs and Boats) if their internal leak sensors are triggered.

- **0: Warn Only.** Displays a warning on the GCS but takes no autonomous action.
- **1: Surface.** (ArduSub). Commands the ROV to climb to the surface immediately.
- **2: Hold.** Stops all movement and maintains current position.

Tuning & Behavior

- **Default:** 0 (Warning).
- **Recommendation:** Set to **1 (Surface)** for ROVs to ensure the vehicle is recoverable if the hull is compromised.
- **Prerequisite:** Requires at least one leak sensor (LEAK1_TYPE) to be enabled.

FS_LONG_ACTN (Long Failsafe Action (Plane))

DEFAULT: 0	RANGE: 0 4

Long Failsafe Action (FS_LONG_ACTN)

Description

`FS_LONG_ACTN` defines the "Plan B" for a sustained radio link failure. While FS_SHORT_ACTN handles brief glitches, the "Long" action is triggered after the connection has been dead for several seconds (see FS_LONG_TIMEOUT).

- **0: Continue (Standard).** If in AUTO mode, the plane keeps flying its mission. If in manual, it triggers RTL.
- **1: RTL.** The plane returns home immediately regardless of mode.
- **2: FBWA.** The plane level the wings and holds altitude (Circle mode fallback).
- **3: Parachute.** Deploys the parachute if configured.

Tuning & Behavior

- **Default:** 0.
- **Safety:** For most users, setting this to **1 (RTL)** is the safest choice to ensure the plane comes back if the radio fails.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





FS_LONG_TIMEOUT (Long Failsafe Timeout)

DEFAULT: 5.0	RANGE: 2 120	UNITS: s

Long Failsafe Timeout (FS_LONG_TIMEOUT)

Description

`FS_LONG_TIMEOUT` sets the duration of signal loss required to trigger the FS_LONG_ACTN.

- **Standard Setting:** 5 seconds.
- **Long Range:** If you are flying at the extreme edge of your radio range where "link blips" are common, you may increase this to **10s** or **20s** to avoid unnecessary RTLs.

Tuning & Behavior

- **Default:** 5.0.
- **Warning:** Increasing this value means the drone will continue flying its last known command for longer without pilot input. Use with caution.

FS_OPTIONS (Failsafe Options)

DEFAULT: 16	RANGE: 0 63

Failsafe Options (FS_OPTIONS)

Description

`FS_OPTIONS` allows you to customize exactly when the drone should "Ignore" a radio or GCS failure and keep flying.

- **Bit 0 (1): RC Continue if Auto.** If the drone is on an autonomous mission, it will keep going even if the RC transmitter is turned off.
- **Bit 1 (2): GCS Continue if Auto.** Keeps the mission going if the telemetry link is lost.
- **Bit 4 (16): GCS Continue if Pilot Control.** (Default). If you are using a GCS to control the drone but still have your RC radio connected, losing the GCS link will not trigger a failsafe.

Tuning & Behavior

- **Default Value:** 16.
- **Safety:** Only enable Bits 0 or 1 if you are an expert and have verified that your drone has a fully autonomous landing/safety plan.

FS_PILOT_INPUT (Pilot Input Failsafe Action)

DEFAULT: 0	RANGE: 0 2

Pilot Input Failsafe Action (FS_PILOT_INPUT)

Description

`FS_PILOT_INPUT` defines the vehicle's "Dead Man's Switch" behavior.

This is primarily used for **ArduSub** (underwater ROVs). Since these vehicles are often tethered, a loss of communication can happen. If the flight controller doesn't see any stick movement or heartbeat from the pilot for a certain amount of time, it assumes the pilot is incapacitated or the link is broken.

- **0: Disarm/Disable.** Stops motors immediately.
- **1: Hold.** (Recommended). Maintains current depth and position.
- **2: Surface.** Commanded ascent to the surface.

Tuning & Behavior

- **Default:** 0.
- **Requirement:** Requires FS_PILOT_TIMEOUT to be set.
- **Safety:** For ROVs, **Hold (1)** is usually the safest option to prevent the tether from becoming tangled in a prop while the vehicle "Ghost drifts."

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





FS_PILOT_TIMEOUT (Pilot Input Failsafe Timeout)

DEFAULT: 3.0	RANGE: 0.1 10.0	UNITS: s

Pilot Input Failsafe Timeout (FS_PILOT_TIMEOUT)

Description

`FS_PILOT_TIMEOUT` sets the "Grace Period" for radio or joystick interruptions.

If the autopilot does not receive a new stick command within this many seconds, it triggers the FS_PILOT_INPUT action.

Tuning & Behavior

- **Default:** 3.0 seconds.
- **Recommendation:** Leave at **3.0**. A value too short (e.g. 0.5) might trigger the failsafe if the telemetry link has a brief, normal hiccup. A value too long (e.g. 10.0) might allow the drone to drift too far before reacting to a lost link.

FS_PRESS_ENABLE (Internal Pressure Failsafe Enable)

DEFAULT: 0	RANGE: 0 3

Internal Pressure Failsafe Enable (FS_PRESS_ENABLE)

Description

`FS_PRESS_ENABLE` is a hardware protection feature for **ArduSub** ROVs.

If an ROV has an internal pressure sensor, an increase in pressure can indicate a structural failure or a fast-expanding gas (e.g., from a failing battery). This parameter defines the emergency response.

- **0: Disabled.**
- **1: Warn Only.**
- **2: Surface.**

Tuning & Behavior

- **Default:** 0.
- **Requirement:** Requires a valid internal pressure sensor and a configured FS_PRESS_MAX threshold.

FS_PRESS_MAX (Internal Pressure Failsafe)

DEFAULT: 105000	RANGE: 0 200000	UNITS: Pa

Internal Pressure Failsafe (FS_PRESS_MAX)

Description

`FS_PRESS_MAX` is a specialized safety setting for **ArduSub** ROVs.

It monitors the pressure sensor inside the ROV's dry housing. If the pressure rises above this limit, it usually indicates a leak or an impending structural failure. The autopilot can then trigger an emergency ascent or warning to protect the electronics.

Tuning & Behavior

- **Default:** 105,000 Pa (Standard atmospheric pressure is approx 101,325 Pa).
- **Calibration:** Set this slightly above your normal sealed housing pressure.

FS_SCR (Failsafe OSD Screen (Alias))

DEFAULT: 0	RANGE: 0 4

Failsafe OSD Screen (FS_SCR)

Note: This parameter is an alias for OSD_FS_SCR.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





FS_SHORT_ACTN (Short Failsafe Action (Plane))

DEFAULT: 0	RANGE: 0 2

Short Failsafe Action (FS_SHORT_ACTN)

Description

FS_SHORT_ACTN defines the immediate response to a radio signal loss. It triggers after FS_SHORT_TIMEOUT (typically 1.5 seconds).

- **0: Circle (Standard).** The plane enters Circle mode and holds its current location.
- **1: RTL.** The plane starts returning home immediately.
- **2: FBWA.** Fallback to level flight.

Tuning & Behavior

- **Default:** 0.
- **Best Practice:** Leave at **0 (Circle)** to allow the link a few seconds to recover before committing to a full RTL return.

FS_SHORT_TIMEOUT (Short Failsafe Timeout)

DEFAULT: 1.5	RANGE: 1 5	UNITS: s

Short Failsafe Timeout (FS_SHORT_TIMEOUT)

Description

FS_SHORT_TIMEOUT defines the delay before the autopilot admits the radio signal is truly gone.

It is designed to filter out "Micro-Glitches" caused by local interference or signal multipathing.

- **Standard Setting:** 1.5 seconds.

Tuning & Behavior

- **Default:** 1.5.
- **Usage:** If your link is very "choppy" but you still want manual control, you can increase this to **2.0s** or **3.0s**, but be aware that the drone will keep its last RC inputs during this time.

FS_TEMP_ENABLE (Internal Temperature Failsafe Enable)

DEFAULT: 0	RANGE: 0 3

Internal Temperature Failsafe Enable (FS_TEMP_ENABLE)

Description

FS_TEMP_ENABLE defines the emergency response if the internal electronics enclosure gets too hot.

- **0: Disabled.**
- **1: Warn Only.**
- **2: Surface.**

Tuning & Behavior

- **Default:** 0.
- **Requirement:** Requires a configured FS_TEMP_MAX threshold.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





FS_TEMP_MAX (Internal Temperature Failsafe)

DEFAULT: 70	RANGE: 0 100	UNITS: degC

Internal Temperature Failsafe (FS_TEMP_MAX)

Description

`FS_TEMP_MAX` protects the vehicle's electronics from overheating.

FS_TERRAIN_ENAB (Terrain Failsafe Enable)

DEFAULT: 0	RANGE: 0 2

Terrain Failsafe Enable (FS_TERRAIN_ENAB)

Description

`FS_TERRAIN_ENAB` defines the "Safety Net" for terrain-following missions.

If you are flying an autonomous mission where the drone's altitude is relative to the ground (Terrain Following), the autopilot requires a steady stream of terrain data (from the SD card or a rangefinder). If this data is lost (e.g., SD card error or flying into an area with no map coverage), this parameter tells the drone how to react.

- **0: Disarm/Land.** (Standard for some vehicles).
- **1: RTL.** Immediately returns home using Barometric (standard) altitude.
- **2: Continue.** (Risky). Switch to barometric altitude and continue the mission at the current height above home.

Tuning & Behavior

- **Default:** 0.
- **Recommendation:** Set to **1 (RTL)** to ensure the drone returns to a safe location if it can no longer "see" the ground profile.

FS_THR_ENABLE (Throttle Failsafe Enable)

DEFAULT: 0	RANGE: 0 5

Throttle Failsafe Enable (FS_THR_ENABLE)

Description

`FS_THR_ENABLE` is the primary failsafe for your radio control link.

If you fly out of range or your transmitter battery dies, your drone needs to know how to react.

- **0: Disabled.** No action. (Dangerous!).
- **1: Enabled (RTL).** The drone will automatically return home and land.
- **3: Enabled (Smart-RTL or RTL).** The drone will try to retrace its steps home.
- **5: Enabled (Land).** The drone will land exactly where it is.

Tuning & Behavior

- **Recommendation:** Always set this to **1 (RTL)** for any outdoor mission.
- **Trigger:** This failsafe is triggered if the throttle channel drops below `FS_THR_VALUE` or if the receiver stops sending data (Radio Loss).

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





FS_THR_VALUE (Throttle Failsafe Threshold)

DEFAULT: 910	RANGE: 910 1100	UNITS: PWM

Throttle Failsafe Threshold (FS_THR_VALUE)

Description

`FS_THR_VALUE` sets the "Alarm Trigger" for your radio connection.

Most RC receivers are configured to pull the throttle channel to a very low value (e.g. 900) if the link to the transmitter is lost. If the autopilot sees the throttle signal drop below this parameter's value, it immediately triggers a Failsafe action (like RTL or Hold).

Tuning & Behavior

- **Default:** 910.
- **Setup:**
  1. Check your normal throttle range in the GCS (usually 1100 to 1900).
  2. Turn off your transmitter and see what the throttle value drops to.
  3. Set `FS_THR_VALUE` to at least **10 units above** that value, but well below your normal minimum throttle (e.g. if transmitter-off is 900, set this to 910).
- **Warning:** If set too high (e.g. 1150), the drone might trigger a failsafe even when you just pull the stick to idle.

FS_VIBE_ENABLE (Vibration Failsafe Enable)

DEFAULT: 1	RANGE: 0 1

Vibration Failsafe Enable (FS_VIBE_ENABLE)

Description

`FS_VIBE_ENABLE` activates a "Limp Mode" for drones suffering from extreme vibration.

If a drone has a damaged propeller or a loose motor, the resulting vibrations can overwhelm the EKF, making the drone fly erratically or drift away. When this failsafe is enabled, the autopilot detects the high vibration and automatically switches to a simpler navigation model that is less accurate but far more resistant to vibration noise.

- **0: Disabled.**
- **1: Enabled (Default).**

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





FAST RATE THREADING (FSTRATE)

Overview

The **FSTRATE** parameter group configures the **Fast-Rate Threading** architecture of the autopilot. This is an advanced performance feature for modern flight controllers (STM32F7 and H7).

Key Concepts

1. Multi-Threaded Control

Standard autopilots run all tasks in a single loop. If a slow task (like writing to the SD card) takes too long, it can "starve" the critical attitude control loop, causing a jittery flight.

- **Fast Rate Enable:** By setting `FSTRATE_ENABLE = 1`, the autopilot creates a separate, high-priority CPU thread dedicated *only* to the high-frequency control loops (Gyro → PID → Motors).

2. Stability

This ensures that no matter how busy the autopilot gets with telemetry or missions, the motors always receive their updates at a perfect, jitter-free frequency (e.g., exactly 400Hz or 800Hz).

Parameter Breakdown

- `FSTRATE_ENABLE`:
  - **0:** Disabled (Single-threaded).
  - **1:** Enabled (Dedicated high-priority loop).

Developer Notes

- **Library:** `AP_Scheduler`.
- **Recommendation:** Highly recommended for all modern H7 boards (Orange Cube, Pixhawk 6X, etc.) to achieve the best possible flight feel.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





FSTRATE_ENABLE (Fast Rate Thread Enable)

DEFAULT: 0	RANGE: 0 1	UNITS: Option

Fast Rate Thread Enable (FSTRATE_ENABLE)

Description

This parameter enables a dedicated, high-priority thread for the main attitude control loop.

By offloading the critical "Fast Loop" (typically 400Hz or higher) to its own thread, ArduPilot can ensure that attitude stabilization is prioritized above all other tasks (like logging, telemetry, or mission logic). This reduces "jitter" in the control loop timing and allows the vehicle to fly smoother and more locked-in.

The Mathematics

When enabled, the scheduler splits the workload:

- **Main Thread:** Handles 10Hz to 50Hz tasks (Navigation, GPS, Logging).
- **Fast Thread:** Handles 400Hz+ tasks (Gyro sampling, Rate PID loops, Motor output).

The update rate of this thread is controlled by `FSTRATE_DIV`, which divides the raw gyro sample rate.

The Engineer's View

In `ArduCopter/Copter.cpp` (and the scheduler logic), this flag triggers the creation of the `FastLoop` thread.

- **0: Disabled.** All code runs in a single main loop. This is safer for very low-end boards but less performant.
- **1: Enabled.** The rate controller runs in parallel with the rest of the code.

Tuning & Behavior

- **Default Value:** 0 (Disabled) on older firmware/boards. Often 1 on modern H7 boards.
- **Recommendation:** Enable this on any modern flight controller (STM32F7/H7). It significantly improves flight performance.
- **Note:** If you see "Scheduler Overrun" errors after enabling this, your loop rate might be set too high for the processor.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





FORWARD/VERTICAL DECOUPLING (FV)

Overview

The **FV** parameter group (specifically `FV_CPLNG_K`) is a specialized configuration for **Vectored ROVs** (Submarines).

Due to the hydrodynamic design of some ROV frames, applying high vertical thrust (climbing/diving) can induce an unwanted pitching moment if the center of drag is not aligned with the center of thrust. This group manages the mathematical decoupling of these forces.

Key Concepts

1. Hydrodynamic Coupling

In a perfectly balanced vehicle, Forward thrust and Vertical thrust are independent. However, in reality, as water flows over the frame, the vehicle may want to pitch up or down.

- `FV_CPLNG_K`: Acts as a gain factor that limits the maximum allowed forward thrust based on the amount of vertical thrust being applied.

Parameter Breakdown

- `FV_CPLNG_K`: Forward/Vertical to Pitch decoupling factor.
  - **0:** Disabled (Full power available on both axes).
  - **1.0 (Default):** Balanced decoupling.

Integration Guide

- **Symptom:** If your ROV pitches up or down uncontrollably when you apply full throttle (vertical) while moving forward, **increase** this value.

Developer Notes

- **Library:** `libraries/AP_Motors/AP_Motors6DOF.cpp`.
- **Context:** Only used in **ArduSub**.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





FV_CPLNG_K (Forward/vertical to pitch decoupling factor)

DEFAULT: 1.0	RANGE: 0.0 1.5	UNITS: null

Forward/vertical to pitch decoupling factor (FV_CPLNG_K)

Description

This parameter is specific to **Vectored ROVs** (like the BlueROV2) running the Sub firmware. It compensates for the physical tendency of some ROV frames to pitch up/down when applying forward or vertical thrust due to drag and center-of-mass offsets.

By adjusting this factor, the mixer limits the output of specific thrusters to prevent the vehicle from pitching destabilizingly during high-speed maneuvers.

The Mathematics

The parameter sets a dynamic limit on the `Forward` thrust component based on the current `Throttle` (Vertical) demand.

$$Limit = 1.0 - (FV_CPLNG_K \times |Throttle|)$$

$$Thrust_Forward_Limited = \text{constrain}(Thrust_Forward, -Limit, Limit)$$

This logic essentially says: "If you are applying full vertical power, reduce the allowed forward power to prevent hydrodynamic instability."

The Engineer's View

In `AP_Motors6DOF::output_armed_stabilizing_vectored()`, this parameter is read into `_forwardVerticalCouplingFactor`.

It is used to calculate `forward_coupling_limit`. This limit is then applied selectively to the rear thrusters (depending on direction of travel) to decouple the forward/vertical hydrodynamic coupling.

- The logic checks `forward_coupling_direction` to identify which thrusters need limiting.

Tuning & Behavior

- **Default Value:** 1.0
- **Range:** 0.0 to 1.5
- **Value 0.0:** Disabled. Full forward thrust allowed regardless of vertical thrust.
- **Value 1.0:** Standard decoupling.
- **Effect of Increasing:** Increases the "safety margin," restricting forward speed more aggressively when vertical thrusters are active.
- **Symptom:** If your ROV does a backflip or pitches violently when you apply full throttle + full forward, **increase** this value.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY

Overview

The **FW** parameter group (specifically `FW_V_FF` and related gains) handles the **Forward Velocity** controller, primarily within the context of **Autonomous Autorotation** for helicopters.

Key Concepts

1. Velocity FeedForward

In a descent (autorotation), the autopilot must maintain a specific forward ground/airspeed to preserve rotor RPM and reach the touchdown point.

- `FW_V_FF`: Provides an immediate pitch command proportional to the target velocity, reducing the burden on the PID error-correction loops.

Parameter Breakdown

- `FW_V_FF`: Forward velocity feed-forward gain.

Developer Notes

- **Library:** `libraries/AC_Autorotation`.
- **Relationship:** Complements the `AROT_` (Autorotation) and `AS_` (Acceleration) groups.

FW_V_FF (Velocity (horizontal) feed forward)

DEFAULT: 0.15	RANGE: 0 1

Velocity (horizontal) feed forward (FW_V_FF)

Description

Feed-forward gain for the horizontal velocity controller during helicopter autorotation glide.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **FWD** parameter group configures voltage compensation for the **Forward Propulsion** motor on **QuadPlane** vehicles.

QuadPlanes often use two separate electrical systems: one for the vertical lift motors (VTOL) and a dedicated high-capacity pack for the forward motor (Fixed-Wing). To ensure consistent thrust as the cruising battery drains, the autopilot needs to know which battery to monitor for the forward motor.

Key Concepts

1. Dual-Battery Voltage Compensation

The autopilot scales the throttle output based on the battery voltage. If the forward motor is on a separate battery, it must be told which monitor instance (`BATT1` vs `BATT2`) to look at.

- `FWD_BAT_IDX` : Selects the monitor index.

Parameter Breakdown

- `FWD_BAT_IDX` : Battery monitor index for forward thrust.
- `FWD_BAT_VOLT_MIN` / `MAX` : The voltage range used to calculate the compensation scaling factor.

Integration Guide

- **Setup:** If you have a single battery for everything, leave these at default.
- **Setup (Dual Pack):** If your forward motor has its own battery monitor, set `FWD_BAT_IDX` to that monitor's ID.

Developer Notes

- **Library:** `ArduPlane/QuadPlane` .
- **Context:** Only relevant for **VTOL / QuadPlane** configurations.

FWD_BAT_IDX (Forward Battery Index)

DEFAULT: 0	RANGE: 0 10

Forward Battery Index (FWD_BAT_IDX)

Description

Selects which battery monitor instance is physically connected to the forward propulsion battery (if different from the vertical lift battery).

FWD_BAT_VOLT_MAX (Forward Battery Voltage Max)

DEFAULT: 0	RANGE: 0 100	UNITS: V

Forward Battery Voltage Max (FWD_BAT_VOLT_MAX)

Description

Defines the "full battery" voltage for the forward propulsion system in a QuadPlane. This allows the flight controller to automatically increase throttle as the battery voltage drops, maintaining consistent forward thrust.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





FWD_BAT_VOLT_MIN (Forward Battery Voltage Min)

DEFAULT: 0	RANGE: 0 100	UNITS: V

Forward Battery Voltage Min (FWD_BAT_VOLT_MIN)

Description

Defines the "empty battery" voltage for the forward propulsion system. Used in conjunction with FWD_BAT_VOLT_MAX for throttle scaling.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY

Overview

The **GCS** parameter group configures the communication and telemetry behavior between the autopilot and the **Ground Control Station** (e.g., Mission Planner, QGroundControl).

While most MAVLink settings are in the `SERIAL` and `MAV` groups, the `GCS` group handles global telemetry options that affect all active MAVLink links.

Key Concepts

1. PID Tuning Streams (`GCS_PID_MASK`)

To tune a drone effectively, the pilot needs to see what the PID controllers are doing in real-time. This parameter enables the broadcast of `PID_TUNING` MAVLink messages.

- **Visualization:** When enabled, the GCS can draw real-time graphs of "Target Rate" vs "Actual Rate" and show the contribution of the P, I, and D terms.

Parameter Breakdown

- `GCS_PID_MASK`: Bitmask of controllers to stream.
  - **Bit 0:** Roll.
  - **Bit 1:** Pitch.
  - **Bit 2:** Yaw.
  - **Bit 3:** Altitude/Position.

Integration Guide

- **Tuning:** Before starting a manual PID tuning session, set `GCS_PID_MASK = 7` (Roll+Pitch+Yaw) to see the performance data on your GCS screen.

Developer Notes

- **Library:** `libraries/GCS_MAVLink`.
- **Performance:** Streaming many PIDs consumes telemetry bandwidth. Turn off when not tuning.

`GCS_PID_MASK` (GCS PID tuning mask)

`DEFAULT: 0`

GCS PID tuning mask (GCS_PID_MASK)

Description

Defines which PID controllers should broadcast their real-time error, proportional, integral, and derivative components over MAVLink (PID_TUNING message). This is essential for ground stations to display real-time PID tuning graphs.

Tuning & Behavior

- **Default Value:** 0 (None)
- **Bitmask (Vehicle dependent):**
  - Bit 0: Roll
  - Bit 1: Pitch
  - Bit 2: Yaw
  - Bit 3: Altitude/Throttle

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **GEN** parameter group configures the **Onboard Generator** subsystem. This is used on hybrid-electric drones that use a small internal combustion engine (ICE) to charge the batteries or power the motors in flight.

ArduPilot can monitor the generator status and, for some models, autonomously control the startup and shutdown sequence.

Key Concepts

1. Generator Types (`GEN_TYPE`)

Selects the driver for the generator system.

- **0:** Disabled.
- **1:** RichenPower.
- **2:** Intelligent Energy (Hydrogen Fuel Cells).

2. Operational Safety (`GEN_OPTIONS`)

A bitmask to control autonomous behavior.

- **Stop on Disarm:** Automatically shuts down the generator when the vehicle disarms.
- **Autostart:** Automatically starts the generator when battery voltage drops below a threshold.

Parameter Breakdown

- `GEN_TYPE`: Driver selection.
- `GEN_OPTIONS`: Behavior flags.

Integration Guide

1. **Hardware:** Connect the generator's serial or PWM control line.
2. **Telemetry:** Ensure battery monitors are configured to see the charging current.

Developer Notes

- **Library:** `libraries/AP_Generator`.

GEN_OPTIONS (Generator Options)

DEFAULT: 0

Generator Options (GEN_OPTIONS)

Description

Configuration bitmask for tailoring the generator's interaction with the flight controller.

Tuning & Behavior

- **Default Value:** 0
- **Bitmask:**
  - Bit 0: Suppress Maintenance-Required Warnings

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





GEN_TYPE (Generator type)

DEFAULT: 0

Generator type (GEN_TYPE)

Description

Enables the generator control module and selects the driver for specific fuel cell or combustion generator hardware.

Tuning & Behavior

- **Default Value:** 0 (Disabled)
- **Values:** 0:Disabled, 1:IE 650w/800w Fuel Cell, 2:IE 2.4kW Fuel Cell, 3:RichenPower
- **Requires reboot to take effect.**

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **GLIDE** parameter group (often appearing as `LAND_GLIDE_` in the full parameter list) configures the **Autonomous Landing Glide Slope** for ArduPlane.

The glide slope is the imaginary path (usually 3 to 10 degrees) that the plane follows from the start of the landing sequence to the touchdown point.

Key Concepts

1. Glide Slope Rebuilding (`GLIDE_SLOPE_THR`)

If the plane is pushed off its planned path by a gust of wind, should it dive back to the original line?

- **Threshold:** If the plane is higher than this value (meters) above the planned slope, the autopilot will "rebuild" the slope from its current position to the target, ensuring a smoother descent.

2. Minimum Slope (`GLIDE_SLOPE_MIN`)

Defines the shallowest angle the plane is allowed to fly during a landing approach. This prevents the autopilot from trying to "float" indefinitely if it has too much energy.

Parameter Breakdown

- `GLIDE_SLOPE_THR`: Altitude error threshold for slope recalculation.
- `GLIDE_SLOPE_MIN`: Minimum allowable glide angle (degrees).

Integration Guide

- **Landing Issues:** If your plane dives aggressively during landing, check if `GLIDE_SLOPE_THR` is too low.
- **Floating:** If your plane consistently overshoots the runway, increase `GLIDE_SLOPE_MIN`.

Developer Notes

- **Library:** `ArduPlane/mode_landing.cpp`.
- **TECS:** Works in tight coordination with the `TECS_` (Total Energy) library to manage the trade-off between altitude and airspeed.

`GLIDE_SLOPE_MIN` (Glide Slope Minimum Altitude)

DEFAULT: 15	RANGE: 0 1000	UNITS: m

Glide Slope Minimum Altitude (GLIDE_SLOPE_MIN)

Description

`GLIDE_SLOPE_MIN` controls how "Smoothly" your plane changes altitude between mission waypoints.

If you have two waypoints with a very small altitude difference (e.g. 5 meters), it might be better to just climb instantly rather than calculating a long, shallow glide slope. This parameter sets that cutoff point.

- **0:** Glide slope logic is always used (or disabled, depending on vehicle).
- **15 (Default):** If the altitude change is less than 15 meters, the plane will try to reach the new altitude as quickly as possible. If it's more than 15 meters, it will calculate a straight line (slope) between the points and follow it precisely.

Tuning & Behavior

- **Usage:** Increase this value if you find your plane "hunting" for altitude during slight variations in terrain-following or waypoint missions.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





GLIDE_SLOPE_THR (Glide slope threshold)

DEFAULT: 5.0	RANGE: 0 100	UNITS: m

Glide slope threshold (GLIDE_SLOPE_THR)

Description

This parameter sets the "re-calculation threshold" for the auto-takeoff and auto-landing glide slopes. If the aircraft finds itself significantly above the planned path (e.g., due to a thermal or overshoot), rather than diving aggressively to regain the original line, the autopilot will compute a *new*, smoother path from its current location to the target if the error exceeds this threshold.

The Mathematics

$$\text{If } (\text{Current Alt} - \text{Path Alt}) > \text{GLIDE_SLOPE_THR} \implies \text{Recalculate Slope}$$

The Engineer's View

Defined in `ArduPlane/Parameters.cpp`.

- **Benefit:** Prevents "porpoising" or aggressive pitch-down commands during takeoff climb-out or landing approach.

Tuning & Behavior

- **Default Value:** 5.0 m
- **Behavior:** Higher values force the plane to stick closer to the original path; lower values allow it to adapt (shallow out) the path more easily.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **GND** parameter group (specifically `GND_EFFECT_COMP`) configures the **Ground Effect Compensation** feature for ArduCopter.

When a multicopter is very close to the ground, the downward air from the propellers (prop wash) creates a high-pressure bubble under the frame. This high pressure can "fool" the barometer into thinking the drone has suddenly dropped in altitude, causing it to jump or bounce during landing/takeoff.

Key Concepts

1. Pressure Compensation

This system uses a model of the vehicle's thrust to estimate the pressure increase caused by the ground and subtracts it from the barometer reading.

- **Trigger:** The compensation is only active when the vehicle is within a certain distance of the ground (based on rangefinder or takeoff altitude).

Parameter Breakdown

- `GND_EFFECT_COMP`:
  - **0:** Disabled.
  - **1:** Enabled.

Integration Guide

- **Symptom:** If your drone hovers stably at 5 meters but "balloons" or jumps when you try to land it, enable `GND_EFFECT_COMP`.

Developer Notes

- **Library:** `libraries/AP_Baro`.
- **Algorithm:** Uses the estimated thrust and the current height above ground to calculate a dynamic pressure offset.

GND_EFFECT_COMP (Ground Effect Compensation Enable/Disable)

DEFAULT: 1	RANGE: 0 1

GND_EFFECT_COMP: Ground Effect Compensation Enable/Disable

Description

Enables or disables the ground effect compensation feature.

Values

Value	Meaning
0	Disabled
1	Enabled

- **Default:** 1

Description

This parameter activates the **Ground Effect Compensation** logic for multicopters.

- **Phenomenon:** When a drone flies close to the ground (usually within one rotor diameter), the air pressure builds up underneath it ("Ground Effect"), increasing lift. This can cause the drone to "float" or become jumpy during landing.
- **Function:** The autopilot detects when it is entering ground effect (based on barometer and throttle/climb rate mismatch) and automatically reduces the target throttle to maintain a smooth descent.
- **Benefit:** Smoother landings and reduced tendency to bounce or balloon upon touchdown.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY

Overview

The **GPIO** parameter group (specifically `GPIO_MASK` or `SERVO_GPIO_MASK`) configures the repurposing of motor/servo outputs for **General Purpose Input/Output (GPIO)** functions.

By default, the pins on your flight controller's servo rail are configured to output high-frequency PWM or DShot signals for motors. This group allows you to "re-task" some of those pins to be simple on/off digital switches (3.3V high/low).

Key Concepts

1. Servo to GPIO Mapping

Pins are assigned using a bitmask. When a bit is set for a specific output channel, the PWM timer for that pin is disabled, and it is handed over to the GPIO controller.

- **Common Uses:**
  - Controlling a Relay.
  - Firing a Camera Shutter.
  - Toggling external LED strips.
  - Input for a push-button (see `BTN` group).

Parameter Breakdown

- `GPIO_MASK`: Bitmask of pins to convert to GPIO.
  - **Bit 0:** Servo 1.
  - **Bit 5:** Servo 6.
  - **Value 32 (2^5):** Converts AUX 1 (on many boards) to GPIO.

Integration Guide

1. **Select Pin:** Identify which servo pin you want to use.
2. **Enable:** Set the corresponding bit in `GPIO_MASK`.
3. **Assign Function:** Use `RELAY_PIN` or `CAM_FEEDBAK_PIN` to reference the digital ID of that pin.

Developer Notes

- **Library:** `libraries/SRV_Channel`.
- **Hardware:** Not all pins on all boards can be converted (depends on the microcontroller's timer allocation). Refer to your board's "Mapping" page on the ArduPilot Wiki.

GPIO_MASK (Servo GPIO Mask)

`DEFAULT: 0`

Servo GPIO Mask (GPIO_MASK)

Description

Allows specific PWM output pins to be repurposed as general-purpose input/output (GPIO) pins. This is useful for controlling external hardware like LEDs, relays, or camera triggers that don't use standard PWM signals.

- **Bitmask corresponds to output channels 1 through 16.**

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **GPS** parameter group configures the **Global Positioning System** (and other GNSS constellations like GLONASS, Galileo, BeiDou). GPS is the primary source for horizontal position, ground velocity, and UTC time for the autopilot.

ArduPilot supports up to 2 simultaneous GPS units (`GPS` and `GPS2`), with advanced logic for blending and redundancy.

Key Concepts

1. Driver Type (`GPS_TYPE`)

Tells ArduPilot which communication protocol to use.

- **1 (Auto):** Autodetects standard u-Blox, MTK, or NMEA.
- **2 (u-Blox):** Forces u-Blox binary protocol (recommended for most).
- **9 (DroneCAN):** For CAN-based GPS units (e.g., Here3, CubePilot).
- **17 (Moving Baseline):** For high-precision RTK Yaw.

2. Multi-GPS Redundancy (`GPS_AUTO_SWITCH`)

- **0 (Disabled):** Use only the primary GPS.
- **1 (Use Best):** Automatically switches to the GPS with the best satellite count and HDOP.
- **2 (Blend):** Merges both GPS solutions into one for smoother navigation.

3. Moving Baseline Yaw (`GPS_MB_...`)

Advanced RTK setup where two GPS units on the same vehicle calculate a relative vector between each other to determine the vehicle's heading (Yaw). This allows for accurate heading even in areas with high magnetic interference where compasses fail.

4. Configuration (`GPS_SAVE_CFG`)

ArduPilot can automatically send configuration commands to u-Blox receivers at boot to set the correct update rate and constellations.

Parameter Breakdown

- `GPS_TYPE`: Primary driver.
- `GPS_PRIMARY`: Which GPS is instance 1 (0 or 1).
- `GPS_GNSS_MODE`: Bitmask to select constellations (GPS, SBAS, Galileo, etc.).
- `GPS_MIN_ELEV`: Minimum satellite elevation (degrees) to ignore satellites near the horizon (noisy).
- `GPS_POS_X/Y/Z`: The physical offset of the GPS antenna from the vehicle center of gravity.

Integration Guide

Standard Setup

1. **Hardware:** Connect GPS to the `GPS` (Serial 3 or 4) port.
2. **Config:** Set `GPS_TYPE = 1` (Auto).
3. **Wait:** Power up outdoors. Check GCS for "3D Fix."

Setting up Dual GPS Blending

1. Connect two GPS units.
2. Set `GPS_TYPE = 1` and `GPS_TYPE2 = 1`.
3. Set `GPS_AUTO_SWITCH = 2` (Blend).
4. Set `GPS_POS_...` and `GPS2_POS_...` offsets accurately.

Developer Notes

- **Library:** `libraries/AP_GPS`.
- **Precision:** Use **RTK-capable** receivers (u-Blox F9P) and an injection source (MavESP8266 or GCS) for centimeter-level accuracy.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





GPS1_CAN_NODEID (Detected CAN Node ID for GPS (GPS 1))

DEFAULT: 0	RANGE: 0 127

Detected CAN Node ID for GPS (GPS1_CAN_NODEID)

Description

`GPS1_CAN_NODEID` displays the DroneCAN (UAVCAN) Node ID that the autopilot has assigned to or detected from the primary GPS unit. This is a **Read-Only** parameter used for diagnostics.

- **Role:** Identifies the specific hardware device on the CAN bus.
- **Usage:** If you need to force the autopilot to use a specific GPS node, set GPS1_CAN_OVRIDE instead.

The Engineer's View

- **Storage:** `AP_GPS::Params::node_id` (Integer).
- **Update Logic:** Updated by the `AP_GPS_DroneCAN` driver when a matching device is found and successfully configured.
- **Persistence:** The value is saved to storage but overwritten on the next boot if detection changes.

Tuning & Behavior

- **Default:** 0 (No device detected yet).
- **Normal Range:** 0-125 (Standard DroneCAN Node IDs).
- **Dependency:** Requires `GPS1_TYPE` = 9 (DroneCAN).

GPS1_CAN_OVRIDE (GPS DroneCAN Node ID Override (Sensor 1))

DEFAULT: 0	RANGE: 0 127

GPS DroneCAN Node ID Override (GPS1_CAN_OVRIDE)

Description

`GPS1_CAN_OVRIDE` allows you to force ArduPilot to use a specific GPS unit as the "Primary" when you have multiple GPS units connected via DroneCAN (UAVCAN).

- **0 (Default):** First-come, first-served. The autopilot uses the first DroneCAN GPS it detects on the bus.
- **1-125:** Locks the primary GPS slot to the hardware with this specific Node ID.

Tuning & Behavior

- **Default:** 0.
- **Usage:** Only set this if you have two identical DroneCAN GPS units and you want to ensure a specific physical one (e.g., the one with the best view of the sky) is always used as Instance 1.
- **Reboot Required:** Yes.

GPS1_COM_PORT (GPS Device physical COM port)

DEFAULT: 0	RANGE: 0 10

GPS Device physical COM port (GPS1_COM_PORT)

Description

`GPS1_COM_PORT` identifies which physical connector *on the GPS hardware itself* is being used to talk to the flight controller.

This is only required for high-end professional GNSS receivers (like Septentrio or Trimble) that have multiple internal UART ports. Most consumer GPS modules (U-Blox) do not need this setting.

Tuning & Behavior

- **Default:** 0.
- **Usage:** If you are using a Septentrio (SBF) or Trimble (GSOF) receiver and the flight controller is plugged into the receiver's second port, set this to **1** or **2** to match the receiver's internal port numbering.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY

GPS1_DELAY_MS (GPS Processing Delay (Sensor 1))

DEFAULT: 0	RANGE: 0 250	UNITS: ms

GPS Processing Delay (GPS1_DELAY_MS)

Description

`GPS1_DELAY_MS` tells the autopilot how "Old" the GPS data is when it arrives at the flight controller.

GPS modules take time to calculate position. By the time the autopilot receives the signal, the drone has already moved slightly. For accurate navigation, the EKF (Extended Kalman Filter) needs to know this delay so it can correctly align the GPS data with the high-speed inertial data (IMU).

- **0 (Default):** Use the driver's hard-coded default for the detected GPS type (e.g. 200ms for u-blox M8N).
- **1-250:** Manual override in milliseconds.

Tuning & Behavior

- **Default:** 0.
- **Recommendation:** Leave at **0** for most users. Only adjust this if you are using an exotic or custom GPS module and you see "Horizontal Position Drift" during aggressive maneuvering.

GPS1_GNSS_MODE (GNSS system configuration)

DEFAULT: 0	RANGE: null

GNSS system configuration

Note: This parameter functions identically to GPS_GNSS_MODE.

GPS1_RATE_MS (NMEA Output rate)

DEFAULT: 100	RANGE: 20 2000	UNITS: ms

NMEA Output rate

Note: This parameter functions identically to GPS_RATE_MS.

GPS1_TYPE (GPS type)

DEFAULT: 0	RANGE: null

GPS type

Note: This parameter functions identically to GPS_TYPE.

GPS2_CAN_NODEID (Detected CAN Node ID for GPS)

DEFAULT: 0	RANGE: null

Detected CAN Node ID for GPS

Note: This parameter functions identically to GPS1_CAN_NODEID.

GPS2_CAN_OVRIDE (DroneCAN GPS NODE ID)

DEFAULT: 0	RANGE: null

DroneCAN GPS NODE ID

Note: This parameter functions identically to GPS1_CAN_OVRIDE.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





GPS2_COM_PORT (GPS physical COM port)

DEFAULT: HAL_GPS_COM_PORT_DEFAULT	RANGE: 0 10

GPS physical COM port

Note: This parameter functions identically to GPS1_COM_PORT.

GPS2_DELAY_MS (GPS delay in milliseconds)

DEFAULT: 0	RANGE: 0 250	UNITS: ms

GPS delay in milliseconds

Note: This parameter functions identically to GPS1_DELAY_MS.

GPS2_GNSS_MODE (GNSS system configuration)

DEFAULT: 0	RANGE: null

GNSS system configuration

Note: This parameter functions identically to GPS_GNSS_MODE.

GPS2_RATE_MS (GPS update rate in milliseconds)

DEFAULT: 200	RANGE: 50 200	UNITS: ms

GPS update rate in milliseconds

Note: This parameter functions identically to GPS_RATE_MS.

GPS2_TYPE (GPS type)

DEFAULT: 0	RANGE: null

GPS type

Note: This parameter functions identically to GPS_TYPE.

GPS_ACC (GPS Accuracy (Invalid/Legacy))

DEFAULT: 0	UNITS: m

GPS Accuracy (GPS_ACC)

Description

This parameter is not found as a standard GPS_ACC in the main codebase.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





GPS_ALT_OFS (GPS Altitude Offset (Invalid/Legacy))

DEFAULT: 0	UNITS: m

GPS Altitude Offset (GPS_ALT_OFS)

Description

This parameter is not found as a standard GPS_ALT_OFS in the main codebase.

GPS_AUTO_CONFIG (Automatic GPS configuration)

DEFAULT: 1	RANGE: 0 2

Automatic GPS configuration (GPS_AUTO_CONFIG)

Description

Allows ArduPilot to automatically send the required initialization commands and baudrate settings to the GPS module.

Tuning & Behavior

- **Default Value:** 1 (Serial only)
- **Values:** 0:Disabled, 1:Enable for Serial GPS, 2:Enable for DroneCAN as well

GPS_AUTO_SWITCH (Automatic Switchover Setting)

DEFAULT: 1	RANGE: 0 4

Automatic Switchover Setting (GPS_AUTO_SWITCH)

Description

Defines the logic for selecting which GPS receiver's data is sent to the EKF when two or more modules are active.

Tuning & Behavior

- **Default Value:** 1 (UseBest)
- **Values:**
  - 0: Use Primary (Strictly use GPS_PRIMARY)
  - 1: UseBest (Select based on highest fix status and sat count)
  - 2: Blend (Mathematically combine data from all receivers)
  - 4: Use primary if 3D fix or better

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY

GPS_BLEND_MASK (Multi GPS Blending Mask)

DEFAULT: 5	RANGE: 0 7

Multi GPS Blending Mask (GPS_BLEND_MASK)

Description

`GPS_BLEND_MASK` controls the "Recipe" for merging two GPS signals into one.

When `GPS_AUTO_SWITCH = 2 (Blend)`, ArduPilot doesn't just pick one GPS; it merges both into a single "Blended" solution. This parameter decides which metrics are used to determine how much to trust each GPS.

- **Bit 0 (1): Horizontal Position Accuracy.** Use the reported horizontal error (hAcc) to weigh the blend.
- **Bit 1 (2): Vertical Position Accuracy.** Use the reported vertical error (vAcc).
- **Bit 2 (4): Speed Accuracy.** Use the velocity error (sAcc).

Tuning & Behavior

- **Default Value:** 5 (Horizontal + Speed).
- **Recommendation:** Leave at **5**. Trusting reported vertical accuracy (Bit 1) can sometimes lead to altitude jumps if one GPS has a sudden change in its altitude noise model.

GPS_BYTELOSS (GPS Byte Loss (Invalid/Legacy))

DEFAULT: 0

GPS Byte Loss (GPS_BYTELOSS)

Description

This parameter is not found as a standard GPS_BYTELOSS in the main codebase.

GPS_DISABLE (GPS Disable (Invalid/Legacy))

DEFAULT: 0

GPS Disable (GPS_DISABLE)

Description

This parameter is not found as a standard GPS_DISABLE in the main codebase.

GPS_DRIFTALT (GPS Drift Altitude (Invalid/Legacy))

DEFAULT: 0	UNITS: m

GPS Drift Altitude (GPS_DRIFTALT)

Description

This parameter is not found as a standard GPS_DRIFTALT in the main codebase.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





GPS_DRV_OPTIONS (driver options)

DEFAULT: 0

driver options (GPS_DRV_OPTIONS)

Description

Advanced configuration for the GPS driver, such as enabling specific UART ports for moving baseline data.

Tuning & Behavior

- **Default Value:** 0
- **Bitmask:**
  - Bit 0: Use UART2 for moving baseline (ublox)
  - Bit 2: Use baudrate 115200
  - Bit 4: Use ellipsoid height instead of AMSL

GPS_GLITCH (GPS Glitch (Invalid/Legacy))

DEFAULT: 0

GPS Glitch (GPS_GLITCH)

Description

This parameter is not found as a standard GPS_GLITCH in the main codebase.

GPS_GNSS_MODE (GNSS constellation mask)

DEFAULT: 0

GNSS constellation mask (GPS_GNSS_MODE)

Description

Specifies which global satellite systems the receiver should use (e.g., GPS, GLONASS, Galileo, BeiDou).

- **A value of 0 usually allows the receiver to use all supported constellations.**

GPS_HDG (GPS Heading)

DEFAULT: 0	UNITS: deg

GPS Heading (GPS_HDG)

Description

The aircraft's actual path over the ground as measured by the primary GPS receiver.

- **ReadOnly.**

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





GPS_HDOP_GOOD (GPS HDOP Threshold (Good Fix))

DEFAULT: 140	RANGE: 100 900

GPS HDOP Threshold (GPS_HDOP_GOOD)

Description

`GPS_HDOP_GOOD` sets the "Trust Limit" for your GPS during pre-arm safety checks.

HDOP (Horizontal Dilution of Precision) is a measure of GPS geometric accuracy. A lower number is better. If your GPS reports an HDOP higher than this value, the autopilot will prevent you from arming in autonomous modes (like Loiter, PosHold, or Auto) because the position data is too imprecise.

- **100:** Extremely strict (Excellent signal required).
- **140 (Default):** Standard for most users.
- **200+:** Relaxed (Allows arming with poorer signals, risky).

Tuning & Behavior

- **Default:** 140 (reported as 1.4 in some GCS).
- **Safety:** Do not increase this value unless you have a very specific reason to fly in an area with poor sky visibility (e.g. narrow canyons). High HDOP can lead to "GPS Glitches" where the drone suddenly jumps several meters.

GPS_HZ (GPS Update Rate (Invalid/Legacy))

DEFAULT: 0	UNITS: Hz

GPS Update Rate (GPS_HZ)

Description

This parameter is not found as a standard GPS_HZ in the main codebase. GPS update rate is typically configured via GPS_RATE_MS.

GPS_INJECT_TO (Destination for GPS_INJECT_DATA)

DEFAULT: 127

Destination for GPS_INJECT_DATA (GPS_INJECT_TO)

Description

Directs the flow of RTK correction data received from the ground station.

- **0: First GPS**
- **1: Second GPS**
- **127: All (Broadcast to all active receivers)**

GPS_JAM (GPS Jamming (Invalid/Legacy))

DEFAULT: 0

GPS Jamming (GPS_JAM)

Description

This parameter is not found as a standard GPS_JAM in the main codebase.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





GPS_LAG_MS (User-defined delay)

DEFAULT: 0	RANGE: 0 250	UNITS: ms

User-defined delay (GPS_LAG_MS)

Description

Allows the user to manually specify the latency of the GPS data. If set to 0, ArduPilot will use a hardcoded delay based on the detected GPS type.

GPS_LOCKTIME (GPS Lock Time (Invalid/Legacy))

DEFAULT: 0	UNITS: s

GPS Lock Time (GPS_LOCKTIME)

Description

This parameter is not found as a standard GPS_LOCKTIME in the main codebase.

GPS_MB1_OFS_X (Moving Base X Offset (GPS 1))

DEFAULT: 0	RANGE: -5 5	UNITS: m

Moving Base X Offset (GPS_MB1_OFS_X)

Description

`GPS_MB1_OFS_X` is the critical measurement for GPS-based yaw (heading).

When using two GPS units for heading (Moving Baseline), the autopilot calculates the drone's orientation by comparing the position of the "Rover" antenna to the "Base" antenna. To do this accurately, it must know the exact physical distance between them in the body frame.

- **Positive X:** The Base antenna is in front of the Rover antenna.
- **Negative X:** The Base antenna is behind the Rover antenna.

Tuning & Behavior

- **Default Value:** 0.
- **Accuracy:** This measurement must be accurate to within **1-2 centimeters** for the heading to be reliable. Use a tape measure or calipers.
- **Orientation:** This is the offset *relative to the other antenna*, not relative to the center of gravity.

GPS_MB1_OFS_Y (Moving Base Y Offset (GPS 1))

DEFAULT: 0	RANGE: -5 5	UNITS: m

Moving Base Y Offset (GPS_MB1_OFS_Y)

Description

`GPS_MB1_OFS_Y` defines the lateral (left/right) distance between the two antennas used for GPS-Yaw.

- **Positive Y:** The Base antenna is to the right of the Rover antenna.
- **Negative Y:** The Base antenna is to the left of the Rover antenna.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





GPS_MB1_OFS_Z (Moving Base Z Offset (GPS 1))

DEFAULT: 0	RANGE: -5 5	UNITS: m

Moving Base Z Offset (GPS_MB1_OFS_Z)

Description

`GPS_MB1_OFS_Z` defines the vertical distance between the two GPS antennas.

- **Positive Z:** The Base antenna is lower than the Rover antenna.
- **Negative Z:** The Base antenna is higher than the Rover antenna.

GPS_MB1_TYPE (Moving Base Type (GPS 1))

DEFAULT: 0	RANGE: 0 1

Moving Base Type (GPS_MB1_TYPE)

Description

`GPS_MB1_TYPE` enables GPS-based yaw (heading) without a compass.

This is part of the **Moving Baseline** feature, where two GPS receivers (one "Base" and one "Rover") are mounted on the drone. By calculating the precise vector between them, ArduPilot can determine the drone's heading with extreme accuracy, regardless of magnetic interference.

- **0: Relative to alternate GPS instance (Default).** The two GPS units on the drone talk to each other to find the heading.
- **1: Relative to Custom Base.** Used for more complex setups with external base stations.

Tuning & Behavior

- **Default Value:** 0.
- **Requirement:** Requires two RTK-capable GPS modules (like u-blox F9P).
- **Configuration:** You must also set the offsets between the two antennas using `GPS_MB1_OFS_X/Y/Z`.

GPS_MB2_TYPE (Moving Base Type (GPS 2))

DEFAULT: 0	RANGE: 0 1

Moving Base Type (GPS_MB2_TYPE)

Description

`GPS_MB2_TYPE` configures the second GPS unit for dual-GPS heading (GPS-Yaw).

See GPS_MB1_TYPE for detailed theory of operation. This parameter is used when you have a second independent moving-baseline pair or a complex 3-antenna setup.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





GPS_MB_ONLY_PORT (Moving Baseline CAN Port Selection)

DEFAULT: 0	RANGE: 0 2

Moving Baseline CAN Port Selection (GPS_MB_ONLY_PORT)

Description

`GPS_MB_ONLY_PORT` is used when setting up a **Moving Baseline** GPS system (where two GPS units work together to calculate the drone's heading without a compass).

This parameter tells the autopilot (or peripheral node) which CAN port should be used to send the RTK correction data between the two GPS units. Using a dedicated port ensures that these high-frequency corrections don't interfere with other vital telemetry on the main CAN bus.

- **0: Auto.**
- **1: CAN Port 1.**
- **2: CAN Port 2.**

Tuning & Behavior

- **Default:** 0.
- **Usage:** Only used on **AP_Periph** CAN nodes when configuring an integrated dual-GPS system over CAN.

GPS_MIN_ELEV (Minimum elevation)

DEFAULT: -100	RANGE: -100 90	UNITS: deg

Minimum elevation (GPS_MIN_ELEV)

Description

Rejects satellites that are too close to the horizon, where signal multi-pathing and atmospheric distortion are highest.

- **-100 uses the GPS module's internal default.**

GPS_NAVFILTER (Navigation filter setting)

DEFAULT: 8	RANGE: 0 8

Navigation filter setting (GPS_NAVFILTER)

Description

Configures the GPS module's internal dynamic model. Setting this correctly for your vehicle type (e.g., Airborne 4G) helps the GPS maintain a more stable fix during high-acceleration maneuvers.

Tuning & Behavior

- **Default Value:** 8 (Airborne 4G)
- **Values:** 0:Portable, 2:Stationary, 3:Pedestrian, 4:Automotive, 5:Sea, 6:Airborne1G, 7:Airborne2G, 8:Airborne4G

GPS_NOISE (GPS Noise (Invalid/Legacy))

DEFAULT: 0

GPS Noise (GPS_NOISE)

Description

This parameter is not found as a standard GPS_NOISE in the main codebase.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





GPS_NUMSATS (GPS Number of Satellites)

DEFAULT: 0

GPS Number of Satellites (GPS_NUMSATS)

Description

Displays the count of satellites providing signals for the primary GPS receiver. A higher count generally leads to a more reliable position fix.

- **ReadOnly: Automatically reported by the GPS module.**

GPS_PORT (GPS Serial Port (Invalid/Legacy))

DEFAULT: 0

GPS Serial Port (GPS_PORT)

Description

This parameter is not found as a top-level GPS_PORT in the main codebase. GPS port selection is handled via SERIALx_PROTOCOL.

GPS_POS (GPS Antenna Offset)

DEFAULT: 0	UNITS: m

GPS Antenna Offset (GPS_POS)

Description

Specifies the physical location of the GPS antenna in body-frame coordinates (X:Forward, Y:Right, Z:Down). Providing accurate offsets improves position and velocity estimation, especially during aggressive maneuvers.

GPS_PRIMARY (Primary GPS)

DEFAULT: 0	RANGE: 0 1

Primary GPS (GPS_PRIMARY)

Description

Determines which GPS module (1st or 2nd) is treated as the primary source when GPS_AUTO_SWITCH is disabled or in specific priority modes.

- **0: First GPS**
- **1: Second GPS**

GPS_RATE_MS (Desired update rate)

DEFAULT: 200	RANGE: 50 200	UNITS: ms

Desired update rate (GPS_RATE_MS)

Description

The target update period for the GPS receiver.

- **200ms = 5Hz (Standard)**
- **100ms = 10Hz (High performance)**

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





GPS_RAW_DATA (GPS Raw Data Logging)

DEFAULT: 0	RANGE: 0 2

GPS Raw Data Logging (GPS_RAW_DATA)

Description

`GPS_RAW_DATA` enables the autopilot to record the raw satellite measurements (carrier phase, pseudorange) into the internal DataFlash log.

This data is used for **Post-Processed Kinematics (PPK)**. If you need centimeter-level accuracy for mapping but don't have a real-time RTK link, you can record this raw data and use software (like RTKLIB) later to calculate an extremely precise flight path.

- **0: Disabled.** Standard NMEA/Binary position data only.
- **1: Enabled.** Logs raw data to the flight controller's SD card.
- **2: Device Specific.** (Septentrio). Commands the GPS unit itself to start/stop its internal logging synchronized with vehicle arming.

Tuning & Behavior

- **Warning:** Raw GPS data creates very large log files. Ensure you have a fast, high-capacity SD card.
- **Requirement:** Requires a GPS module that supports raw output (e.g. u-blox "P" or "F" series, or professional Septentrio/Trimble units).

GPS_SAVE_CFG (Save GPS Configuration)

DEFAULT: 2	RANGE: 0 2

Save GPS Configuration (GPS_SAVE_CFG)

Description

`GPS_SAVE_CFG` determines if the flight controller should permanently save the settings (Baud rate, update frequency, constellations) into the GPS module's own flash memory.

- **0: Do Not Save.** Autopilot configures the GPS on every boot, but the GPS reverts to factory defaults if powered off.
- **1: Save.** Autopilot saves the settings to the GPS flash.
- **2: Save only when needed (Default).** Autopilot only writes to the GPS flash if the configuration has changed.

Tuning & Behavior

- **Default:** 2.
- **Advantage:** Saving the configuration can make the "GPS Lock" and "Detection" phase of the boot process slightly faster, as the GPS doesn't need to be fully reconfigured from scratch every time.
- **Note:** This is primarily supported by **u-blox** (6 series and newer) modules.

GPS_SBAS_MODE (SBAS Mode)

DEFAULT: 2	RANGE: 0 2

SBAS Mode (GPS_SBAS_MODE)

Description

Enables or disables the use of SBAS corrections (like WAAS or EGNOS) to improve GPS accuracy.

- **0: Disabled**
- **1: Enabled**
- **2: No Change (Don't reconfigure the module)**

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





GPS_SBP_LOGMASK (Swift Binary Protocol Logging Mask)

DEFAULT: -256

Swift Binary Protocol Logging Mask (GPS_SBP_LOGMASK)

Description

`GPS_SBP_LOGMASK` is a diagnostic tool for **Swift Navigation** GPS receivers (Piksi/Duro).

These receivers use the Swift Binary Protocol (SBP). This mask allows developers and advanced users to filter exactly which data types from the receiver are recorded in the ArduPilot onboard logs. This helps manage log size while capturing essential troubleshooting information.

Tuning & Behavior

- **Default:** -256 (0xFF00).
- **Usage:** For most users, leave at the default value. If instructed by Swift Navigation support or an ArduPilot developer, you may need to change this to capture specific low-level GNSS data.
- **Impact:** Setting this to record high-rate data will significantly increase the CPU load and SD card bandwidth usage.

GPS_TYPE (GPS type)

DEFAULT: 1

GPS type (GPS_TYPE)

Description

Enables the primary GPS and selects the specific driver for the connected hardware (e.g., u-blox, NMEA, DroneCAN).

Tuning & Behavior

- **Default Value:** 1 (u-blox)
- **Common Values:** 1:u-blox, 2:MTK, 5:NMEA, 9:DroneCAN, 14:MAVLink, 100:SITL
- **Requires reboot after changing to initialize the new driver.**

GPS_VERR (GPS Vertical Error (Invalid/Legacy))

DEFAULT: 0	UNITS: m

GPS Vertical Error (GPS_VERR)

Description

This parameter is not found as a standard GPS_VERR in the main codebase.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **GRIP** parameter group configures the **Gripper / Cargo Release** subsystem. This allows the vehicle to carry, grab, and release payloads using various mechanisms (Servo, EPM, or DroneCAN).

Key Concepts

1. Gripper Types (`GRIP_TYPE`)

Defines the hardware interface for the gripper.

- **1 (Servo):** Standard PWM servo that moves between two positions.
- **2 (EPM):** Electro-Permanent Magnet (e.g., OpenEPM). Uses a short pulse to magnetize/de-magnetize.
- **3 (DroneCAN):** CAN-based gripper devices.

2. Grab and Release Positions

For servo-based grippers, defines the PWM endpoints.

- `GRIP_GRAB` : PWM value to close the gripper.
- `GRIP_RELEASE` : PWM value to open the gripper.
- `GRIP_NEUTRAL` : PWM value for idle/disengaged state.

3. Automation (`GRIP_AUTOCLOSE`)

Can automatically close the gripper when a landing or proximity to a target is detected.

Parameter Breakdown

- `GRIP_ENABLE` : Master switch.
- `GRIP_TYPE` : Hardware driver selection.
- `GRIP_CAN_ID` : Node ID for DroneCAN grippers.

Integration Guide

1. **Pin:** Map a servo output to `Gripper` (Function 28).
2. **Enable:** Set `GRIP_ENABLE = 1` and `GRIP_TYPE` .
3. **Calibrate:** Set the `GRAB` and `RELEASE` PWM values so the mechanism operates without binding.
4. **Control:** Map an RC channel to `Gripper` (RC Option 19) to control it manually.

Developer Notes

- **Library:** `libraries/AP_Gripper` .

GRIP_AUTOCLOSE (Gripper Autoclose time)

DEFAULT: 0.0	RANGE: 0.25 255	UNITS: s

Gripper Autoclose time (GRIP_AUTOCLOSE)

Description

Automates the closure of the gripper after a successful release. This is often used to return a mechanical claw to its idle/stowed position.

- **0 disables automatic closure.**

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





GRIP_CAN_ID (EPM UAVCAN Hardpoint ID)

DEFAULT: 0	RANGE: 0 255

EPM UAVCAN Hardpoint ID (GRIP_CAN_ID)

Description

Uniquely identifies the EPM module on the DroneCAN bus.

GRIP_ENABLE (Gripper Enable/Disable)

DEFAULT: 0	RANGE: 0 1

Gripper Enable/Disable (GRIP_ENABLE)

Description

Master switch for the onboard gripper system, used for delivering or picking up cargo.

Tuning & Behavior

- **Default Value:** 0 (Disabled)
- **Requires reboot to make other GRIP_ parameters visible.**

GRIP_GRAB (Gripper Grab PWM)

DEFAULT: 1900	RANGE: 1000 2000	UNITS: PWM

Gripper Grab PWM (GRIP_GRAB)

Description

The PWM signal value that triggers the "Closed" or "Grab" position of the gripper.

GRIP_NEUTRAL (Neutral PWM)

DEFAULT: 1500	RANGE: 1000 2000	UNITS: PWM

Neutral PWM (GRIP_NEUTRAL)

Description

The idle position for the gripper servo. For EPM, this is the "neutral" state where no power is being applied to the magnets.

GRIP_REGRAB (EPM Gripper Regrab interval)

DEFAULT: 0	RANGE: 0 255	UNITS: s

EPM Gripper Regrab interval (GRIP_REGRAB)

Description

For Electro-Permanent Magnets (EPM), defines how often the flight controller should send a "Grab" pulse to ensure the cargo remains securely attached, compensating for physical jostling.

- **0 disables automatic regrabbing.**

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY

GRIP_RELEASE (Gripper Release PWM)

DEFAULT: 1100	RANGE: 1000 2000	UNITS: PWM

Gripper Release PWM (GRIP_RELEASE)

Description

The PWM signal value that triggers the "Open" or "Release" position of the gripper.

GRIP_TYPE (Gripper Type)

DEFAULT: 0	RANGE: 0 2

Gripper Type (GRIP_TYPE)

Description

Defines the mechanical interface for the gripper.

- **0: None**
- **1: Servo (Standard PWM servo mechanism)**
- **2: EPM (Electro-Permanent Magnet)**

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





GROUND STEERING CONFIGURATION (GROUND)

Overview

The **GROUND** parameter group configures the **Ground Steering** logic for **ArduPlane**.

This system manages the transition from using the Rudder (airflow-based steering) to the Nose/Tail Wheel (mechanical ground-based steering) during takeoff and landing.

Key Concepts

1. Ground Steering Alt (`GROUND_STEER_ALT`)

Defines the altitude (meters) above which ground steering is disabled.

- **Takeoff:** Once the plane climbs above this height, the steering wheel is centered (or follows rudder) to reduce drag and prevent a sudden jerk if the wheels touch back down.

2. Steering Rate (`GROUND_STEER_DPS`)

Limits the maximum rotation rate (degrees per second) of the steering servo.

- **Purpose:** Prevents high-speed "tank slappers" or tip-overs by preventing the pilot (or autopilot) from slamming the steering wheel to full lock at high taxi speeds.

Parameter Breakdown

- `GROUND_STEER_ALT`: Transition altitude (m).
- `GROUND_STEER_DPS`: Max steering slew rate (deg/s).

Integration Guide

- **Nose Wheel:** Map your nose-wheel servo to `GroundSteering` (Function 39) rather than `Rudder` to utilize these specific ground-handling features.

Developer Notes

- **Library:** `ArduPlane/servos.cpp`.
- **Context:** Only relevant for **ArduPlane**.

GROUND_STEER_ALT (Ground steer altitude)

DEFAULT: 0	RANGE: -100 100	UNITS: m

Ground steer altitude (GROUND_STEER_ALT)

Description

Defines the altitude threshold during takeoff at which the aircraft transitions from ground-based steering (nose wheel or tail wheel) to aerodynamic steering (rudder and ailerons).

- **0 (default) disables the altitude-based cutoff.**

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





GROUND_STEER_DPS (Ground steer rate)

DEFAULT: 90	RANGE: 10 360	UNITS: deg/s

Ground steer rate (GROUND_STEER_DPS)

Description

This parameter limits how fast the nose wheel (or tail wheel) servo can move. By slewing the steering command, it prevents the pilot from commanding an instantaneous, sharp turn that could cause the aircraft to tip over (wingtip strike) or skid during high-speed taxiing.

The Mathematics

Rate Limiter:

$$\frac{\Delta \text{Steer Angle}}{\Delta t} \leq \text{GROUND_STEER_DPS}$$

The Engineer's View

Defined in `ArduPlane/Parameters.cpp`.

Tuning & Behavior

- **Default Value:** 90 deg/s
- **Adjustment:** Decrease for top-heavy aircraft or narrow gear widths. Increase for agile ground handling at low speeds.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **GUID** parameter group configures the behavior of the **Guided** flight mode. Guided mode is primarily used for external control by a Ground Control Station (GCS) or a Companion Computer (e.g., "Click to Fly Here" or ROS navigation).

Key Concepts

1. Guided Options (`GUID_OPTIONS`)

A bitmask that alters Guided mode logic.

- **Allow Arming:** Whether the vehicle can be armed while in Guided mode.
- **Ignore Pilot Stick:** If set, moving the RC sticks won't override the Guided mission (Advanced).
- **Face Next WP:** Controls yaw behavior during guided movement.

2. Timeout (`GUID_TIMEOUT`)

Defines how long the autopilot will wait for a new Guided command (e.g., from a script or companion computer) before taking failsafe action (usually Hover or RTL). This is critical for safety in the event of a companion computer crash.

Parameter Breakdown

- `GUID_OPTIONS` : Bitmask of behaviors.
- `GUID_TIMEOUT` : Command timeout (seconds).

Integration Guide

- **Companion Computers:** Set `GUID_TIMEOUT` to a value slightly higher than your navigation loop rate (e.g., if ROS sends targets at 5Hz, set timeout to 0.5s or 1.0s).

Developer Notes

- **Library:** `ArduCopter/mode_guided.cpp` , `ArduPlane/mode_guided.cpp` .

`GUID_OPTIONS` (Guided mode options)

`DEFAULT: 0`

Guided mode options (GUID_OPTIONS)

Description

Configuration bitmask for specific behaviors in Guided mode (Copter).

Tuning & Behavior

- **Default Value:** 0
- **Bitmask:**
  - Bit 0: Allow Arming (in Guided)
  - Bit 1: Allow Takeoff
  - Bit 2: Ignore pilot yaw
  - Bit 3: SCurve for Waypoints

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





GUID_TIMEOUT (Guided mode timeout)

DEFAULT: 3.0	RANGE: 0 10	UNITS: s

Guided mode timeout (GUID_TIMEOUT)

Description

Defines the maximum time the flight controller will wait for a new offboard control message (e.g., from a companion computer) before taking safety action.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

Key Concepts

GUIDED_D (Guided Mode D Gain)

DEFAULT: 0	RANGE: 0 1

Guided Mode D Gain (GUIDED_D)

Description

Damping factor for heading corrections in Plane Guided mode.

GUIDED_D_FF (Guided Mode D-FF)

DEFAULT: 0	RANGE: 0 1

Guided Mode D-FF (GUIDED_D_FF)

Description

Advanced derivative feed-forward gain for the heading controller.

GUIDED_FF (Guided Mode FeedForward)

DEFAULT: 0	RANGE: 0 1

Guided Mode FeedForward (GUIDED_FF)

Description

Feed-forward gain for heading commands in Plane Guided mode.

GUIDED_FLTD (Guided Mode Derivative Filter)

DEFAULT: 0	RANGE: 0 100	UNITS: Hz

Guided Mode Derivative Filter (GUIDED_FLTD)

Description

Low-pass filter frequency for the derivative component of the heading controller in Plane Guided mode.

GUIDED_FLTE (Guided Mode Error Filter)

DEFAULT: 0	RANGE: 0 100	UNITS: Hz

Guided Mode Error Filter (GUIDED_FLTE)

Description

Low-pass filter frequency for the heading error in Plane Guided mode.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





GUIDED_FLTT (Guided Mode Target Filter)

DEFAULT: 0	RANGE: 0 100	UNITS: Hz

Guided Mode Target Filter (GUIDED_FLTT)

Description

Low-pass filter frequency for the target heading in Plane Guided mode.

GUIDED_I (Guided Mode I Gain)

DEFAULT: 0	RANGE: 0 1

Guided Mode I Gain (GUIDED_I)

Description

Corrects for long-term heading errors in Plane Guided mode.

GUIDED_IMAX (Guided Mode I-Max)

DEFAULT: 10	RANGE: 0 4500	UNITS: cdeg

Guided Mode I-Max (GUIDED_IMAX)

Description

Limits the maximum authority of the integral term in Plane Guided mode.

GUIDED_NEF (Guided Mode Notch Error Filter)

DEFAULT: 0

Guided Mode Notch Error Filter (GUIDED_NEF)

Description

Advanced notch filter settings for the heading controller.

GUIDED_NTF (Guided Mode Notch Target Filter)

DEFAULT: 0

Guided Mode Notch Target Filter (GUIDED_NTF)

Description

Advanced notch filter settings for the heading controller target.

GUIDED_P (Guided Mode P Gain)

DEFAULT: 1.0	RANGE: 0 10

Guided Mode P Gain (GUIDED_P)

Description

Determines the responsiveness of the aircraft's heading correction when following offboard commands in Plane Guided mode.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





GUIDED_PDMX (Guided Mode P-D Max)

DEFAULT: 0	RANGE: 0 1

Guided Mode P-D Max (GUIDED_PDMX)

Description

Limits the combined contribution of the proportional and derivative terms.

GUIDED_SMAX (Guided Mode Slew Max)

DEFAULT: 0	RANGE: 0 100

Guided Mode Slew Max (GUIDED_SMAX)

Description

Limits the maximum rate of change for the heading command in Plane Guided mode.

GUIDED_TIMEOUT (Guided mode timeout (Invalid/Legacy))

DEFAULT: 3.0	RANGE: 0 10	UNITS: s

Guided mode timeout (GUIDED_TIMEOUT)

Description

This parameter is not found as GUIDED_TIMEOUT in the main codebase. It likely refers to GUID_TIMEOUT.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **GYR** parameter group (often appearing as `INS_GYR_` in the full parameter list) handles the low-level settings for the **Gyroscopes** (Angular Rate Sensors).

Gyros are the most critical sensors for flight. They measure the vehicle's rotation rate and are the primary input for the high-frequency attitude control loops.

Key Concepts

1. Gyro Calibration (`GYR_CAL`)

Toggles the automatic gyro calibration at boot.

- **Recommendation:** Keep enabled. The autopilot measures the sensor drift while the vehicle is stationary on the ground and subtracts it.

2. Sensor Identification (`GYR_ID`)

Unique hardware identifier for the gyroscope chip, allowing the system to track specific calibrations and health across reboots.

Parameter Breakdown

- `GYR_ID`: Unique hardware ID.
- `GYR_CAL`: Calibration trigger/state.

Integration Guide

- **Startup: DO NOT MOVE** the vehicle while it is booting up. The "Gyro Calibration" occurs during the first few seconds of power-on. Movement during this phase will result in "Bad Gyro Health" or an uncommanded drift in flight.

Developer Notes

- **Library:** `libraries/AP_InertialSensor`.
- **Redundancy:** Most boards have 2 or 3 gyros; the `INS_` library handles the voting and health monitoring between them.

GYR_CAL (Gyro Calibration (Invalid/Legacy))

```
DEFAULT: 0
```

Gyro Calibration (GYR_CAL)

Description

This parameter is not found as a standard top-level GYR_CAL in the main codebase. Gyro calibration is typically handled during boot or via specialized MAVLink commands.

GYR_ID (Gyro ID)

```
DEFAULT: 0
```

Gyro ID (GYR_ID)

Description

Automatically detected hardware ID for the primary gyroscope.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

Key Concepts

GYRO_FILTER (Gyro Filter Frequency)

DEFAULT: 20	RANGE: 0 250	UNITS: Hz

Gyro Filter Frequency (GYRO_FILTER)

Description

The low-pass filter frequency for the gyroscopes. This helps to remove high-frequency noise and vibration from the angular rate data, leading to smoother flight but slightly more latency.

- **Default Value:** 20 Hz (Typical)

GYRO_RATE (Gyro sampling rate)

DEFAULT: 0	RANGE: 0 4000	UNITS: Hz

Gyro sampling rate (GYRO_RATE)

Description

Defines the hardware sampling frequency for the gyroscopes.

- **0: Auto-detect (Recommended)**

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **H** parameter group (often appearing as `H_` or `HELI_`) is the comprehensive configuration set for **Traditional Helicopters** (Single Main Rotor + Tail Rotor, or Tandem).

ArduPilot's helicopter support is world-class, handling complex swashplate mixing, governor control, and autorotation.

Key Concepts

1. Swashplate Mixing (`H_SW_...`)

Defines the mechanical arrangement of the servos driving the main rotor.

- `H_SW_TYPE`: Supports H3, H1, H4, and others.
- `H_SW_COL_DIR`: Reverses the collective pitch direction.

2. Rotor Speed Control (RSC)

Manages the motor/engine power to maintain a constant rotor RPM.

- `H_RSC_MODE`: 1 (Interlocked PWM), 2 (Set-point), 3 (Throttle Curve), or 4 (Governor).
- `H_RSC_SETPOINT`: The target rotor speed.

3. Tail Rotor Configuration (`H_TAIL_...`)

- `H_TAIL_TYPE`: 0 (Servo), 1 (Servo with yaw-to-collective mix), 2 (Direct Drive Fixed Pitch - DDFP).
- `H_COL2YAW`: Collective-to-Yaw compensation (pre-compensates for torque changes).

4. Collective Limits (`H_COL_...`)

- `H_COL_MIN` / `MAX`: The allowable range of collective pitch (e.g., -3 degrees to +12 degrees).

Parameter Breakdown

- `H_RSC_MODE`: The "throttle" logic for the main rotor.
- `H_COL_HOVER`: The collective pitch required for a stable hover (learned or manual).
- `H_FLYBAR_MODE`: Enables logic for legacy mechanical flybar heads (0 for Flybarless).

Integration Guide

- **Safety: Always use a Motor Interlock switch.** Helicopters are physically dangerous; the Interlock ensures the rotor only spins when commanded.
- **Servo Check:** Use the `H_SV_TEST` feature to verify swashplate movement before mounting blades.

Developer Notes

- **Library:** `libraries/AP_Motors/AP_MotorsHeli_Single.cpp`, `AP_MotorsHeli_Tandem.cpp`.
- **Complexity:** Helicopter tuning is significantly more complex than Multicopter tuning and relies heavily on accurate collective/yaw pre-compensation.

H_COL2YAW (Collective-Yaw Mixing)

DEFAULT: 0	RANGE: -2 2

Collective-Yaw Mixing (H_COL2YAW)

Description

Compensates for the changing torque of the main rotor as collective pitch increases. By automatically adding a pre-calculated amount of tail rotor thrust, this reduces the work required by the yaw PID controller and results in a more stable heading during aggressive climbs.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





H_COL2_MAX (Tiltrotor maximum VTOL angle)

DEFAULT: 45	RANGE: 20 80	UNITS: deg

Tiltrotor maximum VTOL angle

Note: This parameter functions identically to H_COL_MAX.

H_COL2_MIN (Swash 2 Minimum Collective Pitch)

DEFAULT: AP_MOTORS_HELI_DUAL_COLLECTIVE2_MIN	RANGE: 1000 2000	UNITS: PWM

Swash 2 Minimum Collective Pitch

Note: This parameter functions identically to H_COL_MIN.

H_COL_ANG_MAX (Collective Blade Pitch Angle Maximum)

DEFAULT: 10	RANGE: 5 20	UNITS: deg

Collective Blade Pitch Angle Maximum (H_COL_ANG_MAX)

Description

The physical blade pitch angle (in degrees) when the swashplate is at its maximum position. This must be measured during setup.

H_COL_ANG_MIN (Collective Blade Pitch Angle Minimum)

DEFAULT: -10	RANGE: -20 0	UNITS: deg

Collective Blade Pitch Angle Minimum (H_COL_ANG_MIN)

Description

The physical blade pitch angle (in degrees) when the swashplate is at its minimum position. This must be measured during setup.

H_COL_HOVER (Collective Hover Value)

DEFAULT: 0.5	RANGE: 0.3 0.8

Collective Hover Value (H_COL_HOVER)

Description

The specific collective pitch setting required to maintain a stationary hover. This serves as the center-point for altitude control logic.

H_COL_LAND_MIN (Collective Pitch Minimum when Landed)

DEFAULT: -2	RANGE: -5 0	UNITS: deg

Collective Pitch Minimum when Landed (H_COL_LAND_MIN)

Description

Restricts the minimum collective pitch while the aircraft is detected as "Landed" in automated modes (like AltHold). This prevents the aircraft from being forced onto the ground with excessive negative pitch.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





H_COL_MAX (Maximum Collective Pitch)

DEFAULT: 1750	RANGE: 1000 2000	UNITS: PWM

Maximum Collective Pitch (H_COL_MAX)

Description

Defines the maximum physical travel limit of the swashplate for collective pitch control.

H_COL_MIN (Minimum Collective Pitch)

DEFAULT: 1250	RANGE: 1000 2000	UNITS: PWM

Minimum Collective Pitch (H_COL_MIN)

Description

Defines the minimum physical travel limit of the swashplate for collective pitch control.

H_COL_ZERO_THRST (Collective Blade Pitch at Zero Thrust)

DEFAULT: 0.0	RANGE: -5 0	UNITS: deg

Collective Blade Pitch at Zero Thrust (H_COL_ZERO_THRST)

Description

The blade pitch angle (typically 0 degrees for symmetrical blades) at which the main rotor produces zero lift.

H_CYC_MAX (Maximum Cyclic Pitch Angle)

DEFAULT: 4500	RANGE: 0 4500

Maximum Cyclic Pitch Angle (H_CYC_MAX)

Description

Defines the maximum angular authority of the swashplate for roll and pitch maneuvers.

- **Typically adjusted to achieve 6-7 degrees of physical blade pitch change.**

H_DCP_SCALER (Dual Cyclic Pitch Scaler)

DEFAULT: 0.25	RANGE: 0 1.0

Dual Cyclic Pitch Scaler (H_DCP_SCALER)

Description

`H_DCP_SCALER` is used for Tandem (two-rotor) helicopters like the CH-47 Chinook.

In a tandem helicopter, yaw is controlled by tilting the two swashplates in opposite directions (Differential Cyclic Pitch). This parameter scales that effect.

- **Tuning:** Adjust so that the yaw response is balanced and does not cause unwanted rolling or pitching.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





H_DCP_TRIM (Direct Cyclic Pitch Trim)

DEFAULT: 0	RANGE: -0.1 0.1

Direct Cyclic Pitch Trim (H_DCP_TRIM)

Description

`H_DCP_TRIM` provides a static offset to the cyclic pitch.

H_DCP_YAW (Direct Cyclic Pitch Yaw)

DEFAULT: 0	RANGE: -1 1

Direct Cyclic Pitch Yaw (H_DCP_YAW)

Description

`H_DCP_YAW` adds a tail kick when you move the cyclic stick.

This is a feed-forward term. If you slam the stick forward, the main rotor torque changes. This parameter tells the tail to compensate instantly, rather than waiting for the gyro to detect a yaw error.

H_DDFP_BAT_IDX (DDVP Battery Index)

DEFAULT: 0	RANGE: 0 10

DDVP Battery Index (H_DDFP_BAT_IDX)

Description

`H_DDFP_BAT_IDX` tells the tail mixer which battery to watch.

If you have a separate battery for the tail motor (uncommon) or if your main battery is on Monitor 2, set this index accordingly.

- **0:** Battery Monitor 1.
- **1:** Battery Monitor 2.

H_DDFP_BAT_V_MAX (DDVP Battery Voltage Max)

DEFAULT: 0	RANGE: 0 100	UNITS: V

DDVP Battery Voltage Max (H_DDFP_BAT_V_MAX)

Description

`H_DDFP_BAT_V_MAX` sets the "Full Charge" reference.

When the battery is at this voltage, the mixer applies 100% of the requested throttle.

Tuning & Behavior

- **Default Value:** 0 (Disabled).
- **Recommendation:** Set to 4.2V * Cell Count (e.g. 25.2V for 6S).

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





H_DDFP_BAT_V_MIN (DDVP Battery Voltage Min)

DEFAULT: 0	RANGE: 0 100	UNITS: V

DDVP Battery Voltage Min (H_DDFP_BAT_V_MIN)

Description

`H_DDFP_BAT_V_MIN` sets the "Empty" reference.

As the battery drops to this level, the mixer boosts the PWM signal to the ESC to compensate for the lower voltage, keeping the RPM consistent.

Tuning & Behavior

- **Default Value:** 0.
- **Recommendation:** Set to 3.3V * Cell Count.

H_DDFP_SPIN_MAX (DDFP Tail Motor Spin Maximum)

DEFAULT: 0.95	RANGE: 0.8 1.0

DDFP Tail Motor Spin Maximum (H_DDFP_SPIN_MAX)

Description

`H_DDFP_SPIN_MAX` defines the saturation point for the tail motor.

H_DDFP_SPIN_MIN (DDFP Tail Motor Spin Minimum)

DEFAULT: 0.15	RANGE: 0 0.3

DDFP Tail Motor Spin Minimum (H_DDFP_SPIN_MIN)

Description

`H_DDFP_SPIN_MIN` sets the lower limit for a fixed-pitch tail motor.

If you are using a standard drone motor for your tail, this is the point where it produces reliable thrust. Setting this correctly ensures the tail doesn't "Stall" or stop spinning during low-torque maneuvers.

H_DDFP_THST_EXPO (DDVP Tail Thrust Expo)

DEFAULT: 0	RANGE: 0 1

DDVP Tail Thrust Expo (H_DDFP_THST_EXPO)

Description

`H_DDFP_THST_EXPO` linearizes the thrust of the tail motor.

Propeller thrust is roughly proportional to `RPM^2`. This parameter applies the inverse curve so that a 50% yaw command results in 50% thrust, not 25%.

Tuning & Behavior

- **0.65:** Typical for 5-inch props.
- **0:** Linear output (bad for thrust).

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





H_DUAL_MODE (Dual Heli Rotor Mode)

DEFAULT: 0	RANGE: 0 2

Dual Heli Rotor Mode (H_DUAL_MODE)

Description

`H_DUAL_MODE` tells ArduPilot what kind of multi-rotor helicopter you are flying.

Standard helicopters have one main rotor and one tail rotor. "Dual" helicopters use two large rotors to generate lift.

- **0: Tandem.** Rotors are arranged front-to-back (like a Chinook CH-47).
- **1: Coaxial.** Two rotors stacked on top of each other on a single shaft.
- **2: Intermeshing.** Rotors are side-by-side and tilt slightly towards each other (like a K-MAX).

Tuning & Behavior

- **Default Value:** 0.
- **Reboot Required:** Yes.

H_FLYBAR_MODE (Flybar Mode Selector)

DEFAULT: 0	RANGE: 0 1

Flybar Mode Selector (H_FLYBAR_MODE)

Description

Tells the attitude controller if the aircraft has a physical flybar.

- **0: NoFlybar (Modern digital FBL system)**
- **1: Flybar (Legacy mechanical stabilization)**

H_GYR_GAIN (External Gyro Gain)

DEFAULT: 0	RANGE: 0 1000	UNITS: PWM

External Gyro Gain (H_GYR_GAIN)

Description

Specifies the gain value sent to an external tail-lock gyroscope.

- **Only used if H_TAIL_TYPE is set to 1 (Servo with External Gyro).**

H_GYR_GAIN_ACRO (ACRO External Gyro Gain)

DEFAULT: 0	RANGE: 0 1000	UNITS: PWM

ACRO External Gyro Gain (H_GYR_GAIN_ACRO)

Description

`H_GYR_GAIN_ACRO` allows you to have a separate tail gyro sensitivity for aerobatic flying.

In `ACRO` mode, pilots often want a more aggressive or direct response from the tail compared to stabilized modes. This parameter overrides the standard H_GYR_GAIN when you switch to ACRO.

- **0 (Default):** Use the standard `H_GYR_GAIN` for all modes.
- **> 0:** Use this specific value when in ACRO mode.

Tuning & Behavior

- **Recommendation:** Only used if `H_TAIL_TYPE = 1` (External Gyro).

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





H_HOVER_LEARN (Hover Value Learning)

DEFAULT: 2	RANGE: 0 2

Hover Value Learning (H_HOVER_LEARN)

Description

Controls whether the flight controller automatically updates the H_COL_HOVER parameter based on actual flight data.

- **0: Disabled**
- **1: Learn (Volatile)**
- **2: Learn and Save (Permanent)**

H_OFFSET (Heli Cyclic Offset)

DEFAULT: 0	RANGE: -0.1 0.1

Heli Cyclic Offset (H_OFFSET)

Description

`H_OFFSET` provides a global trim for the swashplate.

H_OPTIONS (Heli_Options)

DEFAULT: 1

Heli_Options (H_OPTIONS)

Description

Configuration bitmask for helicopter flight behaviors.

- **Bit 0: Use Leaky I (Recommended for most setups to prevent integrator windup on the ground).**

H_RSC_AROT_ENBL (Enable Autorotation Handling)

DEFAULT: 0	RANGE: 0 1

Enable Autorotation Handling (H_RSC_AROT_ENBL)

Description

`H_RSC_AROT_ENBL` activates specialized logic for "Dead-Stick" landings on helicopters.

Autorotation is a maneuver where the main rotor is driven by the upward flow of air during a descent, allowing the helicopter to land safely even if the motor fails. This parameter enables the autopilot to recognize an autorotation state and correctly manage the transition back to powered flight (Bailout) if the engine is restarted in mid-air.

Tuning & Behavior

- **Default Value:** 0 (Disabled).
- **Recommendation:** Enable this if you practice autorotations or have a high-value helicopter where a mid-air motor restart is possible.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





H_RSC_AROT_IDLE (Autorotation Idle Throttle)

DEFAULT: 0	RANGE: 0 40	UNITS: %

Autorotation Idle Throttle (H_RSC_AROT_IDLE)

Description

`H_RSC_AROT_IDLE` keeps the motor "Spinning but not Engaging" during an autorotation.

This is primarily for combustion engines or electric ESCs with a "bailout window." It ensures the engine doesn't drop to a true idle or stop completely, allowing for a much faster re-engagement of power if needed.

H_RSC_AROT_RAMP (Autorotation Bailout Ramp Time)

DEFAULT: 2.0	RANGE: 0.1 10.0	UNITS: s

Autorotation Bailout Ramp Time (H_RSC_AROT_RAMP)

Description

`H_RSC_AROT_RAMP` controls the "Power-Up Speed" after an autorotation.

If you abort an autorotation (Bailout), you want power back quickly, but not so fast that it snaps the main gears or causes a tail kick.

- **ESC with Bailout Mode:** Set this to **0.1s**. The ESC will handle its own fast-ramp.
- **Standard ESC:** Set to **2.0s - 4.0s**.

H_RSC_AROT_RUNUP (Autorotation Bailout Runup Time)

DEFAULT: 3.0	RANGE: 1.0 10.0	UNITS: s

Autorotation Bailout Runup Time (H_RSC_AROT_RUNUP)

Description

`H_RSC_AROT_RUNUP` is a safety timer for autonomous flight.

It tells the autopilot how long it must wait for the head speed to recover before it is allowed to increase the collective pitch. If the autopilot tries to climb before the rotor is back up to speed, the extra drag will "Bog" the motor and likely cause a crash.

Tuning & Behavior

- **Default Value:** 3.0 seconds.
- **Constraint:** Must be at least 1 second longer than H_RSC_AROT_RAMP.

H_RSC_CLDWN_TIME (Cooldown Time)

DEFAULT: 0	RANGE: 0 120	UNITS: s

Cooldown Time (H_RSC_CLDWN_TIME)

Description

Automates the engine cooldown procedure for combustion or turbine engines. After landing and disarming, the engine will run at an elevated idle for this many seconds before shutting down.

- **0 disables the cooldown feature.**

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





H_RSC_CRITICAL (Critical Rotor Speed)

DEFAULT: 50	RANGE: 0 100	UNITS: %

Critical Rotor Speed (H_RSC_CRITICAL)

Description

A safety threshold representing the minimum RPM required to maintain flight. If the (estimated or measured) rotor speed falls below this value, the autopilot will declare a loss of power.

- **Commonly set so that the threshold is crossed approximately 3 seconds after a power loss.**

H_RSC_GOV_COMP (Governor Torque Compensator)

DEFAULT: 25	RANGE: 0 70	UNITS: %

Governor Torque Compensator (H_RSC_GOV_COMP)

Description

Adjusts the sensitivity of the governor's altitude compensation.

H_RSC_GOV_DROOP (Governor Droop Compensator)

DEFAULT: 25	RANGE: 0 100	UNITS: %

Governor Droop Compensator (H_RSC_GOV_DROOP)

Description

Controls how aggressively the governor increases throttle when a drop in rotor RPM is detected due to increased load (e.g., during a climb).

H_RSC_GOV_FF (Governor Feedforward)

DEFAULT: 50	RANGE: 0 100	UNITS: %

Governor Feedforward (H_RSC_GOV_FF)

Description

Provides an immediate throttle increase when collective pitch is increased, anticipating the load before the RPM actually drops.

H_RSC_GOV_RANGE (Governor Operational Range)

DEFAULT: 100	RANGE: 50 200	UNITS: RPM

Governor Operational Range (H_RSC_GOV_RANGE)

Description

Defines the safety window for the internal governor. If the measured RPM deviates from the target by more than this amount, the governor will automatically disengage and fall back to the throttle curve.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY

H_RSC_GOV_RPM (Rotor RPM Setting)

DEFAULT: 1500	RANGE: 800 3500	UNITS: RPM

Rotor RPM Setting (H_RSC_GOV_RPM)

Description

The target operating speed for the rotors when using the AutoThrottle mode. The flight controller will attempt to maintain this exact RPM using closed-loop control.

- **Requires a functional RPM sensor.**

H_RSC_GOV_TORQUE (Governor Torque Limiter)

DEFAULT: 30	RANGE: 10 60	UNITS: %

Governor Torque Limiter (H_RSC_GOV_TORQUE)

Description

Limits how quickly the engine can increase torque during the initial transition to governor control, ensuring a smooth engagement of the AutoThrottle.

H_RSC_IDLE (Throttle Output at Idle)

DEFAULT: 0	RANGE: 0 50	UNITS: %

Throttle Output at Idle (H_RSC_IDLE)

Description

Defines the engine idle throttle for combustion-powered helicopters. For electric helicopters, this is typically set to 0.

H_RSC_MODE (Rotor Speed Control Mode)

DEFAULT: 1	RANGE: 1 4

Rotor Speed Control Mode (H_RSC_MODE)

Description

Configures how the flight controller determines the throttle output for the main rotor.

- **1: RC Passthrough (Direct control from transmitter)**
- **2: External Gov SetPoint (Uses H_RSC_SETPOINT)**
- **3: Throttle Curve (Uses 5-point curve based on collective)**
- **4: AutoThrottle (Advanced internal governor requiring RPM sensor)**

H_RSC_RAMP_TIME (Throttle Ramp Time)

DEFAULT: 1	RANGE: 0 60	UNITS: s

Throttle Ramp Time (H_RSC_RAMP_TIME)

Description

Controls how quickly the throttle signal increases when starting the motor. A slower ramp prevents sudden torque spikes that could damage the gears or drive belts.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





H_RSC_RUNUP_TIME (Rotor Runup Time)

DEFAULT: 10	RANGE: 0 60	UNITS: s

Rotor Runup Time (H_RSC_RUNUP_TIME)

Description

Specifies the physical time required for the rotors to reach full speed. This is used by the flight controller to ensure that it doesn't attempt to take off or enter stabilized modes before the rotors have sufficient inertia.

- **Must be set longer than H_RSC_RAMP_TIME.**

H_RSC_SETPOINT (External Motor Governor Setpoint)

DEFAULT: 70	RANGE: 0 100	UNITS: %

External Motor Governor Setpoint (H_RSC_SETPOINT)

Description

Defines the target throttle percentage sent to an external Electronic Speed Controller (ESC) or governor. This value is used when the pilot enables the "Motor Interlock" (throttle hold off).

- **Typical values range from 60% to 80% depending on the desired head speed.**

H_RSC_SLEWRATE (Throttle Slew Rate)

DEFAULT: 0	RANGE: 0 500	UNITS: %/s

Throttle Slew Rate (H_RSC_SLEWRATE)

Description

Limits the maximum speed of throttle signal changes. This prevents sudden engine surging or mechanical strain.

- **0: Unlimited**
- **100: Full range in 1 second**

H_RSC_THRCRV_0 (Throttle Curve at 0% Collective)

DEFAULT: 25	RANGE: 0 100	UNITS: %

Throttle Curve at 0% Coll (H_RSC_THRCRV_0)

Description

Part of a 5-point throttle curve (0, 25, 50, 75, 100). Defines the motor power required when collective pitch is at its minimum.

H_RSC_THRCRV_100 (Throttle Curve at 100% Collective)

DEFAULT: 100	RANGE: 0 100	UNITS: %

Throttle Curve at 100% Coll (H_RSC_THRCRV_100)

Description

Final point of the 5-point throttle curve.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





H_RSC_THRCRV_25 (Throttle Curve at 25% Collective)

DEFAULT: 32	RANGE: 0 100	UNITS: %

Throttle Curve at 25% Coll (H_RSC_THRCRV_25)

Description

Second point of the 5-point throttle curve.

H_RSC_THRCRV_50 (Throttle Curve at 50% Collective)

DEFAULT: 38	RANGE: 0 100	UNITS: %

Throttle Curve at 50% Coll (H_RSC_THRCRV_50)

Description

Middle point of the 5-point throttle curve.

H_RSC_THRCRV_75 (Throttle Curve at 75% Collective)

DEFAULT: 50	RANGE: 0 100	UNITS: %

Throttle Curve at 75% Coll (H_RSC_THRCRV_75)

Description

Fourth point of the 5-point throttle curve.

H_SV_MAN (Manual Servo Mode)

DEFAULT: 0	RANGE: 0 4

Manual Servo Mode (H_SV_MAN)

Description

A setup tool that forces the swashplate servos into specific positions (Max, Min, or Passthrough) to assist in mechanical leveling and linkage adjustment.

- **WARNING: Must be set to 0 (Disabled) before attempting flight.**

H_SV_TEST (Boot-up Servo Test Cycles)

DEFAULT: 0	RANGE: 0 10

Boot-up Servo Test Cycles (H_SV_TEST)

Description

Commands the swashplate servos to perform a pre-defined movement sequence upon power-up, allowing the pilot to visually confirm that all linkages are functioning correctly.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





H_SW2_COL_DIR (Swashplate Collective Control Direction)

DEFAULT: COLLECTIVE_DIRECTION_NORMAL	RANGE: null	UNITS: null

Swashplate Collective Control Direction

Note: This parameter functions identically to H_SW_COL_DIR.

H_SW2_H3_ENABLE (Swashplate 2 Type)

DEFAULT: 0	RANGE: 0 1

Swashplate 2 Type (H_SW2_H3_ENABLE)

Description

`H_SW2_H3_ENABLE` configures the physical servo layout for the second rotor on a Dual-Rotor helicopter (Tandem, Coaxial, or Intermeshing).

- **0:** Non-CCPM (Standard separate servos).
- **1:** H3-120 (3 servos arranged at 120-degree intervals).

Tuning & Behavior

- **Default Value:** 0.
- **Recommendation:** Match the physical layout of your second swashplate.

H_SW2_H3_PHANG (Swashplate 2 Phase Angle)

DEFAULT: 0	RANGE: -30 30	UNITS: deg

Swashplate 2 Phase Angle (H_SW2_H3_PHANG)

Description

`H_SW2_H3_PHANG` corrects the timing of the second rotor's control inputs.

Because of gyroscopic precession, pushing the swashplate "Forward" doesn't always make the blades tilt "Forward." This parameter adds an offset (Lead/Lag) to align the physical tilt with the intended movement.

Tuning & Behavior

- **Default Value:** 0.
- **Tuning:** Hover the heli and push the Pitch stick. If the heli also rolls, adjust this value until the roll component is gone.

H_SW2_H3_SV1_POS (Antenna Z position offset)

DEFAULT: 0.0f	RANGE: -5 5	UNITS: m

Antenna Z position offset

Note: This parameter functions identically to H_SW_H3_SV1_POS.

H_SW2_H3_SV2_POS (Antenna Z position offset)

DEFAULT: 0.0f	RANGE: -5 5	UNITS: m

Antenna Z position offset

Note: This parameter functions identically to H_SW_H3_SV2_POS.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





H_SW2_H3_SV3_POS (Antenna Z position offset)

DEFAULT: 0.0f	RANGE: -5 5	UNITS: m

Antenna Z position offset

Note: This parameter functions identically to H_SW_H3_SV3_POS.

H_SW2_LIN_SVO (Linearize Swashplate Servo Mechanical Throw)

DEFAULT: 0	RANGE: null	UNITS: null

Linearize Swashplate Servo Mechanical Throw

Note: This parameter functions identically to H_SW_LIN_SVO.

H_SW2_TYPE (Tiltrotor type)

DEFAULT: TILT_TYPE_CONTINUOUS	RANGE: null	UNITS: null

Tiltrotor type

Note: This parameter functions identically to H_SW_TYPE.

H_SW_H3_ENABLE (H3 Swashplate Enable)

DEFAULT: 0	RANGE: 0 1

H3 Swashplate Enable (H_SW_H3_ENABLE)

Description

`H_SW_H3_ENABLE` activates advanced setup for 3-servo (H3) swashplates.

Normally, ArduPilot assumes a standard 120° swashplate layout. If your helicopter has a custom mechanical layout where the servos are not at the standard positions, you enable this parameter to manually define the angle of each servo using `H_SW_H3_SV1_POS` etc.

- **0: Default.** Use the layout selected in `H_SW_TYPE`.
- **1: Manual.** Use custom servo angles.

H_SW_H3_PHANG (H3 Generic Phase Angle Comp)

DEFAULT: 0	RANGE: -30 30	UNITS: deg

H3 Generic Phase Angle Comp (H_SW_H3_PHANG)

Description

`H_SW_H3_PHANG` corrects for "Control Coupling" – a common issue on helicopters where a pitch command causes a roll, or vice versa.

Because of gyroscopic precession, a force applied to the main rotor blades takes effect 90 degrees later in the rotation. If your swashplate linkages are not perfectly aligned with the blade grips, the drone will "drift" sideways when you tell it to go forward. This parameter rotates the virtual control axes to fix that alignment error.

Tuning & Behavior

- **Default Value:** 0.
- **Tuning:** If you push the pitch stick forward and the heli rolls to the right, add a few degrees of phase angle until the movement is purely longitudinal.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





H_SW_H3_SV1_POS (H3 Servo 1 Position)

DEFAULT: -60	RANGE: -180 180	UNITS: deg

H3 Servo 1 Position (H_SW_H3_SV1_POS)

Description

`H_SW_H3_SV1_POS` defines the physical location of the first swashplate servo on a 3-servo head.

Angles are measured from the nose (0°).

- **-60:** Standard position for many RC helicopters (Rear-Left).
- **60:** Standard position (Rear-Right).
- **180:** Standard position (Front).

Tuning & Behavior

- **Usage:** Only used if H_SW_H3_ENABLE is set to 1.

H_SW_H3_SV2_POS (Swashplate Servo 2 Position)

DEFAULT: 120	RANGE: -180 180	UNITS: deg

Swashplate Servo 2 Position (H_SW_H3_SV2_POS)

Description

`H_SW_H3_SV2_POS` defines the location of the second servo.

H_SW_H3_SV3_POS (Swashplate Servo 3 Position)

DEFAULT: 240	RANGE: -180 180	UNITS: deg

Swashplate Servo 3 Position (H_SW_H3_SV3_POS)

Description

`H_SW_H3_SV3_POS` defines the location of the third servo.

H_SW_TYPE (Swashplate Type)

DEFAULT: 0	RANGE: 0 5

Swashplate Type (H_SW_TYPE)

Description

Defines the physical layout of the swashplate servos (e.g., 120-degree CCPM vs. 90-degree).

- **0: H3 Generic (3 servos)**
- **1: H1 (Non-CCPM)**
- **3: H3_120 (Standard 120-degree layout)**

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





H_TAIL_SPEED (DDVP Tail ESC speed)

DEFAULT: 0	RANGE: 0 100	UNITS: %

DDVP Tail ESC speed (H_TAIL_SPEED)

Description

`H_TAIL_SPEED` sets the target RPM for a **Direct Drive Variable Pitch (DDVP)** tail.

In this setup, the tail rotor is spun by its own motor at a constant speed, while a servo changes the blade pitch to control yaw. This parameter defines how much power is sent to the tail motor ESC.

Tuning & Behavior

- **Default Value:** 0.
- **Recommendation:** Set high enough to ensure the tail has enough authority to counter main rotor torque during high-load maneuvers, but not so high that you waste battery or stress the motor.

H_TAIL_TYPE (Tail Type)

DEFAULT: 0	RANGE: 0 4

Tail Type (H_TAIL_TYPE)

Description

Defines the hardware configuration used for yaw control on a single-rotor helicopter.

- **0: Servo only (Standard mechanical tail)**
- **1: Servo with External Gyro**
- **2: DirectDrive VarPitch (Motor on tail, pitch servo controlled)**
- **3: DirectDrive FixedPitch CW (Motor speed controls yaw, clockwise rotor)**
- **4: DirectDrive FixedPitch CCW (Motor speed controls yaw, counter-clockwise rotor)**

H_YAW_REV_EXPO (Yaw Reverse Exponential)

DEFAULT: 0	RANGE: 0 1

Yaw Reverse Exponential (H_YAW_REV_EXPO)

Description

`H_YAW_REV_EXPO` smooths out the tail response when the helicopter is flying backward.

H_YAW_SCALER (Yaw Axis Gain Scaler)

DEFAULT: 1.0	RANGE: 0 2

Yaw Axis Gain Scaler (H_YAW_SCALER)

Description

`H_YAW_SCALER` is specific to **Dual Rotor** helicopters (like Coaxial or Tandem designs).

It provides a master gain for the yaw axis. Because dual-rotor helis use complex torque-matching or differential pitch to yaw, the standard PID gains might not have enough range. This parameter allows you to scale the entire output to get the desired yaw rate.

Tuning & Behavior

- **Default Value:** 1.0.
- **Recommendation:** If the helicopter is lazy in yaw even with high P-gains, increase this to **1.5**.
- **Safety:** Only active for Dual Heli builds.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





H_YAW_TRIM (Tail Rotor Trim)

DEFAULT: 0.0	RANGE: 0 1

Tail Rotor Trim (H_YAW_TRIM)

Description

Provides a static offset to the tail rotor output (primarily for DDFP tails) to account for constant main-rotor torque during hover.

Linearize Swashplate Servo Mechanical Throw ()

H_SW_LIN_SVO: Linearize Swashplate Servo Mechanical Throw

Description

This linearizes the swashplate servo's mechanical output to account for nonlinear output due to arm rotation. This requires a specific setup procedure to work properly. The servo arm must be centered on the mechanical throw at the servo trim position and the servo trim position kept as close to 1500 as possible. Leveling the swashplate can only be done through the pitch links. See the ardupilot wiki for more details on setup.

Values

Value	Meaning
0	Disabled
1	Enabled

Description

Enables compensation for the non-linear mechanical movement of rotary servo arms driving the swashplate. When enabled, it adjusts the servo output to produce linear swashplate movement relative to the input, provided the mechanical setup is correct (arms centered at 1500 trim).

Source Code

ardupilot/libraries/AP_Motors/AP_MotorsHeli_Swash.cpp

Swashplate Collective Control Direction ()

H_SW_COL_DIR: Swashplate Collective Control Direction

Description

Direction collective moves for positive pitch. 0 for Normal, 1 for Reversed.

Values

Value	Meaning
0	Normal
1	Reversed

Description

This parameter controls the direction of collective pitch movement. It reverses the collective input to the swashplate mixing.

- **Normal (0):** Standard collective direction.
- **Reversed (1):** Inverted collective direction.

In the code, when set to **Reversed**, the collective input is inverted (`collective = 1 - collective`).

Source Code

ardupilot/libraries/AP_Motors/AP_MotorsHeli_Swash.cpp

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **HARDPOINT** parameter group (specifically within **AP_Periph**) configures the DroneCAN **Hardpoint** subsystem.

A "Hardpoint" is a designated mechanical output used for payload release, crop spraying, or robotic actuation that is commanded via the CAN bus rather than a dedicated PWM wire from the main autopilot.

Key Concepts

1. Hardpoint ID (`HARDPOINT_ID`)

Assigns a unique identifier to the peripheral output. The main autopilot sends a MAVLink or DroneCAN command specifying "Activate Hardpoint 1," and only the device with `HARDPOINT_ID = 1` will actuate.

Parameter Breakdown

- `HARDPOINT_ID`: Identifier (0-15).
- `HARDPOINT_RATE`: The frequency (Hz) at which the hardpoint status is reported back to the autopilot.

Integration Guide

1. **Node Setup:** On your `AP_Periph` node (e.g., a CAN-to-PWM converter), set `HARDPOINT_ID`.
2. **Autopilot:** Use the `Do_Hardpoint` mission command or a GCS trigger to actuate the device.

Developer Notes

- **Library:** `Tools/AP_Periph`.
- **Message:** Uses the `uavcan.equipment.hardpoint.Status` and `Command` messages.

`HARDPOINT_ID` (Hardpoint ID)

DEFAULT: 0	RANGE: 0 15

Hardpoint ID (HARDPOINT_ID)

Description

This parameter assigns a specific ID to the PWM output of a DroneCAN peripheral, designating it as a "Hardpoint" (e.g., a weapon release, magnet, or gripper). The flight controller can then actuate this specific output by sending a matching DroneCAN hardpoint command.

The Mathematics

ID Matching:

$$\text{Actuate} \iff \text{Command.ID} == \text{HARDPOINT_ID}$$

The Engineer's View

Defined in `Tools/AP_Periph/Parameters.cpp`. Enables the `uavcan.equipment.hardpoint` subscriber.

Tuning & Behavior

- **Default Value:** 0
- **Usage:** Set unique IDs for different mechanisms (e.g., Left Wing Release = 1, Right Wing Release = 2).

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY

HARDPOINT_RATE: Hardpoint PWM rate

Description

This parameter sets the rate at which DroneCAN Hardpoint commands are broadcast when a PWM input is detected. This applies when the AP_Periph node is configured to read a PWM input (on a designated `PWM_HARDPOINT_PIN`) and forward it as a DroneCAN `uavcan.equipment.hardpoint.Command` message.

Values

- **Range:** 10 to 100
- **Units:** Hz
- **Increment:** 1

Description

If your AP_Periph device is wired to read a PWM signal (e.g., from an RC receiver) to trigger a Hardpoint, this parameter controls how frequently that trigger command is sent over the CAN bus while the PWM signal is active.

- **Higher Rate:** Updates the command status more frequently, potentially useful for rapid-response mechanisms, but consumes more CAN bus bandwidth.
- **Lower Rate:** Conserves bandwidth but increases latency for the command update.

Source Code

ardupilot/Tools/AP_Periph/Parameters.cpp

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **HDG** parameter group (specifically `HDG_CORR_EN` and `HDG_CORR_WT`) configures the **Heading Correction** logic, primarily within the context of **Autonomous Docking** (`DOCK` mode) for ArduRover.

During the final stages of a docking approach, the vehicle must align its heading perfectly with the docking station entrance to avoid a collision.

Key Concepts

1. Correction Enable (`HDG_CORR_EN`)

Toggles the alignment correction logic. When enabled, the rover prioritizes heading alignment over direct-to-waypoint navigation during the "Final Approach" phase.

2. Weighting (`HDG_CORR_WT`)

Defines the strength of the correction.

- **Higher values:** Vehicle will maneuver more aggressively to line up the heading.
- **Lower values:** Smoother transition but may result in "crooked" docking.

Developer Notes

- **Library:** `Rover/mode_dock.cpp`.
- **Context:** Only relevant for **ArduRover**.

Dock mode heading correction enable/disable ()

HDG_CORR_EN: Dock mode heading correction enable/disable

Description

When enabled, the autopilot modifies the path to approach the target head-on along the desired line of approach (`DOCK_DIR`) in Dock mode.

Values

Value	Meaning
0	Disabled
1	Enabled

Description

This parameter forces the Rover (or Boat) to align with a specific compass heading (`DOCK_DIR`) *before* it reaches the dock, ensuring a straight-in approach. This is useful for docking into a slip or connecting to a charging station where the angle of arrival is critical.

- **Enabled (1):** The vehicle generates a virtual waypoint along the approach vector to steer the vehicle onto the correct line.
- **Disabled (0):** The vehicle drives directly towards the dock from its current position, regardless of the angle.

Source Code

ardupilot/Rover/mode_dock.cpp

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





HDG_CORR_WT: Dock mode heading correction weight

Description

This value describes how aggressively the vehicle tries to correct its heading to be on the desired line of approach.

Values

- **Range:** 0.00 to 0.90
- **Increment:** 0.05
- **Default:** 0.75

Description

This parameter controls the position of the "virtual target" used to align the vehicle with the dock. It acts as a "carrot on a stick" placed along the desired approach line between the vehicle and the dock.

- **Value:** Represents the fraction of the distance from the vehicle to the dock where the virtual target is placed (relative to the dock).
  - **0.75 (Default):** The target is placed at 75% of the distance to the dock (effectively 25% ahead of the vehicle) along the approach line.
  - **Higher Value (closer to 1.0):** The target is placed closer to the vehicle's current distance but projected onto the line. This might result in smoother but slower convergence to the line.
  - **Lower Value:** The target is placed closer to the dock. This forces a sharper, more aggressive turn to get onto the line quickly.

Source Code

ardupilot/Rover/mode_dock.cpp

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **HEEL** parameter group (specifically `HEEL_MAX`) configures the safety limits for **ArduRover Sailboats**.

"Heeling" refers to the lean angle of a sailboat caused by wind pressure on the sails. While some heel is normal, excessive heel can lead to a capsize or lose rudder effectiveness.

Key Concepts

1. Heel Limit (`HEEL_MAX`)

This is a safety threshold. If the sailboat leans (heels) past this angle (degrees), the autopilot will automatically **ease the sails** (release the sheets) to dump wind pressure and right the boat.

- **Purpose:** Capsize prevention.

Parameter Breakdown

- `HEEL_MAX`: Maximum allowable heel angle in degrees.
  - **0:** Disabled (Danger!).
  - **20-30:** Common for many small monohulls.

Integration Guide

- **Tuning:** Set this slightly below the angle where your hull's deck begins to submerge or where you lose steering authority.

Developer Notes

- **Library:** `Rover/sailboat.cpp`.
- **Context:** Requires an accurate Roll measurement from the IMU.

Sailing maximum heel angle ()

HEEL_MAX: Sailing maximum heel angle

Description

When in auto sail trim modes, the heel (roll angle) will be limited to this value using PID control.

Values

- **Range:** 0 to 90
- **Units:** deg
- **Increment:** 1
- **Default:** 15

Description

This parameter sets the safety limit for the sailboat's heel (tipping) angle. If the boat tilts more than this angle due to strong wind, the autopilot will automatically release the mainsheet (or depower the wingsail) to bring the boat back upright.

- **Primary Use:** Preventing capsize and reducing excessive weather helm (tendency to turn into the wind) which causes drag.
- **Mechanism:** The value acts as a setpoint for a PID controller in the Attitude Control library. When the actual heel exceeds `HEEL_MAX`, the controller calculates a "sheet out" value to reduce sail power.

Source Code

ardupilot/Rover/sailboat.cpp

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **HGT** parameter group (specifically `HGT_OVR` or `FLOW_HGT_OVR`) configures the manual height reference for **Optical Flow** sensors.

Key Concepts

1. Flow Height Requirement

Optical flow sensors cannot determine absolute speed without knowing the distance to the ground.

- **Normal:** The system uses a Rangefinder (Lidar) to get live height data.
- **Override (`HGT_OVR`):** If no rangefinder is present, you can enter a static height (in meters) into this parameter.
- **Warning:** If you use an override, the Flow speed calculation will only be accurate when you are flying at exactly that height. If you fly higher, the drone will think it's moving slower than it actually is, potentially causing instability.

Parameter Breakdown

- `HGT_OVR`: Static height for flow calculation (meters).
  - **0:** Use Rangefinder data (Default).

Developer Notes

- **Library:** `libraries/AP_OpticalFlow`.
- **Usage:** Only intended for bench testing or very specific research environments where a fixed altitude is maintained via other means.

Height override of sensor above ground ()

HGT_OVR: Height override of sensor above ground

Description

This is used in rover vehicles where the sensor is a fixed height above the ground.

Values

- **Range:** 0 to 2
- **Units:** m
- **Increment:** 0.01
- **Default:** 0.0f

Description

For Rovers using an Optical Flow sensor, this parameter tells the autopilot exactly how high the sensor is mounted above the ground.

- **Why it matters:** Optical flow sensors measure how fast the texture of the ground moves across their view (angular velocity). To convert this into actual vehicle speed (linear velocity), the autopilot must know the distance to the ground.
- **Calculation:** $Velocity = FlowRate \times Height$.
- **Rover vs. Copter:** Copters use a rangefinder (Lidar/Sonar) because their height changes. Rovers stay on the ground, so a fixed value can be used instead of a rangefinder.

Source Code

ardupilot/libraries/AP_OpticalFlow/AP_OpticalFlow.cpp

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **HOME** parameter group (specifically `HOME_RESET_ALT`) configures how the autopilot handles the reference altitude for the **Home Position**.

The Home point is the location where the vehicle armed (or where GPS lock was first obtained). It is the target for the Return to Launch (RTL) mode.

Key Concepts

1. Altitude Resets

On startup, the autopilot sets the current altitude to 0. However, if the barometer drifts significantly while waiting for GPS lock, the "Home" altitude might be offset.

- `HOME_RESET_ALT`: (ArduPlane) Defines the altitude change (threshold) required to trigger an automatic reset of the 0m reference before takeoff.

Parameter Breakdown

- `HOME_RESET_ALT`: Reset threshold in meters.

Developer Notes

- **Library:** `libraries/AP_AHRS`.
- **Safety:** Accurate home altitude is critical for ensuring the vehicle clears obstacles during the return flight and lands at the correct height.

Home reset altitude threshold ()

HOME_RESET_ALT: Home reset altitude threshold

Description

When the aircraft is within this altitude of the home waypoint while disarmed, it will automatically update the home position. Set to 0 to continuously reset it.

Values

- **Range:** -1 to 127
- **Units:** m
- **Default:** 0

Description

This parameter controls whether and when the autopilot updates its "Home" position while the aircraft is on the ground (disarmed).

- **0 (Default):** Continuously update Home to the current GPS position while disarmed. The altitude is also continuously re-zeroed.
- **-1:** Never automatically update Home. Home is set once when GPS lock is first acquired.
- **> 0:** Update Home only if the aircraft's altitude is within this many meters of the initial startup altitude. If you move the aircraft significantly up or down (e.g., carrying it up a hill), home updating stops to prevent the barometer from being re-zeroed at the new height.

Source Code

ardupilot/ArduPlane/Parameters.cpp

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **ICE** parameter group configures the management of **Internal Combustion Engines** (Gas/Nitro). This library automates the starting sequence, idle control, and health monitoring of liquid-fueled engines.

In ArduPilot, an engine is managed via a dedicated "Starter" output and an "Ignition/Throttle" output.

Key Concepts

1. Automated Starting

The autopilot can manage the electric starter motor to crank the engine until a sustained RPM is detected.

- `ICE_STARTER_TIME`: How long to run the starter motor per attempt.
- `ICE_STRT_MX_RTRY`: Number of attempts before giving up.
- `ICE_START_PCT`: Throttle percentage applied during cranking.

2. Idle Management

Once started, the engine must maintain an "Idle" state without stalling.

- `ICE_IDLE_RPM`: The target RPM for idling on the ground.
- `ICE_IDLE_PCT`: The baseline throttle percentage to maintain idle.

3. RPM Monitoring

Requires an RPM sensor (Hall Effect or Optical) to be configured.

- `ICE_RPM_CHAN`: Maps which RPM sensor instance (1 or 2) corresponds to the engine.

Parameter Breakdown

- `ICE_ENABLE`: Master switch.
- `ICE_START_DELAY`: Delay between enabling ignition and cranking the starter.
- `ICE_REDLINE_RPM`: Maximum safe RPM (used for warnings/governor).

Integration Guide

1. **Outputs:** Map a servo to `Ignition` (Function 67) and `Starter` (Function 68).
2. **Sensors:** Ensure `RPM_TYPE` is set and providing a valid signal.
3. **Tuning:** Adjust `ICE_IDLE_PCT` so the engine stays running reliably but the drone/plane doesn't creep forward on the ground.

Developer Notes

- **Library:** `libraries/AP_ICEngine`.
- **State Machine:** Transitions through `IDLE`, `STARTING`, `RUNNING`, and `STOPPING`.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





ICE_ENABLE (ICEngine Enable)

DEFAULT: 0	RANGE: 0 1

ICEngine Enable (ICE_ENABLE)

Description

`ICE_ENABLE` activates the dedicated control logic for gas and nitro engines.

When enabled, ArduPilot moves beyond simple throttle control and activates a formal state machine to manage the **Ignition** and **Starter** sequences. It allows the autopilot to autonomously start the engine for a mission, maintain a steady idle on the ground, and monitor for "uncommanded stops" (flame-outs) using an RPM sensor.

Tuning & Behavior

- **Default Value:** 0 (Disabled).
- **Reboot Required:** Yes.
- **Integration:** Once enabled, you must assign servo functions to your ignition (`k_ignition`) and starter (`k_starter`) channels.

ICE_IDLE_DB (Deadband for Idle Governor)

DEFAULT: 50	RANGE: 0 1000	UNITS: RPM

ICE_IDLE_DB: Deadband for Idle Governor

Description

This configures the deadband that is tolerated before adjusting the idle setpoint.

Values

- **Range:** 0 to 1000
- **Units:** RPM
- **Default:** 50

Description

This parameter stabilizes the automatic idle control for internal combustion engines.

- **Function:** If the engine RPM is within `+/- ICE_IDLE_DB` of the target idle speed, the governor will not make any adjustments to the throttle servo.
- **Benefit:** Prevents the throttle servo from "hunting" or jittering constantly in response to minor, natural RPM fluctuations of a gas engine.

ICE_IDLE_PCT (Throttle Percentage for Engine Idle)

DEFAULT: 0	RANGE: 0 100	UNITS: %

Throttle Percentage for Engine Idle (ICE_IDLE_PCT)

Description

`ICE_IDLE_PCT` sets the "Floor" for the throttle command.

Once the engine is running, the autopilot will never drop the throttle below this value, even if the mission commands 0% thrust. This prevents the engine from stalling (dying) during low-power segments of a flight, such as a steep descent or landing approach.

Tuning & Behavior

- **Default Value:** 0%.
- **Recommendation:** Set this to the lowest throttle percentage that results in a reliable, steady idle.
- **Interaction:** If `ICE_IDLE_RPM` is also set, the autopilot will use the governor to dynamically adjust this floor.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





ICE_IDLE_RPM (RPM Setpoint for Idle Governor)

DEFAULT: -1	RANGE: -1 10000	UNITS: RPM

RPM Setpoint for Idle Governor (ICE_IDLE_RPM)

Description

`ICE_IDLE_RPM` enables the automated "Idle Governor."

Without this parameter (`-1`), the drone just holds a fixed throttle percentage (`ICE_IDLE_PCT`) when idle. With it enabled, the autopilot monitors the real-time RPM and dynamically adjusts the throttle to maintain this exact speed. This compensates for changes in air temperature, altitude, and engine warm-up.

Tuning & Behavior

- **Default Value:** -1 (Disabled).
- **Requirement:** Requires a functional RPM sensor (`RPM_CHAN` must be set).
- **Note:** Use ICE_IDLE_SLEW to control how quickly the governor reacts to RPM changes.

ICE_IDLE_SLEW (Slew Rate for idle control)

DEFAULT: 1	RANGE: 1 100	UNITS: %/s

ICE_IDLE_SLEW: Slew Rate for idle control

Description

This configures the slewrate used to adjust the idle setpoint in percentage points per second.

Values

- **Range:** 1 to 100
- **Units:** %/s
- **Default:** 1

Description

This parameter controls how quickly the engine governor can change the idle throttle position.

- **Function:** It limits the rate of change of the throttle servo output when the governor is active.
- **Safety:** Prevents the servo from slamming open or closed too quickly, which could cause the engine to stall or surge unexpectedly.
- **Tuning:** Increase this value if the governor is too slow to catch a dropping RPM (e.g., when disengaging a load).

ICE_OPTIONS (ICE Options)

DEFAULT: 0	RANGE: 0 31

ICE Options (ICE_OPTIONS)

Description

`ICE_OPTIONS` provides a set of "Safety Toggles" for engine operation.

- **Bit 0 (1): Disable ignition in RC failsafe.**
  If you lose radio link, the autopilot instantly cuts the ignition. Safer for large vehicles.
- **Bit 2 (4): Throttle control in MANUAL while disarmed.**
  Allows the pilot to test throttle movement and adjust the carburetor even when the motors are "disarmed."
- **Bit 3 (8): Disable while disarmed.**
  Prevents the engine from starting unless the flight controller is fully armed.
- **Bit 4 (16): Crank direction Reverse.**
  Reverses the direction of the starter motor (requires hardware support, e.g., TCA9554).

Tuning & Behavior

- **Default Value:** 0.
- **Recommendation:** Enable **Bit 0** for high-risk operations to ensure the engine stops if the controller is no longer in command.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





ICE_REDLINE_RPM (RPM Redline Limit)

DEFAULT: 0	RANGE: 0 2000000	UNITS: RPM

RPM Redline Limit (ICE_REDLINE_RPM)

Description

`ICE_REDLINE_RPM` is a safety governor that prevents engine destruction.

If the engine RPM exceeds this limit (e.g., during a steep dive or if the propeller detaches), ArduPilot will override the mission throttle and force it lower to bring the RPM back into a safe range.

Tuning & Behavior

- **Default Value:** 0 (Disabled).
- **Recommendation:** Set to the manufacturer's recommended maximum continuous RPM.
- **Action:** When active, the autopilot reduces the effective throttle output until the RPM falls below the limit.

ICE_RPM_CHAN (RPM instance channel to use)

DEFAULT: 0	RANGE: 0 4

ICE_RPM_CHAN: RPM instance channel to use

Description

This is which of the RPM instances to use for detecting the RPM of the engine.

Values

- **Range:** 0 to 4
- **Default:** 0

Description

This parameter links the Internal Combustion Engine (ICE) controller to a specific RPM sensor.

- **0:** Use the first detected RPM sensor instance.
- **1:** Use RPM1.
- **2:** Use RPM2.
- **Setup:** You must have an RPM sensor configured (using the `RPM_` parameters) before the ICE controller can see any data.

ICE_RPM_THRESH (RPM threshold)

DEFAULT: 100	RANGE: 100 100000	UNITS: RPM

RPM threshold (ICE_RPM_THRESH)

Description

`ICE_RPM_THRESH` is the "Run Signal" for the autopilot.

When the RPM sensor reports a speed higher than this value, ArduPilot transitions from the `STARTING` state to the `RUNNING` state and immediately turns off the starter motor. If the RPM falls below this value while the engine is supposed to be running, a "Uncommanded Stop" error is triggered, and the autopilot may attempt an in-flight restart.

Tuning & Behavior

- **Default Value:** 100 RPM.
- **Recommendation:** Set to a value roughly **50% of your idle RPM**. If your engine idles at 1200 RPM, set this to **600**. This ensures the starter stays engaged until the engine has truly fired and stabilized.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





ICE_STARTCHN_MIN (Input channel for engine start minimum PWM)

DEFAULT: 0	RANGE: 0 1300	UNITS: PWM

ICE_STARTCHN_MIN: Input channel for engine start minimum PWM

Description

This is a minimum PWM value for engine start channel for an engine stop to be commanded.

Values

- **Range:** 0 to 1300
- **Units:** PWM (Microseconds)
- **Default:** 0

Description

This parameter sets a noise floor for the Engine Start switch.

- **Function:** If you have assigned an RC Channel to `ICE_START_CHAN`, the autopilot monitors it.
- **Logic:** To command an engine stop (Kill), the PWM value must be **Low** (below 1300) but **High enough** to be valid (above `ICE_STARTCHN_MIN`).
- **Use Case:** Prevents a failsafe event or a glitch (where PWM might drop momentarily to 0 or 800) from accidentally killing the engine in flight.

ICE_STARTER_TIME (Time to run starter)

DEFAULT: 3	RANGE: 0.1 5.0	UNITS: s

Time to run starter (ICE_STARTER_TIME)

Description

`ICE_STARTER_TIME` sets the "Cranking Duration."

It defines how long the starter motor remains powered during a start attempt. If the engine reaches the `ICE_RPM_THRESH` before this timer expires, the starter is immediately disengaged. If the timer expires and the engine is not running, the system will wait (`ICE_START_DELAY`) and then try again.

Tuning & Behavior

- **Default Value:** 3 seconds.
- **Recommendation:** Set to the minimum time required for your engine to fire and reach a self-sustaining RPM. Cranking for too long can overheat the starter motor or drain the battery.

ICE_START_DELAY (Time to wait between starts)

DEFAULT: 2	RANGE: 1 10	UNITS: s

Time to wait between starts (ICE_START_DELAY)

Description

`ICE_START_DELAY` is a cooldown and recovery timer for the engine start sequence.

After a failed start attempt, the autopilot waits for this duration before trying again. This delay protects the starter motor from over-working and allows the fuel system to re-prime if necessary.

Tuning & Behavior

- **Default Value:** 2 seconds.
- **Recommendation:** Set high enough to allow the starter motor to stop spinning completely before the next engagement.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





ICE_START_PCT (Throttle Percentage for Engine Start)

DEFAULT: 5	RANGE: 0 100	UNITS: %

Throttle Percentage for Engine Start (ICE_START_PCT)

Description

`ICE_START_PCT` defines the "Start Throttle" – the position the carburetor or butterfly valve moves to when the starter motor is spinning.

Most gas engines require a small amount of throttle opening to draw in enough air and fuel to fire. If this value is too low, the engine will crank but never catch. If it is too high, the engine might "Scream" or surge immediately upon starting, which can be dangerous on the ground.

Tuning & Behavior

- **Default Value:** 5%.
- **Tuning:** Increase this in 2% increments if the engine fails to start while cranking.
- **Safety:** Ensure the drone is secured or the propeller is clear, as the engine will immediately move to this power level during the start sequence.

ICE_STRT_MX_RTRY (Maximum Number of Retries)

DEFAULT: 0	RANGE: 0 127

Maximum Number of Retries (ICE_STRT_MX_RTRY)

Description

`ICE_STRT_MX_RTRY` prevents the drone from burning out its starter motor or draining its battery on a stubborn engine.

If the engine fails to start after a series of attempts, this parameter sets the limit. Once reached, the autopilot stops trying and sends a "Critical: Engine failed to start" message to the GCS.

- **0 (Default):** Infinite retries.
- **3:** The system will try to start 3 times, then give up.

Tuning & Behavior

- **Recommendation:** Set to **3 or 5** for autonomous missions to ensure a mechanical issue doesn't lead to a total electrical failure.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **IM** parameter group configures the **Input Mapping** and collective curves for **Traditional Helicopters**.

Helicopters require complex nonlinear mapping between the pilot's collective stick and the physical pitch of the main rotor blades. This group manages the "Feel" of the collective in different flight modes.

Key Concepts

1. Collective Curves (`IM_STB_COL_x`)

In Stabilize and Acro modes, the collective stick doesn't command altitude; it commands blade pitch directly.

- **4-Point Curve:** The curve is defined by four points corresponding to Stick positions (0%, 40%, 60%, 100%).
- `IM_STB_COL_1`: Pitch at minimum stick.
- `IM_STB_COL_4`: Pitch at maximum stick.

2. Acro Expo (`IM_ACRO_COL_EXP`)

Adds exponential "feel" to the collective stick in Acro mode, allowing for more precise hover control around the center stick.

Parameter Breakdown

- `IM_STB_COL_1`..`4`: Curve points (normalized 0-1000).
- `IM_ACRO_COL_EXP`: Exponential factor.

Integration Guide

- **Tuning:** Adjust these curves to achieve a linear climb/descent feel that matches your motor power and blade efficiency.

Developer Notes

- **Library:** `libraries/AC_AttitudeControl/AC_AttitudeControl_Heli.cpp`.
- **Context:** Only relevant for **Helicopters**.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Acro Mode Collective Expo ()

IM_ACRO_COL_EXP: Acro Mode Collective Expo

Description

Used to soften collective pitch inputs near center point in Acro mode.

Values

- **Range:** 0 to 1.0
- **Default:** 0
- **Increment:** 0.1

Value	Meaning
0	Disabled (Linear)
0.1	Very Low
0.2	Low
0.3	Medium
0.4	High
0.5	Very High

Description

This parameter adds "Expo" (Exponential curvature) to the collective pitch stick when flying in Acro mode.

- **Effect:** It reduces the sensitivity of the collective stick around the center position (mid-stick).
- **Purpose:** Allows for more precise control of blade pitch near 0 degrees (typically mid-stick in Acro setup) while still allowing full positive and negative pitch at the stick extremes. This is useful for 3D flight maneuvers where precise management of zero-lift is required.

Source Code

ardupilot/libraries/AC_InputManager/AC_InputManager_Heli.cpp

Stabilize Collective High ()

IM_STB_COL_4: Stabilize Collective High

Description

Helicopter's maximum collective pitch setting at full collective stick input in Stabilize mode. Set this as a percent of collective range given by `H_COL_MAX` minus `H_COL_MIN`.

Values

- **Range:** 0 to 100
- **Units:** %
- **Increment:** 1
- **Default:** 100

Description

This parameter defines the top point of the collective pitch curve used in **Stabilize** mode.

- **Input:** 100% (Stick fully up).
- **Output:** The value of `IM_STB_COL_4` (as a percentage of the total collective range).
- **Usage:** Typically set to 100% to allow full climb performance, but can be reduced if the pilot wants to limit the maximum climb rate in Stabilize mode.

Source Code

ardupilot/libraries/AC_InputManager/AC_InputManager_Heli.cpp

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





`Stabilize Collective Low ()`

IM_STB_COL_1: Stabilize Collective Low

Description

Helicopter's minimum collective pitch setting at zero collective stick input in Stabilize mode. Set this as a percent of collective range given by `H_COL_MAX` minus `H_COL_MIN`.

Values

- **Range:** 0 to 100
- **Units:** %
- **Increment:** 1
- **Default:** 0

Description

This parameter defines the bottom point of the collective pitch curve used in **Stabilize** mode.

- **Input:** 0% (Stick fully down).
- **Output:** The value of `IM_STB_COL_1` (as a percentage of the total collective range defined by `H_COL_MIN` and `H_COL_MAX`).
- **Usage:** Typically set to 0% for normal setups, or higher if you want to limit negative pitch (descent rate) in Stabilize mode to prevent accidental rapid descents by beginners.

Source Code

ardupilot/libraries/AC_InputManager/AC_InputManager_Heli.cpp

`Stabilize Collective Mid-High ()`

IM_STB_COL_3: Stabilize Collective Mid-High

Description

Helicopter's collective pitch setting at mid-high (60%) collective stick input in Stabilize mode. Set this as a percent of collective range given by `H_COL_MAX` minus `H_COL_MIN`.

Values

- **Range:** 0 to 100
- **Units:** %
- **Increment:** 1
- **Default:** 60

Description

This parameter defines the third point of the collective pitch curve used in **Stabilize** mode.

- **Input:** 60% (Stick slightly above center).
- **Output:** The value of `IM_STB_COL_3` (as a percentage of the total collective range).
- **Usage:** Typically set to a value that provides a hover or gentle climb. This, along with `IM_STB_COL_2`, creates the "hover band" sensitivity.

Source Code

ardupilot/libraries/AC_InputManager/AC_InputManager_Heli.cpp

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





`Stabilize Collective Mid-Low ()`

IM_STB_COL_2: Stabilize Collective Mid-Low

Description

Helicopter's collective pitch setting at mid-low (40%) collective stick input in Stabilize mode. Set this as a percent of collective range given by `H_COL_MAX` minus `H_COL_MIN`.

Values

- **Range:** 0 to 100
- **Units:** %
- **Increment:** 1
- **Default:** 40

Description

This parameter defines the second point of the collective pitch curve used in **Stabilize** mode.

- **Input:** 40% (Stick slightly below center).
- **Output:** The value of `IM_STB_COL_2` (as a percentage of the total collective range).
- **Usage:** Typically set to a value that provides a gentle descent or hover-capable pitch, depending on pilot preference. This helps define the transition from minimum pitch to hover pitch.

Source Code

ardupilot/libraries/AC_InputManager/AC_InputManager_Heli.cpp

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **IMU** parameter group (often appearing as `INS_` in the full parameter list) configures the global behavior of the **Inertial Measurement Units**.

This group handles physical offsets and timing constants that apply to all internal sensors.

Key Concepts

1. Lever-Arm Compensation (`IMU_POS_X/Y/Z`)

Defines the distance (in meters) from the center of gravity (CG) of the vehicle to the physical location of the IMU chip on the circuit board.

- **Importance:** If the IMU is far from the CG (e.g., on the tail of a long drone), the centripetal acceleration during a turn will bias the EKF. Setting these offsets allows the math to "shift" the measurement back to the CG.

2. Sample Rate (`IMU_SAMPLE_RATE`)

Defines the internal frequency (Hz) at which raw sensor data is pulled from the IMU hardware.

- **Standard:** 1000Hz (1kHz).
- **High-End:** 2000Hz or 4000Hz (reduces aliasing on high-vibration frames).

Parameter Breakdown

- `IMU_POS_X/Y/Z`: Physical offset from CG (meters).
- `IMU_SAMPLE_RATE`: Internal hardware sampling frequency.

Developer Notes

- **Library:** `libraries/AP_InertialSensor`.
- **Alias:** In most firmwares, these are mapped to `INS_POS_...` and `INS_GYRO_RATE`.

IMU Offsets ()

IMU_POS: IMU Offsets

Description

XYZ position of the IMU accelerometer relative to the body frame origin.

Values

- **Units:** m
- **Format:** Vector3 (X, Y, Z)
- **Default:** 0, 0, 0

Description

This parameter is used in **SITL (Software In The Loop) simulation** only. It defines the physical location of the simulated IMU sensor relative to the center of the simulated vehicle.

- **Purpose:** Used to simulate "lever arm" effects. When the vehicle rotates, an IMU not at the center of rotation measures centripetal and tangential acceleration. Setting this parameter allows developers to verify that the EKF (Extended Kalman Filter) correctly compensates for these offsets.
- **X:** Forward offset (positive forward).
- **Y:** Right offset (positive right).
- **Z:** Down offset (positive down).

Source Code

ardupilot/libraries/SITL/SITL.cpp

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





IMU_SAMPLE_RATE: IMU Sample Rate

Description

Sets the sample rate for the Inertial Measurement Unit (IMU) on the AP_Periph node.

Values

- **Range:** 0 to 1000
- **Units:** Hz
- **Default:** 0

Description

This parameter configures the loop rate of the Inertial Sensor driver on the peripheral.

- **0:** Use the driver's default rate.
- **>0:** Force a specific sample rate in Hz. Higher rates allow for better vibration filtering (if the hardware supports it) but consume more CPU resources on the peripheral microcontroller.

Source Code

ardupilot/Tools/AP_Periph/Parameters.cpp

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY



Overview

The **INITIAL** parameter group (specifically `INITIAL_MODE`) defines the behavior of the autopilot immediately after boot-up.

Key Concepts

1. Boot-Up Mode

When you power on the vehicle, it must enter a safe and predictable state before the pilot takes control.

- `INITIAL_MODE`: (ArduRover) Selects which drive mode is active at startup.
  - **Default:** Usually 0 (Manual) or 1 (Hold).
  - **Safety:** Setting this to "Hold" ensures the rover won't move even if the transmitter is sending a non-centered signal at boot.

Parameter Breakdown

- `INITIAL_MODE`: Mode ID to enter on boot.

Developer Notes

- **Vehicle: ArduRover**.
- **Related:** Plane and Copter typically boot into `STABILIZE` or `MANUAL` by default and do not have a dedicated parameter for this.

Initial driving mode ()

INITIAL_MODE: Initial driving mode

Description

This selects the mode to start in on boot. This is useful for when you want to start in AUTO mode on boot without a receiver. Usually used in combination with when `AUTO_TRIGGER_PIN` or `AUTO_KICKSTART`.

Values

Value	Meaning
0	Manual
1	Acro
3	Steering
4	Hold
5	Loiter
6	Follow
7	Simple
8	Dock
9	Circle
10	Auto
11	RTL
12	SmartRTL
15	Guided

- **Default:** 0 (Manual)

Description

This parameter determines the vehicle's state immediately after the flight controller boots up.

- **Standard Use:** Keep at **0 (Manual)** for normal operations where you turn on the rover and then switch modes using your RC transmitter.
- **Autonomous/No-RC Use:** Set to **10 (Auto)** if you want the rover to immediately begin a mission upon power-up (often used with `AUTO_TRIGGER_PIN` to require a button press, or `AUTO_KICKSTART` to require a push before the motor engages).

Source Code

ardupilot/Rover/Parameters.cpp

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **INJECT** parameter group (specifically `INJECT_TO` or `GPS_INJECT_TO`) configures the routing of **RTK Correction** data.

Key Concepts

1. RTK Routing

In a dual-GPS system, the RTK correction data (RTCM) arrives from the Ground Station over the telemetry link. The autopilot must decide which physical GPS receiver to send this data to.

- `INJECT_TO`:
  - **0:** Send to GPS 1.
  - **1:** Send to GPS 2.
  - **127:** Send to all connected GPS units.

Developer Notes

- **Library:** `libraries/AP_GPS`.
- **Context:** Critical for centimeter-level positioning accuracy.

INJECT_TO (GPS Data Injection Target)

DEFAULT: 127	RANGE: 0 127	UNITS: Index

GPS Data Injection Target (INJECT_TO)

Description

This parameter controls the routing of "GPS_INJECT_DATA" MAVLink packets. These packets are typically used to send RTK (Real-Time Kinematic) corrections from a Ground Control Station (GCS) to the autopilot, enabling centimeter-level positioning accuracy.

If your vehicle has multiple GPS sensors, you can use this parameter to decide if the corrections should go to a specific GPS instance (e.g., just the primary one) or be "broadcast" to all connected GPS modules simultaneously.

The Mathematics

The value maps to the internal GPS instance index (i):

- **0:** GPS 1 (First instance).
- **1:** GPS 2 (Second instance).
- **127:** All GPS instances (Broadcast mode).

The Engineer's View

In `AP_GPS::handle_msg()` (`libraries/AP_GPS/AP_GPS.cpp`):

1. The system receives raw serial data from the MAVLink `GPS_INJECT_DATA` or `GPS2_RTK` packets.
2. It checks the `_inject_to` parameter.
3. If set to **127** (`GPS_RTK_INJECT_TO_ALL`), it iterates through every detected GPS backend and calls `inject_data()`.
4. If set to a specific index (0 or 1), it sends the data *only* to that specific backend.
5. This ensures that if you have two different GPS models (e.g., a u-blox and a Septentrio) that require different correction formats, you can target the corrections appropriately (though most users simply broadcast to all).

Tuning & Behavior

- **Default Value:** 127 (All).
- **Recommended:** Leave at **127** unless you have a specific dual-GPS setup where the sensors are of different types and sharing corrections causes errors.
- **RTK Fix:** If your GCS shows "RTK Fixed" or "RTK Float" but only on one GPS, check this parameter to ensure it is set to include the second GPS instance if needed.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





INERTIAL NAVIGATION SYSTEM (INS)

Overview

The **INS** parameter group is the core configuration for the **Inertial Sensors** (Accelerometers and Gyroscopes). This library is responsible for sampling raw sensor data, applying calibrations, and filtering the results before they are consumed by the EKF (State Estimator) and PID controllers.

This is the most critical system for flight stability. If the INS data is noisy or incorrectly calibrated, the vehicle cannot fly.

Key Concepts

1. Calibration (Offsets & Scaling)

Every IMU has manufacturing imperfections (e.g., when the drone is still, the accel might read 0.05m/s² instead of 0).

- **Offsets (`_OFFS`):** Shift the zero point.
- **Scaling (`_SCAL`):** Ensures that 1G of gravity actually measures as 9.81m/s².

2. Harmonic Notch Filters (`INS_HNTCH_...`)

This is a modern ArduPilot feature that uses a high-Q notch filter to "erase" motor vibration noise.

- **Dynamic Tracking:** The filter can follow the motor RPM in real-time (via ESC telemetry or throttle) to target the exact frequency of the propeller noise.
- **Result:** Allows for much higher PID gains and a "locked-in" flight feel.

3. Temperature Calibration (`INS_TCAL_...`)

MEMS sensors drift as they heat up. The `TCAL` system applies complex polynomial curves to counteract this drift, ensuring a stable horizon from a cold boot to a hot flight.

4. Fast Sampling (`INS_FAST_SAMPLE`)

Enables pulling data from sensors at 8kHz or 16kHz for better aliasing rejection.

Parameter Breakdown

- `INS_ACC_FILTER`: Low-pass filter for accelerometers (Hz).
- `INS_GYRO_FILTER`: Low-pass filter for gyroscopes (Hz).
- `INS_ENABLE_MASK`: Selects active IMUs.
- `INS_HNTCH_ENABLE`: Toggles the harmonic notch.

Integration Guide

Accelerometer Calibration

1. **Level:** Place vehicle on a perfectly level surface. Click "Calibrate Level."
2. **6-Point:** Follow the GCS prompts to place the vehicle on each of its 6 sides. This populates the `INS_ACCn_OFFS` and `INS_ACCn_SCAL` values.

Tuning the Notch Filter

1. **Log:** Fly and record raw IMU data (`INS_LOG_BAT_MASK = 1`).
2. **Analyze:** Use the Mission Planner "FFT" tool to find the noise peak (e.g., 180Hz).
3. **Config:** Set `INS_HNTCH_FREQ` to 180 and `INS_HNTCH_BW` to 90.

Developer Notes

- **Library:** `libraries/AP_InertialSensor`.
- **Criticality:** High. The `INS` loop runs at the highest priority in the firmware.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





INS4_ACC_CALTEMP (Accelerometer 4 Calibration Temperature)

DEFAULT: -300	RANGE: -50 100	UNITS: degC

Accelerometer 4 Calibration Temperature (INS4_ACC_CALTEMP)

Description

`INS4_ACC_CALTEMP` records the temperature of the 4th IMU when its accelerometer calibration was performed.

This is part of ArduPilot's support for high-redundancy flight controllers that have 4 or more IMUs. The autopilot uses this value to monitor thermal drift and apply corrections relative to this calibration baseline.

INS4_ACC_ID (Accelerometer ID)

DEFAULT: 0	RANGE: null

Accelerometer ID

Note: This parameter functions identically to INS_ACC_ID.

INS4_GYR_CALTEMP (Gyroscope 4 Calibration Temperature)

DEFAULT: -300	RANGE: -50 100	UNITS: degC

Gyroscope 4 Calibration Temperature (INS4_GYR_CALTEMP)

Description

`INS4_GYR_CALTEMP` records the temperature of the 4th IMU when its gyroscope calibration was performed.

INS4_GYR_ID (Gyro ID)

DEFAULT: 0	RANGE: null

Gyro ID

Note: This parameter functions identically to INS_GYR_ID.

INS4_USE (Use first IMU for attitude, velocity and position estimates)

DEFAULT: 1	RANGE: null

Use first IMU for attitude, velocity and position estimates

Note: This parameter functions identically to INS_USE.

INS5_ACC_CALTEMP (Accelerometer 5 Calibration Temperature)

Accelerometer 5 Calibration Temperature

Note: This parameter functions identically to INS4_ACC_CALTEMP.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





INS5_ACC_ID (Accelerometer ID)

DEFAULT: 0	RANGE: null

Accelerometer ID

Note: This parameter functions identically to INS_ACC_ID.

INS5_GYR_CALTEMP (Gyroscope 5 Calibration Temperature)

Gyroscope 5 Calibration Temperature

Note: This parameter functions identically to INS4_GYR_CALTEMP.

INS5_GYR_ID (Gyro ID)

DEFAULT: 0	RANGE: null

Gyro ID

Note: This parameter functions identically to INS_GYR_ID.

INS5_USE (Use first IMU for attitude, velocity and position estimates)

DEFAULT: 1	RANGE: null

Use first IMU for attitude, velocity and position estimates

Note: This parameter functions identically to INS_USE.

INS_ACC1_CALTEMP (Accelerometer 1 Calibration Temperature)

DEFAULT: -300	RANGE: -50 100	UNITS: degC

Accelerometer 1 Calibration Temperature (INS_ACC1_CALTEMP)

Description

`INS_ACC1_CALTEMP` records the temperature of the flight controller when you performed the 6-point accelerometer calibration.

The EKF uses this as a reference point. If the current temperature deviates significantly from this value, the system may apply thermal compensation corrections (if `INS_TCAL` is enabled) to maintain accuracy.

Tuning & Behavior

- **Default Value:** -300 (Uncalibrated).
- **Usage:** Do not edit manually. This is updated automatically when you run the calibration routine.

INS_ACC2OFFS (null)

DEFAULT: 0	RANGE: null	UNITS: null

None

Note: This parameter functions identically to INS_ACCOFFS.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





INS_ACC2SCAL (null)

DEFAULT: 1.0	RANGE: null	UNITS: null

None

Note: This parameter functions identically to INS_ACCSCAL.

INS_ACC2_CALTEMP (Calibration temperature for 2nd accelerometer)

DEFAULT: -300	RANGE: null	UNITS: degC

Calibration temperature for 2nd accelerometer

Note: This parameter functions identically to INS_ACC1_CALTEMP.

INS_ACC2_ID (null)

DEFAULT: 0	RANGE: null	UNITS: null

None

Note: This parameter functions identically to INS_ACC_ID.

INS_ACC3OFFS (null)

DEFAULT: 0	RANGE: null	UNITS: null

None

Note: This parameter functions identically to INS_ACCOFFS.

INS_ACC3SCAL (null)

DEFAULT: 1.0	RANGE: null	UNITS: null

None

Note: This parameter functions identically to INS_ACCSCAL.

INS_ACC3_CALTEMP (Calibration temperature for 3rd accelerometer)

DEFAULT: -300	RANGE: null	UNITS: degC

Calibration temperature for 3rd accelerometer

Note: This parameter functions identically to INS_ACC1_CALTEMP.

INS_ACC3_ID (null)

DEFAULT: 0	RANGE: null	UNITS: null

None

Note: This parameter functions identically to INS_ACC_ID.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





INS_ACCEL_FILTER (Accelerometer Low Pass Filter)

DEFAULT: 10	RANGE: 0 250	UNITS: Hz

Accelerometer Low Pass Filter (INS_ACCEL_FILTER)

Description

`INS_ACCEL_FILTER` removes vibration noise from the accelerometer data.

Unlike the Gyro (which is for attitude), the Accelerometer is primarily for position and climb rate. It is very sensitive to motor vibration.

The Engineer's View

Defined in `AP_InertialSensor.cpp`.
The filtered accelerometer data is used by the EKF to calculate velocity and position increments.

Tuning & Behavior

- **Default Value:** 10 Hz.
- **Recommendation:** Usually leave at **10-20 Hz**. Setting it higher may improve position hold in high winds, but only if your frame is exceptionally clean (low vibration).

INS_ACCOFFS (Accelerometer Z Offset)

DEFAULT: 0	RANGE: -3.5 3.5	UNITS: m/s^2

Accelerometer Z Offset (INS_ACCOFFS)

Description

`INS_ACCOFFS` is the result of the accelerometer calibration.

It represents the constant error (bias) in the Z-axis measurement. The flight controller subtracts this value from the raw sensor reading to get the true acceleration.

Tuning & Behavior

- **Default Value:** 0.
- **Usage:** Do not edit manually. Updated by the `AccelCal` routine.

INS_ACCSCAL (Accelerometer Z Scale Factor)

DEFAULT: 1.0	RANGE: 0.8 1.2

Accelerometer Z Scale Factor (INS_ACCSCAL)

Description

`INS_ACCSCAL` corrects the "Gain" of the accelerometer.

If you tip the drone 90 degrees, the Z-axis should read 0G. If it reads 0.1G, the scale factor is wrong. This parameter is calculated automatically during the 6-point calibration to ensuring that 1G really means 1G.

Tuning & Behavior

- **Default Value:** 1.0 (No correction).
- **Usage:** Automatically updated during calibration.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





INS_ACC_BODYFIX (Accelerometer Body Fix)

DEFAULT: 2	RANGE: 0 2

Accelerometer Body Fix (INS_ACC_BODYFIX)

Description

`INS_ACC_BODYFIX` corrects for physical misalignment of the flight controller.

If your board is mounted slightly crooked, the accelerometers will report a constant error. This parameter enables a correction matrix that rotates the accelerometer vector to match the vehicle's body frame.

Tuning & Behavior

- **0:** Disabled.
- **1:** Use AHRS Trim.
- **2:** Use Board Orientation.

INS_ACC_ID (Accelerometer ID)

DEFAULT: 0

Accelerometer ID (INS_ACC_ID)

Description

`INS_ACC_ID` is a Read-Only parameter that stores the unique hardware ID of the first accelerometer (Accel 1).

It is used to identify the sensor and associate calibration data (offsets/scaling) with the specific physical hardware.

The Mathematics

See COMPASS_DEV_ID for ID packing details (Bus type, Bus number, Address).

The Engineer's View

Populated during `AP_InertialSensor::init()`.
If set to 0, the system re-detects the sensor on the next boot.
Usually, this ID matches `INS_GYR_ID` if the accelerometer and gyroscope are part of the same 6-axis IMU chip (e.g., MPU6000, ICM20602), though the `dev_type` bits might differ slightly depending on the driver implementation.

Tuning & Behavior

- **Default Value:** 0 (Auto-Detect)
- **Recommendation:** Do not touch. If you replace the flight controller, this will update automatically.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





INS_ENABLE_MASK (IMU enable mask)

DEFAULT: 127	RANGE: 0 127

IMU enable mask (INS_ENABLE_MASK)

Description

`INS_ENABLE_MASK` allows precise control over which Inertial Measurement Units (IMUs) are detected and initialized by the autopilot at boot.

ArduPilot can support multiple IMUs (typically 3 on high-end boards like the Cube Orange). If a specific IMU sensor hardware is faulty (e.g., "IMU 2 Calibration Failed" persistent error) or is known to be noisy/defective on a particular board revision, you can use this mask to completely hide it from the flight controller.

The Mathematics

$$\text{Enabled IMUs} = \sum_k = 0^{6} (2^k \text{ is set}) \rightarrow \text{IMU } k+1$$

- **Bit 0 (1):** IMU 1
- **Bit 1 (2):** IMU 2
- **Bit 2 (4):** IMU 3
- **...**

Default is **127 (0x7F)**, which enables the first 7 possible IMUs (usually only 2 or 3 exist).

The Engineer's View

Used in `AP_InertialSensor::detect_backends()`.
The driver check `ADD_BACKEND` verifies `(1U << probe_count) & enable_mask`. If the bit is 0, the driver probe is skipped entirely. This is cleaner than `INS_USE_x = 0` because it saves RAM and CPU by not even allocating the backend driver structures.

Tuning & Behavior

- **Default Value:** 127 (All Enabled)
- **Use Case:**
  - **Faulty Sensor:** If the GCS says "Bad Gyro Health" for IMU 3 and recalibration doesn't fix it, set `INS_ENABLE_MASK` to **3** (1 + 2) to disable IMU 3 permanently.
  - **Cube Isolated IMU:** On the Cube Black/Orange, the isolated IMU (IMU 3) is sometimes disabled for specific industrial applications where the vibration mount causes issues.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





INS_FAST_SAMPLE (Fast sampling mask)

DEFAULT: 1	RANGE: 0 7

Fast sampling mask (INS_FAST_SAMPLE)

Description

INS_FAST_SAMPLE is a bitmask that determines which IMUs are allowed to run at the higher INS_GYRO_RATE (e.g., 2kHz, 4kHz, 8kHz).

- **Bit 0:** IMU 1
- **Bit 1:** IMU 2
- **Bit 2:** IMU 3

If a bit is set (1), that IMU attempts to run at INS_GYRO_RATE.
If a bit is clear (0), that IMU runs at the standard 1kHz (or default backend rate).

The Mathematics

$$\text{Sample Rate}_i = \begin{cases} \text{INS_GYRO_RATE} & \text{if bit } i \text{ is set} \\ 1\text{kHz} & \text{otherwise} \end{cases}$$

The Engineer's View

Default behavior depends on the board (HAL_DEFAULT_INS_FAST_SAMPLE).
Typically, the primary IMU (Bit 0) is enabled for fast sampling to feed the low-latency control loop. Secondary IMUs might be left slow to save CPU or reduce noise aliasing.

Tuning & Behavior

- **Default Value:** 1 (IMU 1 Fast)
- **Recommendation:**
  - **1 (IMU 1 only):** Good for most setups.
  - **3 (IMU 1 & 2):** If you want redundancy and EKF lane switching to have equal performance on both sensors.
  - **7 (All):** Max performance, max CPU usage.

INS_GYR1_CALTEMP (Gyroscope 1 Calibration Temperature)

DEFAULT: -300	RANGE: -50 100	UNITS: degC

Gyroscope 1 Calibration Temperature (INS_GYR1_CALTEMP)

Description

INS_GYR1_CALTEMP records the board temperature during gyro calibration.

Gyroscopes are sensitive to temperature changes. Knowing the temperature at which the zero-offset was determined allows the flight controller to better predict and correct for bias drift as the drone warms up or cools down.

Tuning & Behavior

- **Default Value:** -300 (Uncalibrated).
- **Usage:** Automatically updated during calibration.

INS_GYR2OFFS (null)

DEFAULT: 0	RANGE: null	UNITS: null

None

Note: This parameter functions identically to INS_GYROFFS.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





INS_GYR2_CALTEMP (Calibration temperature for 2nd gyroscope)

DEFAULT: -300	RANGE: null	UNITS: degC

Calibration temperature for 2nd gyroscope

Note: This parameter functions identically to INS_GYR1_CALTEMP.

INS_GYR2_ID (null)

DEFAULT: 0	RANGE: null	UNITS: null

None

Note: This parameter functions identically to INS_GYR_ID.

INS_GYR3OFFS (null)

DEFAULT: 0	RANGE: null	UNITS: null

None

Note: This parameter functions identically to INS_GYROFFS.

INS_GYR3_CALTEMP (Calibration temperature for 3rd gyroscope)

DEFAULT: -300	RANGE: null	UNITS: degC

Calibration temperature for 3rd gyroscope

Note: This parameter functions identically to INS_GYR1_CALTEMP.

INS_GYR3_ID (null)

DEFAULT: 0	RANGE: null	UNITS: null

None

Note: This parameter functions identically to INS_GYR_ID.

INS_GYROFFS (Gyroscope Z Offset)

DEFAULT: 0	RANGE: -0.1 0.1	UNITS: rad/s

Gyroscope Z Offset (INS_GYROFFS)

Description

`INS_GYROFFS` is the result of the gyro calibration.

When you boot the drone, it must sit perfectly still for a few seconds. During this time, the flight controller averages the gyro readings to find the "Zero" point. This value is stored here to correct for sensor drift.

Tuning & Behavior

- **Default Value:** 0.
- **Usage:** Updated automatically on every boot. Do not change manually.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY

INS_GYRO_FILTER (Gyroscope Low Pass Filter)

DEFAULT: 20	RANGE: 0 250	UNITS: Hz

Gyroscope Low Pass Filter (INS_GYRO_FILTER)

Description

`INS_GYRO_FILTER` sets the "Smoothness" of the raw gyroscope data.

All sensors produce high-frequency electrical and mechanical noise (vibration). This low-pass filter removes that noise before the autopilot uses the data for attitude control.

- **Higher Frequency (e.g., 40Hz):** Less delay (lower latency), but more noise gets through. Best for small, clean racing quads.
- **Lower Frequency (e.g., 10Hz):** Very smooth data, but adds delay. Best for large, vibey industrial drones.

The Mathematics

This is a standard 2nd-order software Butterworth filter. The signal y is calculated from the input x using the cutoff frequency f_c:

$$y_n = \text{Butterworth}(x_n, f_c)$$

Lowering f_c increases the group delay (latency) of the signal, which can destabilize the PID controller if set too low.

The Engineer's View

Defined in `AP_InertialSensor.cpp`.
The filtered gyro data is the primary input to the Rate PID loop. Delay here is "bad" for performance but necessary for stability.

Tuning & Behavior

- **Default Value:** 20 Hz.
- **Small Quads (5-7 inch):** Try **30Hz to 60Hz**.
- **Large Quads (15 inch+):** Try **10Hz to 20Hz**.
- **Harmonic Notch:** If you have persistent motor noise, do not lower this filter excessively; use the `INS_HNTCH` (Harmonic Notch) system instead.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY

INS_GYRO_RATE (Gyro rate for IMUs with Fast Sampling enabled)

DEFAULT: 0	RANGE: 0 3

Gyro rate for IMUs with Fast Sampling enabled (INS_GYRO_RATE)

Description

`INS_GYRO_RATE` sets the loop rate of the sensor backend. This is *not* the main flight control loop rate (which is usually 400Hz or 800Hz), but the rate at which the sensor driver reads data from the IMU chip.

- **0: 1kHz (Default).** Standard rate. Good for older boards.
- **1: 2kHz.** Required for good Harmonic Notch filtering on most quads.
- **2: 4kHz.** Higher precision, lower latency. Requires a fast CPU (F7/H7).
- **3: 8kHz.** Ultra-low latency. Only for high-performance H7 boards.

The Mathematics

The sensor backend runs a loop:

1. Read Gyro at `INS_GYRO_RATE`.
2. Apply Hardware LPF (if configured on chip).
3. Apply Harmonic Notch Filter.
4. Apply `INS_GYRO_FILTER` (Software LPF).
5. Downsample to Main Loop Rate (e.g., average 8 samples if running 8kHz sensor and 1kHz loop).

The Engineer's View

Sets `_fast_sampling_rate` in `AP_InertialSensor`.
To use the Harmonic Notch Filter effectively, the sampling rate must be at least **2x** the highest frequency you want to filter (Nyquist Theorem).
If you want to filter noise at 400Hz (2nd harmonic of a 200Hz motor), you need at least 800Hz sampling. In practice, 2kHz is the recommended minimum for notch filtering to avoid aliasing.

Tuning & Behavior

- **Default Value:** 0 (1kHz)
- **Recommendation:**
  - **Set to 1 (2kHz)** or **2 (4kHz)** on modern boards (Cube Orange, Matek H743).
  - This provides a dense stream of data for the filters to work with, resulting in a cleaner signal for the PID loop.
- **Warning:** Higher rates consume more CPU. Check `LOAD` in MAVLink messages. If it exceeds 80%, lower this rate.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





INS_GYR_CAL (Gyro Calibration scheme)

DEFAULT: 1	RANGE: 0 1

Gyro Calibration scheme (INS_GYR_CAL)

Description

`INS_GYR_CAL` determines if the autopilot should re-calibrate the gyroscope offsets every time it boots.

- **0: Never.** Relies on saved offsets. Fast boot, but risky if temperature changes significantly.
- **1: Start-up only (Default).** Calibrates every time you plug in the battery. Requires the drone to be **still** during startup.

The Mathematics

The calibration routine calculates the mean value of the gyro output while the vehicle is stationary (assumed 0 rotation) and subtracts this as a bias.
$$ \vec{\omega}*{cal} = |vec{|omega}*{raw} - \vec{\omega}_{bias} $$

The Engineer's View

If set to **1**, `AP_InertialSensor::init()` calls `_init_gyro()` which blocks startup until the gyros are stable. This is why you see "Calibrating IMU..." on the HUD.
If set to **0**, it skips this and uses the `INS_GYR_OFFS` parameters stored in EEPROM.

Tuning & Behavior

- **Default Value:** 1
- **Recommendation:** Leave at **1**.
- **Boat Mode:** If launching from a moving boat, you **must** set this to **0** (Never), otherwise the calibration will fail or learn a huge bias (thinking the boat's motion is "still"), causing the drone to flip on takeoff. You must perform a good calibration on land first.

INS_GYR_ID (Gyro ID)

DEFAULT: 0

Gyro ID (INS_GYR_ID)

Description

`INS_GYR_ID` is a Read-Only parameter that stores the unique hardware ID of the first gyroscope (Gyro 1).

See INS_ACC_ID for details on ID structure.

The Mathematics

See COMPASS_DEV_ID.

The Engineer's View

See INS_ACC_ID.

Tuning & Behavior

- **Default Value:** 0 (Auto-Detect)

INS_HNTC2_ATT (Harmonic Notch Filter attenuation)

DEFAULT: 40	RANGE: 5 50	UNITS: dB

Harmonic Notch Filter attenuation

Note: This parameter functions identically to INS_HNTCH_ATT for a secondary notch instance.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





INS_HNTC2_BW (Harmonic Notch Filter bandwidth)

DEFAULT: 40	RANGE: 5 250	UNITS: Hz

Harmonic Notch Filter bandwidth

Note: This parameter functions identically to INS_HNTCH_BW for a secondary notch instance.

INS_HNTC2_ENABLE (Enable pullup after altitude wait)

DEFAULT: 0	RANGE: null	UNITS: null

Enable pullup after altitude wait

Note: This parameter functions identically to INS_HNTCH_ENABLE for a secondary notch instance.

INS_HNTC2_FM_RAT (Throttle notch min freqency ratio)

DEFAULT: 1.0	RANGE: 0.1 1.0	UNITS: null

Throttle notch min freqency ratio

Note: This parameter functions identically to INS_HNTCH_FM_RAT for a secondary notch instance.

INS_HNTC2_FREQ (Video Transmitter Frequency)

DEFAULT: 0	RANGE: 1000 6000	UNITS: null

Video Transmitter Frequency

Note: This parameter functions identically to INS_HNTCH_FREQ for a secondary notch instance.

INS_HNTC2_HMNCS (Harmonic Notch Filter harmonics)

DEFAULT: 3	RANGE: null	UNITS: null

Harmonic Notch Filter harmonics

Note: This parameter functions identically to INS_HNTCH_HMNCS for a secondary notch instance.

INS_HNTC2_MODE (Rotor Speed Control Mode)

DEFAULT: (int8_t	RANGE: null	UNITS: null

Rotor Speed Control Mode

Note: This parameter functions identically to INS_HNTCH_MODE.

INS_HNTC2_OPTS (Harmonic Notch Filter options)

DEFAULT: 0	RANGE: null	UNITS: null

Harmonic Notch Filter options

Note: This parameter functions identically to INS_HNTCH_OPTS for a secondary notch instance.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





INS_HNTC2_REF (Harmonic Notch Filter reference value)

DEFAULT: 0	RANGE: 0.0 1.0	UNITS: null

Harmonic Notch Filter reference value

Note: This parameter functions identically to INS_HNTCH_REF for a secondary notch instance.

INS_HNTC3_ATT (Harmonic Notch Filter attenuation)

DEFAULT: 40	RANGE: 5 50	UNITS: dB

Harmonic Notch Filter attenuation

Note: This parameter functions identically to INS_HNTCH_ATT for a secondary notch instance.

INS_HNTC3_BW (Harmonic Notch Filter bandwidth)

DEFAULT: 40	RANGE: 5 250	UNITS: Hz

Harmonic Notch Filter bandwidth

Note: This parameter functions identically to INS_HNTCH_BW for a secondary notch instance.

INS_HNTC3_ENABLE (Enable pullup after altitude wait)

DEFAULT: 0	RANGE: null	UNITS: null

Enable pullup after altitude wait

Note: This parameter functions identically to INS_HNTCH_ENABLE for a secondary notch instance.

INS_HNTC3_FM_RAT (Throttle notch min freqency ratio)

DEFAULT: 1.0	RANGE: 0.1 1.0	UNITS: null

Throttle notch min freqency ratio

Note: This parameter functions identically to INS_HNTCH_FM_RAT for a secondary notch instance.

INS_HNTC3_FREQ (Video Transmitter Frequency)

DEFAULT: 0	RANGE: 1000 6000	UNITS: null

Video Transmitter Frequency

Note: This parameter functions identically to INS_HNTCH_FREQ for a secondary notch instance.

INS_HNTC3_HMNCS (Harmonic Notch Filter harmonics)

DEFAULT: 3	RANGE: null	UNITS: null

Harmonic Notch Filter harmonics

Note: This parameter functions identically to INS_HNTCH_HMNCS for a secondary notch instance.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





INS_HNTC3_MODE (Rotor Speed Control Mode)

DEFAULT: (int8_t	RANGE: null	UNITS: null

Rotor Speed Control Mode

Note: This parameter functions identically to INS_HNTCH_MODE.

INS_HNTC3_OPTS (Harmonic Notch Filter options)

DEFAULT: 0	RANGE: null	UNITS: null

Harmonic Notch Filter options

Note: This parameter functions identically to INS_HNTCH_OPTS for a secondary notch instance.

INS_HNTC3_REF (Harmonic Notch Filter reference value)

DEFAULT: 0	RANGE: 0.0 1.0	UNITS: null

Harmonic Notch Filter reference value

Note: This parameter functions identically to INS_HNTCH_REF for a secondary notch instance.

INS_HNTC4_ATT (Harmonic Notch Filter 4 Attenuation)

DEFAULT: 40	RANGE: 10 100	UNITS: dB

Harmonic Notch Filter 4 Attenuation (INS_HNTC4_ATT)

Description

`INS_HNTC4_ATT` determines how aggressively the fourth filter removes noise.

Tuning & Behavior

- **Default Value:** 40 dB.
- **Recommendation:** Leave at **40**.

INS_HNTC4_BW (Harmonic Notch Filter 4 Bandwidth)

DEFAULT: 40	RANGE: 10 250	UNITS: Hz

Harmonic Notch Filter 4 Bandwidth (INS_HNTC4_BW)

Description

`INS_HNTC4_BW` sets the width of the notch for the fourth filter.

Tuning & Behavior

- **Default Value:** 40 Hz.
- **Recommendation:** Set to **half of INS_HNTC4_FREQ**.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





INS_HNTC4_ENABLE (Harmonic Notch Filter 4 Enable)

DEFAULT: 0	RANGE: 0 1

Harmonic Notch Filter 4 Enable (INS_HNTC4_ENABLE)

Description

`INS_HNTC4_ENABLE` turns on an additional dynamic filter.

Most drones only need one or two notch filters. However, complex multi-rotor setups (like a dodecacopter) or frames with multiple vibration sources might need a fourth filter to clean up the gyro signal completely.

Tuning & Behavior

- **0:** Disabled.
- **1:** Enabled.

INS_HNTC4_FM_RAT (Throttle notch min freqency ratio)

DEFAULT: 1.0	RANGE: 0.1 1.0	UNITS: null

Throttle notch min freqency ratio

Note: This parameter functions identically to INS_HNTCH_FM_RAT.

INS_HNTC4_FREQ (Harmonic Notch Filter 4 Frequency)

DEFAULT: 80	RANGE: 10 500	UNITS: Hz

Harmonic Notch Filter 4 Frequency (INS_HNTC4_FREQ)

Description

`INS_HNTC4_FREQ` sets the target frequency for the fourth notch filter.

ArduPilot allows you to stack multiple notch filters. You might use Filter 1 (`HNTCH`) for the motors, and Filter 2 (`HNTC2`) for a specific frame resonance. Filter 4 provides yet another layer of surgical noise removal.

- **Static Mode:** This is the fixed frequency to remove.
- **Dynamic Mode:** This is the reference frequency at the reference throttle/RPM.

Tuning & Behavior

- **Default Value:** 80 Hz.
- **Recommendation:** Use this only if you have a complex noise profile with multiple independent peaks that Filters 1, 2, and 3 couldn't handle.

INS_HNTC4_HMNCS (Harmonic Notch Filter 4 Harmonics)

DEFAULT: 1	RANGE: 0 31

Harmonic Notch Filter 4 Harmonics (INS_HNTC4_HMNCS)

Description

`INS_HNTC4_HMNCS` tells the filter to target multiples of the base frequency.

- **Bit 0 (1):** Fundamental frequency (1x).
- **Bit 1 (2):** 2nd Harmonic (2x).
- **Bit 2 (4):** 3rd Harmonic (3x).

Tuning & Behavior

- **Default Value:** 1 (Base freq only).
- **Recommendation:** Set to **3** (1st + 2nd) or **7** (1st + 2nd + 3rd) if your FFT shows multiple spikes.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





INS_HNTC4_MODE (Harmonic Notch Filter 4 Dynamic Mode)

DEFAULT: 0	RANGE: 0 4

Harmonic Notch Filter 4 Dynamic Mode (INS_HNTC4_MODE)

Description

`INS_HNTC4_MODE` tells the fourth filter how to move.

- **0:** Fixed frequency.
- **1:** Throttle-based tracking.
- **2:** RPM Sensor.
- **3:** ESC Telemetry.
- **4:** Dynamic FFT.

Tuning & Behavior

- **Default Value:** 0.

INS_HNTC4_OPTS (Harmonic Notch Filter options)

DEFAULT: 0	RANGE: null	UNITS: null

Harmonic Notch Filter options

Note: This parameter functions identically to INS_HNTCH_OPTS.

INS_HNTC4_REF (Harmonic Notch Filter 4 Reference)

DEFAULT: 0	RANGE: 0 1.0

Harmonic Notch Filter 4 Reference (INS_HNTC4_REF)

Description

`INS_HNTC4_REF` scales the dynamic frequency for the fourth filter.

Tuning & Behavior

- **Default Value:** 0.
- **Usage:** Set to Hover Throttle if using Mode 1.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





INS_HNTCH_ATT (Harmonic Notch Filter attenuation)

DEFAULT: 40	RANGE: 5 50	UNITS: dB

Harmonic Notch Filter attenuation (INS_HNTCH_ATT)

Description

`INS_HNTCH_ATT` sets how "deep" the cut is. A higher dB value means more noise removal.

- **15-20 dB:** Mild filtering. Good for slightly noisy frames.
- **40 dB (Default):** Deep filtering. Removes almost 99% of the noise at the center frequency.
- **> 40 dB:** Very sharp, deep notch. Can cause phase issues if the bandwidth is too narrow.

The Mathematics

Attenuation A determines the filter coefficient calculations.

$$\text{Gain} = 10^{-A/20}$$

At 40dB, the gain at the center frequency is 0.01.

The Engineer's View

Used in `NotchFilter::calculate_A_and_Q`.
Higher attenuation increases the "sharpness" of the phase transition around the notch.

Tuning & Behavior

- **Default Value:** 40 dB
- **Recommendation:** Default is usually fine.
  - If the filter is causing latency issues (oscillations), try reducing this to 20 or 30 dB.
  - If noise is still getting through, increasing this rarely helps; instead, increase `INS_HNTCH_BW` (Bandwidth).

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





INS_HNTCH_BW (Harmonic Notch Bandwidth)

DEFAULT: 40	RANGE: 5 250	UNITS: Hz

Harmonic Notch Bandwidth (INS_HNTCH_BW)

Description

`INS_HNTCH_BW` sets the "width" of the frequency gap that the notch filter removes.

Imagine the Notch Filter as a "V" shape cut out of the vibration graph.

- **The Bottom of the V** is the Center Frequency (`INS_HNTCH_FREQ`).
- **The Width of the V** is controlled by this Bandwidth parameter.

A wider bandwidth catches more noise (especially if your motors are changing speed rapidly), but it also introduces more **Latency (Delay)** into the flight controller. Too much delay will make the drone wobble or oscillate.

The Mathematics

The quality factor (Q) of the filter is derived from the ratio of the center frequency (f_c) to the bandwidth (BW):

$$Q = \frac{f_c}{BW}$$

A lower bandwidth results in a higher Q (sharper notch, less delay).
A higher bandwidth results in a lower Q (wider notch, more delay).

For dynamic harmonic notch filters, the bandwidth scales with the frequency to maintain a constant Q. This means if your motor frequency doubles, the notch becomes twice as wide to catch the noise.

The Engineer's View

Defined in `AP_InertialSensor`.
This parameter sets the -3dB cutoff points relative to the center frequency.
If `INS_HNTCH_BW` is set to 40Hz and `INS_HNTCH_FREQ` is 80Hz:

- The filter will aggressively attenuate noise between approximately 60Hz and 100Hz.
- The phase lag introduced by the filter is directly proportional to the depth and width of the notch.

Tuning & Behavior

- **Default Value:** 40 Hz.
- **Golden Rule:** Set this to **half of your Center Frequency**.
  - If `INS_HNTCH_FREQ` = 80Hz, set `INS_HNTCH_BW` = 40Hz.
  - If `INS_HNTCH_FREQ` = 200Hz (small racing quad), set `INS_HNTCH_BW` = 100Hz.
- **Warning:** Setting `INS_HNTCH_BW` larger than `INS_HNTCH_FREQ` is invalid and can cause filter instability.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





INS_HNTCH_ENABLE (Harmonic Notch Filter Enable)

DEFAULT: 0	RANGE: 0 1

Harmonic Notch Filter Enable (INS_HNTCH_ENABLE)

Description

`INS_HNTCH_ENABLE` activates the most advanced vibration-fighting tool in ArduPilot: the **Harmonic Notch Filter**.

Standard low-pass filters are like a broad-brush; they smooth everything. A Notch Filter is like a scalpel; it targets one specific frequency (the one caused by your spinning motors) and removes it while leaving everything else untouched.

Enabling this allows the drone to ignore the "buzz" of its own motors, leading to much cooler motors and smoother flight.

The Engineer's View

Defined in `AP_InertialSensor.cpp`.
When enabled, the autopilot allocates resources for a dynamically-tuned notch filter on the gyroscope data path. This filter is placed *after* the raw sampling but *before* the Rate PIDs.

Tuning & Behavior

- **Default Value:** 0 (Disabled)
- **Recommendation:** Every serious drone pilot should enable this. It is a game-changer for flight quality.
- **Reboot Required:** Yes.

INS_HNTCH_FM_RAT (Throttle notch min freqency ratio)

DEFAULT: 1.0	RANGE: 0.1 1.0

Throttle notch min freqency ratio (INS_HNTCH_FM_RAT)

Description

`INS_HNTCH_FM_RAT` prevents the filter frequency from dropping too low when you lower the throttle.

- **1.0 (Default):** The filter never drops below `INS_HNTCH_FREQ`.
- **< 1.0:** The filter can drop to `FREQ * FM_RAT`.

This is important because filtering at very low frequencies (e.g., < 40Hz) introduces massive phase lag (delay) that can destabilize the drone.

The Mathematics

$$f_min_limit = \text{INS_HNTCH_FREQ} \times \text{INS_HNTCH_FM_RAT}$$

$$f_actual = \max(f_calculated, f_min_limit)$$

The Engineer's View

Used in `HarmonicNotchFilter::update()`.
It clamps the lower bound of the center frequency.

Tuning & Behavior

- **Default Value:** 1.0
- **Range:** 0.1 - 1.0
- **Recommendation:** Leave at **1.0** if using Throttle Reference.
- **Exception:** If you set `INS_HNTCH_FREQ` to your *hover* frequency (Throttle Mode), you might want to allow the filter to drop slightly during descent (low throttle). Set to **0.7** to allow it to track down to 70% of hover RPM.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





INS_HNTCH_FREQ (Harmonic Notch Center Frequency)

DEFAULT: 80	RANGE: 10 500	UNITS: Hz

Harmonic Notch Center Frequency (INS_HNTCH_FREQ)

Description

`INS_HNTCH_FREQ` defines the primary frequency of the noise you want to remove.

For most drones, this corresponds to the rotation speed of the motors (in rotations per second) during a steady hover. By targeting this specific frequency, the filter removes the vibration caused by the spinning props before it reaches the flight controller.

The Mathematics

$$\text{Frequency (Hz)} = \frac{\text{RPM}}{60}$$

If your motors hover at 6000 RPM:

$$\text{INS_HNTCH_FREQ} = \frac{6000}{60} = 100 \text{ Hz}$$

If you use Dynamic Harmonic Notch (e.g. Throttle or ESC based), this parameter acts as the **minimum lower bound** or the reference frequency for the configuration.

The Engineer's View

This parameter sets the f_c (center cutoff) for the digital filter.
In "Static Mode" (`INS_HNTCH_MODE` = 0), the filter stays fixed at this frequency.
In "Dynamic Mode", the filter moves. However, `INS_HNTCH_FREQ` is still used to calculate the initial Quality Factor (Q) in conjunction with `INS_HNTCH_BW`.

Tuning & Behavior

- **Default Value:** 80 Hz.
- **How to find it:
    1. Set `INS_LOG_BAT_MASK` = 1 (IMU Pre-Filter).
    2. Do a hover flight.
    3. Download the DataFlash log.
    4. Run an FFT (Fast Fourier Transform) on the `GYRO[0].GyrX` data.
    5. Look for the biggest spike in the graph. That X-axis value is your Frequency.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





INS_HNTCH_HMNCS (Harmonic Notch Filter harmonics)

DEFAULT: 3	RANGE: 0 65535

Harmonic Notch Filter harmonics (INS_HNTCH_HMNCS)

Description

`INS_HNTCH_HMNCS` is a bitmask that tells the filter which multiples of the base frequency to target.

- **Bit 0 (1):** 1st Harmonic (Base Frequency).
- **Bit 1 (2):** 2nd Harmonic ($2 \times F$).
- **Bit 2 (4):** 3rd Harmonic ($3 \times F$).
- **...**

Default is **3 (1 + 2)**, meaning it filters the Base Frequency and the 2nd Harmonic.

The Mathematics

$$\text{Active Filters} = \sum_k = 0^{15} (2^k \text{ is set}) \rightarrow \text{Filter at } (k+1) \cdot f_0$$

The Engineer's View

Each set bit allocates a new `NotchFilter` object.
Allocating too many harmonics (e.g., setting it to 255) consumes CPU and memory.
For typical multicopters, the energy is concentrated in the 1st, 2nd, and sometimes 4th harmonic.

Tuning & Behavior

- **Default Value:** 3 (1st and 2nd Harmonics)
- **Recommendation:**
  - **3 (1st + 2nd):** Good for most quads.
  - **1 (1st only):** If CPU is tight or noise is simple.
  - **7 (1st + 2nd + 3rd):** If spectrogram shows a 3rd peak.
  - **5 (1st + 3rd):** Sometimes useful for specific props.
  - **127 (All):** Don't do this.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





INS_HNTCH_MODE (Harmonic Notch Filter Dynamic Mode)

DEFAULT: 1	RANGE: 0 4

Harmonic Notch Filter Dynamic Mode (INS_HNTCH_MODE)

Description

`INS_HNTCH_MODE` tells the filter how to find the noise. Motor noise moves as you change throttle. This parameter selects the "Sensor" used to track that movement.

- **0: Fixed.** The filter sits at `INS_HNTCH_FREQ` and never moves. Useful for frame resonance that doesn't change.
- **1: Throttle (Default).** Uses the estimated throttle output. If you throttle up, the filter frequency rises. Simple and effective for most drones without RPM sensors.
- **2: RPM Sensor.** Uses real-time data from an RPM sensor. Extremely accurate.
- **3: ESC Telemetry.** Uses RPM data reported by BLHeli_32/AM32 ESCs via DShot. The "Gold Standard" for filtering.
- **4: Dynamic FFT.** Uses a Fast Fourier Transform to "listen" to the noise and find the peak automatically. CPU intensive but requires no external sensors.

The Mathematics

Throttle Mode (1):
$$ f_{center} = \text{FREQ} \cdot \frac{\text{Throttle}*{current} - |*text{Throttle}*{min}}{\text{REF} - \text{Throttle}*{min}} + f*{min} $$
(Simplified linear model).

ESC/RPM Mode (2/3):

$$f_center = \text{RPM}_average / 60$$

The Engineer's View

This maps to the `_tracking_mode` member of `HarmonicNotchFilterParams`.

- **Throttle:** Low latency, but an "estimate". Requires tuning `INS_HNTCH_REF`.
- **ESC Telemetry:** High accuracy, but can have transport latency.
- **FFT:** High latency (window size dependent), but requires zero tuning of reference values.

Tuning & Behavior

- **Default Value:** 1 (Throttle)
- **Recommendation:**
  - If you have **BLHeli_32/AM32 ESCs**, set to **3 (ESC Telemetry)**.
  - If you have a powerful H7 board (Cube Orange, Durandal), set to **4 (FFT)**.
  - Otherwise, stick with **1 (Throttle)** and tune `INS_HNTCH_REF` carefully.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY

INS_HNTCH_OPTS (Harmonic Notch Filter options)

DEFAULT: 0	RANGE: 0 31

Harmonic Notch Filter options (INS_HNTCH_OPTS)

Description

`INS_HNTCH_OPTS` provides advanced tweaks to how the filter operates.

- **Bit 0 (1): Double Notch.** Puts two notches side-by-side to create a wider, flatter stopband. Good for broad noise peaks.
- **Bit 1 (2): Multi-Source.** Used with FFT or ESC telemetry to track multiple independent peaks (e.g., different motors spinning at different speeds) instead of just one average frequency.
- **Bit 2 (4): Update at loop rate.** Updates the filter center frequency every main loop cycle instead of subsampled. Higher CPU usage, better tracking.
- **Bit 3 (8): Enable on All IMUs.** Forces the filter to run on all gyros, not just the primary. **Crucial** for EKF3 affinity/lane switching to work correctly.
- **Bit 4 (16): Triple Notch.** Even wider than Double.

The Mathematics

Double Notch: Instead of one filter at f_0, it places two at $f_0 \pm \delta$, creating a "W" shape (or flat bottom) response.

Multi-Source: Allocates distinct filters for each frequency input source.

The Engineer's View

- **Bit 3 (8) - EnableOnAllIMUs:** Highly recommended for redundancy. If the primary gyro fails and EKF switches to the secondary, you want that secondary to be filtered too.
- **Bit 1 (2) - Multi-Source:** Essential for **Dynamic FFT** setups to track individual motor peaks.

Tuning & Behavior

- **Default Value:** 0
- **Recommendation:**
  - **Set Bit 3 (8) → Value 8.** (Filter all IMUs).
  - **If using FFT:** Set Bit 1 (2) + Bit 4 (16) (Triple Notch) often works well → Value 2 + 16 = 18 (or 2+8 = 10).
  - **If using ESC Telemetry:** Set Bit 1 (2) → Value 2 (Track individual motors).

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





INS_HNTCH_REF (Harmonic Notch Filter reference value)

DEFAULT: 0	RANGE: 0.0 1.0

Harmonic Notch Filter reference value (INS_HNTCH_REF)

Description

`INS_HNTCH_REF` tells the filter how to interpret the dynamic sensor data.

- **Throttle Mode (1):** This is your **Hover Throttle**.
  - If you hover at 35% throttle, set this to **0.35**.
  - The filter assumes that at 0.35 throttle, the noise is at `INS_HNTCH_FREQ`.
- **ESC/RPM Mode (2/3):** This is a **Scaler**.
  - Usually set to **1.0** (No scaling).
  - If you have a helicopter where the RPM sensor reports Motor RPM but the noise is at Rotor RPM (gear ratio), you can use this to scale it. Or if using FFT, this is unused.

The Mathematics

Throttle Mode:

$$\text{Scaling Factor} = \frac{\text{Current Throttle}}{\text{INS_HNTCH_REF}}$$

$$f_notch = \text{INS_HNTCH_FREQ} \times \sqrt{\text{Scaling Factor}}$$

(Note: The relationship is often modeled as square root for RPM vs Throttle).

The Engineer's View

If set to **0** in dynamic modes, the filter effectively becomes static or disabled (depending on code path).
In **FFT Mode**, this parameter is typically ignored, but setting it to a non-zero value is sometimes required to "arm" the dynamic logic in older firmwares.

Tuning & Behavior

- **Default Value:** 0
- **Range:** 0.0 - 1.0
- **Critical for Throttle Mode:** If this is wrong, the notch will move incorrectly. Ideally, perform a "Hover Learn" flight or check `MOT_THST_HOVER` and copy that value here.

INS_LOG_BAT_CNT (Batch Sampler Sample Count)

DEFAULT: 1024	RANGE: 512 16384

Batch Sampler Sample Count (INS_LOG_BAT_CNT)

Description

`INS_LOG_BAT_CNT` determines the resolution of your FFT graph.

A larger sample count means you can see finer frequency details (e.g., distinguishing between 100Hz and 102Hz). However, larger batches take longer to write to the SD card, potentially causing log gaps.

Tuning & Behavior

- **Default Value:** 1024 samples.
- **Recommendation:** 1024 is sufficient for most tuning. Use **4096** for precision analysis.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





INS_LOG_BAT_LGCT (Batch Sampler Log Count)

DEFAULT: 0	RANGE: 0 1000

Batch Sampler Log Count (INS_LOG_BAT_LGCT)

Description

`INS_LOG_BAT_LGCT` acts as a limit switch for the data recorder.

Since batch logs are huge, recording them for an entire 30-minute flight would generate gigabytes of data and potentially crash the logger. This parameter stops the sampler after capturing a set number of snapshots.

Tuning & Behavior

- **Default Value:** 0 (Unlimited/Disabled?). Wait, usually 0 means disabled?
  - *Correction:* In modern ArduPilot, 0 usually means "Disabled" for the *trigger*, or "Unlimited" depending on context. For this parameter, it typically means "Don't stop until disarmed."
- **Recommendation:** Set to **0** for general flights. Set to **10** if you only need a quick sample of the hover.

INS_LOG_BAT_LGIN (Batch Sampler Log Interval)

DEFAULT: 20	RANGE: 0 10000	UNITS: ms

Batch Sampler Log Interval (INS_LOG_BAT_LGIN)

Description

`INS_LOG_BAT_LGIN` controls the "Frame Rate" of the vibration recorder.

It sets the delay between finishing one batch and starting the next.

Tuning & Behavior

- **Default Value:** 20 ms.
- **0:** Disabled (or continuous if supported).
- **Recommendation:** Keep default. Faster logging uses more SD card bandwidth.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





INS_LOG_BAT_MASK (Sensor Bitmask)

DEFAULT: 0	RANGE: 0 511

Sensor Bitmask (INS_LOG_BAT_MASK)

Description

`INS_LOG_BAT_MASK` enables "Batch Sampling". This is a special high-rate logging mode used primarily for **FFT (Fast Fourier Transform) Analysis**.

When enabled, ArduPilot logs short bursts of high-frequency gyro data (e.g., at 1kHz or 2kHz) to the SD card. This data is critical for identifying motor noise frequencies so you can tune the Harmonic Notch Filters.

- **0:** Disabled (Normal logging).
- **1:** Log IMU 1 Gyro.
- **3 (1+2):** Log IMU 1 & 2 Gyros (Recommended for tuning).

The Mathematics

$$\text{Active Samplers} = \sum_k = 0^n (2^k \text{ is set}) \rightarrow \text{Sensor } k$$

- **Bit 0 (1):** IMU 1 Gyro
- **Bit 1 (2):** IMU 2 Gyro
- **Bit 2 (4):** IMU 3 Gyro
- **Bit 3 (8):** IMU 1 Accel
- **Bit 4 (16):** IMU 2 Accel
- ...

The Engineer's View

Used in `AP_InertialSensor::BatchSampler::init()`.
Batch sampling creates massive log files. A 5-minute flight with `INS_LOG_BAT_MASK = 1` can produce a 100MB+ binary log. The data is stored in `ISBH` and `ISBD` messages.

Tuning & Behavior

- **Default Value:** 0 (Disabled)
- **Recommendation:**
  - **For Tuning:** Set to **1** (or 3) for a single hover flight to capture noise data.
  - **For Normal Flight:** Set to **0**. Leaving this on wastes SD card space and CPU cycles (though it runs at low priority).
- **How to use:**
  1. Set `INS_LOG_BAT_MASK = 1`.
  2. Set `INS_LOG_BAT_OPT = 0` (Pre-filter) to see raw noise.
  3. Fly/Hover.
  4. Download log.
  5. Use Mission Planner's "FFT" button to analyze the noise profile.

INS_LOG_BAT_OPT (Batch Sampling Options)

DEFAULT: 0	RANGE: 0 31

Batch Sampling Options (INS_LOG_BAT_OPT)

Description

`INS_LOG_BAT_OPT` configures the high-speed data recorder for vibration analysis.

To tune the Harmonic Notch Filter, you need to know exactly what frequencies your motors are vibrating at. Standard logs are too slow to capture this. The "Batch Sampler" captures short bursts of high-speed gyro data (1kHz+) and writes them to the log.

- **Bit 0:** Sensor Selection. (0 = Gyro, 1 = Accel).
- **Bit 1:** Message Type. (0 = Post-Filter, 1 = Pre-Filter).
- **Bit 2:** Sample Rate. (0 = 1kHz, 1 = 2kHz+).

Tuning & Behavior

- **Default Value:** 0 (Gyro, Post-Filter).
- **Recommendation:** Set to **0** for standard Gyro FFT analysis. Set to **2** (Bit 1) to see the "Raw" noise before any filters are applied.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





INS_POS (IMU Position Offset)

DEFAULT: 0	RANGE: -5 5	UNITS: m

IMU Position Offset (INS_POS)

Description

`INS_POS` (often appearing as `INS_POS_X/Y/Z`) tells the autopilot exactly where the primary flight controller IMU is mounted relative to the drone's balance point (Center of Gravity).

If the flight controller is mounted far away from the center (e.g., at the front of a long airplane), when the plane pitches, the IMU will experience an "Up/Down" acceleration that isn't actually happening to the whole plane. Correctly setting these offsets allows the EKF to mathematically "move" the IMU back to the center of gravity, resulting in a much cleaner and more accurate position estimate.

Tuning & Behavior

- **Default:** 0.
- **Measurement:** Measure the distance from the center of the flight controller to the vehicle's CoG.
  - **X:** Positive = Forward.
  - **Y:** Positive = Right.
  - **Z:** Positive = Down.
- **Recommendation:** Leave at **0** if the flight controller is within 5-10cm of the center. Accuracy is only critical for high-performance navigation or very large frames.

INS_POS1 (IMU 1 Position Offset)

DEFAULT: 0 0 0	RANGE: -5 5	UNITS: m

IMU 1 Position Offset (INS_POS1)

Description

`INS_POS1` defines the physical location of the primary accelerometer.

The EKF assumes the IMU is at the center of the vehicle. If the flight controller is mounted far from the Center of Gravity (CoG), rotation can be misinterpreted as acceleration (centripetal force). This parameter allows the EKF to correct for that lever-arm effect.

Tuning & Behavior

- **Default Value:** 0,0,0.
- **Usage:** Measure the distance from the CoG to the center of the flight controller in meters.
- **X:** Positive Forward.
- **Y:** Positive Right.
- **Z:** Positive Down.

INS_POS2 (null)

DEFAULT: 0.0f	RANGE: null	UNITS: null

None

Note: This parameter functions identically to INS_POS1.

INS_POS3 (null)

DEFAULT: 0.0f	RANGE: null	UNITS: null

None

Note: This parameter functions identically to INS_POS1.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





INS_RAW_LOG_OPT (Raw IMU Logging Options)

DEFAULT: 0	RANGE: 0 65535

Raw IMU Logging Options (INS_RAW_LOG_OPT)

Description

`INS_RAW_LOG_OPT` enables high-speed data recording from the IMUs.

By default, ArduPilot logs filtered IMU data at a reduced rate. This parameter allows you to log the raw, unfiltered sensor data at full speed (up to 4kHz depending on hardware). This is critical for FFT analysis, notch filter tuning, and diagnosing high-frequency vibration issues.

Tuning & Behavior

- **0:** Disabled.
- **1:** Log Primary Gyro/Accel.
- **2:** Log Secondary Gyro/Accel.
- **4:** Log Tertiary Gyro/Accel.
- **7:** Log All IMUs.

Recommendation

Only enable this for short debugging flights. Raw logging consumes significant SD card bandwidth and can fill the logs quickly.

INS_SAMPLE_RATE (IMU Sample Rate)

DEFAULT: 1	RANGE: 1 3	UNITS: kHz

IMU Sample Rate (INS_SAMPLE_RATE)

Description

`INS_SAMPLE_RATE` defines how fast the flight controller talks to the accelerometer and gyroscope chips.

A higher sample rate allows for better vibration filtering (aliasing prevention) and more responsive flight control. However, it also increases the CPU load on the autopilot.

- **1: 1 kHz (Standard).** Suitable for most F4 and F7 flight controllers.
- **2: 2 kHz.**
- **3: 8 kHz.** Only recommended for high-performance H7 flight controllers and high-end IMUs (like ICM series).

Tuning & Behavior

- **Default:** 1 kHz.
- **Recommendation:** Only increase this if your CPU load (`PM_CPU_Usage`) remains low (below 50%) and you are looking for the absolute best flight performance on a racing quad.

INS_STILL_THRESH (INS Stillness Threshold)

DEFAULT: 0.1	RANGE: 0.05 0.5	UNITS: m/s^2

INS Stillness Threshold (INS_STILL_THRESH)

Description

`INS_STILL_THRESH` defines "Motionless."

When calibrating gyroscopes or performing pre-arm checks, the autopilot needs to know if the drone is sitting still. If the accelerometer variance is below this threshold, the system is assumed to be stationary.

Tuning & Behavior

- **Default Value:** 0.1 m/s^2.
- **Increase:** If you are on a boat or a swaying platform.
- **Decrease:** For laboratory precision calibration.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





INS_TCAL1_ACC1 (IMU1 Temperature Cal Accel 1)

DEFAULT: 0	RANGE: -10 10

IMU1 Temperature Cal Accel 1 (INS_TCAL1_ACC1)

Description

`INS_TCAL1_ACC1` is the linear component of the temperature compensation curve for the first IMU's accelerometer.

As the flight controller heats up (due to the STM32 processor or sunny weather), the accelerometer reading can drift. This parameter corrects that drift based on the difference between the current temperature and the calibration reference temperature (35°C).

The Mathematics

The correction applied to the sensor reading is a 3rd-order polynomial:

$$\text{Correction} = \left(C_1 \Delta T + C_2 \Delta T^2 + C_3 \Delta T^3\right) \times 10^{-6}$$

Where:

- C_1 is `INS_TCAL1_ACC1`
- $\Delta T = T_current - 35°C$
- The factor 10^{-6} scales the parameter values (which are stored as large integers for easier GCS editing) back to their physical units.

The Engineer's View

Defined in `AP_InertialSensor_tempcal.cpp`.
This parameter is learned automatically when you run the "Temperature Calibration" routine. You should generally **not** edit this manually unless you are manually fitting a curve to logged data.

Tuning & Behavior

- **Default Value:** 0.
- **Recommendation:** Perform a temperature calibration if you see the EKF attitude drift significantly during the first few minutes of flight (warm-up).

INS_TCAL1_ACC2 (IMU1 Temperature Cal Accel 2)

DEFAULT: 0	RANGE: -10 10

IMU1 Temperature Cal Accel 2 (INS_TCAL1_ACC2)

Description

`INS_TCAL1_ACC2` is the quadratic component of the temperature compensation curve.

The Mathematics

$$\text{Correction} = \left(C_1 \Delta T + C_2 \Delta T^2 + C_3 \Delta T^3\right) \times 10^{-6}$$

Where C_2 is `INS_TCAL1_ACC2`.

Tuning & Behavior

- **Default Value:** 0.
- **Recommendation:** Do not edit manually.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





INS_TCAL1_ACC3 (IMU1 Temperature Cal Accel 3)

DEFAULT: 0	RANGE: -10 10

IMU1 Temperature Cal Accel 3 (INS_TCAL1_ACC3)

Description

`INS_TCAL1_ACC3` is the cubic component of the temperature compensation curve.

The Mathematics

$$\text{Correction} = \left(C_1\Delta T + C_2\Delta T^2 + C_3\Delta T^3\right) \times 10^{-6}$$

Where C_3 is `INS_TCAL1_ACC3`.

Tuning & Behavior

- **Default Value:** 0.
- **Recommendation:** Do not edit manually.

INS_TCAL1_ENABLE (IMU1 Temperature Calibration Enable)

DEFAULT: 0	RANGE: 0 2

IMU1 Temperature Calibration Enable (INS_TCAL1_ENABLE)

Description

`INS_TCAL1_ENABLE` turns on the thermal compensation logic.

IMUs drift when they get hot. If you have performed a temperature calibration (heating the flight controller while it is stationary), setting this to **1** will apply the learned corrections.

Tuning & Behavior

- **0:** Disabled.
- **1:** Enabled (Use learned values).
- **2:** Learn Calibration (Start learning process).

INS_TCAL1_GYR1 (IMU1 Temperature Cal Gyro 1)

DEFAULT: 0	RANGE: -10 10

IMU1 Temperature Cal Gyro 1 (INS_TCAL1_GYR1)

Description

`INS_TCAL1_GYR1` is the linear component of the temperature compensation curve for the first IMU's gyroscope.

As the flight controller heats up, the gyro bias drifts. This parameter corrects that drift.

The Mathematics

$$\text{Correction} = \left(C_1\Delta T + C_2\Delta T^2 + C_3\Delta T^3\right) \times 10^{-6}$$

Where C_1 is `INS_TCAL1_GYR1`.

Tuning & Behavior

- **Default Value:** 0.
- **Recommendation:** Do not edit manually. Perform a temperature calibration.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





INS_TCAL1_GYR2 (IMU1 Temperature Cal Gyro 2)

DEFAULT: 0	RANGE: -10 10

IMU1 Temperature Cal Gyro 2 (INS_TCAL1_GYR2)

Description

`INS_TCAL1_GYR2` is the quadratic component of the temperature compensation curve for the gyro.

The Mathematics

$$\text{Correction} = \left(C_1 \Delta T + C_2 \Delta T^2 + C_3 \Delta T^3\right) \times 10^{-6}$$

Where C_2 is `INS_TCAL1_GYR2`.

Tuning & Behavior

- **Default Value:** 0.

INS_TCAL1_GYR3 (IMU1 Temperature Cal Gyro 3)

DEFAULT: 0	RANGE: -10 10

IMU1 Temperature Cal Gyro 3 (INS_TCAL1_GYR3)

Description

`INS_TCAL1_GYR3` is the cubic component of the temperature compensation curve for the gyro.

The Mathematics

$$\text{Correction} = \left(C_1 \Delta T + C_2 \Delta T^2 + C_3 \Delta T^3\right) \times 10^{-6}$$

Where C_3 is `INS_TCAL1_GYR3`.

Tuning & Behavior

- **Default Value:** 0.

INS_TCAL1_TMAX (IMU1 Temperature Calibration Max)

DEFAULT: 0	RANGE: 0 85	UNITS: degC

IMU1 Temperature Calibration Max (INS_TCAL1_TMAX)

Description

`INS_TCAL1_TMAX` records the hottest temperature reached during calibration.

If the flight controller gets hotter than this during flight, the EKF will stop using the temperature compensation to avoid extrapolating into unknown territory.

Tuning & Behavior

- **Default Value:** 0 (Uncalibrated).
- **Recommendation:** Do not edit manually.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





INS_TCAL1_TMIN (IMU1 Temperature Calibration Min)

DEFAULT: 0	RANGE: -40 85	UNITS: degC

IMU1 Temperature Calibration Min (INS_TCAL1_TMIN)

Description

`INS_TCAL1_TMIN` records the coldest temperature from the calibration.

Tuning & Behavior

- **Default Value:** 0.

INS_TCAL2_ENABLE (Enable pullup after altitude wait)

DEFAULT: 0	RANGE: null	UNITS: null

Enable pullup after altitude wait

Note: This parameter functions identically to INS_TCAL1_ENABLE.

INS_TCAL2_TMAX (Temperature calibration max)

DEFAULT: 70	RANGE: -70 80	UNITS: degC

Temperature calibration max

Note: This parameter functions identically to INS_TCAL1_TMAX.

INS_TCAL2_TMIN (Temperature calibration min)

DEFAULT: 0	RANGE: -70 80	UNITS: degC

Temperature calibration min

Note: This parameter functions identically to INS_TCAL1_TMIN.

INS_TCAL3_ENABLE (Enable pullup after altitude wait)

DEFAULT: 0	RANGE: null	UNITS: null

Enable pullup after altitude wait

Note: This parameter functions identically to INS_TCAL1_ENABLE.

INS_TCAL3_TMAX (Temperature calibration max)

DEFAULT: 70	RANGE: -70 80	UNITS: degC

Temperature calibration max

Note: This parameter functions identically to INS_TCAL1_TMAX.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





INS_TCAL3_TMIN (Temperature calibration min)

DEFAULT: 0	RANGE: -70 80	UNITS: degC

Temperature calibration min

Note: This parameter functions identically to INS_TCAL1_TMIN.

INS_TCAL_ACC1 (Accelerometer 1 Temp Coeff)

DEFAULT: 0	RANGE: -10 10

Accelerometer 1 Temp Coeff (INS_TCAL_ACC1)

Description

`INS_TCAL_ACC1` is a learned calibration value.

During a "Temperature Calibration," the flight controller heats up (either naturally or via `BRD_HEAT_TARG`) while stationary. It measures how the accelerometer bias shifts with temperature and calculates this slope coefficient to subtract the thermal error in flight.

Tuning & Behavior

- **Default Value:** 0 (No compensation).
- **Usage:** Automatically populated by the temp calibration routine.

INS_TCAL_ACC2 (Accelerometer 2nd order temperature coefficient Z axis)

DEFAULT: 0	RANGE: null	UNITS: null

Accelerometer 2nd order temperature coefficient Z axis

Note: This parameter functions identically to INS_TCAL_ACC1.

INS_TCAL_ACC3 (Accelerometer 3rd order temperature coefficient Z axis)

DEFAULT: 0	RANGE: null	UNITS: null

Accelerometer 3rd order temperature coefficient Z axis

Note: This parameter functions identically to INS_TCAL_ACC1.

INS_TCAL_GYR1 (Gyroscope 1 Temp Coeff)

DEFAULT: 0	RANGE: -10 10

Gyroscope 1 Temp Coeff (INS_TCAL_GYR1)

Description

`INS_TCAL_GYR1` is the thermal slope for the gyroscope.

Similar to `INS_TCAL_ACC1`, this value allows the EKF to predict and remove gyro drift caused by temperature changes, improving attitude estimation accuracy during long flights where the board temperature varies.

Tuning & Behavior

- **Default Value:** 0.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





INS_TCAL_GYR2 (Gyroscope 2nd order temperature coefficient Z axis)

DEFAULT: 0	RANGE: null	UNITS: null

Gyroscope 2nd order temperature coefficient Z axis

Note: This parameter functions identically to INS_TCAL_GYR1.

INS_TCAL_GYR3 (Gyroscope 3rd order temperature coefficient Z axis)

DEFAULT: 0	RANGE: null	UNITS: null

Gyroscope 3rd order temperature coefficient Z axis

Note: This parameter functions identically to INS_TCAL_GYR1.

INS_TCAL_OPTIONS (Temperature Calibration Options)

DEFAULT: 0	RANGE: 0 3

Temperature Calibration Options (INS_TCAL_OPTIONS)

Description

`INS_TCAL_OPTIONS` controls how the learned temperature calibration data is stored.

- **Bit 0:** Persist Accel Calibration. Saves the accel offset/scale to the bootloader sector (if supported) so it survives a full parameter reset.
- **Bit 1:** Persist Temp Calibration. Saves the `INS_TCAL` parameters to the bootloader sector.

Tuning & Behavior

- **Default Value:** 0.

INS_TRIM_OPTION (Accel cal trim option)

DEFAULT: 1	RANGE: 0 2

Accel cal trim option (INS_TRIM_OPTION)

Description

`INS_TRIM_OPTION` determines how ArduPilot calculates the AHRS Trim (Level Horizon) during the Accelerometer Calibration.

- **0: Don't adjust trims.** The calibration only calculates scaling/offsets for the sensor. The trim (level) remains unchanged.
- **1: Assume first orientation was level (Default).** The vehicle is assumed to be perfectly level when you start the calibration.
- **2: Assume ACC_BODYFIX is aligned.** Assumes the flight controller is perfectly aligned with the frame, and calculating trims is unnecessary.

The Mathematics

The calibration solves for offsets $\vec{b}$ and scale factors S.
$$ \vec{a}_{cal} = S (\vec{a}_{raw} - \vec{b}) $$
Then, it calculates the rotation matrix R_trim required to make the first sample match the gravity vector $\vec{g} = [0, 0, 1]^T$.

The Engineer's View

Used in `AP_InertialSensor::_calculate_trim()`.
If set to **1**, the "Level" step is implicitly done as part of the 6-point calibration.
If set to **0**, you must run a separate "Calibrate Level" command after the 6-point calibration.

Tuning & Behavior

- **Default Value:** 1
- **Recommendation:** Leave at **1** for most users. Just ensure the drone is level when you click "Calibrate Accelerometers".

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





INS_USE (Use first IMU)

DEFAULT: 1	RANGE: 0 1

Use first IMU (INS_USE)

Description

`INS_USE` toggles the **first** IMU (gyro/accel) for state estimation.

- **1 (Enabled):** The EKF can use this sensor for flight.
- **0 (Disabled):** The sensor is initialized and logged, but the EKF will ignore its data.

The Mathematics

The EKF runs multiple "lanes" (instances). If `INS_USE` is 0, the lane corresponding to this sensor is flagged as "Not Active" or "Unhealthy" (depending on implementation details in EKF2/EKF3).

The Engineer's View

Used in `AP_InertialSensor::use_gyro()` and `use_accel()`.
If set to 0, the driver still runs (unlike `INS_ENABLE_MASK`), but `get_gyro_health()` effectively returns false for the purpose of fusion.

Tuning & Behavior

- **Default Value:** 1
- **Use Case:** If IMU 1 is noisy (e.g., resonance on the specific chip), you can set `INS_USE = 0` to force the EKF to rely on IMU 2 and 3.

INS_USE2 (null)

DEFAULT: 1	RANGE: null	UNITS: null

None

Note: This parameter functions identically to INS_USE.

INS_USE3 (null)

DEFAULT: 1	RANGE: null	UNITS: null

None

Note: This parameter functions identically to INS_USE.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **INTRVAL** parameter group (specifically `INTRVAL_MIN` or `CAM_INTRVAL_MIN`) sets time-based constraints for camera triggering.

Key Concepts

1. Minimum Trigger Interval

Many digital cameras (DSLRs, mirrorless) require time to process an image and clear their buffer before they can take another photo.

- `INTRVAL_MIN`: The minimum time in seconds (or milliseconds, check vehicle defaults) between successive shutter commands.
- **Purpose:** Prevents the autopilot from "spamming" the camera, which can cause the camera to lock up, miss shots, or result in corrupt images.

Developer Notes

- **Library:** `libraries/AP_Camera`.
- **Usage:** Crucial for high-speed mapping missions where waypoints are close together.

Camera minimum time interval between photos ()

INTRVAL_MIN: Camera minimum time interval between photos

Description

Postpone shooting if previous picture was taken less than this many seconds ago.

Values

- **Range:** 0 to 10
- **Units:** s
- **Default:** 0

Description

This parameter enforces a minimum "cooldown" period between camera triggers.

- **Purpose:** Useful for cameras that need time to save a photo to the SD card before taking another one. If the flight controller tries to trigger the camera faster than this rate (e.g., due to high speed and low `CAM_TRIGG_DIST`), the trigger command will be skipped or delayed until the interval has passed.
- **0:** Disabled (triggers as fast as commanded).

Source Code

ardupilot/libraries/AP_Camera/AP_Camera_Params.cpp

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





JERK LIMIT CONFIGURATION (JERK)

Overview

The **JERK** parameter group (often appearing as `PSC_JERK_` in the full parameter list) configures the smoothness of the vehicle's motion.

"Jerk" is the rate of change of acceleration (the 3rd derivative of position). By limiting jerk, the autopilot ensures that the vehicle doesn't "snap" its movements, resulting in a more fluid, organic, and cinematographic flight feel.

Key Concepts

1. Motion Kinematics

- **Velocity:** Speed.
- **Acceleration:** How fast speed changes.
- **Jerk:** How fast acceleration changes.

2. S-Curve Path Planning

ArduPilot uses **S-Curves** for navigation. S-Curves ensure that velocity increases and decreases smoothly by strictly controlling the jerk levels.

- `JERK_XY`: Horizontal jerk limit.
- `JERK_Z`: Vertical jerk limit.

Parameter Breakdown

- `JERK_XY`: Maximum horizontal jerk (m/s/s/s).
  - **Higher values:** Snappier, more aggressive waypoint turns.
  - **Lower values:** Smoother, smoother starts and stops.

Integration Guide

- **Cinematography:** Decrease these values for ultra-smooth pans and steady shots.
- **Racing/Agility:** Increase these values to allow the drone to "snap" onto its new heading/velocity quickly.

Developer Notes

- **Library:** `libraries/AC_PrecLand`, `libraries/AR_AttitudeControl`, and `AC_PosControl`.
- **Constraint:** These parameters limit the output of the position controller's path generator.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





`Jerk limit for the horizontal kinematic input shaping ()`

JERK_XY: Jerk limit for the horizontal kinematic input shaping

Description

Jerk limit of the horizontal kinematic path generation used to determine how quickly the aircraft varies the acceleration target.

Values

- **Range:** 1 to 20
- **Units:** m/s^2/s
- **Increment:** 1
- **Default:** 5.0

Description

This parameter controls the smoothness of horizontal acceleration changes in autonomous modes (like Loiter, Auto, Guided). "Jerk" is the rate of change of acceleration.

- **Lower Value (e.g., 2-4):** Acceleration ramps up and down slowly. The vehicle will feel very smooth and fluid but may feel "loose" or slow to react to stick inputs in Loiter.
- **Higher Value (e.g., 10-15):** Acceleration changes rapidly. The vehicle will feel responsive and "snappy" but may be jerky or twitchy if set too high for the physical capabilities of the airframe.

Source Code

ardupilot/libraries/AC_AttitudeControl/AC_PosControl.cpp

`Jerk limit for the vertical kinematic input shaping ()`

JERK_Z: Jerk limit for the vertical kinematic input shaping

Description

Jerk limit of the vertical kinematic path generation used to determine how quickly the aircraft varies the acceleration target.

Values

- **Range:** 5 to 50
- **Units:** m/s^2/s
- **Increment:** 1
- **Default:** 5.0

Description

This parameter controls the smoothness of vertical (Z-axis) acceleration changes in autonomous modes. It effectively limits how fast the autopilot can change the climb or descent rate command.

- **Lower Value:** Smoother transitions between climbing, holding altitude, and descending. Prevents sudden "jumps" in motor output.
- **Higher Value:** Faster reaction to altitude change requests, but may cause the vehicle to feel "hard" or jerky in altitude changes.

Source Code

ardupilot/libraries/AC_AttitudeControl/AC_PosControl.cpp

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **JS** parameter group configures the behavior of a **USB Joystick or Gamepad** when used as the primary control interface. This is most common in **ArduSub** (ROV) and some Ground Vehicle setups where a traditional RC transmitter is not used.

The joystick commands are sent from the GCS (Mission Planner/QGC) to the autopilot via MAVLink `MANUAL_CONTROL` messages.

Key Concepts

1. Gain Control (`JS_GAIN_...`)

Because joysticks often have shorter travel than RC sticks, or are used in sensitive environments, the system uses a "Gain" to scale the inputs.

- `JS_GAIN_DEFAULT`: The multiplier applied to all sticks at startup.
- `JS_GAIN_STEPS`: The number of increments used when the pilot uses "Gain Up/Down" buttons on the gamepad.

2. Thrust Scaling (`JS_THR_GAIN`)

Specific scaling for the throttle/vertical axis, allowing for fine depth control while maintaining high horizontal maneuverability.

Parameter Breakdown

- `JS_GAIN_DEFAULT`: Starting gain (e.g., 0.5 = 50% power).
- `JS_GAIN_MAX`: Maximum allowable gain.
- `JS_GAIN_MIN`: Minimum gain.

Integration Guide

1. **Hardware:** Plug a gamepad into your PC running the GCS.
2. **Calibrate:** Perform "Joystick Calibration" in the GCS.
3. **Functions:** Map gamepad buttons to ArduPilot functions (e.g., `Arm`, `Mode Change`, `Gripper`).
4. **Tuning:** Use the gain buttons in-mission to adjust sensitivity based on current conditions (e.g., lower gain for precise work near a structure).

Developer Notes

- **Library:** `ArduSub` (mostly).
- **Logic:** Input is received via `GCS_MAVLINK::handle_manual_control()`.

JS_GAIN_DEFAULT (Joystick Default Gain)

DEFAULT: 0.5	RANGE: 0.1 1.0

Joystick Default Gain (JS_GAIN_DEFAULT)

Description

`JS_GAIN_DEFAULT` defines the "Responsiveness" of the vehicle when using a USB joystick (most common in **ArduSub** ROV applications).

It acts as a master multiplier for all control sticks.

- **0.5 (Default):** 50% sensitivity. Moving the stick all the way only commands 50% power. This is excellent for beginners or precision maneuvering.
- **1.0:** 100% sensitivity. Full stick movement commands full power.

Tuning & Behavior

- **Default:** 0.5.
- **Usage:** If you find the vehicle is too "twitchy" or difficult to control smoothly at boot, decrease this value. If you always find yourself needing more speed, increase it.
- **Live Adjustment:** In ArduSub, this gain is often adjusted in real-time during a dive using buttons on the joystick, but it will revert to this default value when the vehicle is rebooted.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





JS_GAIN_MAX (Joystick Maximum Gain)

DEFAULT: 1.0	RANGE: 0.1 1.0

Joystick Maximum Gain (JS_GAIN_MAX)

Description

`JS_GAIN_MAX` sets the upper boundary for the real-time gain adjustment feature.

In many applications (like ROVs), pilots use joystick buttons to "Turn up" or "Turn down" the sensitivity during a mission. This parameter ensures that even if you click the "Increase Gain" button many times, the sensitivity will never exceed this safe limit.

Tuning & Behavior

- **Default:** 1.0 (Allow full 100% sensitivity).
- **Safety:** If you find that the vehicle is dangerously fast or unstable at 100% gain, reduce this value to **0.8** or **0.7** to "Cap" the maximum performance.

JS_GAIN_MIN (Joystick Minimum Gain)

DEFAULT: 0.25	RANGE: 0.1 0.8

Joystick Minimum Gain (JS_GAIN_MIN)

Description

`JS_GAIN_MIN` sets the lower boundary for the real-time gain adjustment feature.

This ensures that even if you click the "Decrease Gain" button many times, the vehicle's sensitivity will never drop so low that it becomes unresponsive. This prevents a pilot from accidentally "Disabling" their controls while trying to slow down for precision work.

Tuning & Behavior

- **Default:** 0.25 (Minimum 25% sensitivity).
- **Usage:** If you need extremely fine precision (e.g. using a robotic arm on an ROV), you might lower this to **0.1** (10%).

JS_GAIN_STEPS (Joystick Gain Steps)

DEFAULT: 4	RANGE: 1 10	UNITS: Steps

Joystick Gain Steps (JS_GAIN_STEPS)

Description

`JS_GAIN_STEPS` determines how many times you have to click a button to go from JS_GAIN_MIN to JS_GAIN_MAX.

It sets the "Resolution" of the in-flight gain adjustment.

- **Default:** 4 steps. (e.g., 25%, 50%, 75%, 100%).
- **Usage:** Increase this if you want finer control over your vehicle's speed and responsiveness during a mission.

JS_LIGHTS_STEPS (Joystick Lights Steps)

DEFAULT: 8	RANGE: 1 20	UNITS: Steps

Joystick Lights Steps (JS_LIGHTS_STEPS)

Description

`JS_LIGHTS_STEPS` sets the "Brightness Increments" for ROV external lights.

When you use a joystick button to dim or brighten your underwater lights, this parameter defines how much the brightness changes per click.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





JS_THR_GAIN (Joystick Throttle Gain)

DEFAULT: 1.0	RANGE: 0.1 1.0

Joystick Throttle Gain (JS_THR_GAIN)

Description

`JS_THR_GAIN` allows you to independently scale the throttle sensitivity without affecting the rotation or lateral axes.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **KDE** parameter group (specifically `KDE_NPOLE`) configures the interface for **KDE Direct CAN ESCs**.

Key Concepts

1. Pole Count

To calculate accurate RPM from a brushless motor, the ESC (and the autopilot) must know how many permanent magnets (poles) are in the motor bell.

- `KDE_NPOLE`: Sets the number of poles for the connected KDE motors.
  - **Calculation:** Total magnetic poles (usually an even number like 14, 28, etc.).

Developer Notes

- **Library:** `libraries/AP_KDECAN`.
- **Context:** Only relevant if using KDE Direct specialized CAN ESCs.

Number of motor poles ()

KDE_NPOLE: Number of motor poles

Description

Sets the number of motor poles (magnetic poles on the bell) to calculate the correct RPM value from the KDECAN ESC telemetry.

Values

- **Range:** 2 to 255
- **Increment:** 2
- **Default:** 14

Description

This parameter ensures the RPM reported by your KDE Direct CAN ESCs is accurate. ESCs measure "electrical RPM" based on the commutations. To get physical RPM, the autopilot needs to divide the electrical RPM by half the number of motor poles.

- **Check your motor specs:** Look for "Magnetic Poles" or "Magnet Poles" in your motor's datasheet.
- **Common Values:**
  - 14 (Standard for many heavy-lift motors like KDE Direct)
  - 12, 22, 24, 28 (Other common configurations)

Source Code

ardupilot/libraries/AP_KDECAN/AP_KDECAN.cpp

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **KFF** parameter group (often appearing as `KFF_` or `MIX_`) configures **Feedforward** mixing for ArduPlane.

Feedforward provides an immediate control surface movement proportional to the pilot's stick input (or another control axis), bypassing the slower PID error-correction loops.

Key Concepts

1. Rudder Mixing (`KFF_RDDRMIX`)

Used on aircraft without ailerons ("Rudder Only").

- **Function:** It mixes the Roll controller output into the Rudder servo. This allows the autopilot to turn the plane using only the rudder.

2. Pitch Mixing (`KFF_THR2PTCH`)

Mixes throttle into the pitch axis.

- **Function:** If your plane pitches up when you add power (due to motor thrust line), this parameter can automatically add down-elevator to compensate.

Parameter Breakdown

- `KFF_RDDRMIX`: Roll to Rudder feedforward gain.
- `KFF_THR2PTCH`: Throttle to Pitch feedforward gain.

Developer Notes

- **Library:** `ArduPlane/servos.cpp`.

KFF_RDDRMIX (Rudder to Aileron Mix)

DEFAULT: 0.5	RANGE: 0 1.0

Rudder to Aileron Mix (KFF_RDDRMIX)

Description

`KFF_RDDRMIX` automates "Coordinated Turns" for airplanes.

When a plane rolls into a turn using ailerons, it often experiences "Adverse Yaw" (the tail swinging the wrong way). This parameter automatically adds a proportional amount of rudder to keep the nose pointed into the wind during the roll.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





KFF_THR2PTCH (Throttle to Pitch Mix)

DEFAULT: 0	RANGE: -5 5	UNITS: deg

Throttle to Pitch Mix (KFF_THR2PTCH)

Description

`KFF_THR2PTCH` compensates for the physical design of an aircraft where the motor's thrust "pushes" the nose up or down.

Many planes, especially those with high or low mounted motors, tend to pitch up when you add power. To fly level, you have to manually push the elevator down. This parameter automates that correction. If you set it to **2.0**, then at 100% throttle, the autopilot will automatically command a 2-degree pitch-down correction.

Tuning & Behavior

- **Positive Values:** Pitch the nose **UP** as throttle increases.
- **Negative Values:** Pitch the nose **DOWN** as throttle increases.
- **Calibration:** Fly the plane in MANUAL mode. Add power and see which way the nose moves. Adjust this parameter until the plane maintains a level attitude when changing throttle.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **LAND** parameter group configures the final stages of flight: **Landing** and **Auto-Landing**.

Effective landing parameters ensure the vehicle touches down softly, stays straight on the runway, and disarms promptly after contact.

Key Concepts

1. Vertical Speed (Copter)

- `LAND_SPEED`: The slow descent rate (m/s) used during the final few meters of touchdown.
- `LAND_SPEED_HIGH`: The faster descent rate used for the initial descent from altitude before slowing down for the final touchdown.

2. Plane Auto-Landing

- `LAND_FLARE_ALT`: The altitude (meters) at which the plane begins to "Flare" (pitch up to stall onto the runway).
- `LAND_PITCH_DEG`: The target pitch angle during the flare.
- `LAND_ABORT_DEG`: If the pilot (or autopilot) decides to abort the landing, this is the pitch angle it will climb at.

3. Deep Stall Landing (`LAND_DS_...`)

(Advanced Plane feature) Used for landing in very tight spaces by intentionally stalling the wing at a high angle of attack.

Parameter Breakdown

- `LAND_TYPE`: Selection of landing logic (e.g., standard vs deepstall).
- `LAND_DISARMDELAY`: How long to wait after touchdown before automatically disarming.
- `LAND_OPTIONS`: Bitmask for specific behaviors (e.g., allow repositioning with sticks during auto-land).

Integration Guide

- **Soft Landing:** If your copter bounces on landing, reduce `LAND_SPEED`.
- **Landing Flare:** If your plane hits the runway nose-first, increase `LAND_FLARE_ALT` or `LAND_PITCH_DEG`.

Developer Notes

- **Library:** `libraries/AP_Landing` (Shared), `ArduCopter/mode_land.cpp`, `ArduPlane/mode_landing.cpp`.

LAND_ABORT_DEG (Land Abort Angle)

DEFAULT: 0	RANGE: 0 90	UNITS: deg

Land Abort Angle (LAND_ABORT_DEG)

Description

`LAND_ABORT_DEG` prevents the drone from landing if it is pitching too violently.

If the drone is fighting strong wind and pitching up/down more than this angle, it assumes the landing is unsafe and will climb back up (Go Around).

Tuning & Behavior

- **Default Value:** 0 (Disabled).
- **Recommendation:** Set to **15 degrees** for fixed-wing or quadplanes to prevent propeller strikes.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





LAND_ABORT_THR (Land Abort Throttle)

DEFAULT: 0	RANGE: 0 100	UNITS: %

Land Abort Throttle (LAND_ABORT_THR)

Description

`LAND_ABORT_THR` triggers a Go-Around if the motor demand is too high.

If the drone is descending but needs 90% throttle to maintain the path (due to wind shear or a heavy payload), it's dangerous to continue. This parameter forces an abort.

Tuning & Behavior

- **Default Value:** 0 (Disabled).
- **Recommendation:** Set to **90%** to ensure you always have headroom to climb out.

LAND_ALT_LOW (Land Altitude Low)

DEFAULT: 10	RANGE: 0 30	UNITS: m

Land Altitude Low (LAND_ALT_LOW)

Description

`LAND_ALT_LOW` determines the "Slow Down Height" for automatic landings.

When a drone is landing automatically (e.g., during a Return-to-Launch), it usually descends quickly to save battery. However, descending fast into the ground would cause a crash. `LAND_ALT_LOW` sets the altitude where the drone transitions from a fast "Transit Descent" to a slow, gentle "Touchdown Descent."

The Mathematics

The autopilot manages the descent rate (v_z) based on the current altitude (h):

- If $h > \text{LAND_ALT_LOW}$, $v_z = \text{WPNAV_SPEED_DN}$.
- If $h \leq \text{LAND_ALT_LOW}$, v_z is linearly ramped down until it reaches `LAND_SPEED` at the ground.

The Engineer's View

Defined in the vehicle-specific `Parameters.cpp` (e.g., `ArduCopter`).
This parameter is critical for preventing "ground effect" instability. If the drone descends too fast into its own propwash near the ground, it can wobble or lose control. By slowing down at a specific height, the flight controller has more time to stabilize the airframe before contact.

Tuning & Behavior

- **Default Value:** 10 meters.
- **High Performance:** For large industrial drones, you may want to increase this to **15 or 20 meters** to ensure a very stable approach.
- **Small Quads:** For racing or small freestyle quads, you can lower this to **5 meters** for a faster recovery.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY



LAND_DISARMDELAY (Land Disarm Delay)

DEFAULT: 20	RANGE: 0 127	UNITS: s

Land Disarm Delay (LAND_DISARMDELAY)

Description

`LAND_DISARMDELAY` is the "settling time" for the Land Detector.

After the sensors think the drone has touched the ground (Velocity = 0, Throttle = Min), the autopilot waits this many seconds to be *sure* before killing the motors.

Tuning & Behavior

- **Default Value:** 20 seconds (Fixed Wing). For Copter, this is usually `DISARM_DELAY` (default 2s).
- **Recommendation:**
  - **Fixed Wing:** 20s allows for taxiing or stopping.
  - **Copter:** 2s prevents tip-overs.

LAND_DS_ABORTALT (Deepstall Abort Altitude)

DEFAULT: 0	RANGE: 0 50	UNITS: m

Deepstall Abort Altitude (LAND_DS_ABORTALT)

Description

`LAND_DS_ABORTALT` defines the "Point of No Return" for deepstall landings.

Once the plane has stalled and is falling, it takes time and altitude to recover into forward flight. If you trigger an abort (Go Around) when the plane is too low, it will hit the ground before it can regain airspeed. This parameter prevents the autopilot from trying to recover if it is below this safety height.

Tuning & Behavior

- **Default Value:** 0 (Always allowed to try).
- **Recommendation:** Set to the altitude it takes your plane to recover from a full stall plus 10 meters buffer (e.g. **20m**).

LAND_DS_AIL_SCL (Deepstall Aileron Gain Scaling)

DEFAULT: 1.0	RANGE: 0 2.0

Deepstall Aileron Gain Scaling (LAND_DS_AIL_SCL)

Description

`LAND_DS_AIL_SCL` adjusts how much the ailerons move to steer the plane while it is stalling.

In a deep stall, the wing is fully stalled, so ailerons might work differently (often reversed or less effective). This parameter allows you to tweak the steering aggression specifically for this flight phase without messing up your normal flight PID gains.

Tuning & Behavior

- **Default Value:** 1.0 (No change).
- **Recommendation:** If the plane oscillates in roll while falling, reduce to **0.5**.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





LAND_DS_APP_EXT (Deepstall Approach Extension)

DEFAULT: 50	RANGE: 10 200	UNITS: m

Deepstall Approach Extension (LAND_DS_APP_EXT)

Description

`LAND_DS_APP_EXT` adds a "Straightaway" to the deepstall landing path.

Before the plane enters a stall, it needs to be perfectly lined up with the target and flying level. This parameter ensures the plane flies in a straight line for at least this distance before pulling the elevator up to stall.

Tuning & Behavior

- **Default Value:** 50 meters.
- **Recommendation:** Increase to **100m** if your plane has high inertia and needs more time to settle onto its final heading.

LAND_DS_ARSP_MAX (Deepstall Enabled Airspeed)

DEFAULT: 15.0	RANGE: 5 20	UNITS: m/s

Deepstall Enabled Airspeed (LAND_DS_ARSP_MAX)

Description

`LAND_DS_ARSP_MAX` defines the "Entry Speed" for the deep stall maneuver.

When the plane slows down past this speed, the deep stall controller wakes up and starts blending in its own steering logic.

Tuning & Behavior

- **Default Value:** 15.0 m/s.
- **Recommendation:** Set slightly above your stall speed.

LAND_DS_ARSP_MIN (Deepstall Minimum Derating Airspeed)

DEFAULT: 10.0	RANGE: 5 20	UNITS: m/s

Deepstall Minimum Derating Airspeed (LAND_DS_ARSP_MIN)

Description

`LAND_DS_ARSP_MIN` defines the "Fully Stalled" speed.

Between `ARSP_MAX` and `ARSP_MIN`, the control authority is mixed. Below `ARSP_MIN`, the deep stall controller is in full command.

Tuning & Behavior

- **Default Value:** 10.0 m/s.
- **Recommendation:** Set to your deep-stall vertical descent speed converted to forward speed equivalent, or just below stall speed.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





LAND_DS_D (Deepstall Steering D Gain)

DEFAULT: 0	RANGE: 0 0.5

Deepstall Steering D Gain (LAND_DS_D)

Description

`LAND_DS_D` dampens the steering response to prevent oscillations.

Tuning & Behavior

- **Default Value:** 0.
- **Recommendation:** Increase if you see rapid wobbling.

LAND_DS_ELEV_PWM (Deepstall Elevator PWM)

DEFAULT: 1500	RANGE: 900 2100	UNITS: PWM

Deepstall Elevator PWM (LAND_DS_ELEV_PWM)

Description

`LAND_DS_ELEV_PWM` is the "Stall Position."

This is the exact servo signal sent to the elevator to hold the nose up (or down, for inverted deep stall).

Tuning & Behavior

- **Default Value:** 1500 (Neutral).
- **Recommendation:** Set to your maximum up-elevator PWM (e.g., 1900 or 1100 depending on reversal). Ensure it is enough to keep the wing fully stalled.

LAND_DS_I (Deepstall Steering I Gain)

DEFAULT: 0	RANGE: 0 5

Deepstall Steering I Gain (LAND_DS_I)

Description

`LAND_DS_I` corrects for wind drift during the descent.

Tuning & Behavior

- **Default Value:** 0.
- **Recommendation:** Use sparingly to correct steady-state heading errors.

LAND_DS_L1 (Deepstall L1 Period)

DEFAULT: 30.0	RANGE: 5 50	UNITS: s

Deepstall L1 Period (LAND_DS_L1)

Description

`LAND_DS_L1` defines how aggressively the plane tries to return to the path.

A shorter period means the plane will turn sharper to get back on track.

Tuning & Behavior

- **Default Value:** 30.0 s.
- **Recommendation:** Since deep stall happens slowly, a long period like 30s is usually appropriate.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





LAND_DS_L1_I (Deepstall L1 Integrator Gain)

DEFAULT: 0	RANGE: 0 1

Deepstall L1 Integrator Gain (LAND_DS_L1_I)

Description

`LAND_DS_L1_I` helps the plane stay on the line in a crosswind.

Tuning & Behavior

- **Default Value:** 0.
- **Recommendation:** Increase if the plane consistently lands downwind of the target point.

LAND_DS_L1_TCON (Deepstall L1 Time Constant)

DEFAULT: 0.4	RANGE: 0.1 1.0	UNITS: s

Deepstall L1 Time Constant (LAND_DS_L1_TCON)

Description

`LAND_DS_L1_TCON` defines how aggressively a fixed-wing aircraft should correct its course during a **Deep Stall** landing.

In a deep stall landing, the plane intentionally stalls its wings to drop vertically onto a target. Because the wings have very little lift and airflow is turbulent, standard steering is less effective. This parameter tunes the "L1" navigation controller's responsiveness in this specific state.

- **Lower Values (0.1 - 0.4):** More aggressive corrections. The plane will try very hard to stay on the line, but might wobble.
- **Higher Values (0.5 - 1.0):** Smoother, lazier corrections. The plane will wander more but flight will be more stable.

Tuning & Behavior

- **Default:** 0.4.
- **Usage:** Only relevant for planes configured for Deep Stall landings (e.g. some foam hobby planes and specialized UAVs).

LAND_DS_P (Deepstall Steering P Gain)

DEFAULT: 0	RANGE: 0 5

Deepstall Steering P Gain (LAND_DS_P)

Description

`LAND_DS_P` controls how aggressively the plane tries to steer towards the landing point while falling.

Tuning & Behavior

- **Default Value:** 0.
- **Recommendation:** Tune carefully. Too high, and the plane will rock violently. Too low, and it drifts with the wind.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





LAND_DS_SLEW_SPD (Deepstall Slew Speed)

DEFAULT: 0.5	RANGE: 0 2	UNITS: s

Deepstall Slew Speed (LAND_DS_SLEW_SPD)

Description

`LAND_DS_SLEW_SPD` controls how fast the plane "pops up" into the stall.

- **Fast (0.1s):** Violent entry. Good for precise spot landing but high stress on the airframe.
- **Slow (1.0s):** Gentle entry. The plane will balloon up before falling.

Tuning & Behavior

- **Default Value:** 0.5 s.
- **Recommendation:** Adjust to get a clean break into the stall without zooming up too high.

LAND_DS_SLOPE_A (Deepstall Slope A)

DEFAULT: 1

LAND_DS_SLOPE_A: Deepstall Slope A

Description

The wind coefficient for the deepstall landing distance calculation (Distance = A * Wind + B).

Values

- **Default:** 1

Description

This parameter is part of the **DeepStall** landing system. It helps predict how far the plane will travel horizontally during the deepstall descent based on the wind speed.

- **Formula:** `Travel Distance = (LAND_DS_SLOPE_A * Wind_Speed) + LAND_DS_SLOPE_B`
- **Function:** The autopilot uses this estimate to decide exactly when to trigger the stall so that the plane lands on the target point.
- **Tuning:** Requires empirical testing. Measure the landing distance vs. wind speed over several flights to determine the slope.

LAND_DS_SLOPE_B (Deepstall Slope B)

DEFAULT: 1	UNITS: m

LAND_DS_SLOPE_B: Deepstall Slope B

Description

The offset coefficient for the deepstall landing distance calculation (Distance = A * Wind + B).

Values

- **Default:** 1
- **Units:** m

Description

This parameter represents the **base horizontal distance** the aircraft travels during a deepstall descent in zero wind conditions.

- **Formula:** `Travel Distance = (LAND_DS_SLOPE_A * Wind_Speed) + LAND_DS_SLOPE_B`
- **Function:** It accounts for the forward travel due to the plane's drag profile and entry speed.
- **Tuning:** Perform test deepstalls in calm conditions to determine this baseline distance.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





LAND_DS_V_DWN (Deepstall Sink Rate)

DEFAULT: 2	RANGE: 0 20	UNITS: m/s

Deepstall Sink Rate (LAND_DS_V_DWN)

Description

`LAND_DS_V_DWN` defines how fast the plane falls once it has stalled.

Tuning & Behavior

- **Default Value:** 2.0 m/s.
- **Recommendation:** High-drag planes (large wings) will fall slower. High-weight planes will fall faster.

LAND_DS_V_FWD (Deepstall Forward Velocity)

DEFAULT: 1	RANGE: 0 20	UNITS: m/s

Deepstall Forward Velocity (LAND_DS_V_FWD)

Description

`LAND_DS_V_FWD` is a prediction parameter for Deepstall landings (where the plane purposefully stalls to drop vertically).

Because the aircraft has no lift and very little control, the autopilot needs to "guess" how much it will drift forward while falling. This allows it to calculate the exact point in the air to trigger the stall so that the drone lands on the target.

The Mathematics

The predicted travel distance D is calculated using the ratio of forward speed V_fwd to sink rate V_dwn:

$$D_drift = \frac{V_fwd \times \text{Height}}{V_dwn} + \text{Stall Entry Distance}$$

Tuning & Behavior

- **Default Value:** 1.0 m/s.
- **Tuning:** If the plane consistently lands **Short** of the target, decrease this value. If it lands **Long**, increase it.

LAND_DS_YAW_LIM (Deepstall Yaw Rate Limit)

DEFAULT: 10	RANGE: 0 90	UNITS: deg/s

Deepstall Yaw Rate Limit (LAND_DS_YAW_LIM)

Description

`LAND_DS_YAW_LIM` prevents the plane from spinning like a top while falling.

In a deepstall, the rudder is often less effective or can cause roll coupling. This parameter limits how hard the L1 controller tries to steer the heading.

Tuning & Behavior

- **Default Value:** 10 deg/s.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





LAND_FLAP_PERCNT (Landing Flap Percentage)

DEFAULT: 0	RANGE: 0 100	UNITS: %

Landing Flap Percentage (LAND_FLAP_PERCNT)

Description

`LAND_FLAP_PERCNT` sets the flap position for the final approach.

- **0:** Use the standard `FLAP_1_PERCENT` or automatic flap logic.
- **1-100:** Override with a specific landing setting (e.g. Full Flaps).

Tuning & Behavior

- **Default Value:** 0.

LAND_FLARE_AIM (Land Flare Aim Altitude)

DEFAULT: 0	RANGE: -1 1	UNITS: m

Land Flare Aim Altitude (LAND_FLARE_AIM)

Description

`LAND_FLARE_AIM` aims the flare path slightly underground.

By aiming for an altitude of **-0.5 meters** (half a meter below the runway), you ensure that the plane positively touches down rather than floating indefinitely just a few inches above the ground.

Tuning & Behavior

- **Default Value:** 0 m.
- **Recommendation:** Set to **-0.5** or **-1.0** to reduce "Float" and force a touchdown.

LAND_FLARE_ALT (Land Flare Altitude)

DEFAULT: 3	RANGE: 0 10	UNITS: m

Land Flare Altitude (LAND_FLARE_ALT)

Description

`LAND_FLARE_ALT` sets the "Pull Up" height for a fixed-wing autonomous landing.

As the plane approaches the runway, it flies a descending path. At this altitude, it stops following the path and starts the "Flare" (pulling the nose up to bleed off speed and descend very slowly until touchdown).

Tuning & Behavior

- **Default Value:** 3 meters.
- **Recommendation:** Set to **2-4 meters** for most small UAVs.
- **Too Low:** The plane will hit the ground before it can slow down (Hard landing).
- **Too High:** The plane will "Float" above the runway for too long and might stall and drop.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





LAND_FLARE_SEC (Land Flare Time)

DEFAULT: 2.0	RANGE: 0 5	UNITS: s

Land Flare Time (LAND_FLARE_SEC)

Description

`LAND_FLARE_SEC` is a time-based backup for starting the landing flare.

The autopilot calculates when it *expects* to hit the ground based on its current sink rate. If that time becomes less than this value, it will start the flare even if it hasn't reached `LAND_FLARE_ALT` yet. This helps in high-sink-rate situations where the plane might otherwise hit the ground before the altitude-based trigger can react.

Tuning & Behavior

- **Default Value:** 2.0 seconds.

LAND_OPTIONS (Landing Options)

DEFAULT: 0	RANGE: 0 7

Landing Options (LAND_OPTIONS)

Description

`LAND_OPTIONS` tweaks the landing logic.

- **Bit 0:** Use Rangefinder for Flare. Requires a Lidar. Much more precise than Baro.
- **Bit 1:** Idle throttle during Pre-Flare.

Tuning & Behavior

- **Default Value:** 0.

LAND_PF_ALT (Land Pre-Flare Altitude)

DEFAULT: 0	RANGE: 0 30	UNITS: m

Land Pre-Flare Altitude (LAND_PF_ALT)

Description

`LAND_PF_ALT` adds a "Slow Down" zone before the flare.

At this altitude, the plane will reduce its airspeed target to `LAND_PF_ARSPD`. This allows high-speed aircraft to bleed off energy safely before they are right on top of the runway.

Tuning & Behavior

- **Default Value:** 0 (Disabled).
- **Recommendation:** Set to **10-15m** for fast jets or slippery gliders.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





LAND_PF_ARSPD (Land Pre-Flare Airspeed)

DEFAULT: 0	RANGE: 0 30	UNITS: m/s

Land Pre-Flare Airspeed (LAND_PF_ARSPD)

Description

`LAND_PF_ARSPD` sets the speed limit for the "Final Approach."

This is typically slower than cruise but faster than stall speed. It bridges the gap between flying speed and landing speed.

Tuning & Behavior

- **Default Value:** 0 (Disabled).

LAND_PF_SEC (Land Pre-Flare Time)

DEFAULT: 0	RANGE: 0 10	UNITS: s

Land Pre-Flare Time (LAND_PF_SEC)

Description

`LAND_PF_SEC` triggers the pre-flare based on time-to-impact.

Tuning & Behavior

- **Default Value:** 0 (Disabled).

LAND_PITCH_DEG (Land Flare Pitch Angle)

DEFAULT: 0	RANGE: 0 15	UNITS: deg

Land Flare Pitch Angle (LAND_PITCH_DEG)

Description

`LAND_PITCH_DEG` sets the minimum nose-up attitude for touchdown.

This ensures the plane lands on its main wheels (for tricycle gear) or in a 3-point attitude (for taildraggers), protecting the propeller from striking the runway.

Tuning & Behavior

- **Default Value:** 0.
- **Recommendation:** Set to **3-5 degrees** for most airframes.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





LAND_REPOSITION (Pilot Land Reposition Enable)

DEFAULT: 1	RANGE: 0 1

Pilot Land Reposition Enable (LAND_REPOSITION)

Description

`LAND_REPOSITION` allows you to steer the drone while it is landing itself.

If the drone is performing an automatic land (e.g. at the end of a Return-to-Launch), it might head toward a patch of long grass or a small puddle. If this parameter is **Enabled (1)**, you can use your transmitter sticks to "nudge" the drone left, right, forward, or backward without cancelling the landing sequence. The drone will continue its vertical descent while moving horizontally based on your inputs.

- **1: Enabled (Default).** Active pilot steering allowed during LAND.
- **0: Disabled.** The drone will land exactly where it is, ignoring pilot sticks.

Tuning & Behavior

- **Safety:** This is a vital safety feature. It allows you to avoid last-second obstacles on the ground that the GPS might not be precise enough to miss.

LAND_SLOPE_RCALC (Land Slope Recalculate Altitude)

DEFAULT: 2.0	RANGE: 0 5	UNITS: m

Land Slope Recalculate Altitude (LAND_SLOPE_RCALC)

Description

`LAND_SLOPE_RCALC` is a correction threshold.

If the plane is too high or too low on the approach, the autopilot will try to fly back to the ideal line. However, once the plane gets very close to the ground (below this altitude), it stops trying to correct the slope and commits to the current path to avoid erratic pitch changes just before touchdown.

Tuning & Behavior

- **Default Value:** 2.0 m.
- **Recommendation:** Keep low. Recalculating the slope at 20m is fine; recalculating at 0.5m might cause a crash.

LAND_SPEED (Land Descent Speed)

DEFAULT: 50	RANGE: 30 200	UNITS: cm/s

Land Descent Speed (LAND_SPEED)

Description

`LAND_SPEED` is the "Final Contact Speed" for the drone. It is the vertical velocity used from `LAND_ALT_LOW` until the drone detects it has touched the ground.

The Engineer's View

In the ArduCopter control loop, once the `LAND_ALT_LOW` threshold is crossed, the position controller switches to a target velocity equal to this parameter. The "Land Detector" state machine then waits for:

1. **Vertical Velocity** to be near zero.
2. **Throttle** to be at minimum.
   Once these are met for a specific duration (`LAND_DISARMDELAY`), the drone disarms.

Tuning & Behavior

- **Default Value:** 50 cm/s (0.5 m/s).
- **The "Bounce" Issue:** If your drone bounces back into the air upon touching down, your `LAND_SPEED` is likely too high, or your landing gear is too springy. Reduce to **30-40 cm/s**.
- **Surface Consistency:** If landing on tall grass or soft sand, you may need a slightly higher speed to ensure the land detector triggers reliably.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





LAND_SPEED_HIGH (Land Descent Speed High)

DEFAULT: 0	RANGE: 0 500	UNITS: cm/s

Land Descent Speed High (LAND_SPEED_HIGH)

Description

`LAND_SPEED_HIGH` allows the drone to descend faster from high altitudes during an RTL or LAND mission, then slow down once it gets closer to the ground.

- **0 (Default):** The drone uses the standard `WPNAV_SPEED_DN` for the entire descent until `LAND_ALT_LOW`.
- **Value (e.g., 200):** The drone will descend at 2 m/s until it hits the low altitude threshold.

Tuning & Behavior

- **Default Value:** 0
- **Recommendation:** If you frequently fly at high altitudes (e.g., 100m+), set this to **200 or 300** to save battery during RTL.

LAND_THEN_NEUTRL (Land Then Neutralize Servos)

DEFAULT: 0	RANGE: 0 1

Land Then Neutralize Servos (LAND_THEN_NEUTRL)

Description

`LAND_THEN_NEUTRL` protects your servos after the flight.

Once the plane has landed and disarmed, the autopilot usually holds the last known control positions (e.g., elevator up). This parameter forces all surfaces to their center (neutral) position.

Tuning & Behavior

- **0:** Disabled (Hold last position).
- **1:** Enabled (Center all surfaces).
- **Recommendation:** Enable this to prevent servo buzz on the ground.

LAND_THR_SLEW (Landing Throttle Slew Rate)

DEFAULT: 0	RANGE: 0 100	UNITS: %/s

Landing Throttle Slew Rate (LAND_THR_SLEW)

Description

`LAND_THR_SLEW` prevents the throttle from "pumping" during the approach.

If TECS is fighting turbulence, it might command rapid throttle changes. This parameter forces the throttle to change smoothly, which is better for scale aircraft and passenger comfort.

Tuning & Behavior

- **Default Value:** 0 (Disabled/Use `THR_SLEW`).
- **Recommendation:** Set to **20-30%** if you hear the motor surging on final approach.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





LAND_TYPE (Default Landing Type)

DEFAULT: 0	RANGE: 0 1

Default Landing Type (LAND_TYPE)

Description

`LAND_TYPE` sets the default behavior for the `LAND` flight mode.

- **0:** Standard fixed-wing landing (Glide slope → Flare).
- **1:** Deep Stall landing (Pop up → Vertical descent).

Tuning & Behavior

- **Default Value:** 0.
- **Deep Stall:** Requires `LAND_DS_*` parameters to be configured first.

LAND_WIND_COMP (Land Wind Compensation)

DEFAULT: 0	RANGE: 0 100	UNITS: %

Land Wind Compensation (LAND_WIND_COMP)

Description

`LAND_WIND_COMP` adds a safety margin for windy days.

If you are landing into a 20 m/s headwind, the ground speed will be very low. If the wind suddenly stops (wind shear), the plane might stall. This parameter tells the autopilot to fly faster than normal when it detects strong wind.

Tuning & Behavior

- **Default Value:** 0 (Disabled).
- **50%:** If wind is 10 m/s, add 5 m/s to the landing speed.
- **Recommendation:** Set to **30-50%** for reliable all-weather operation.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **LEAD** parameter group (specifically `LEAD_RLL` and `LEAD_PTCH`) configures the legacy **Lead Filter**.

Key Concepts

1. GPS Lag Compensation

Before the invention of high-frequency GPS (10Hz+) and modern EKF (Kalman Filters), ArduPilot used a "Lead Filter" to project the vehicle's position forward in time. This compensated for the inherent 1-second lag found in old NMEA GPS receivers.

Developer Notes

- **Status: Legacy.** These parameters have essentially no effect when using modern EKF2 or EKF3 estimators, as the EKF handles sensor delay compensation internally with much higher precision.

Mount Pitch stabilization lead time ()

LEAD_PTCH: Mount Pitch stabilization lead time

Description

Servo mount pitch angle output leads the vehicle angle by this amount of time based on current pitch rate. Increase until the servo is responsive but does not overshoot.

Values

- **Range:** 0.0 to 0.2
- **Units:** s
- **Increment:** 0.005
- **Default:** 0.0f

Description

This parameter is a feed-forward term for the camera mount's stabilization. It helps compensate for servo lag.

- **How it works:** It uses the vehicle's current pitch *rate* (degrees per second) to predict where the pitch angle will be in `LEAD_PTCH` seconds and drives the servo to that future position immediately.
- **Tuning:**
  - **Too Low:** The camera mount reacts too slowly to vehicle pitch changes (video looks shaky/laggy).
  - **Too High:** The camera mount over-corrects or jitters (overshoot).

Source Code

ardupilot/libraries/AP_Mount/AP_Mount_Params.cpp

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





LEAD_RLL: Mount Roll stabilization lead time

Description

Servo mount roll angle output leads the vehicle angle by this amount of time based on current roll rate. Increase until the servo is responsive but does not overshoot.

Values

- **Range:** 0.0 to 0.2
- **Units:** s
- **Increment:** 0.005
- **Default:** 0.0f

Description

This parameter is a feed-forward term for the camera mount's roll stabilization. It helps compensate for servo lag during rolling maneuvers.

- **How it works:** It uses the vehicle's current roll *rate* (degrees per second) to predict where the roll angle will be in `LEAD_RLL` seconds and drives the servo to that future position immediately.
- **Tuning:**
  - **Too Low:** The camera mount reacts too slowly to vehicle roll changes.
  - **Too High:** The camera mount over-corrects or jitters.

Source Code

ardupilot/libraries/AP_Mount/AP_Mount_Params.cpp

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **LEAK** parameter group configures the **Leak Detection** system for **ArduSub** (ROVs).

Water ingress is the primary failure mode for underwater vehicles. This system monitors up to 3 physical leak sensors and triggers a failsafe alert (and optionally an autonomous surfacing maneuver) if water is detected.

Key Concepts

1. Sensor Logic

- `LEAKn_TYPE`: 1=Digital (Pin high/low).
- `LEAKn_LOGIC`: Defines if "Wet" is High (3.3V) or Low (0V).
- `LEAKn_PIN`: The GPIO pin on the flight controller connected to the sensor.

Integration Guide

1. **Hardware:** Place leak probes at the lowest point of the electronics enclosure.
2. **Config:** Set the pin and logic based on your sensor (e.g., BlueRobotics SOS probe).
3. **Failsafe:** Configure the GCS to alert the pilot immediately on a leak trigger.

Developer Notes

- **Library:** `libraries/AP_LeakDetector`.
- **Context:** Only relevant for **ArduSub**.

LEAK1_LOGIC (Leak Sensor Logic (Sensor 1))

DEFAULT: 0	RANGE: 0 1

Leak Sensor Logic (LEAK1_LOGIC)

Description

`LEAK1_LOGIC` calibrates the autopilot for your specific leak sensor hardware.

- **0: Low when dry (Default).** The sensor reports a logical 0 when no water is present.
- **1: High when dry.** The sensor reports a logical 1 when dry.

Tuning & Behavior

- **Calibration:** If your GCS reports a "LEAK" error even when the ROV is completely dry, try toggling this parameter.
- **Safety:** Always test your leak sensor manually (e.g. by bridging the contacts with a damp cloth) before diving to ensure the autopilot correctly detects the alarm state.

LEAK1_PIN (Leak Detector Pin (Sensor 1))

DEFAULT: -1	RANGE: -1 103

Leak Detector Pin (LEAK1_PIN)

Description

`LEAK1_PIN` sets the physical pin on the flight controller where the leak sensor's signal is connected.

- **-1: Not used.**
- **0-9: Analog Pin Number.**
- **50-55: AUX Out Pins.**

Tuning & Behavior

- **Default:** -1.
- **Action:** Ensure the pin matches your hardware wiring.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





LEAK1_TYPE (Leak Detector Type (Sensor 1))

DEFAULT: 0	RANGE: 0 1

Leak Detector Type (LEAK1_TYPE)

Description

`LEAK1_TYPE` enables the first leak detection sensor. This is a critical safety feature for ROVs (ArduSub) and boats to detect water ingress before it causes a short circuit or catastrophic failure.

- **0: Disabled.**
- **1: Analog.** For sensors that output a varying voltage (common for simple resistive probes).
- **2: Digital.** For sensors that output a simple HIGH/LOW signal.

Tuning & Behavior

- **Reboot Required:** Yes.
- **Integration:** Once enabled, you must also set the physical pin number using LEAK1_PIN.
- **Safety:** When a leak is detected, ArduPilot will trigger a GCS warning and can be configured to perform a failsafe action (like surfacing or stopping).

LEAK2_LOGIC (Default reading of leak detector when dry)

DEFAULT: 0	RANGE: null

Default reading of leak detector when dry

Note: This parameter functions identically to LEAK1_LOGIC.

LEAK2_PIN (Pin that leak detector is connected to)

DEFAULT: -1	RANGE: null

Pin that leak detector is connected to

Note: This parameter functions identically to LEAK1_PIN.

LEAK2_TYPE (Leak detector pin type (analog/digital))

DEFAULT: DISABLED	RANGE: null

Leak detector pin type (analog/digital)

Note: This parameter functions identically to LEAK1_TYPE.

LEAK3_LOGIC (Default reading of leak detector when dry)

DEFAULT: 0	RANGE: null

Default reading of leak detector when dry

Note: This parameter functions identically to LEAK1_LOGIC.

LEAK3_PIN (Pin that leak detector is connected to)

DEFAULT: -1	RANGE: null

Pin that leak detector is connected to

Note: This parameter functions identically to LEAK1_PIN.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





LEAK3_TYPE (Leak detector pin type (analog/digital))

DEFAULT: DISABLED	RANGE: null

Leak detector pin type (analog/digital)

Note: This parameter functions identically to LEAK1_TYPE.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **LED** parameter group (specifically `LED_BRIGHTNESS`) configures the on-board **Status LEDs**. This is primarily used in **AP_Periph** firmware for DroneCAN nodes (like GPS modules).

Key Concepts

1. Brightness Control

Allows the user to adjust the intensity of the RGB status LEDs to match ambient lighting conditions or to reduce power consumption on battery-limited devices.

- `LED_BRIGHTNESS`:
  - **100:** Full brightness.
  - **10-50:** Dimmed for night operations or stealth.
  - **0:** Disabled.

Developer Notes

- **Library:** `Tools/AP_Periph` and `libraries/AP_Notify`.
- **Note:** Main autopilot LED brightness is typically handled in the `NTF_` group.

LED_BRIGHTNESS (LED Brightness)

DEFAULT: 100	RANGE: 0 100	UNITS: %

LED Brightness (LED_BRIGHTNESS)

Description

This parameter controls the brightness intensity of the RGB status LED on the DroneCAN peripheral (e.g., GPS/Compass module). It allows you to dim the LED for stealth or power saving, or turn it off completely.

The Mathematics

$$\text{Duty Cycle} \propto \text{LED_BRIGHTNESS}\%$$

The Engineer's View

Defined in `Tools/AP_Periph/Parameters.cpp`. Used by the `ToshibaLED` or `NeoPixel` driver on the peripheral node.

Tuning & Behavior

- **Default Value:** 100% (High)
- **0:** LED Off.
- **10-50:** Dimmed.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY

LEFT BUTTON CONFIGURATION (LEFT)

Overview

The **LEFT** parameter group (specifically `LEFT_LONG`) configures the behavior of the **Left Shoulder Button** on specialized **Toy Mode** controllers (gamepad-style).

Key Concepts

1. Toy Mode Mapping

In Toy Mode, the controller buttons are mapped to specific ArduPilot flight modes or actions.

- `LEFT_LONG`: Defines what happens when the left button is held for more than 1 second.
  - **Common Actions:** Change flight mode, trigger photo, or perform a flip.

Developer Notes

- **Library:** `ArduCopter/toy_mode.cpp`.
- **Context:** Only relevant if using specific SkyViper or similar gamepad-driven hardware.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Tmode left long action ()

LEFT_LONG: Tmode left long action

Description

This is the action taken when the left (Mode) button is long-pressed in Toy Mode.

Values

Value	Meaning
0	None
1	Take Photo
2	Toggle Video
3	Mode Acro
4	Mode AltHold
5	Mode Auto
6	Mode Loiter
7	Mode RTL
8	Mode Circle
9	Mode Land
10	Mode Drift
11	Mode Sport
12	Mode AutoTune
13	Mode PosHold
14	Mode Brake
15	Mode Throw
16	Flip
17	Mode Stabilize
18	Disarm
19	Toggle Mode
20	Arm-Land-RTL
21	Toggle Simple Mode
22	Toggle Super Simple Mode
23	Motor Load Test
24	Mode FlowHold

- **Default:** 0 (None)

Description

This parameter assigns a specific function to the "Long Press" action of the Left Button (typically the "Mode" button) on a supported Toy Mode controller (e.g., SkyViper or similar gamepad-style controllers running ArduPilot).

Source Code

ardupilot/ArduCopter/toy_mode.cpp

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **LEVEL** parameter group (specifically `LEVEL_ROLL_LIMIT`) configures the maximum bank angle during critical flight phases for **ArduPlane**.

Key Concepts

1. Ground Clearance

During Auto-Takeoff and the Flare portion of a landing, the plane is very close to the runway. Banking too steeply during these moments can cause a wingtip to strike the ground, resulting in a cartwheel crash.

- `LEVEL_ROLL_LIMIT`: Limits the bank angle (degrees) when altitude is very low (< 5 meters).
  - **Recommendation:** Keep small (e.g., 5-10 degrees).

Developer Notes

- **Library:** `ArduPlane/mode_landing.cpp`.
- **Context:** Only relevant for fixed-wing aircraft.

LEVEL_ROLL_LIMIT (Level flight roll limit)

DEFAULT: 5	RANGE: 0 45	UNITS: deg

Level flight roll limit (LEVEL_ROLL_LIMIT)

Description

This safety parameter defines the maximum allowable roll (bank) angle during critical near-ground flight phases, specifically Auto-Takeoff and Landing (during the flare). By limiting the bank angle, it prevents the wingtips from striking the ground/runway, which could cause a crash.

The Mathematics

During takeoff (below a certain altitude) and landing flare:

$$|\phi_demand| \leq \text{LEVEL_ROLL_LIMIT}$$

The Engineer's View

Defined in `ArduPlane/Parameters.cpp`.

- **Takeoff:** Used when altitude is < 5m or during the initial climb.
- **Landing:** Used during the `FLARE` stage of an auto-landing.
- **Zero Logic:** If set to 0, it completely disables the heading hold controller during these phases, which effectively means the roll is not actively controlled to level (dangerous).

Tuning & Behavior

- **Default Value:** 5 deg
- **Range:** 0 - 45
- **Recommended:** Keep small (5-10 deg) to ensure wing clearance.
- **Warning:** Do not set to 0 unless you understand the implications of disabling heading hold during takeoff/landing.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





LANDING GEAR CONFIGURATION (LGR)

Overview

The **LGR** parameter group configures the **Retractable Landing Gear** management system.

ArduPilot can autonomously manage the deployment and retraction of landing gear to improve aerodynamics in flight and ensure safety during landing.

Key Concepts

1. Automatic Deployment (`LGR_DEPLOY_ALT`)

The landing gear will automatically deploy (descend) when the vehicle descends below this altitude (meters) above ground.

2. Automatic Retraction (`LGR_RETRACT_ALT`)

The landing gear will automatically retract (lift) when the vehicle climbs above this altitude after takeoff.

3. Weight on Wheels (WOW)

Some advanced airframes use a sensor (`LGR_WOW_PIN`) to detect when the aircraft is actually touching the ground, preventing the gear from retracting while the vehicle's weight is on it.

Parameter Breakdown

- `LGR_ENABLE`: Master switch.
- `LGR_STARTUP`: Behavior at boot (Deploy, Retract, or No Change).
- `LGR_OPTIONS`: Bitmask for behavior (e.g., override deployment during emergency landing).

Integration Guide

1. **Pin:** Map a servo output to `Landing Gear` (Function 29).
2. **Enable:** Set `LGR_ENABLE = 1`.
3. **Config:** Set the deploy/retract altitudes based on your operational needs.
4. **RC:** Map an RC channel to `Landing Gear` (RC Option 29) to allow manual override.

Developer Notes

- **Library:** `libraries/AP_LandingGear`.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





LGR_DEPLOY_PIN (Chassis deployment feedback pin)

DEFAULT: -1

LGR_DEPLOY_PIN: Chassis deployment feedback pin

Description

Pin number to use for detection of gear deployment. If set to -1, feedback is disabled.

Values

- **Range:** -1 to 55 (GPIO Pin Number)
- **Default:** -1 (Disabled)

Description

This parameter assigns a GPIO (General Purpose Input/Output) pin to read a limit switch or sensor that detects when the landing gear is fully deployed (down and locked).

- **Usage:** Connect a microswitch or hall sensor to this pin.
- **Function:** When the pin enters the state defined by `LGR_DEPLOY_POL`, the autopilot confirms the gear is down. This can be used to prevent landing if the gear fails to deploy.

Source Code

ardupilot/libraries/AP_LandingGear/AP_LandingGear.cpp

LGR_DEPLOY_POL (Landing Gear Deployment Polarity)

DEFAULT: 0	RANGE: 0 1

Landing Gear Deployment Polarity (LGR_DEPLOY_POL)

Description

`LGR_DEPLOY_POL` defines the logic of the landing gear deployment signal.

- **0: Normal.**
- **1: Inverted.**

Tuning & Behavior

- **Usage:** If your landing gear retracts when it should deploy (and vice versa), toggle this parameter. This is often necessary when using different brands of electronic retracts.

LGR_ENABLE (Landing Gear Enable)

DEFAULT: 0	RANGE: 0 1

Landing Gear Enable (LGR_ENABLE)

Description

`LGR_ENABLE` activates the logic for controlling retractable landing gear.

When enabled, you can control the gear manually via an auxiliary switch or allow the autopilot to handle it automatically during takeoff and landing.

- **1: Enabled.** Enables landing gear logic and makes other LGR parameters visible.
- **0: Disabled.** Landing gear system is inactive.

Tuning & Behavior

- **Default:** 0.
- **Setup:** You must also assign a motor output to the landing gear function by setting `SERVOx_FUNCTION = 29`.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





LGR_OPTIONS (Landing Gear Options)

DEFAULT: 3	RANGE: 0 3	UNITS: Bitmask

Landing Gear Options (LGR_OPTIONS)

Description

LGR_OPTIONS defines when the autopilot should move the landing gear without pilot intervention.

- **Bit 0 (1): Retract after Takeoff.** Automatically pulls the gear up once a safe climb is established.
- **Bit 1 (2): Deploy during Land.** Automatically lowers the gear when the aircraft begins its final approach or reaches a specific altitude.

Tuning & Behavior

- **Default:** 3 (Both Auto-Retract and Auto-Deploy enabled).
- **Altitude Trigger:** Use LGR_DEPLOY_ALT and LGR_RETRACT_ALT to fine-tune exactly when these actions occur.

LGR_STARTUP (Landing Gear Startup Position)

DEFAULT: 0	RANGE: 0 2

Landing Gear Startup Position (LGR_STARTUP)

Description

LGR_STARTUP determines the physical position of the landing gear immediately after the flight controller boots up.

This is a safety-critical setting. If the gear defaults to "Retracted" while the drone is sitting on the ground, it could cause damage to the gimbal or airframe.

- **0: WaitForPilotInput (Default).** The gear stays exactly where it was until the pilot flips the gear switch on the transmitter.
- **1: Retract.** The gear will immediately attempt to pull up upon power-on. (Not recommended for ground starts!)
- **2: Deploy.** The gear will immediately attempt to extend upon power-on.

Tuning & Behavior

- **Recommendation:** Use **0 (Wait for Input)** or **2 (Deploy)** for most multirotors.

LGR_WOW_PIN (Weight on Wheels (WOW) Feedback Pin)

DEFAULT: -1	RANGE: -1 103

Weight on Wheels (WOW) Feedback Pin (LGR_WOW_PIN)

Description

LGR_WOW_PIN tells the autopilot if the drone is physically sitting on its landing gear.

This requires a physical switch or sensor (Weight-on-Wheels / WOW) mounted on the landing gear strut. When the gear is compressed by the weight of the aircraft, the switch triggers. This provides a high-certainty "On Ground" signal that the autopilot can use for safety, such as preventing the motors from spooling up or disabling high-gain stabilization while taxiing.

Tuning & Behavior

- **-1:** Disabled (Default).
- **0-103:** Physical GPIO pin number.
- **Requirement:** Requires LGR_WOW_POL to be set correctly for your specific switch type (Normally Open vs Normally Closed).

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY

LGR_WOW_POL (Weight-On-Wheels Polarity)

DEFAULT: 0	RANGE: 0 1

Weight-On-Wheels Polarity (LGR_WOW_POL)

Description

`LGR_WOW_POL` calibrates the autopilot for your specific Weight-On-Wheels (WOW) sensor hardware.

A WOW sensor (like a microswitch on the landing strut) tells the autopilot if the aircraft is physically supported by the ground.

- **0: Low.** Signal is 0V when there is weight on the wheels.
- **1: High.** Signal is 3.3V/5V when there is weight on the wheels.

Landing gear deployment altitude ()

LGR_DEPLOY_ALT: Landing gear deployment altitude

Description

Altitude where the landing gear will be deployed. This should be lower than the `LGR_RETRACT_ALT`. If zero, altitude is not used for deploying landing gear. Only applies when the vehicle is armed.

Values

- **Range:** 0 to 1000
- **Units:** m
- **Increment:** 1
- **Default:** 0

Description

This parameter automates landing gear deployment based on altitude.

- **Function:** When the vehicle descends below this altitude (in meters above home), the landing gear will automatically deploy.
- **Safety:** Useful for ensuring gear is down for landing even if the pilot forgets.
- **Constraint:** Must be set lower than `LGR_RETRACT_ALT` (Retract Altitude) to prevent rapid toggling (hysteresis).

Source Code

ardupilot/libraries/AP_LandingGear/AP_LandingGear.cpp

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





LGR_RETRACT_ALT: Landing gear retract altitude

Description

Altitude where the landing gear will be retracted. This should be higher than the `LGR_DEPLOY_ALT`. If zero, altitude is not used for retracting landing gear. Only applies when the vehicle is armed.

Values

- **Range:** 0 to 1000
- **Units:** m
- **Increment:** 1
- **Default:** 0

Description

This parameter automates landing gear retraction based on altitude.

- **Function:** When the vehicle climbs above this altitude (in meters above home), the landing gear will automatically retract.
- **Usage:** Set this to a safe height (e.g., 10-20 meters) to automatically clean up the airframe after takeoff.
- **Constraint:** Must be higher than `LGR_DEPLOY_ALT` to prevent rapid toggling.

Source Code

ardupilot/libraries/AP_LandingGear/AP_LandingGear.cpp

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **LOAD** parameter group is specific to the **Toy Mode** motor verification routines. It is used primarily in factory environments to perform automated burn-in or stress testing of the motors and ESCs.

Key Concepts

1. Test Patterns

The autopilot can execute pre-defined throttle sequences to exercise the motors.

- `LOAD_TYPE`: Selects between constant thrust, sequential motor blips, or synchronized pulses.

Parameter Breakdown

- `LOAD_TYPE`: Pattern selection.
- `LOAD_MUL`: Multiplier to scale the intensity of the test.

Developer Notes

- **Library:** `ArduCopter/toy_mode.cpp`.
- **Caution:** These parameters are not intended for use in normal flight and can cause the drone to behave unexpectedly if activated.

LOAD_FILT (Load test filter)

DEFAULT: 1	RANGE: 0 100

LOAD_FILT: Load test filter

Description

This filters the load test output. A value of 1 means no filter. 2 means values are repeated once. 3 means values are repeated 3 times, etc.

Values

- **Range:** 0 to 100
- **Default:** 1

Description

This parameter is part of the **Toy Mode** motor load testing suite (used primarily for factory endurance testing). It acts as a simple low-pass filter by holding each output value for multiple cycles.

- **1:** Updates motor output every cycle (No filtering).
- **>1:** Holds the motor output constant for this many cycles before updating to the next value in the test sequence. This smooths out rapid changes in the load test profile.

Source Code

ardupilot/ArduCopter/toy_mode.cpp

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





LOAD_MUL (Load test multiplier)

DEFAULT: 1.0	RANGE: 0 1

LOAD_MUL: Load test multiplier

Description

This scales the load test output, as a value between 0 and 1.

Values

- **Range:** 0 to 1
- **Increment:** 0.01
- **Default:** 1.0

Description

This parameter controls the overall intensity of the motor load test.

- **Usage:** It multiplies the PWM values defined in the test pattern (`LOAD_TYPE`) by this factor.
- **Example:** Setting this to 0.5 reduces the power output of the entire test sequence by 50%, useful for running initial tests without stressing the hardware to its maximum.

Source Code

ardupilot/ArduCopter/toy_mode.cpp

LOAD_TYPE (Load test type)

DEFAULT: 1

LOAD_TYPE: Load test type

Description

This sets the type of load test pattern to run.

Values

Value	Meaning
0	Constant Thrust
1	Log Replay 1 (Sequence)
2	Log Replay 2 (Synchronized)

- **Default:** 1 (LogReplay1)

Description

Selects the specific motor output pattern used during the factory load test.

- **Constant Thrust (0):** Runs all motors at a fixed, low throttle (500 PWM above min, scaled by `LOAD_MUL`).
- **Log Replay 1 (1):** Replays a pre-defined sequence of throttle values, cycling through motors.
- **Log Replay 2 (2):** Similar to Log Replay 1 but applies the pattern to all motors simultaneously.

Source Code

ardupilot/ArduCopter/toy_mode.cpp

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





DATAFLASH LOGGING CONFIGURATION (LOG)

Overview

The **LOG** parameter group configures the **DataFlash Logging** system. This is the "Black Box" of the autopilot, recording high-frequency internal data (IMU, PIDs, GPS, Battery) for post-flight analysis.

Logs are essential for tuning, diagnosing crashes, and verifying mission performance.

Key Concepts

1. Storage Backend (`LOG_BACKEND_TYPE`)

Defines where the data is sent.

- **File (1):** Records to the SD Card (Standard).
- **MAVLink (2):** Streams logs to the Ground Station in real-time.
- **Block (4):** Records to on-board SPI Flash memory.

2. Message Filter (`LOG_BITMASK`)

There are hundreds of data messages available. To save SD card space and CPU, you can choose which ones to log.

- **Standard:** Logs basic navigation and attitude data.
- **Fast Attitude:** Logs at the PID loop rate (essential for tuning).
- **Raw IMU:** Logs every single gyro sample (very heavy, used for FFT analysis).

3. Disarmed Logging (`LOG_DISARMED`)

If enabled, the autopilot logs data even when the motors are not spinning. This is great for bench testing but consumes space quickly.

Parameter Breakdown

- `LOG_BITMASK`: Bitmask of active log categories.
- `LOG_BACKEND_TYPE`: Output selection.
- `LOG_FILE_BUFSIZE`: RAM buffer size.
- `LOG_MAX_FILES`: Number of old logs to keep before overwriting.

Integration Guide

Diagnosing a Problem

1. **Enable Raw Logging:** Set `LOG_BITMASK` to include `RawIMU` (check Wiki for exact bit for your version).
2. **Replicate:** Fly and perform the maneuver that caused the issue.
3. **Download:** Use the "DataFlash Logs" tab in Mission Planner to download the `.BIN` file.
4. **Analyze:** Use "Review Log" to graph the data.

Developer Notes

- **Library:** `libraries/AP_Logger`.
- **Speed:** On modern boards, logging can consume a significant portion of CPU time. Use `LOG_BITMASK` judiciously.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





LOG_BACKEND_TYPE (AP_Logger Backend Storage type)

DEFAULT: 1	RANGE: 0 7

AP_Logger Backend Storage type (LOG_BACKEND_TYPE)

Description

`LOG_BACKEND_TYPE` determines where your flight data is physically stored.

Most flight controllers have an SD card, but some specialized hardware might use internal flash chips (Block) or send the log data live over a telemetry radio (MAVLink).

- **Bit 0 (1): File.** Save logs to the SD card (Standard).
- **Bit 1 (2): MAVLink.** Send log data live to your Ground Station. (Requires high bandwidth link like WiFi/LTE).
- **Bit 2 (4): Block.** Save logs to the internal onboard dataflash chip (if available).

Tuning & Behavior

- **Default Value:** Usually 1 (File).
- **Safety:** You can enable multiple backends simultaneously (e.g., set to 5 for File + Block).
- **Reboot Required:** Yes.

LOG_BITMASK (Log Bitmask)

DEFAULT: 176126	RANGE: 0 4294967295

Log Bitmask (LOG_BITMASK)

Description

`LOG_BITMASK` is the "Recording Selector" for your drone's black box.

ArduPilot can log a massive amount of data, but recording everything all the time would fill up your SD card too fast and might overload the CPU. This bitmask allows you to pick exactly what you want to record.

Key Bits

- **Bit 0 (1): ATTITUDE_FAST.** Logs attitude at the fastest possible rate. (Heavy CPU).
- **Bit 1 (2): ATTITUDE_MED.** Logs attitude at a medium rate. (Standard).
- **Bit 2 (4): GPS.** Logs GPS position and satellite health.
- **Bit 3 (8): PM.** Processor Monitoring (CPU load and loop times).
- **Bit 7 (128): IMU.** Raw accelerometer and gyro data.
- **Bit 9 (512): CURRENT.** Battery voltage and amperage.
- **Bit 12 (4096): PID.** Motor controller P-I-D gains and errors. (Essential for tuning).
- **Bit 19 (524288): IMU_RAW.** Extreme high-speed raw IMU data for vibration analysis.

Tuning & Behavior

- **Default Value:** 176126 (Standard set including GPS, ATT_MED, CTUN, NTUN, etc.).
- **Recommendation:**
  - **Tuning PIDs:** Ensure Bit 12 (PID) is enabled.
  - **Troubleshooting Vibrations:** Enable Bit 19 (IMU_RAW) briefly for one flight, then disable it (it generates huge files).

LOG_BLK_RATEMAX (Maximum logging rate for block backend)

DEFAULT: 0	RANGE: 0 1000	UNITS: Hz

Maximum logging rate for block backend (LOG_BLK_RATEMAX)

Description

`LOG_BLK_RATEMAX` sets a speed limit for data being written to the onboard flash chip.

Internal flash chips often have limited capacity and write cycles. By setting a rate limit (e.g., 50Hz), you can ensure you capture enough data for tuning while preventing the chip from filling up too quickly during long flights.

- **0:** No limit.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





LOG_DARM_RATEMAX (Maximum logging rate when disarmed)

DEFAULT: 0	RANGE: 0 1000	UNITS: Hz

Maximum logging rate when disarmed (LOG_DARM_RATEMAX)

Description

`LOG_DARM_RATEMAX` allows you to save SD card space by recording at a lower fidelity while sitting on the ground.

While you are flying, you need high-speed data (400Hz+) to analyze vibrations and PIDs. On the ground, you usually only need a slow update (e.g. 5Hz) to monitor GPS health or battery status.

- **0:** Use the normal backend rate limits.
- **10:** Force 10Hz logging while disarmed.

LOG_DISARMED (Enable logging while disarmed)

DEFAULT: 0	RANGE: 0 3

Enable logging while disarmed (LOG_DISARMED)

Description

`LOG_DISARMED` determines if the "Record" button stays active when the drone is disarmed.

Normally, ArduPilot only logs while you are flying to save SD card space. However, logging on the ground is vital for debugging GPS problems, sensor initialization, or pre-arm failures.

- **0: Disabled (Default).** Only log when armed.
- **1: Enabled.** Log constantly as long as the board has power.
- **2: Disabled on USB.** Log while on battery, but stop when plugged into a computer (to speed up parameter changes).
- **3: Discard if never armed.** Log while disarmed, but delete the file on reboot if the drone didn't actually take off.

Tuning & Behavior

- **Recommendation:** Set to **1** if you are troubleshooting a "Pre-Arm: Needs 3D Fix" or other startup issue. Set to **0** for general use to keep your SD card clean.

LOG_FILE_BUFSIZE (Logging File buffer size max)

DEFAULT: 200	RANGE: 4 200	UNITS: kB

Logging File buffer size max (LOG_FILE_BUFSIZE)

Description

`LOG_FILE_BUFSIZE` helps prevent "Gaps" in your logs.

SD cards are sometimes slow to write. This parameter creates a "Waiting Area" (buffer) in the flight controller's RAM. Data is collected here and then written to the card in one big burst. A larger buffer can handle slower SD cards without losing data.

Tuning & Behavior

- **Default Value:** 200 KB (on high-end boards like Cube Orange).
- **Recommendation:** If you see "Log: GAP" messages in your log files, increase this value to its maximum (200).

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





LOG_FILE_DSRMROT (Stop logging to current file on disarm)

DEFAULT: 0	RANGE: 0 1

Stop logging to current file on disarm (LOG_FILE_DSRMROT)

Description

`LOG_FILE_DSRMROT` helps organize your flight logs.

- **0 (Default):** One log file per power cycle. If you land, disarm, and re-arm without unplugging the battery, everything is saved in one big file.
- **1 (Rotate):** Every flight (each Arm/Disarm cycle) gets its own individual `.BIN` file. This makes it much easier to find the specific flight you are looking for in a long day of testing.

Tuning & Behavior

- **Recommendation:** Set to **1** for easier post-flight analysis.
- **Note:** If LOG_DISARMED is enabled, setting this to 1 will immediately start a new log the moment you land.

LOG_FILE_MB_FREE (Min free MB on SD card)

DEFAULT: 500	RANGE: 10 1000	UNITS: MB

Min free MB on SD card (LOG_FILE_MB_FREE)

Description

`LOG_FILE_MB_FREE` is an "Auto-Cleanup" setting.

To prevent the SD card from becoming completely full (which causes write errors and potentially crashes), ArduPilot automatically deletes your oldest log files when the free space drops below this limit.

- **500 (Default):** Maintain at least 500MB of free space.

Tuning & Behavior

- **Recommendation:** Set this to be larger than your largest typical flight log (e.g. if you fly 1-hour missions that generate 200MB logs, 500MB is a safe limit).

LOG_FILE_RATEMAX (Maximum logging rate for file backend)

DEFAULT: 0	RANGE: 0 1000	UNITS: Hz

Maximum logging rate for file backend (LOG_FILE_RATEMAX)

Description

`LOG_FILE_RATEMAX` prevents your SD card from being overwhelmed by too much data.

High-speed logs (like raw IMU or fast attitude) can generate massive amounts of traffic. If your SD card is slow, the drone might experience "I/O Wait" stutters. This parameter sets a limit on the logging frequency.

- **0 (Default):** No limit. Record everything as fast as the system demands.
- **100:** Limit streaming messages to 100 Hz.

Tuning & Behavior

- **Recommendation:** Leave at **0** unless you see "Log dropped" errors in your GCS or experience flight performance issues due to slow storage media.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





LOG_FILE_TIMEOUT (Timeout before giving up on file writes)

DEFAULT: 5	RANGE: 1 60	UNITS: s

Timeout before giving up on file writes (LOG_FILE_TIMEOUT)

Description

`LOG_FILE_TIMEOUT` detects failing SD cards.

If your SD card is slow or damaged, it might "hang" during a write operation. If it stays unresponsive for longer than this duration, ArduPilot declares the log as failed and stops recording to protect the main flight control CPU from waiting indefinitely.

Tuning & Behavior

- **Default Value:** 5 seconds.
- **Safety:** Do not set this too high. An unresponsive SD card can eventually cause "Internal Error 0x800" (Main Loop Slow) if the system spends too much time waiting for it.

LOG_MAV_BUFSIZE (MAVLink Logging Buffer Size)

DEFAULT: 8	RANGE: 1 100	UNITS: kB

MAVLink Logging Buffer Size (LOG_MAV_BUFSIZE)

Description

This parameter sets the size of the memory buffer (in kilobytes) dedicated to "MAVLink Logging." Unlike standard logging which writes to an SD card, MAVLink logging streams the log data over your telemetry link (WiFi or Radio) directly to the Ground Control Station (GCS).

This is extremely useful for vehicles without SD cards (like some small racing drones) or for real-time analysis of flight data while the aircraft is still in the air. The buffer ensures that brief interruptions in the radio link don't cause gaps in the telemetry log.

The Mathematics

The system allocates a static block of RAM based on this value.

$$\text{RAM_}MAV_Log = LOG_MAV_BUFSIZE \times 1024 \text{ bytes}$$

Because MAVLink logging is sent in small chunks, the internal code calculates how many "Blocks" can fit into this RAM:

$$\text{BlockCount} = \frac{\text{RAM_}MAV_Log}{\text{sizeof(dm_block)}}$$

(where dm_block is typically 92 bytes).

The Engineer's View

In `AP_Logger_MAVLink.cpp`, this parameter determines the capacity of the outgoing message queue.

- **Bandwidth Sensitivity:** MAVLink logging consumes significant telemetry bandwidth. Even with a large buffer, if your radio link is slow (e.g., a 915MHz SiK radio at 57600 baud), the buffer will fill up quickly and data will be dropped.
- **Recommendation:** Only use this feature on high-speed links (WiFi, Ethernet, or high-speed SIYI/Herelink telemetry). On standard slow radios, this feature should typically be disabled or set to a very low rate.

Tuning & Behavior

- **Default Value:** 8 kB.
- **Range:** 1 to 100 kB.
- **When to Increase:** If you are using real-time GCS logging over a fast link and notice "Data Missing" or "Gaps" in your live telemetry charts.
- **When to Decrease:** If you are tight on RAM and not using the MAVLink logging feature.
- **Reboot Required:** Yes. Memory allocation for the logging backend happens at startup.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





LOG_MAV_RATEMAX (Maximum logging rate for mavlink backend)

DEFAULT: 0	RANGE: 0 1000	UNITS: Hz

Maximum logging rate for mavlink backend (LOG_MAV_RATEMAX)

Description

`LOG_MAV_RATEMAX` prevents live logging from choking your telemetry link.

Live MAVLink logging is very bandwidth intensive. If you are using a 915MHz SiK radio, you should set this to a very low value (e.g. 1Hz) or disable the MAVLink backend entirely. If you are using a high-speed WiFi link, you can set it higher.

Tuning & Behavior

- **Default Value:** 0 (Unlimited).
- **Recommendation:** Usually left at **0** because the MAVLink backend is disabled by default in `LOG_BACKEND_TYPE`.

LOG_MAX_FILES (Maximum number of log files)

DEFAULT: 500	RANGE: 2 500

Maximum number of log files (LOG_MAX_FILES)

Description

`LOG_MAX_FILES` limits the number of files in the `/LOGS` directory.

- **Default (500):** ArduPilot keeps up to 500 flights.
- **Safety:** This prevents directory fragmentation and slow boot times caused by having thousands of tiny files on the SD card.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





LOG_RATE (External AHRS Logging Rate)

DEFAULT: 10	RANGE: 1 100	UNITS: Hz

External AHRS Logging Rate (LOG_RATE)

Description

This parameter controls the data recording frequency for "External AHRS" (Attitude and Heading Reference System) devices. When you use a specialized external sensor suite (like a VectorNav or a companion-computer based estimator) to provide attitude data to ArduPilot, this parameter determines how often that external data is saved to your SD card.

Setting an appropriate rate is important for post-flight analysis and digital twin simulations, ensuring you have enough resolution to see fast movements without bloating your log file size.

The Mathematics

The logging interval ($T_interval$) is derived directly from the frequency:

$$T_interval = \frac{1000}{LOG_RATE} \text{ milliseconds}$$

The system checks the time elapsed since the last log write ($t_now - t_last$) and only writes a new entry if the elapsed time is greater than or equal to $T_interval$.

The Engineer's View

In `AP_ExternalAHRS.cpp`, this maps to the `log_rate` member.

- **Update Logic:** Inside the `AP_ExternalAHRS::update()` function, the code enforces the timing constraint:
  `if (log_rate.get() > 0 && now_ms - last_log_ms ≥ uint32_t(1000U/log_rate.get())) { ... }`
- **Data Quality:** Note that this logging rate is independent of the *internal processing rate* of the EKF. Even if you log at 10Hz, the autopilot may still be using the external data at a much higher frequency (e.g., 50Hz or 100Hz) for stabilization.

Tuning & Behavior

- **Default Value:** 10 Hz.
- **Range:** 1 to 100 Hz.
- **General Use: 10Hz** is sufficient for general navigation and flight path review.
- **Acrobatics/Research:** Increase to **50Hz** if you are performing aggressive maneuvers and need to precisely compare the external AHRS performance against ArduPilot's internal EKF.
- **0:** Disables logging for the external AHRS device.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





LOG_RAW (Proximity Raw Logging Enable)

DEFAULT: 0	RANGE: 0 1	UNITS: Option

Proximity Raw Logging Enable (LOG_RAW)

Description

This parameter enables the recording of raw, unfiltered distance data from Proximity sensors (like LiDAR or Ultrasound 360-degree scanners) to the autopilot's data log.

Proximity sensors are the "Eyes" of the drone's obstacle avoidance system. Normally, ArduPilot only logs the processed "Object Locations." Enabling `LOG_RAW` allows you to see every individual distance measurement the sensor reports, which is essential for diagnosing why an avoidance system might be failing, detecting "phantom" obstacles caused by sunlight or dust, and verifying the physical performance of your sensors.

The Mathematics

This is a binary boolean gate (G_log):

$$\text{IF } LOG_RAW == 1 \rightarrow \text{Execute } Write_Proximity_Raw_Message()$$

When enabled, the system generates high-frequency `PRX` log packets containing distance and angle pairs for every detection.

The Engineer's View

In `AP_Proximity.cpp`, this parameter maps to `_raw_log_enable`.

- **Logging Backend:** When enabled, the `AP_Proximity::update()` loop triggers the message creation in the `Log_Write()` function.
- **SD Card Load:** Warning: Raw proximity data can be very high-volume (hundreds of points per second). Enabling this will significantly increase the size of your `.BIN` log files and the load on your SD card.
- **Data Integrity:** This logs the data *before* any of ArduPilot's internal filtering or "Persistence" checks are applied, providing a "Truth" view of the sensor output.

Tuning & Behavior

- **Default Value:** 0 (Disabled).
- **When to Enable:** Specifically during troubleshooting of Obstacle Avoidance or when integrating a new Proximity sensor model.
- **When to Disable:** Standard flight operations. Keep disabled to save log space and reduce CPU overhead.
- **Dependencies:** Requires `PRX_TYPE` to be non-zero and a healthy proximity sensor to be connected.

LOG_REPLAY (Enable logging of information needed for Replay)

DEFAULT: 0	RANGE: 0 1

Enable logging of information needed for Replay (LOG_REPLAY)

Description

`LOG_REPLAY` is the "Diagnostic Mode" for your EKF.

If you are experiencing "EKF Compass Variance" or "EKF Lane Switch" errors and don't know why, you need to enable this. It records the raw sensor data and internal Kalman filter variables with high precision, allowing developers to "Replay" your actual flight through a simulator to find the bug.

Tuning & Behavior

- **Default Value:** 0 (Disabled).
- **Warning:** Enabling this generates **much larger log files** and increases CPU load. Use it only for troubleshooting, not for standard flying.
- **Prerequisite:** Requires LOG_DISARMED to be set to 1 or 2 to capture the critical pre-flight calibration data.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





LOITER MODE CONFIGURATION (LOIT)

Overview

The **LOIT** parameter group configures the behavior of the **Loiter** flight mode. Loiter is the standard GPS-based position-hold mode where the vehicle maintains its location and altitude when the sticks are centered.

Unlike "AltHold," which only holds height, Loiter uses the GPS and EKF to resist wind drift and maintain a fixed point over the ground.

Key Concepts

1. Responsiveness (`LOIT_SPEED` / `LOIT_ACC_MAX`)

Defines how fast the drone moves when you push the sticks.

- `LOIT_SPEED`: Maximum horizontal speed (cm/s).
- `LOIT_ACC_MAX`: Maximum acceleration (cm/s/s). Lower values result in smoother, more cinematic movement.

2. Braking Aggression (`LOIT_BRK_...`)

Configures how the drone stops when the sticks are released.

- `LOIT_BRK_ACCEL`: The deceleration rate.
- `LOIT_BRK_DELAY`: The time (seconds) to wait before applying the brakes.

3. Turning (`LOIT_ANG_MAX`)

Limits the maximum lean angle used to maintain position. This is usually lower than the global `ANGLE_MAX` to prevent aggressive corrections.

Parameter Breakdown

- `LOIT_SPEED`: Top speed in cm/s (e.g., 1250 = 12.5 m/s).
- `LOIT_RADIUS`: The circle radius used for loitering (Plane only).

Integration Guide

- **Cinematography:** Reduce `LOIT_ACC_MAX` to 250 and `LOIT_BRK_JERK` to 250 for very smooth transitions.
- **Wind Performance:** If the drone drifts in high wind, ensure `LOIT_ANG_MAX` is high enough to lean into the wind.

Developer Notes

- **Library:** `ArduCopter/mode_loiter.cpp`.
- **EKF:** Loiter is entirely dependent on a healthy EKF position estimate. If GPS is lost, the vehicle will usually switch to AltHold.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





LOIT_ACC_MAX (Loiter Horizontal Maximum Acceleration)

DEFAULT: 500	RANGE: 50 1000	UNITS: cm/s/s

Loiter Horizontal Maximum Acceleration (LOIT_ACC_MAX)

Description

`LOIT_ACC_MAX` is the "throttle response" setting for your drone's horizontal movement in Loiter mode. It determines how fast the drone can transition from a hover to full speed (`LOIT_SPEED`) when you slam the stick forward.

- **Low Value (e.g. 100):** The drone accelerates very slowly and "lazily." It feels like a massive ship. Ideal for smooth cinematic pans.
- **High Value (e.g. 800):** The drone accelerates instantly and aggressively. It feels like a high-performance sports car.
- **Default (500):** 5.0 m/s² (roughly 0.5G). A standard setting that feels snappy but controllable for most pilots.

The Mathematics

This parameter provides the acceleration limit (a_max) for the pilot-driven path shaper:

$$ \vec{V}*{target}(t) = |vec{V}*{target}(t-dt) + \text{constrain}(\vec{a}*{calc}, -a*{max}, a_{max}) \cdot dt $$

Where:

- a_max is `LOIT_ACC_MAX`.

Physical Consequence: If `LOIT_ACC_MAX` is set to 500, and your drone has a Hover Throttle of 50%, it will have to tilt by about 27 degrees ($\tan^{-1}(500/980)$) to achieve this acceleration. If your `ANGLE_MAX` is lower than this, you will never reach full acceleration.

The Engineer's View

This parameter is the `_accel_cmss` member in `AC_Loiter`.

Unlike **Auto** mode (which uses `WPNAV_ACCEL`), **Loiter** mode uses this parameter to calculate the "Inertia" felt by the pilot. A critical feature of this controller is that **Acceleration is prioritized over Speed**. If the drone is hit by a gust, it will use the full `LOIT_ACC_MAX` authority to hold its position before it even considers the pilot's speed request.

Tuning & Behavior

- **Default Value:** 500 cm/s² (5.0 m/s²)
- **Range:** 50 - 1000 cm/s²
- **Effect of Increasing:** Snappier, more "connected" feel to the sticks. Better performance for manual object tracking.
- **Effect of Decreasing:** Smoother, "floaty" feel. Greatly reduces the chance of accidental jerky movements appearing on camera.

Use Case Recommendations

- **Aerial Cinematography: Decrease to 200 - 300.** Makes every start and stop look smooth and professionally damped.
- **Precision Industrial Work: Keep at 500.** You want the drone to move exactly when you tell it to, with no lazy ramp-up.
- **Racing / Freestyle: Increase to 800 - 1000.** Provides the instant authority needed to change directions at the edge of the flight envelope.

Troubleshooting

- **Scenario:** Drone feels "nervous" or "jittery" in Loiter, even though P-gains are low.
  - **Diagnosis:** `LOIT_ACC_MAX` is too high, causing the drone to over-react to tiny finger twitches.
  - **Fix:** Reduce `LOIT_ACC_MAX` to 250.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY

LOIT_ANG_MAX (Loiter Maximum Lean Angle)

DEFAULT: 0	RANGE: 0 45	UNITS: deg

Loiter Maximum Lean Angle (LOIT_ANG_MAX)

Description

`LOIT_ANG_MAX` is the "tilt limiter" specifically for Loiter mode.

When you fly in **Loiter**, the flight controller is doing two things at once: following your stick commands and fighting the wind. If both of these require a steep lean, the drone might tilt too far and lose altitude. `LOIT_ANG_MAX` allows you to cap the "aggression" of Loiter mode, making it gentler and safer than manual flight modes like **Stabilize**.

- **Set to 0 (Default):** The drone uses the global `ANGLE_MAX` (typically 30-45 degrees).
- **Set > 0:** The drone will never lean further than this angle in Loiter, even if you push the sticks to the limit and a strong wind is blowing.

Physical Meaning: A lower lean angle means a lower maximum speed and slower braking. If you set this to 15 degrees, the drone will feel very calm and stable, but it won't be able to fly fast.

The Mathematics

The controller uses a ternary selector to determine the active limit (θ_limit):

$$\theta_limit = \begin{cases} \text{LOIT_ANG_MAX}, & \text{if LOIT_ANG_MAX} > 0 \\ \text{ANGLE_MAX}, & \text{otherwise} \end{cases}$$

Where θ_limit is in degrees. This limit is applied to the final output of the 2D position and velocity controllers before the lean angle is sent to the attitude controller.

The Engineer's View

This parameter is the `_angle_max` member in `AC_Loiter`.

It is applied at the end of the `AC_Loiter::update()` cycle. One important technical detail is that **this limit includes wind compensation**. If the wind requires 10 degrees of tilt to hold position, and `LOIT_ANG_MAX` is set to 20 degrees, the pilot only has 10 degrees of "leftover" tilt to use for movement. This ensures the drone never exceeds its safety envelope.

Tuning & Behavior

- **Default Value:** 0 (Inherit Global)
- **Range:** 0 - 45 degrees
- **Effect of Increasing:** Allows the drone to fly faster and stop more aggressively in Loiter mode.
- **Effect of Decreasing:** Makes the drone feel much more stable and predictable. Safe for beginners.

Use Case Recommendations

- **Aerial Cinematography: Set to 20.** Ensures the drone never tilts so aggressively that it jerks the gimbal or brings the propellers into the camera's view.
- **Underpowered Drones / Large Payload: Set to 25.** Prevents the drone from leaning so far that it doesn't have enough vertical thrust to stay in the air.
- **Standard build: Keep at 0.** Let the global safety limits handle the drone's behavior.

Troubleshooting

- **Scenario:** My drone hovers perfectly, but it's incredibly slow in Loiter mode, even with `LOIT_SPEED` set to 2000.
  - **Diagnosis:** `LOIT_ANG_MAX` is set too low, preventing the drone from tilting far enough to reach the target speed.
  - **Fix:** Set `LOIT_ANG_MAX` to 0 or increase to 30.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





LOIT_BRK_ACCEL (Loiter Braking Acceleration)

DEFAULT: 250	RANGE: 25 500	UNITS: cm/s/s

Loiter Braking Acceleration (LOIT_BRK_ACCEL)

Description

`LOIT_BRK_ACCEL` determines the "stopping distance" of your drone in Loiter mode. It defines the maximum deceleration the drone is allowed to use when you let go of the control sticks.

Think of it as the difference between a car with standard brakes and one with racing brakes.

- **Low Value (e.g. 50):** The drone will "coast" for a long distance after you release the stick. This looks very graceful on camera but can be dangerous if you need to stop quickly to avoid a tree.
- **High Value (e.g. 400):** The drone will "slam on the brakes," pitching up aggressively to stop in its tracks.
- **Default (250):** 2.5 m/s². A balanced stop that doesn't stress the frame but keeps the drone safe.

The Mathematics

The path shaper calculates the required deceleration (a_brake) to reach zero velocity:

$$a_brake = \text{constrain}\left(\frac{V_current^2}{2 \cdot \text{Dist}_to_target}, 0, \text{LOIT_BRK_ACCEL}\right)$$

If you release the sticks while flying at 10 m/s, and `LOIT_BRK_ACCEL` is 250 (2.5 m/s²), it will take exactly **4 seconds** and **20 meters** for the drone to come to a complete stop.

The Engineer's View

This parameter is the `_brake_accel_cmss` member in `AC_Loiter`.

It is specifically triggered when the sticks return to the **Deadzone** (`THR_DZ`). One advanced technical detail is that **Braking uses a different Jerk limit** (`LOIT_BRK_JERK`). This allows you to have a very high `LOIT_BRK_ACCEL` (for safety) but a low Jerk limit so the *start* of the braking maneuver isn't too violent for the gimbal.

```
// AC_Loiter.cpp
// Logic for entering the "Braking State" and applying _brake_accel_cmss.
```

Tuning & Behavior

- **Default Value:** 250 cm/s² (2.5 m/s²)
- **Range:** 25 - 500 cm/s²
- **Effect of Increasing:** Shorter stopping distance. Drone feels more "responsive" to stick centering.
- **Effect of Decreasing:** Smoother, "lazier" stops. Much better for cinematic "flow."

Use Case Recommendations

- **Cinematography: Decrease to 100 - 150.** This prevents the drone from jerking the horizon every time you finish a move.
- **Precision Industrial Flight: Increase to 350 - 400.** If you are flying 1 meter away from a building, you want the drone to stop *instantly* when you release the stick.
- **Racing: Keep Default or Increase.** You need reliable stopping power to hit gates in Loiter mode.

Troubleshooting

- **Scenario:** I release the stick, and the drone tilts back so hard that the landing gear appears in the camera shot.
  - **Diagnosis:** `LOIT_BRK_ACCEL` is too high, or `LOIT_BRK_JERK` is too high.
  - **Fix:** Reduce `LOIT_BRK_ACCEL` to 150.
- **Scenario:** I release the stick, but the drone keeps drifting for 5-10 meters before stopping.
  - **Diagnosis:** `LOIT_BRK_ACCEL` is too low.
  - **Fix:** Increase `LOIT_BRK_ACCEL` to 300.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





LOIT_BRK_DELAY (Loiter Brake Start Delay)

DEFAULT: 0.1	RANGE: 0 2.0	UNITS: s

Loiter Brake Start Delay (LOIT_BRK_DELAY)

Description

`LOIT_BRK_DELAY` is a small but important "usability" setting for Loiter mode. It determines how long the drone should wait before it starts its active braking maneuver.

When you center your control sticks, the drone can either stop immediately or "coast" for a fraction of a second. This delay is useful because it prevents the drone from "jerking" its brakes if you are just passing the stick through the center point (e.g., when switching from flying forward to flying backward).

- **Low Value (0.0):** The drone brakes the instant the stick enters the deadzone. This is the most responsive setting but can feel "twitchy" for your fingers.
- **High Value (e.g. 1.0):** The drone will coast for a full second before it actively tries to stop. This feels very "loose" and is generally not recommended for precise flight.
- **Default (0.1):** A tiny 100ms buffer that makes stick-centering feel natural without adding noticeable lag.

The Mathematics

This parameter acts as a **Deadzone Timer** (Δt_idle). The Braking state is only entered if:

$$\text{Stick_In_Deadzone_Time} \geq \text{LOIT_BRK_DELAY}$$

If the pilot moves the stick out of the deadzone before the delay expires, the braking state is cancelled and the drone continues to follow the pilot's speed command.

The Engineer's View

This parameter is the `_brake_delay` member in `AC_Loiter`.

It is a simple time-logic gate. In the `AC_Loiter::update()` loop, a counter tracks how many milliseconds have passed since the sticks were last outside the `THR_DZ`. This prevents the "SCurve Shaper" from resetting its target every time a pilot has a shaky finger.

```
// AC_Loiter.cpp
// Timer logic for initiating the brake shaper.
```

Tuning & Behavior

- **Default Value:** 0.1s
- **Range:** 0.0 - 2.0s
- **Effect of Increasing:** The drone feels "smoother" to pilot commands but takes longer to come to a stop.
- **Effect of Decreasing:** The drone stops instantly. Stick feel is much more "robotic."

Use Case Recommendations

- **High-Resolution Photography: Keep at 0.1.** This small buffer prevents tiny finger movements from triggering a gimbal-shaking brake maneuver.
- **Tactical / FPV Loitering: Decrease to 0.0.** You want the absolute maximum responsiveness. If you center the stick, the drone must stop now.
- **Beginner Training: Increase to 0.3.** Makes the drone feel more forgiving of "nervous" stick movements through the center point.

Troubleshooting

- **Scenario:** Drone seems to "hesitate" for a split second after I release the sticks before it finally starts to brake.
  - **Diagnosis:** `LOIT_BRK_DELAY` is set too high.
  - **Fix:** Reduce to 0.1 or 0.0.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY

LOIT_BRK_JERK (Loiter Braking Jerk)

DEFAULT: 500	RANGE: 500 5000	UNITS: cm/s/s/s

Loiter Braking Jerk (LOIT_BRK_JERK)

Description

`LOIT_BRK_JERK` determines the **"Feel"** of the brakes. While `LOIT_BRK_ACCEL` determines the *force* used to stop, `LOIT_BRK_JERK` determines how quickly that force is applied.

- **Low Value (500):** The drone will "ease into" its braking move. This prevents the drone from jerking the camera gimbal when you release the stick.
- **High Value (2000+):** The drone will "snap" into its braking move instantly. This feels very responsive but can be mechanically stressful.

Pro-Tip: If you want your drone to stop fast but not look jerky on camera, use a **High** `LOIT_BRK_ACCEL` combined with a **Low** `LOIT_BRK_JERK`. This allows the drone to reach a high braking power, but ramps that power up smoothly.

The Mathematics

This parameter limits the rate of change of the braking acceleration (a_brake):

$$\left|\frac{da_brake}{dt}\right| \leq \text{LOIT_BRK_JERK}$$

Where:

- LOIT_BRK_JERK is in cm/s³.
- **Example:** If set to 500, it will take 0.5 seconds to reach a full 250 cm/s² braking acceleration.

The Engineer's View

This parameter is the `_brake_jerk_max_cmsss` member in `AC_Loiter`.

It is applied at the transition between the **Pilot Input State** and the **Braking State**. When the pilot centers the sticks, the EKF position at that moment is used as the target, and this jerk-limited shaper is used to calculate the deceleration path.

```
// AC_Loiter.cpp
// Braking shaper logic uses _brake_jerk_max_cmsss to smooth the transition to zero velocity.
```

Tuning & Behavior

- **Default Value:** 500 cm/s³ (0.5 m/s³)
- **Range:** 500 - 5000 cm/s³
- **Effect of Increasing:** The drone reacts faster when you let go of the stick. Stops feel "robotic" and crisp.
- **Effect of Decreasing:** Stops feel "organic" and fluid. Reduces gimbal oscillation during the braking phase.

Use Case Recommendations

- **High-End Cinematography: Keep Default (500).** This is the sweet spot for smooth camera work.
- **Search and Rescue / Industrial: Increase to 1500.** When searching for an object, you want the drone to stop exactly where you saw it, with no lazy "gliding."
- **Small Stiff Frames (Racing): Set to 2500.** These frames handle jerk well and benefit from the faster response.

Troubleshooting

- **Scenario:** When I let go of the stick, the drone makes a sudden "thump" sound from the motors before it starts to slow down.
  - **Diagnosis:** `LOIT_BRK_JERK` is too high, commanding a torque change faster than the frame can handle.
  - **Fix:** Reduce `LOIT_BRK_JERK` to 500.

LOIT_OPTIONS (QLoiter mode options)

DEFAULT: 0	RANGE: 0 1

QLoiter mode options (LOIT_OPTIONS)

Note: This parameter is an alias for Q_LOIT_OPTIONS.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





LOIT_RADIUS (Loiter radius)

DEFAULT: 2	RANGE: 0 20	UNITS: m

Loiter radius (LOIT_RADIUS)

Description

`LOIT_RADIUS` is a **Rover-specific** parameter. It defines an "acceptance circle" around the loiter target.

Unlike a Multicopter (which tries to stay on a single pixel), a Rover (especially a Boat) often has to deal with currents or momentum that make perfect station-keeping difficult. `LOIT_RADIUS` creates a dead-zone where the Rover doesn't constantly fight small errors, saving battery and reducing mechanical wear on the steering/motors.

The Mathematics

The error e is the distance to target.

$$\text{Correction} = \begin{cases} \text{PID}(e) & \text{if } e > \text{LOIT_RADIUS} \\ 0 & \text{if } e \leq \text{LOIT_RADIUS} \end{cases}$$

(Note: Actual implementation uses a smoother blend, but this logic captures the intent).

The Engineer's View

In the Rover codebase, this is handled in `ModeLoiter`. It prevents "hunting" around the target point. For boats, this is often set to match the accuracy of the GPS or the length of the hull.

Tuning & Behavior

- **Default Value:** 2 meters
- **Range:** 0 to 20 meters
- **Recommendation:**
  - **Boats:** Set to 5.0m or higher if in a current. This prevents the motor from constantly pulsing on and off.
  - **Precision Rovers:** Set to 0.5m or 1.0m if using RTK GPS and you need tight positioning.
  - **Sailboats:** This parameter is critical for "heaving-to" or staying within a start box.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





LOIT_SPEED (Loiter Horizontal Maximum Speed)

DEFAULT: 1250	RANGE: 20 3500	UNITS: cm/s

Loiter Horizontal Maximum Speed (LOIT_SPEED)

Description

`LOIT_SPEED` is the "Speed Limiter" for your drone's most common flight mode: **Loiter**.

When you push the pitch or roll stick all the way to the edge in Loiter mode, the drone doesn't just lean over to a fixed angle—it attempts to reach a specific ground speed. `LOIT_SPEED` defines what that maximum speed is.

- **Standard Speed (1250):** 12.5 meters per second (45 km/h). A fast but manageable speed for general flight.
- **High Speed:** Makes the drone feel much more capable during transits, but makes fine position adjustments more difficult because the stick sensitivity is higher.
- **Low Speed:** Excellent for precision inspection or when learning to fly. It effectively makes the drone "un-crashable" from over-speeding.

The Mathematics

The pilot's stick position (S, from -1 to 1) is linearly mapped to a **Target Velocity** (V_target):

$$V_target = S \cdot \text{LOIT_SPEED}$$

This target velocity is then passed to the **S-Curve shaper**, which ensures the drone accelerates smoothly to reach that speed without exceeding the `LOIT_ACC_MAX` limit.

Precision Mode: At very small stick deflections, ArduPilot uses a different sensitivity curve to allow for centimeter-level positioning even if `LOIT_SPEED` is set very high.

The Engineer's View

This parameter is the `_speed_cms` member in `AC_Loiter`.

It is fundamentally different from **Stabilize** mode. In Stabilize, the stick controls **Angle**. In Loiter, the stick controls **Velocity**. This means that if you are flying at `LOIT_SPEED` and you let go of the sticks, the drone will automatically calculate the required "braking acceleration" to bring the velocity back to zero as quickly as possible.

```
// AC_Loiter.cpp
// Stick input scaling logic uses _speed_cms.
```

Tuning & Behavior

- **Default Value:** 1250 cm/s (12.5 m/s)
- **Range:** 20 - 3500 cm/s
- **Effect of Increasing:** Faster travel between points in Loiter. Drone feels more powerful.
- **Effect of Decreasing:** Drone feels "gentle" and safe. Stick movements result in smaller, more controlled speed changes.

Use Case Recommendations

- **Industrial Inspection: Decrease to 500.** When flying near a cell tower or bridge, you want the drone to move slowly even if you accidentally bump the stick.
- **Standard Hobby Flight: Set to 1500 - 2000.** Allows for spirited flying while still maintaining the safety of a GPS-locked hover.
- **Search and Rescue: Increase to 2500.** Speed is critical when searching large areas manually.

Troubleshooting

- **Scenario:** Drone feels very "twitchy" in Loiter—even a tiny stick move makes it lurch forward.
  - **Diagnosis:** `LOIT_SPEED` is set too high for the pilot's skill level or the frame's agility.
  - **Fix:** Reduce `LOIT_SPEED` to 1000.
- **Scenario:** Drone can't fly against a 10 m/s wind in Loiter.
  - **Diagnosis:** `LOIT_SPEED` must be higher than the wind speed to make progress.
  - **Fix:** Increase `LOIT_SPEED` to 1500.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY

LOIT_SPEED_GAIN (Loiter speed gain)

DEFAULT: 0.5	RANGE: 0 5

Loiter speed gain (LOIT_SPEED_GAIN)

Description

`LOIT_SPEED_GAIN` is a **Rover-specific** parameter. It acts like a "Volume Knob" for how aggressively the vehicle tries to get back into its loiter circle.

- **Low Value:** The rover gently rolls back toward the center when it drifts out.
- **High Value:** The rover snaps back quickly, potentially causing "overshoot" or rapid motor pulsing.

The Mathematics

This gain K_loit scales the desired velocity v_des derived from the distance error e:

$$v_des = K_loit \cdot \sqrt{2 \cdot a \cdot e}$$

(Following standard ArduPilot square-root controller logic for stopping distance).

The Engineer's View

Maps to `g2.loiter_speed_gain` in the Rover codebase.
It allows users to tune the responsiveness of loiter without having to touch the lower-level `ATC_SPEED_P` or `ATC_STR_RAT_P` gains, which affect all modes.

Tuning & Behavior

- **Default Value:** 0.5
- **Recommendation:**
  - If the rover "oscillates" (driving back and forth over the loiter point), decrease this to **0.2 or 0.3**.
  - If the rover drifts way too far before reacting, increase this toward **1.0**.

LOIT_TYPE (Loiter type)

DEFAULT: 0	RANGE: 0 2

Loiter type (LOIT_TYPE)

Description

`LOIT_TYPE` is a **Rover-specific** parameter that tells the vehicle how to move when it drifts outside its loiter circle.

- **0: Forward or Reverse.** The rover will take the shortest path back to the center, even if it means driving backward. (Recommended for standard rovers).
- **1: Always Face Forward.** The vehicle will only drive forward to return to the point. It will perform a U-turn if necessary. (Recommended for boats without a reverse gear or large scale rovers).
- **2: Always Face Stern.** The vehicle will only move backward to return. (Rarely used).

The Mathematics

This parameter toggles the logic branch in the steering/throttle controller:

$$\text{TargetHeading} = \begin{cases} \theta_direct & \text{if Type 1} \\ \theta_direct \pm 180^\circ & \text{if Type 0 and } dist < limit \end{cases}$$

The Engineer's View

Controlled in `Rover::ParametersG2::loit_type`.
Affects path planning within `ModeLoiter`. If Type 1 is selected, the "Pivot Turn" logic may be triggered depending on the angle error.

Tuning & Behavior

- **Default Value:** 0 (Shortest path)
- **Recommendation:**
  - **Cars/Mowers:** Use **0**. It's the most efficient for battery life.
  - **Boats:** Use **1** if your motor setup doesn't handle reverse well or if you want the boat to always face the wind/waves while loitering.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **MAG** parameter group (specifically `MAG_ALY`) is a **SITL-only** tool used to simulate magnetic interference.

Note: For physical compass calibration and setup, please refer to the COMPASS group.

Key Concepts

1. Simulated Interference

In the Software-In-The-Loop (SITL) simulator, developers can inject a "Magnetic Anomaly" to see how the EKF and navigation algorithms handle incorrect heading data.

- `MAG_ALY`: A 3D vector (X, Y, Z) in Gauss that is added to the simulated world's magnetic field.
- **Purpose:** Testing the "Compass Variance" failsafe logic and "Compass-less" flight algorithms.

Developer Notes

- **Library:** `libraries/SITL`.
- **Units:** Gauss (Magnetic field strength).

Magnetic Anomaly Vector ()

MAG_ALY: Magnetic Anomaly Vector

Description

Sets a local magnetic anomaly vector (in Gauss) to be added to the earth's magnetic field in the simulator.

Values

- **Units:** Gauss
- **Format:** Vector3 (X, Y, Z)
- **Default:** 0, 0, 0

Description

This parameter allows developers to simulate magnetic interference or local anomalies in the **SITL** environment.

- **X, Y, Z Components:** Represents the anomaly vector added to the base earth magnetic field.
- **Use Case:** Testing compass calibration routines, EKF mag fusion resilience, or simulating flight near large metal structures.
- **Note:** The anomaly strength decays with height based on `MAG_ALY_HGT`.

Source Code

ardupilot/libraries/SITL/SITL.cpp

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **MAN** parameter group configures the "Exponential" (Expo) curves for **ArduPlane** when flying in manual and stabilized modes.

Expo softens the response of the control surfaces around the center of the sticks. This allows the pilot to make tiny, precise corrections for level flight while still having full control authority at the stick limits.

Key Concepts

1. Manual Expo (`MAN_EXPO_...`)

- `MAN_EXPO_ROLL` / `PITCH` / `RUDDER`: Sets the amount of curve for each axis.
  - **0:** Linear (direct mapping).
  - **100:** Maximum softness at center, very aggressive at ends.
  - **Typical:** 30% to 50% is common for a smooth feel.

Parameter Breakdown

- `MAN_EXPO_ROLL`: Roll stick expo (%).
- `MAN_EXPO_PITCH`: Pitch stick expo (%).
- `MAN_EXPO_RUDDER`: Rudder stick expo (%).

Developer Notes

- **Library:** `ArduPlane/servos.cpp`.
- **Context:** Only applies to **Manual**, **Acro**, and **FBWA/B** modes. Does not affect autonomous navigation.

MAN_EXPO_RUDDER (Manual input expo for rudder)

DEFAULT: 0	RANGE: 0 100	UNITS: %

Manual input expo for rudder (MAN_EXPO_RUDDER)

Description

This parameter adds **Exponential** (Expo) to the Rudder stick input when flying in Manual, Acro, or Training modes. Expo softens the response around the center of the stick, allowing for precise small corrections, while maintaining full control authority at the stick limits.

The Mathematics

$$\text{Input_}shaped = (1 - k) \cdot \text{Input} + k \cdot \text{Input}^3$$

Where $k = \frac{\text{MAN_EXPO_RUDDER}}{100}$.

The Engineer's View

Defined in `ArduPlane/Parameters.cpp`. Used in `Plane::read_radio()` to pre-process the RC input before it is passed to the servo mixers. This applies only to the Rudder channel.

Tuning & Behavior

- **Default Value:** 0% (Linear response)
- **Range:** 0 - 100%
- **30-40%:** Typical value for smoother yaw control.
- **Higher Values:** Very soft center, aggressive endpoint response.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





`Manual control expo for roll ()`

MAN_EXPO_ROLL: Manual control expo for roll

Description

Percentage exponential for roll input in MANUAL, ACRO, and TRAINING modes.

Values

- **Range:** 0 to 100
- **Units:** %
- **Increment:** 1
- **Default:** 0

Description

This parameter adds exponential curvature ("expo") to the aileron stick input when flying in manual or semi-manual modes.

- **0%:** Linear response. 50% stick = 50% servo travel.
- **Higher %:** Reduces sensitivity around the center stick position (neutral) while maintaining full throw at the extremes.
- **Benefit:** Allows for smoother fine-control adjustments during level flight while still permitting aggressive maneuvers when the stick is pushed fully.

Source Code

ardupilot/ArduPlane/Parameters.cpp

`Manual input expo for pitch ()`

MAN_EXPO_PITCH: Manual input expo for pitch

Description

Percentage exponential for pitch input in MANUAL, ACRO, and TRAINING modes.

Values

- **Range:** 0 to 100
- **Units:** %
- **Increment:** 1
- **Default:** 0

Description

This parameter adds exponential curvature ("expo") to the elevator stick input when flying in manual or semi-manual modes.

- **0%:** Linear response.
- **Higher %:** Reduces sensitivity around the neutral pitch stick position.
- **Benefit:** Helps prevent "porpoising" (pilot-induced oscillation in pitch) by making small elevator corrections less sensitive, while keeping full authority for loops and sharp turns.

Source Code

ardupilot/ArduPlane/Parameters.cpp

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **MANUAL** parameter group configures the behavior of the **Manual** flight/drive mode, primarily for **ArduRover**.

In Manual mode, the pilot's sticks are passed directly to the motor mixers with minimal autopilot intervention.

Key Concepts

1. Steering Expo (`MANUAL_STR_EXPO`)

Similar to the `MAN` group in Plane, this adds an exponential curve to the Rover's steering stick. It makes the rover less twitchy when driving straight at high speeds.

2. Output Masking (`MANUAL_RCMASK`)

Allows the autopilot to ignore certain RC channels while in Manual mode, forcing those outputs to a fixed value.

Parameter Breakdown

- `MANUAL_STR_EXPO`: Steering exponential percentage.
- `MANUAL_OPTIONS`: Bitmask for specific behaviors (e.g., handling throttle in manual).

Developer Notes

- **Library:** `Rover/mode_manual.cpp`.

MANUAL_OPTIONS (Manual Mode Options)

DEFAULT: 0	RANGE: 0 1	UNITS: Bitmask

Manual Mode Options (MANUAL_OPTIONS)

Description

`MANUAL_OPTIONS` provides specialized behaviors for the **MANUAL** drive mode, primarily on Rovers.

- **Bit 0 (1): Enable Steering Speed Scaling.** When enabled, the steering sensitivity is automatically reduced as the vehicle goes faster. This prevents high-speed "twitchiness" and rollovers caused by sudden sharp turns at full throttle.

Tuning & Behavior

- **Default:** 0 (Full sensitivity at all speeds).
- **Recommendation:** Enable this if you find your Rover difficult to drive straight at high speeds.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY

MANUAL_RCMASK (Manual R/C Pass-Through Mask)

DEFAULT: 0	RANGE: 0 65535	UNITS: Bitmask

Manual R/C Pass-Through Mask (MANUAL_RCMASK)

Description

`MANUAL_RCMASK` allows advanced pilots to use "Old School" transmitter mixing when flying in **MANUAL** mode.

Normally, even in MANUAL mode, ArduPilot performs some processing on your stick inputs to map them to the correct outputs (e.g., mixing Elevons). By setting bits in this mask, you tell the autopilot to completely step out of the way for specific channels. Your transmitter's mixed signals will flow directly to the servos as if the flight controller wasn't there.

- **0 (Default):** Normal ArduPilot behavior.
- **Bitmask:** Each bit represents a channel (Bit 0 = Chan 1, Bit 1 = Chan 2, etc.).

Tuning & Behavior

- **Warning:** This is an **expert-only** parameter. If your transmitter mix is different from your autopilot's stabilized mix (e.g. FBWA), the plane will behave differently depending on the mode.
- **Safety:** Ensure your "Failsafe" behavior is still correct. If the autopilot takes over during a failsafe, it will use its *own* mixing, not your transmitter's.

MANUAL_STR_EXPO (Manual Steering Exponential)

DEFAULT: 0	RANGE: -0.5 0.95

Manual Steering Exponential (MANUAL_STR_EXPO)

Description

`MANUAL_STR_EXPO` makes Rover steering feel less "twitchy" when the control sticks are near the center.

Without Expo (0), the steering is linear—small stick movements result in immediate, sharp turns. This can make high-speed driving difficult. Setting a positive Expo value flattens the curve near the center, providing fine control for straight-line driving and subtle corrections, while still allowing for full lock-to-lock steering when the stick is pushed to the edge.

Tuning & Behavior

- **0:** Linear response (Standard).
- **Positive (0.1 to 0.95):** Softer center-stick feel.
- **Negative (-0.1 to -0.5):** Sharper center-stick feel (Aggressive).
- **Recommendation:** Start with **0.2** for high-speed rovers to improve stability.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **MAV** parameter group configures the frequency and content of the **MAVLink Telemetry** streams.

MAVLink is the language ArduPilot uses to talk to Ground Stations, Companion Computers, and HUDs. Because telemetry links (like Sik Radios) often have limited bandwidth, the autopilot allows you to tune how often specific types of data are sent.

Key Concepts

1. Stream Rates (`MAVn_...`)

You can configure the frequency (Hz) for different categories of messages:

- `EXTRA1`: Attitude data (Roll, Pitch, Yaw).
- `EXTRA2`: VFR HUD data (Speed, Altitude, Throttle).
- `EXTRA3`: System status and AHRS data.
- `POSITION`: GPS coordinates and velocity.
- `RAW_SENS`: Raw accelerometer, gyro, and mag data.

2. System ID (`MAV_SYSID`)

Every vehicle on a network must have a unique ID.

- `MAV_SYSID`: The ID of this vehicle (default 1).
- `MAV_COMPID`: The component ID (default 1 for autopilot).

Parameter Breakdown

- `MAV_SYSID`: System identifier (1-255).
- `MAV1_POSITION`: GPS stream rate (Hz).
- `MAV1_EXTRA1`: Attitude stream rate (Hz).

Integration Guide

Optimizing Bandwidth

1. **High Speed:** If using a USB cable or high-speed radio, you can set rates to 10Hz or 50Hz for smooth HUD movement.
2. **Low Speed:** If using a long-range, low-baud radio, set position to 2Hz and raw sensors to 0Hz to avoid clogging the link.

Developer Notes

- **Library:** `libraries/GCS_MAVLink`.
- **Mapping:** The rates are global, but can be requested dynamically by the GCS via the `REQUEST_DATA_STREAM` message.

MAV1_ADSB (MAVLink ADS-B Stream Rate (Telem 1))

DEFAULT: 5	RANGE: 0 50	UNITS: Hz

MAVLink ADS-B Stream Rate (MAV1_ADSB)

Description

`MAV1_ADSB` sets the update frequency for traffic information on your GCS map.

If your drone is equipped with an ADS-B receiver (like the Ping1090), it detects nearby full-sized aircraft. This parameter determines how often those plane positions are sent to your Ground Control Station.

Tuning & Behavior

- **Default:** 5 Hz (Note: Rover defaults to 0).
- **Safety: 5 Hz** is a good value to ensure you see approaching aircraft with minimal lag on your map.
- **Bandwidth:** This stream only sends data when aircraft are actually detected. It consumes very little bandwidth when the sky is clear.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





MAV1_EXTRA1 (MAVLink Extra 1 Stream Rate (Telem 1))

DEFAULT: 1	RANGE: 0 50	UNITS: Hz

MAVLink Extra 1 Stream Rate (MAV1_EXTRA1)

Description

`MAV1_EXTRA1` sets the update frequency for orientation and tuning data.

This stream group includes:

- `ATTITUDE` (Roll, Pitch, Yaw).
- `SIMSTATE` (SITL only).
- `AHRS2` (Secondary AHRS state).
- `PID_TUNING` (Live data for GCS tuning graphs).

Tuning & Behavior

- **Default:** 1 Hz.
- **Smooth HUD:** If you want a smooth artificial horizon on your GCS or OSD, set this to **4 Hz** or **10 Hz**.
- **PID Tuning:** Set to **20 Hz** or **50 Hz** temporarily when using the GCS PID graphs to capture high-frequency oscillations.
- **Bandwidth:** Higher rates consume more bandwidth. On a 57.6k SiK radio, avoid setting this above 10 Hz if other streams are active.

MAV1_EXTRA2 (MAVLink Extra 2 Stream Rate (Telem 1))

DEFAULT: 1	RANGE: 0 50	UNITS: Hz

MAVLink Extra 2 Stream Rate (MAV1_EXTRA2)

Description

`MAV1_EXTRA2` sets the update frequency for the "Dashboard" data.

This stream group primarily contains the `VFR_HUD` message, which includes:

- Airspeed.
- Groundspeed.
- Heading.
- Throttle percentage.
- Altitude.
- Climb rate.

Tuning & Behavior

- **Default:** 1 Hz.
- **Recommendation:** Set to **2 Hz** or **5 Hz** for a more responsive display of speed and altitude on your GCS dashboard.
- **Notes:** This message is the primary source of data for the "HUD" instrument in Mission Planner and QGroundControl.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





MAV1_EXTRA3 (MAVLink Extra 3 Stream Rate (Telem 1))

DEFAULT: 1	RANGE: 0 50	UNITS: Hz

MAVLink Extra 3 Stream Rate (MAV1_EXTRA3)

Description

`MAV1_EXTRA3` sets the update frequency for a variety of "Housekeeping" and secondary sensor messages.

This stream group includes:

- `SYSTEM_TIME` and `AHRS` status.
- `BATTERY_STATUS` (Voltage, Current, Capacity).
- `WIND` (Direction and Speed).
- `VIBRATION` (Live clipping and vibe levels).
- `RPM` and `ESC_TELEMETRY`.
- `RANGEFINDER` and `OPTICAL_FLOW`.

Tuning & Behavior

- **Default:** 1 Hz.
- **Recommendation: 1 Hz** or **2 Hz** is usually sufficient for these values, as they do not change rapidly.
- **Bandwidth:** This group is quite large (many different messages). Increasing the rate can significantly impact bandwidth on low-speed telemetry links.

MAV1_EXT_STAT (MAVLink Extended Status Stream Rate)

DEFAULT: 1	RANGE: 0 50	UNITS: Hz

MAVLink Extended Status Stream Rate (MAV1_EXT_STAT)

Description

`MAV1_EXT_STAT` determines how often the drone reports its "Vital Signs" to the Ground Control Station.

This parameter covers a large group of essential status messages, including:

- **SYS_STATUS:** Battery voltage, current, and sensor health.
- **GPS_RAW_INT:** Number of satellites, HDOP, and fix type.
- **MEMINFO:** CPU load and free memory.
- **NAV_CONTROLLER_OUTPUT:** Target vs. actual headings and altitude.

Tuning & Behavior

- **Default:** 1 Hz.
- **Recommendation:** Set to **2 or 5 Hz** for a more responsive GCS dashboard.
- **Note:** This is one of the most important stream rates for pilot situational awareness. If it is too low, your battery and GPS readings will look "stuck" or update very infrequently.

MAV1_PARAMS (MAVLink Parameter Stream Rate)

DEFAULT: 10	RANGE: 0 50	UNITS: Hz

MAVLink Parameter Stream Rate (MAV1_PARAMS)

Description

`MAV1_PARAMS` defines how fast ArduPilot "shouts" its settings to your Ground Control Station (GCS).

When you open the parameter list in Mission Planner or QGroundControl, this rate determines how quickly the data fills up your screen. If the rate is too low, the GCS will feel sluggish. If it is too high, it might consume too much bandwidth on a slow telemetry radio link.

Tuning & Behavior

- **Default:** 10 Hz.
- **Recommendation:** Leave at **10** for most setups. If you are using a high-speed link (WiFi or Ethernet), you can increase this to **50** for instant parameter refreshes.
- **Context:** This is the rate for `PARAM_VALUE` messages.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





MAV1_POSITION (MAVLink Position Stream Rate (Telem 1))

DEFAULT: 1	RANGE: 0 50	UNITS: Hz

MAVLink Position Stream Rate (MAV1_POSITION)

Description

`MAV1_POSITION` sets the update frequency for the vehicle's location data.

This stream group includes:

- `GLOBAL_POSITION_INT` (Latitude, Longitude, Altitude, Velocity).
- `LOCAL_POSITION_NED` (Position relative to the home location).

Tuning & Behavior

- **Default:** 1 Hz.
- **Map Smoothness:** If your drone "jumps" on the GCS map or in the HUD, increase this to **2 Hz** or **5 Hz**.
- **Bandwidth:** This is one of the most important streams for basic navigation. Ensure it has a higher priority than "Extra" data groups if bandwidth is limited.

MAV1_RAW_CTRL (MAVLink Raw Control Stream Rate)

DEFAULT: 0	RANGE: 0 50	UNITS: Hz

MAVLink Raw Control Stream Rate (MAV1_RAW_CTRL)

Description

`MAV1_RAW_CTRL` defines the frequency for specialized raw control data messages.

In most modern ArduPilot versions, this stream is **Unused** or reserved for Hardware-In-The-Loop (HIL) simulations and developer-level diagnostic analysis.

Tuning & Behavior

- **Default:** 0 Hz.
- **Recommendation:** Leave at **0** unless specifically instructed by a developer or when setting up a custom simulation environment that requires high-rate raw actuator data.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





MAV1_RAW_SENS (Raw sensor stream rate (Telem 1))

DEFAULT: 1	RANGE: 0 50	UNITS: Hz

Raw sensor stream rate (MAV1_RAW_SENS)

Description

`MAV1_RAW_SENS` sets how many times per second the autopilot sends "Raw Sensor" data to the Ground Control Station (GCS) or Companion Computer.

This includes:

- `RAW_IMU` (Accelerometers, Gyroscopes, Magnetometers)
- `SCALED_PRESSURE` (Barometer)
- `SCALED_PRESSURE2`
- `SCALED_PRESSURE3`
- `AIRSPEED`

The Engineer's View

This controls the `SR1_RAW_SENS` stream group.
High rates are essential for off-board Visual Inertial Odometry (VIO) or FFT analysis, but consume significant telemetry bandwidth.

Tuning & Behavior

- **Default:** 1 Hz.
- **Telemetry (SiK Radio):** Keep at **1-2 Hz** to save bandwidth.
- **Companion Computer (VIO): 50 Hz** (or higher if supported) is required for accurate state estimation.
- **Log Replay:** If you are recording tlogs for replay, higher rates give smoother playback but huge file sizes.

MAV1_RC_CHAN (MAVLink RC Channel Stream Rate)

DEFAULT: 0	RANGE: 0 50	UNITS: Hz

MAVLink RC Channel Stream Rate (MAV1_RC_CHAN)

Description

`MAV1_RC_CHAN` controls how often the autopilot reports your stick positions and motor outputs to the Ground Control Station.

This is essential for viewing the "Radio Calibration" bars or "Servo Output" diagnostics in your GCS while the drone is powered on. It also affects the refresh rate of any RC-based widgets or graphs on your dashboard.

Tuning & Behavior

- **Default:** 0 Hz (Only sends when requested or at a very low base rate).
- **Recommendation:** Set to **2 or 5 Hz** if you want to see smooth movements on the GCS radio bars.
- **Context:** Covers `SERVO_OUTPUT_RAW` and `RC_CHANNELS` messages.

MAV_OPTIONS (quadplane options)

DEFAULT: 0	RANGE: null	UNITS: null

quadplane options

Note: This parameter functions identically to MAV1_OPTIONS.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





MAV_SYSID (Slung Payload MAVLink System ID)

DEFAULT: 2	RANGE: 1 255

Slung Payload MAVLink System ID (MAV_SYSID)

Description

`MAV_SYSID` is used when simulating or controlling a **Slung Payload** (a weight hanging from the drone via a cable).

In a MAVLink network, every independent object (the drone, the GCS, the payload) needs its own System ID. This parameter identifies the payload itself, allowing ground station software to track the payload's position and orientation separately from the drone's.

Tuning & Behavior

- **Default:** 2.
- **Usage:** Only relevant for advanced missions involving autonomous slung load delivery or research. Most users can leave this at the default.

MAV_TELEM_DELAY (Telemetry startup delay)

DEFAULT: 0	RANGE: 0 30	UNITS: s

MAV_TELEM_DELAY: Telemetry startup delay

Description

The amount of time (in seconds) to delay radio telemetry to prevent an Xbee bricking on power up.

Values

- **Range:** 0 to 30
- **Units:** s
- **Increment:** 1
- **Default:** 0

Description

This parameter adds a pause before the telemetry ports are initialized during boot.

- **Purpose:** Some older telemetry radios (specifically certain XBee modules) could enter a "bricked" or unresponsive state if they received data immediately upon power-up while their internal bootloader was running. This delay ensures the flight controller waits until the radio is fully booted before sending any MAVLink data.
- **Modern Hardware:** Most modern radios (SiK, RFD900, WiFi) do not require this, so it can typically be left at 0.

Source Code

ardupilot/Rover/Parameters.cpp

MAV_UPDATE_RATE (Slung Payload Update Rate)

DEFAULT: 1	RANGE: 1 10	UNITS: Hz

Slung Payload Update Rate (MAV_UPDATE_RATE)

Description

`MAV_UPDATE_RATE` defines how often the simulated or real slung payload sends its data to the flight controller or ground station.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **MAX** parameter group (often appearing as `PSC_MAX_` in the full parameter list) defines the hard **Kinematic Limits** for the Position Controller.

These limits ensure that the autopilot never requests a velocity that exceeds the physical capabilities of the vehicle or the desires of the pilot for smoothness.

Key Concepts

1. Velocity Limits (`MAX_VEL_...`)

- `MAX_VEL_XY` : The maximum horizontal speed allowed in autonomous modes (Auto, Guided).
- `MAX_VEL_Z` : The maximum climb and descent rate.

2. Position Error Limits (`MAX_POS_...`)

Defines the maximum allowed distance between the vehicle's actual position and its "Target" position before the controller stops trying to close the gap instantly (preventing violent corrections).

Parameter Breakdown

- `MAX_VEL_XY` : Horizontal velocity limit (m/s).
- `MAX_VEL_Z` : Vertical velocity limit (m/s).

Developer Notes

- **Library:** `libraries/AC_PosControl`.
- **Context:** These are the "Inner" limits of the navigation system. The higher-level mission speed (`WP_NAV_SPEED`) is usually capped by these.

MAX_POS_XY (Max XY Position Delta (Blimp))

DEFAULT: 0.2	RANGE: 0.1 5.0	UNITS: m/s

Max XY Position Delta (MAX_POS_XY)

Description

`MAX_POS_XY` is a safety and smoothing parameter for **ArduBlimp**.

It limits how far the blimp's target position can "Jump" in a single update cycle. Because blimps have very high inertia and move slowly, sudden large changes in requested position can cause the control loops to become unstable. This parameter ensures that the requested path remains smooth and achievable by the blimp's low-thrust propulsion.

Tuning & Behavior

- **Default:** 0.2 m/s.
- **Usage:** For larger, slower blimps, you may need to reduce this value to **0.1** to ensure smooth autonomous movement.

MAX_POS_YAW (Max Yaw Position Delta (Blimp))

DEFAULT: 0.3	RANGE: 0.1 5.0	UNITS: rad/s

Max Yaw Position Delta (MAX_POS_YAW)

Description

`MAX_POS_YAW` limits how fast the blimp's heading target can rotate. This is crucial for avoiding large, oscillating yaw movements in airships with high rotational inertia.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





MAX_POS_Z (Max Vertical Position Delta (Blimp))

DEFAULT: 0.2	RANGE: 0.1 5.0	UNITS: m/s

Max Vertical Position Delta (MAX_POS_Z)

Description

`MAX_POS_Z` limits how fast the vertical target position can move in **ArduBlimp**. It ensures that altitude changes are smooth and don't overwhelm the blimp's limited vertical thrust.

MAX_VEL_XY (Max Horizontal Velocity Delta (Blimp))

DEFAULT: 0.5	RANGE: 0.1 5.0	UNITS: m/s^2

Max Horizontal Velocity Delta (MAX_VEL_XY)

Description

`MAX_VEL_XY` acts as an acceleration limit for the blimp's horizontal flight. It prevents the autopilot from demanding sudden bursts of speed that would cause the blimp to pitch or swing violently.

MAX_VEL_YAW (Max Yaw Velocity Delta (Blimp))

DEFAULT: 0.3	RANGE: 0.1 5.0	UNITS: rad/s^2

Max Yaw Velocity Delta (MAX_VEL_YAW)

Description

`MAX_VEL_YAW` limits the rotational acceleration of the blimp. This prevents the propulsion system from trying to snap the blimp's heading too aggressively, which could damage the frame or motors.

MAX_VEL_Z (Max Vertical Velocity Delta (Blimp))

DEFAULT: 0.5	RANGE: 0.1 5.0	UNITS: m/s^2

Max Vertical Velocity Delta (MAX_VEL_Z)

Description

`MAX_VEL_Z` limits vertical acceleration. It ensures that the blimp's transition between climbing and descending is handled smoothly by the buoyancy and motor control logic.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **MB** parameter group (specifically `MB_CAN_PORT`) configures the **GPS Moving Baseline** subsystem.

Moving Baseline is an advanced RTK technique where two GPS receivers on the same drone (a "Base" and a "Rover") calculate a precise relative position vector between each other. This vector provides an extremely accurate **Yaw (Heading)** source that is immune to magnetic interference.

Key Concepts

1. Inter-Receiver Communication

The two GPS units must talk to each other to exchange raw measurement data.

- `MB_CAN_PORT`: Selects the CAN bus port used for this inter-GPS communication if you are using DroneCAN GPS units (like the Here3+ or F9P CAN).

Developer Notes

- **Library:** `libraries/AP_GPS`.
- **Requirement:** Requires two RTK-capable GPS receivers (e.g., u-Blox F9P).
- **Benefit:** Allows flying in high-interference areas (near metal roofs, power lines) where a standard compass would fail.

MB_CAN_PORT (Moving Baseline CAN Port option)

`DEFAULT: 0`

MB_CAN_PORT: Moving Baseline CAN Port option

Description

Autoselect dedicated CAN port on which moving baseline data will be transmitted.

Values

Value	Meaning
0	Sends moving baseline data on all ports
1	Auto select remaining port for transmitting Moving baseline Data

- **Default:** 0

Description

This parameter manages the routing of high-bandwidth RTK "Moving Baseline" data between two GPS units (typically for GPS-based Yaw).

- **0 (Broadcast):** The moving baseline correction data is sent out on all active CAN interfaces. This is the simplest configuration but consumes more bus bandwidth.
- **1 (Auto-Select):** The system attempts to find a specific, dedicated CAN port to send the data on, keeping the other ports free for other traffic. This is useful on devices with multiple CAN buses to segregate the heavy RTK traffic.

Source Code

ardupilot/Tools/AP_Periph/Parameters.cpp

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **MIN** parameter group is a collection of minimum safety and operational thresholds for various subsystems.

Key Concepts

1. Terrain Following (`MIN_CM`)

(ArduPlane) Defines the minimum clearance altitude (in centimeters) the autopilot will maintain above the ground when using **Terrain Following** (TERRAIN) logic.

2. Tracker Reversal (`MIN_REVERSE_TIME`)

(Antenna Tracker) For trackers that don't have continuous 360-degree rotation, they must occasionally "unwind." This parameter sets the minimum time (seconds) to continue the unwinding motion once a limit is hit, preventing rapid oscillation at the end-stop.

Developer Notes

- **Context:** `MIN_CM` is associated with `AP_Terrain`. `MIN_REVERSE_TIME` is specific to `AntennaTracker`.

`MIN_CM` (Rangefinder minimum distance)

DEFAULT: 20	UNITS: cm

MIN_CM: Rangefinder minimum distance

Description

Minimum distance in centimeters that the rangefinder can reliably read.

Values

- **Units:** cm
- **Increment:** 1
- **Default:** 20

Description

This parameter defines the "blind spot" of the rangefinder.

- **Function:** Any reading returned by the sensor that is less than this value will be treated as invalid (out of range).
- **Why it matters:** Most acoustic (Sonar) and optical (Lidar) rangefinders cannot measure distance zero. They have a minimum range (e.g., 20cm). If the vehicle is on the ground and the sensor reads noise below this threshold, filtering out these values prevents the autopilot from thinking it's flying underground or getting bad altitude data.

Source Code

ardupilot/libraries/AP_RangeFinder/AP_RangeFinder_Params.cpp

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





MIN_REVERSE_TIME (Minimum time to apply a yaw reversal)

DEFAULT: 1	RANGE: 0 20	UNITS: s

Minimum time to apply a yaw reversal (MIN_REVERSE_TIME)

Description

This parameter is specific to Antenna Trackers with mechanical stops (non-continuous rotation). When the tracker hits its yaw limit (e.g., 360 degrees), it must stop and reverse direction ("unwind") to re-acquire the target. This parameter forces the unwinding motion to continue for a minimum duration, preventing the tracker from getting stuck toggling back and forth at the limit.

The Mathematics

Hysteresis Timer:

$$\text{If (Limit Hit)} \implies \text{Reverse State} = \text{True}$$

$$\text{If (Time in Reverse} < \text{MIN_REVERSE_TIME)} \implies \text{Keep Reversing}$$

The Engineer's View

Defined in `AntennaTracker/Parameters.cpp`. Handles the "Unwrap" logic for servos.

- **Lag Compensation:** Mechanical systems have inertia and lag; this timer ensures the movement is significant enough to clear the stop.

Tuning & Behavior

- **Default Value:** 1 s
- **Increase:** If the tracker "chatters" or oscillates at the end-stop.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY

Overview

The **MIS** parameter group configures how the autopilot executes and manages **Waypoints** and **Missions**.

This group controls the high-level logic of the Auto flight mode.

Key Concepts

1. Mission Completion (`MIS_DONE_BEHAVE`)

Defines what the vehicle does after the last waypoint in the mission is reached.

- **0:** Stay in Auto (Loiter at last WP).
- **1:** Switch to RTL (Return to Launch).
- **2:** (Rover) Switch to Hold.

2. Restart Logic (`MIS_RESTART`)

Determines if a mission resumes from the last waypoint or starts from the beginning if the mode is toggled.

- **0 (Resume):** If you switch from Auto to Loiter and back to Auto, the vehicle continues where it left off.
- **1 (Restart):** Every time you enter Auto mode, the mission starts from Waypoint 1.

Parameter Breakdown

- `MIS_TOTAL`: Read-only count of mission items.
- `MIS_OPTIONS`: Bitmask for specialized behaviors (e.g., skip first waypoint).

Integration Guide

- **Safe Ops:** Set `MIS_DONE_BEHAVE = 1` (RTL) for long-range missions to ensure the drone comes home once its work is done.

Developer Notes

- **Library:** `libraries/AP_Mission`.

MIS_DONE_BEHAVE (Mission done behavior)

DEFAULT: 0	RANGE: 0 3

Mission done behavior (MIS_DONE_BEHAVE)

Description

`MIS_DONE_BEHAVE` is a **Rover/Boat-specific** parameter. It tells the vehicle what to do when the mission "finish line" is crossed.

- **0: Hold (Default).** The vehicle stops and enters **Hold** mode.
- **1: Loiter.** (Recommended for Boats). The vehicle stays at the final location, fighting wind or current.
- **2: Acro.** Switches to Acro mode.
- **3: Manual.** Switches to Manual mode.

The Engineer's View

Controlled in the Rover vehicle logic.
For boats, **Loiter (1)** is almost always better than **Hold (0)**, because a boat in "Hold" mode will drift away with the tide, whereas in "Loiter" it will use its motors to stay on the final waypoint.

Tuning & Behavior

- **Default Value:** 0 (Hold)
- **Recommendation:**
  - **Cars/Mowers:** Use **0 (Hold)**.
  - **Boats:** Use **1 (Loiter)**.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





MIS_OPTIONS (Mission Options)

DEFAULT: 0	RANGE: 0 7

Mission Options (MIS_OPTIONS)

Description

`MIS_OPTIONS` provides a set of toggle-able rules for how missions are handled.

- **Bit 0 (1): Clear on Reboot.** If enabled, the drone will wipe its mission memory every time you power it off. (Useful for data security or ensuring you don't accidentally fly an old mission).
- **Bit 2 (4): ContinueAfterLand.** (Copter Only). If a mission includes a `LAND` command, the drone will stay armed and continue to the next waypoint (usually a takeoff) after landing. Without this, the mission ends at the landing.

The Engineer's View

Used in `AP_Mission::init()` (for clear on boot) and `AP_Mission::continue_after_land_check_for_takeoff()`.

Tuning & Behavior

- **Default Value:** 0
- **Recommendation:** Usually leave at **0**. Use **Bit 2 (4)** only if your mission requires multiple stop-and-go landings (e.g., package delivery).

MIS_RESTART (Mission Restart)

DEFAULT: 0	RANGE: 0 1

Mission Restart (MIS_RESTART)

Description

`MIS_RESTART` determines the drone's "Memory" when you switch into **Auto** mode.

- **0: Resume (Default).** If you were mid-mission, switched to manual to avoid an obstacle, and then switched back to Auto, the drone will continue from where it left off (the last active waypoint).
- **1: Restart.** Every time you enter Auto mode, the drone will fly back to Waypoint #1 and start the mission from the beginning.

The Engineer's View

Defined in `AP_Mission.cpp`.
When the vehicle enters `MISSION_RUNNING` state, the `start_or_resume()` function is called. This parameter acts as the logic gate between `start()` (reset indices) and `resume()` (keep current index).

Tuning & Behavior

- **Default Value:** 0 (Resume)
- **Recommendation:** Keep at **0**. It is much safer to resume a mission than to have the drone unexpectedly turn around and fly back to the start of a large field.
- **Note:** If the mission has reached the final waypoint and completed, entering Auto mode will always restart from #1 regardless of this parameter.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





MIS_TOTAL (Total Mission Commands)

DEFAULT: 0	RANGE: 0 32766	UNITS: Commands

Total Mission Commands (MIS_TOTAL)

Description

This parameter serves as a counter for the total number of waypoints and mission commands currently saved in the autopilot's onboard memory. It tells the flight controller exactly how many steps are in the current flight plan.

Important: This is a read-only parameter managed by the system. Users should never manually edit this value. The Ground Control Station (GCS) updates it automatically whenever you upload or clear a mission.

The Mathematics

The value N represents the count of valid mission items:

$$N = \text{MIS_TOTAL}$$

The valid range of command indices is $[0, N-1]$, where index 0 is typically the "Home" location.

The maximum possible value depends on the available EEPROM or FRAM storage on your specific flight controller board.

The Engineer's View

In `AP_Mission.cpp`, this parameter maps to `_cmd_total`.

- **Storage Logic:** When `write_cmd_to_storage()` is called (e.g., during a mission upload), this counter is incremented.
- **Sanity Check:** During `init()`, the code checks if `_cmd_total` exceeds the calculated maximum capacity of the storage device (`_commands_max`). If it does (indicating corruption), the mission is wiped, and `_cmd_total` is reset to 0.
- **Truncation:** The `truncate()` function uses this parameter to logically "delete" items from the end of the mission list by simply reducing the count.

Tuning & Behavior

- **Default Value:** 0 (Empty Mission).
- **Update:** Automatically updated by GCS operations.
- **Troubleshooting:** If your vehicle refuses to enter AUTO mode, check `MIS_TOTAL`. If it is 0 or 1 (just Home), the vehicle knows it has no mission to fly.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY

Overview

The **MIXING** parameter group (specifically `MIXING_GAIN` and `MIXING_OFFSET`) configures the output scaling for **Elevon** and **V-Tail** aircraft.

In these airframes, two surfaces handle two different control axes (e.g., Elevons handle Roll AND Pitch). The mixer combines the demands from the controllers into physical servo movements.

Key Concepts

1. Mixing Gain (`MIXING_GAIN`)

The multiplier applied to the mixed output.

- **Purpose:** If your control surfaces don't move enough at full stick, increase this. If they move too much (binding), decrease it.

2. Reflex / Offset (`MIXING_OFFSET`)

Adds a constant bias to the surfaces.

- **Purpose:** Many flying wings require "Reflex" (upward elevator trim) to maintain stable flight. Using this parameter allows you to add that reflex without affecting the center-point of the PID loops.

Parameter Breakdown

- `MIXING_GAIN`: Output multiplier (typically 0.5 to 1.0).
- `MIXING_OFFSET`: Neutral bias (%).

Developer Notes

- **Library:** `ArduPlane/servos.cpp`.
- **Context:** Only used when `FRAME_TYPE` or `SERVOx_FUNCTION` indicates a mixed output (Elevon/VTail).

MIXING_OFFSET (Mixing Offset)

DEFAULT: 0	RANGE: -100 100	UNITS: %

Mixing Offset (MIXING_OFFSET)

Description

This parameter adds a fixed offset (bias) to the control surfaces when using V-Tail or Elevon mixing. It is primarily used to provide **Reflex** (upward elevator trim) on flying wings without reducing the available servo travel range, or to correct for minor build asymmetries.

The Mathematics

$$\text{Servo_out} = \text{Mixer}(\text{Pitch}, \text{Roll}) + \text{MIXING_OFFSET}$$

The Engineer's View

Defined in `ArduPlane/Parameters.cpp`.

- **Positive:** Typically adds up-elevator (reflex) on elevons.
- **Negative:** Adds down-elevator.

Tuning & Behavior

- **Default Value:** 0%
- **Flying Wings:** Often requires +5% to +10% reflex for stability.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





`Mixing Gain ()`

MIXING_GAIN: Mixing Gain

Description

The gain for the V-tail and elevon output mixers.

Values

- **Range:** 0.5 to 1.2
- **Default:** 0.5

Description

This parameter adjusts the servo travel limits when using software mixing (e.g., Flying Wings with Elevons or V-Tail planes).

- **Default (0.5):** Ensures that even if you apply full Pitch AND full Roll simultaneously, the servos will not hit their mechanical limits (saturate). `0.5 * Pitch + 0.5 * Roll = 1.0 (Full Throw)`.
- **Higher Values (e.g. 1.0):** Gives more authority to individual inputs but risks "clipping" if you apply both inputs at once. Useful if your control surfaces aren't moving enough to control the plane.

Source Code

ardupilot/ArduPlane/Parameters.cpp

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





MOUNT / GIMBAL CONFIGURATION (MNT)

Overview

The **MNT** parameter group configures the **Camera Mount** (Gimbal) subsystem. ArduPilot supports up to 2 mounts (`MNT1` and `MNT2`).

The mount system allows the autopilot to stabilize the camera against the vehicle's movement and point the camera at specific targets (ROI).

Key Concepts

1. Mount Types (`MNT_TYPE`)

Defines how the autopilot talks to the gimbal hardware.

- **1 (Servo):** PWM servos for Tilt, Roll, and Yaw. Stabilization is done by the flight controller.
- **3 (AlexMos):** Serial protocol for AlexMos/SimpleBGC gimbals.
- **4 (STorM32):** Serial protocol for STorM32 gimbals.
- **6 (DJI):** For DJI Zenmuse gimbals (via converter).
- **8 (DroneCAN):** CAN-based gimbals (e.g., Gremsy CAN).

2. Stabilization

If using a simple Servo gimbal (`Type=1`), ArduPilot can use its IMU data to keep the camera level.

- `MNT_LEAD_PTCH` / `RLL`: Predictive lag compensation.

3. Angles and Limits (`MNT_PITCH_MIN / MAX`)

Defines the mechanical range of motion of the gimbal.

- **Pitch -90:** Camera pointing straight down.
- **Pitch 0:** Camera pointing straight forward.

Parameter Breakdown

- `MNT1_TYPE`: Active driver.
- `MNT1_DEFLT_MODE`: Behavior at boot (Retracted, Neutral, MAVLink, RC Target).
- `MNT1_RC_RATE`: How fast the camera moves when you move the RC sticks (deg/s).

Integration Guide

1. **Hardware:** Mount the gimbal. If serial, connect to a Telem port.
2. **Enable:** Set `MNT1_TYPE`.
3. **Port:** (If serial) Set `SERIALx_PROTOCOL = 7` (AlexMos) or `8` (STorM32).
4. **Verify:** Tilt the drone; the camera should remain level relative to the horizon.

Developer Notes

- **Library:** `libraries/AP_Mount`.
- **Interaction:** Tightly integrated with the `CAM` group for automated photo triggering during survey missions.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Camera Mount instance ()

MNT_INST: Camera Mount instance

Description

Mount instance camera is associated with. 0 means camera and mount have identical instance numbers (e.g., Camera 1 → Mount 1).

Values

- **Range:** 0 to 2
- **Default:** 0

Description

This parameter links a configured Camera (`CAM` or `CAM2`) to a specific Gimbal/Mount (`MNT` or `MNT2`).

- **0:** Default mapping (Camera 1 uses Mount 1, Camera 2 uses Mount 2).
- **1:** This Camera uses Mount 1.
- **2:** This Camera uses Mount 2.

Source Code

ardupilot/libraries/AP_Camera/AP_Camera_Params.cpp

MNT1_DEFLT_MODE (Mount Default Mode (Instance 1))

DEFAULT: 3	RANGE: 0 6

Mount Default Mode (MNT1_DEFLT_MODE)

Description

`MNT1_DEFLT_MODE` sets the fallback behavior for the gimbal. It determines where the camera points when the drone first powers on, or when an autonomous mission ends and returns control to the pilot.

- **0: Retracted.** Gimbal moves to its stowed position.
- **1: Neutral.** Gimbal points straight forward.
- **2: MAVLink.** Waits for a command from the GCS.
- **3: RC Targeting.** (Default). The gimbal follows the position of the knobs/sliders on your transmitter.
- **4: GPS Point.** Points at a specific location on the map.

Tuning & Behavior

- **Default:** 3 (RC Targeting).
- **Recommendation:** Set to **3** if you want manual control of the camera tilt during flight. Set to **6 (Home Location)** if you want the camera to automatically track the takeoff point.

MNT1_DEVID (Mount Device ID (Instance 1))

DEFAULT: 0

Mount Device ID (MNT1_DEVID)

Description

`MNT1_DEVID` is a **Read-Only** diagnostic parameter that stores the unique hardware ID of the detected gimbal. It encodes the driver type, the communication bus (e.g. DroneCAN, I2C), and the bus address.

- **Role:** Verifies that the autopilot has successfully connected to the external gimbal hardware.
- **Usage:** If this is 0, the autopilot has failed to detect the gimbal specified in MNT1_TYPE.

Tuning & Behavior

- **Default:** 0.
- **Action:** No user adjustment is possible. This is used by the system to ensure it is talking to the correct device instance.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





MNT1_LEAD_PTCH (Mount Pitch Lead Time)

DEFAULT: 0	RANGE: 0 0.2	UNITS: s

Mount Pitch Lead Time (MNT1_LEAD_PTCH)

Description

`MNT1_LEAD_PTCH` compensates for the "Lag" in your gimbal servos.

When the drone pitches forward quickly, there is a small delay before the servo actually moves to counteract it. This can result in the horizon tilting momentarily in your video. This parameter tells the autopilot to "Predict" where the vehicle is going based on its current rotation speed and command the servo slightly ahead of time.

Tuning & Behavior

- **Default:** 0.
- **Recommendation:** Increase in small steps (e.g. **0.01s** or **0.02s**).
- **Optimal Setting:** The horizon should stay perfectly level even during fast pitch maneuvers.
- **Warning:** If you set this too high, the gimbal will "Overshoot" or vibrate during fast movements.

MNT1_LEAD_RLL (Mount Roll Lead Time)

DEFAULT: 0	RANGE: 0 0.2	UNITS: s

Mount Roll Lead Time (MNT1_LEAD_RLL)

Description

`MNT1_LEAD_RLL` compensates for mechanical and signal lag on your gimbal's roll axis.

It is similar to MNT1_LEAD_PTCH but for banking movements. If your video horizon "leans" during a turn and then slowly levels out, increasing this value will make the gimbal react faster to keep the camera flat.

Tuning & Behavior

- **Default:** 0.
- **Recommendation:** Start with **0.01s**.
- **Optimal Setting:** The horizon should stay perfectly horizontal even during aggressive "S-turns" or bank-and-yank maneuvers.

MNT1_OPTIONS (Mount Options (Instance 1))

DEFAULT: 0	RANGE: 0 1

Mount Options (MNT1_OPTIONS)

Description

`MNT1_OPTIONS` is a bitmask to enable specific Gimbal behaviors.

- **Bit 0 (1): Lock state from previous mode.** If you manually lock the gimbal's heading (using an RC switch) and then change the flight mode, the gimbal will stay locked in its current position rather than snapping back to its default orientation.

Tuning & Behavior

- **Default:** 0.
- **Usage:** Enable Bit 0 if you are performing cinematic shots where you want the camera to stay pointed at a subject even if the drone's autopilot behavior changes.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





MNT1_PITCH_MAX (Mount Pitch Maximum (Instance 1))

DEFAULT: 20	RANGE: -90 90	UNITS: deg

Mount Pitch Maximum (MNT1_PITCH_MAX)

Description

`MNT1_PITCH_MAX` defines the upper mechanical limit of your gimbal.

Setting this correctly prevents the gimbal from trying to tilt past its physical stop, which can cause motor overheating, "gimbal flip," or mechanical damage.

- **Default:** 20 degrees (pointing slightly above the horizon).
- **Recommendation:** Match this to your gimbal's physical capability.

Tuning & Behavior

- **Default:** 20.
- **Constraint:** This value must be greater than MNT1_PITCH_MIN.
- **Usage:** For mapping missions (straight down), you might set this to 0 to keep the camera strictly at or below the horizon.

MNT1_PITCH_MIN (Mount Pitch Minimum (Instance 1))

DEFAULT: -90	RANGE: -90 90	UNITS: deg

Mount Pitch Minimum (MNT1_PITCH_MIN)

Description

`MNT1_PITCH_MIN` defines the lower mechanical limit of your gimbal.

- **-90: Straight Down.** (Standard for mapping and FPV photography).
- **0: Horizon.** (Prevents the gimbal from looking down at the ground).

Tuning & Behavior

- **Default:** -90.
- **Action:** Ensure your gimbal can physically reach the value set here without straining the motors.

MNT1_RC_RATE (Mount RC Rate (Instance 1))

DEFAULT: 0	RANGE: 0 90	UNITS: deg/s

Mount RC Rate (MNT1_RC_RATE)

Description

`MNT1_RC_RATE` determines how fast your camera gimbal tilts or pans when you move the knobs on your transmitter.

- **Non-Zero Value (1-90):** Enables **Rate Control**. Moving the knob away from the center makes the gimbal start moving in that direction. The farther you move the knob, the faster it rotates (up to this limit). Returning the knob to the center stops the movement.
- **Zero (0):** Enables **Angle Control**. The gimbal's angle directly matches the position of the knob (e.g., knob at 10 o'clock = camera at -45 degrees).

Tuning & Behavior

- **Default:** 0 (Angle Control).
- **Recommendation:** Use **Rate Control** (e.g., 20 or 30 deg/s) for cinematic filming, as it allows for much smoother starts and stops. Use **Angle Control** (0) if you need the camera to instantly point where your switch/knob is located.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





MNT1_ROLL_MAX (Mount Roll Maximum (Instance 1))

DEFAULT: 30	RANGE: -180 180	UNITS: deg

Mount Roll Maximum (MNT1_ROLL_MAX)

Description

`MNT1_ROLL_MAX` sets the upper boundary for the gimbal's roll axis.

This is used to prevent the gimbal from hitting its physical stops when the drone is banking hard. It is particularly important for 2-axis gimbals or 3-axis gimbals with limited roll travel.

- **Units:** Degrees.
- **Default:** 30.

Tuning & Behavior

- **Default:** 30.
- **Constraint:** This value should match the physical limit of your gimbal. If the gimbal starts "stuttering" or making a high-pitched noise during a hard turn, your roll limits are likely too wide.

MNT1_ROLL_MIN (Mount Roll Minimum (Instance 1))

DEFAULT: -30	RANGE: -180 180	UNITS: deg

Mount Roll Minimum (MNT1_ROLL_MIN)

Description

`MNT1_ROLL_MIN` sets the lower boundary for the gimbal's roll axis.

This is the negative counterpart to MNT1_ROLL_MAX. It ensures the gimbal doesn't try to bank too far in the opposite direction.

- **Units:** Degrees.
- **Default:** -30.

Tuning & Behavior

- **Default:** -30.
- **Safety:** Always verify that your gimbal can physically reach these angles by manually banking the drone while it is powered on.

MNT1_SYSID_DFLT (Mount Target System ID)

DEFAULT: 0	RANGE: 0 255

Mount Target System ID (MNT1_SYSID_DFLT)

Description

`MNT1_SYSID_DFLT` specifies the "Network Address" of the gimbal.

In a MAVLink network (where multiple components like the autopilot, gimbal, and camera communicate), this ID allows the autopilot to send tilt/pan commands specifically to the correct gimbal controller.

- **0 (Default):** The autopilot will attempt to automatically find the first gimbal on the network.
- **1-255:** Manually locks the autopilot to a specific MAVLink system ID.

Tuning & Behavior

- **Default:** 0.
- **Usage:** Only set this if you have multiple gimbals (e.g. forward and downward) and the autopilot is consistently trying to control the wrong one.
- **Reboot Required:** Yes.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





MNT1_TYPE (Mount Type (Instance 1))

DEFAULT: 0	RANGE: 0 13

Mount Type (MNT1_TYPE)

Description

`MNT1_TYPE` enables gimbal control for your primary camera or payload. ArduPilot supports a wide variety of gimbals, from simple servos to high-end serial gimbals with integrated cameras.

- **0: None.** Gimbal control is disabled.
- **1: Servo.** For DIY gimbals controlled via PWM outputs on the flight controller.
- **3: Alexmos Serial.** For BaseCam/SimpleBGC controllers.
- **4: SToRM32 MAVLink.**
- **6: Gremsy.** Professional serial gimbals.
- **8: Siyi.** For Siyi ZR10/A8/ZR30 and similar integrated gimbals.
- **11: Viewpro.**

The Engineer's View

Defined in `AP_Mount`.
Enabling a mount type allocates the corresponding driver backend. The backend is responsible for receiving target angles from the autopilot (or RC passthrough) and commanding the hardware.

Tuning & Behavior

- **Default Value:** 0 (Disabled).
- **Reboot Required:** Yes. Mount drivers are allocated at boot.
- **Integration:** Once enabled, you must also configure the default mode (MNT1_DEFLT_MODE) and limits (MNT1_PITCH_MIN, etc.).

MNT1_YAW_MAX (Mount Yaw Maximum (Instance 1))

DEFAULT: 180	RANGE: -180 180	UNITS: deg

Mount Yaw Maximum (MNT1_YAW_MAX)

Description

`MNT1_YAW_MAX` defines how far the gimbal is allowed to rotate (pan) to the right.

- **For 360-degree Gimbals:** Set to 180 (allowing full rotation).
- **For Limited-travel Gimbals:** Set to the physical stop (e.g. 45 or 90 degrees).

Tuning & Behavior

- **Default:** 180.
- **Cable Safety:** Ensure that your camera cables can handle the rotation set here without being pulled out or tangled.

MNT1_YAW_MIN (Mount Yaw Minimum (Instance 1))

DEFAULT: -180	RANGE: -180 180	UNITS: deg

Mount Yaw Minimum (MNT1_YAW_MIN)

Description

`MNT1_YAW_MIN` defines how far the gimbal is allowed to rotate (pan) to the left.

This is the negative counterpart to MNT1_YAW_MAX.

- **Units:** Degrees.
- **Default:** -180 (Full left rotation).

Tuning & Behavior

- **Default:** -180.
- **Alignment:** Ensure that 0 degrees corresponds to the camera pointing directly forward along the vehicle's centerline.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





MNT2_DEFLT_MODE (Mount default operating mode)

DEFAULT: MAV_MOUNT_MODE_RC_TARGETING	RANGE: null

Mount default operating mode

Note: This parameter functions identically to MNT1_DEFLT_MODE.

MNT2_DEVID (Mount Device ID)

DEFAULT: 0	RANGE: null

Mount Device ID

Note: This parameter functions identically to MNT1_DEVID.

MNT2_LEAD_PTCH (Mount Pitch stabilization lead time)

DEFAULT: 0.0f	RANGE: 0.0 0.2	UNITS: s

Mount Pitch stabilization lead time

Note: This parameter functions identically to MNT1_LEAD_PTCH.

MNT2_LEAD_RLL (Mount Roll stabilization lead time)

DEFAULT: 0.0f	RANGE: 0.0 0.2	UNITS: s

Mount Roll stabilization lead time

Note: This parameter functions identically to MNT1_LEAD_RLL.

MNT2_OPTIONS (Mount options)

DEFAULT: 0	RANGE: null

Mount options

Note: This parameter functions identically to MNT1_OPTIONS.

MNT2_PITCH_MAX (Mount Pitch angle maximum)

DEFAULT: 20	RANGE: -90 90	UNITS: deg

Mount Pitch angle maximum

Note: This parameter functions identically to MNT1_PITCH_MAX.

MNT2_PITCH_MIN (Mount Pitch angle minimum)

DEFAULT: -90	RANGE: -90 90	UNITS: deg

Mount Pitch angle minimum

Note: This parameter functions identically to MNT1_PITCH_MIN.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





MNT2_RC_RATE (Mount RC Rate)

DEFAULT: 0	RANGE: 0 90	UNITS: deg/s

Mount RC Rate

Note: This parameter functions identically to MNT1_RC_RATE.

MNT2_ROLL_MAX (Mount Roll angle maximum)

DEFAULT: 30	RANGE: -180 180	UNITS: deg

Mount Roll angle maximum

Note: This parameter functions identically to MNT1_ROLL_MAX.

MNT2_ROLL_MIN (Mount Roll angle minimum)

DEFAULT: -30	RANGE: -180 180	UNITS: deg

Mount Roll angle minimum

Note: This parameter functions identically to MNT1_ROLL_MIN.

MNT2_SYSID_DFLT (Mount Target sysID)

DEFAULT: 0	RANGE: null

Mount Target sysID

Note: This parameter functions identically to MNT1_SYSID_DFLT.

MNT2_TYPE (Mount Type)

DEFAULT: 0	RANGE: null

Mount Type

Note: This parameter functions identically to MNT1_TYPE.

MNT2_YAW_MAX (Mount Yaw angle maximum)

DEFAULT: 180	RANGE: -180 180	UNITS: deg

Mount Yaw angle maximum

Note: This parameter functions identically to MNT1_YAW_MAX.

MNT2_YAW_MIN (Mount Yaw angle minimum)

DEFAULT: -180	RANGE: -180 180	UNITS: deg

Mount Yaw angle minimum

Note: This parameter functions identically to MNT1_YAW_MIN.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





MNT_NEUTRAL (Mount Neutral Position)

DEFAULT: 0	RANGE: -180 180	UNITS: deg

Mount Neutral Position (MNT_NEUTRAL)

Description

`MNT_NEUTRAL` defines the "Center" position of the gimbal.

This is used when the gimbal mode is set to **Neutral**. Typically, this is set to (0, 0, 0) so that the camera points straight ahead along the drone's longitudinal axis.

Tuning & Behavior

- **Default:** 0.
- **Usage:** If your gimbal is mounted slightly crooked, you can use these parameters to "Trim" the neutral position so that the camera is perfectly centered when the autopilot is in Neutral mode.

MNT_RETRACT (Mount Retracted Position)

DEFAULT: 0	RANGE: -180 180	UNITS: deg

Mount Retracted Position (MNT_RETRACT)

Description

`MNT_RETRACT` defines the "Parked" position of the gimbal.

This is used when the gimbal mode is set to **Retracted**. For many professional drones, this position points the camera inward or upward into a protective cavity to prevent the lens from being scratched or the motors from being strained during landing or transport.

Tuning & Behavior

- **Default:** 0 (Centered).
- **Usage:** Set the Roll, Pitch, and Yaw values to match your gimbal's specific stowing requirement.
- **Safety:** Ensure the retracted position does not cause the gimbal to hit the airframe or stretch internal cables.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **MODE** parameter group (specifically `MODE_CH`) configures the primary method for switching between vehicle behaviors.

Key Concepts

1. The Mode Channel

ArduPilot uses a designated RC input channel to select the active flight/drive mode.

- `MODE_CH`: Specifies the RC channel number (1-16).
  - **Default:** Usually 5 (for Copter/Plane) or 8 (for some Rover setups).
  - **Logic:** The PWM value of this channel is divided into 6 segments, which are mapped to the flight modes defined in the `FLTMODE_` (or `MODE1`..`MODE6`) parameters.

Integration Guide

- **Setup:** Ensure your RC transmitter is mixing switches to the channel specified in `MODE_CH`.
- **Verification:** Move the switches and verify the mode name changes in your Ground Control Station HUD.

Developer Notes

- **Library:** `RC_Channel::read_mode_switch()`.

MODE_CH (Mode channel)

`DEFAULT: MODE_CHANNEL`

Mode channel (MODE_CH)

Description

The **Mode Channel** parameter selects which Radio Control (RC) input channel is used to switch between flight/drive modes (e.g., Manual, Acro, Auto, Loiter) on the vehicle. This is a fundamental setup step that maps a physical switch on your transmitter to the vehicle's mode selection logic. The value corresponds to the RC channel number (1-16).

The Mathematics

The logic performs a direct mapping from the Pulse Width Modulation (PWM) value of the selected channel to a mode index.

$$\text{Selected Mode} = f(\text{PWM}(\text{MODE_CH}))$$

The PWM range (typically 1000us to 2000us) is divided into 6 segments, allowing for up to 6 different modes to be selected via a single channel (often mixed with switches on the transmitter).

The Engineer's View

This parameter is defined in `Rover/Parameters.cpp` as a `GSCALAR` (Global Scalar). It is read into the `mode_channel` variable. The `RC_Channel::read_mode_switch()` function checks the PWM input of the specified channel against standard thresholds (1230, 1360, 1490, 1620, 1750) to determine the active mode index (0-5).

Tuning & Behavior

- **Default Value:** 0 (Disabled) or vehicle specific default.
- **Range:** 0 - 16
- **0:** Disabled (Mode switching via RC is turned off; vehicle remains in initial mode or controlled via GCS).
- **1-16:** The RC input channel number to use for mode switching.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY

MOTOR CONFIGURATION & THRUST LOGIC (MOT)

Overview

The **MOT** parameter group is the core configuration for the **Motor Mixer** and **Thrust Linearization** system in multicopters, helicopters, and blimps.

This library translates the high-level PID demands (Roll, Pitch, Yaw, Throttle) into specific electrical signals sent to the ESCs. It also handles "Voltage Compensation," ensuring the drone feels the same at the end of a battery pack as it does at the start.

Key Concepts

1. PWM & Signal Type (`MOT_PWM_TYPE`)

Defines the communication protocol with the ESCs.

- **0:** Normal PWM (50-490Hz).
- **4:** DShot150.
- **6:** DShot600.

2. Spin Levels (`MOT_SPIN_ARM` / `MIN` / `MAX`)

- `SPIN_ARM`: The speed motors spin when armed but throttle is zero.
- `SPIN_MIN`: The lowest speed allowed in flight (ensures motors don't stall during aggressive descents).
- `SPIN_MAX`: Usually 0.95 (95%) to leave headroom for attitude control at full throttle.

3. Thrust Linearization (`MOT_THST_EXPO`)

Brushless motors do not produce linear thrust (50% PWM is not 50% thrust).

- `THST_EXPO`: A curve factor (0 to 1) that "flattens" the thrust response so the PID controllers have consistent authority across the entire throttle range.

4. Voltage Compensation (`MOT_BAT_VOLT_...`)

As battery voltage drops, the same PWM signal produces less thrust. The autopilot automatically increases the signal to compensate.

- `MOT_BAT_VOLT_MAX`: Voltage of a full pack.
- `MOT_BAT_VOLT_MIN`: Voltage of an empty pack.

Parameter Breakdown

- `MOT_THST_HOVER`: The learned or manual throttle value (0.0 - 1.0) required to maintain hover.
- `MOT_SLEWRATE`: Limits how fast the throttle can increase (prevents current spikes).
- `MOT_YAW_HEADROOM`: Reserves a portion of motor power for yaw control even at full throttle.

Integration Guide

1. **ESC Calibration:** Not needed for DShot. For PWM, must match `MOT_PWM_MIN` and `MAX`.
2. **Hover Tune:** Hover the vehicle in a stable mode (AltHold/Loiter). Check `MOT_THST_HOVER` in the logs and set it manually if `MOT_HOVER_LEARN` is disabled.
3. **Vibration:** If the vehicle is "jumpy," check if `MOT_THST_EXPO` is too high for your motor/prop combo.

Developer Notes

- **Library:** `libraries/AP_Motors`.
- **Hierarchy:** `AP_MotorsMatrix` (Multicopter), `AP_MotorsHeli` (Helicopter).

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





MOT1_DIRECTION (Motor 1 Normal or Reverse)

DEFAULT: 1	RANGE: -1 1

Motor 1 Normal or Reverse (MOT1_DIRECTION)

Description

`MOT1_DIRECTION` allows you to reverse the spin direction of Motor 1 via software. This is useful for "Sub" (Submarine) vehicles or Rovers where rewiring is difficult.

- **1: Normal.**
- **-1: Reverse.**

The Engineer's View

This parameter is primarily used by `AP_Motors6DOF` (Sub) and `AP_MotorsUGV` (Rover).
For Multicopters (Copter/QuadPlane), motor direction is usually fixed by the frame type and wiring, or reversed via DShot commands (`SERVO_BLH_REV`). This parameter might not have an effect on standard Copters unless using a specific frame class that supports it.

Tuning & Behavior

- **Default Value:** 1
- **Usage:** Change to -1 if the motor is spinning the wrong way and you cannot swap two phase wires.

MOT2_DIRECTION (Motor normal or reverse)

DEFAULT: 1	RANGE: null

Motor normal or reverse

Note: This parameter functions identically to MOT1_DIRECTION.

MOT3_DIRECTION (Motor normal or reverse)

DEFAULT: 1	RANGE: null

Motor normal or reverse

Note: This parameter functions identically to MOT1_DIRECTION.

MOT4_DIRECTION (Motor normal or reverse)

DEFAULT: 1	RANGE: null

Motor normal or reverse

Note: This parameter functions identically to MOT1_DIRECTION.

MOT5_DIRECTION (Motor normal or reverse)

DEFAULT: 1	RANGE: null

Motor normal or reverse

Note: This parameter functions identically to MOT1_DIRECTION.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





MOT6_DIRECTION (Motor normal or reverse)

DEFAULT: 1	RANGE: null

Motor normal or reverse

Note: This parameter functions identically to MOT1_DIRECTION.

MOT7_DIRECTION (Motor normal or reverse)

DEFAULT: 1	RANGE: null

Motor normal or reverse

Note: This parameter functions identically to MOT1_DIRECTION.

MOT8_DIRECTION (Motor normal or reverse)

DEFAULT: 1	RANGE: null

Motor normal or reverse

Note: This parameter functions identically to MOT1_DIRECTION.

MOT9_DIRECTION (Motor normal or reverse)

DEFAULT: 1	RANGE: null

Motor normal or reverse

Note: This parameter functions identically to MOT1_DIRECTION.

MOT_BAT_CURR_MAX (Motor Current Max)

DEFAULT: 0	RANGE: 0 200	UNITS: A

Motor Current Max (MOT_BAT_CURR_MAX)

Description

`MOT_BAT_CURR_MAX` is a safety limiter for your battery and PDB.

If the total current drawn by the motors exceeds this value, ArduPilot will actively reduce the maximum throttle to bring the current back down.

Tuning & Behavior

- **Default Value:** 0 (Disabled).
- **Recommendation:** Set to **80% of your battery's C-Rating** or your PDB's max current.
- **Safety:** This response is not instant (filtered by `MOT_BAT_CURR_TC`), so it won't stop a millisecond spike, but it will prevent sustained over-current.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





MOT_BAT_CURR_TC (Battery Current Time Constant)

DEFAULT: 5.0	RANGE: 0 10.0	UNITS: s

Battery Current Time Constant (MOT_BAT_CURR_TC)

Description

`MOT_BAT_CURR_TC` determines how quickly the autopilot reacts to an over-current situation.

ArduPilot has a "Current Limiter" that reduces throttle if the drone pulls more Amps than your battery or ESCs can handle. This parameter acts as a "Buffer" – a higher value allows brief current spikes (like a fast punch-out) without cutting power, while still protecting against sustained high current that could melt wires or blow a fuse.

Tuning & Behavior

- **Default:** 5.0 seconds.
- **High Value (e.g. 10.0):** Very slow reaction. Good for high-performance drones that need occasional bursts of power.
- **Low Value (e.g. 1.0):** Fast reaction. Best for protecting fragile batteries or electronics from any significant current surge.
- **Dependencies:** Only active if `MOT_BAT_CURR_MAX` is configured.

MOT_BAT_IDX (Battery Compensation Index)

DEFAULT: 0	RANGE: 0 9

Battery Compensation Index (MOT_BAT_IDX)

Description

`MOT_BAT_IDX` tells the autopilot which battery it should look at to decide how much to scale the motor power (compensation).

If you have multiple batteries with separate monitors (e.g., BATT1 for motors, BATT2 for electronics), you must set this to the index of the battery that is actually powering the ESCs/Motors.

- **0: Battery 1.** (Standard).
- **1: Battery 2.**
- **etc.**

Tuning & Behavior

- **Default:** 0.
- **Important:** If you set this to an index that is not enabled (e.g., set to 1 but only BATT1 is configured), the autopilot will not be able to perform voltage compensation, and your throttle feel may vary during the flight.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY

MOT_BAT_VOLT_MAX (Battery voltage compensation maximum voltage)

DEFAULT: 0	RANGE: 6 53	UNITS: V

Battery voltage compensation maximum voltage (MOT_BAT_VOLT_MAX)

Description

`MOT_BAT_VOLT_MAX` helps your drone feel consistent from the start to the end of a battery pack.

At 16.8V (full charge), your motors produce more thrust for 50% throttle than they do at 14.0V (empty). This parameter tells the autopilot what "Full Voltage" looks like so it can scale the throttle output down slightly at the start of the flight to match the performance at the end.

The Mathematics

$$\text{Scale Factor} = \frac{\text{Current Voltage}}{\text{MOT_BAT_VOLT_MAX}}$$

Tuning & Behavior

- **Default Value:** 0 (Disabled).
- **Recommendation:** Set to **4.2V * Cell Count**.
  - **4S:** 16.8V
  - **6S:** 25.2V
- **Benefit:** Makes your PID tuning valid for the entire flight, preventing "oscillations at takeoff" (due to high voltage) and "sluggishness at landing" (due to low voltage).

MOT_BAT_VOLT_MIN (Battery voltage compensation minimum voltage)

DEFAULT: 0	RANGE: 6 42	UNITS: V

Battery voltage compensation minimum voltage (MOT_BAT_VOLT_MIN)

Description

`MOT_BAT_VOLT_MIN` is the "Empty" reference for thrust scaling.

Tuning & Behavior

- **Default Value:** 0 (Disabled).
- **Recommendation:** Set to **3.3V * Cell Count**.
  - **4S:** 13.2V
  - **6S:** 19.8V

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY

MOT_BOOST_SCALE (Booster Motor Scale Factor)

DEFAULT: 0	RANGE: 0 5.0

Booster Motor Scale Factor (MOT_BOOST_SCALE)

Description

`MOT_BOOST_SCALE` is used for hybrid aircraft that have dedicated "Thrust" motors in addition to their lift motors (e.g. some Tailsitters or QuadPlanes).

It defines how much the booster motor responds when you push the throttle. The booster motor's output is calculated as:

$$\text{Booster Output} = \text{Main Throttle} \times \text{MOT_BOOST_SCALE}$$

- **0:** Disabled.
- **1.0:** The booster motor mirrors the throttle exactly.
- **> 1.0:** The booster motor provides more power than the main lift motors at the same stick position.

Tuning & Behavior

- **Default:** 0.
- **Usage:** Use this to balance your forward-flight speed against your climb-rate performance.

MOT_FV_CPLNG_K (6-DOF Pitch Decoupling Gain)

DEFAULT: 1.0	RANGE: 0 1.5

6-DOF Pitch Decoupling Gain (MOT_FV_CPLNG_K)

Description

`MOT_FV_CPLNG_K` is an advanced tuning parameter for **6-DOF** (Omnidirectional) multicopters.

In a traditional drone, moving forward requires pitching the nose down. In a 6-DOF drone, the motors are tilted so you can move forward while staying perfectly level. However, applying forward thrust often creates a slight, unintended "Nod" in the pitch axis due to the motor geometry. This parameter applies a mathematical correction to "cancel out" that coupling.

- **0:** Disabled.
- **1.2:** Typical "Normal" value for most 6-DOF frames.

Tuning & Behavior

- **Default:** 1.0.
- **Calibration:** Fly your 6-DOF drone and command a pure forward translation. If the drone pitches down as it accelerates, increase this value. If it pitches up, decrease it.
- **Context:** Only used if your motor layout is configured for 6-DOF control.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





MOT_HOVER_LEARN (Hover Value Learning)

DEFAULT: 2	RANGE: 0 2

Hover Value Learning (MOT_HOVER_LEARN)

Description

`MOT_HOVER_LEARN` allows the drone to figure out how much throttle is needed to hover and save that value.

- **0: Disabled.** Uses the manually set `MOT_THST_HOVER` and never changes it.
- **1: Learn.** Updates the hover estimate in RAM while flying, but **does not save** it to EEPROM on disarm. Good for testing payloads without messing up your base tune.
- **2: Learn and Save (Default).** Continuously updates the hover estimate and saves it to `MOT_THST_HOVER` when you disarm.

The Mathematics

The autopilot monitors the average throttle output while in "AltHold" or "Loiter" when the vertical velocity is near zero. It passes this through a slow low-pass filter to update the `MOT_THST_HOVER` value.

The Engineer's View

Knowing the correct hover throttle is critical for the Altitude Controller (`PSC_ACCZ_P` and `PSC_ACCZ_I`). If the hover throttle is wrong, the drone will jump up or drop down when engaging AltHold, and the altitude hold performance will be "bouncy".

Tuning & Behavior

- **Default Value:** 2 (Learn and Save)
- **Recommendation:** Leave at **2** for most users.
- **Payload Drones:** If you constantly swap between heavy payloads (camera) and no payload, set to **1** (Learn). This way, the drone learns the weight for the current flight, but reverts to the "base weight" setting when you reboot, preventing it from taking off like a rocket if you remove the payload but it "remembered" the heavy throttle settings.

MOT_PWM_FREQ (Motor PWM Frequency (Brushed))

DEFAULT: 16	RANGE: 1 20	UNITS: kHz

Motor PWM Frequency (MOT_PWM_FREQ)

Description

`MOT_PWM_FREQ` defines the switching frequency of the H-Bridge or ESC when controlling **Brushed** motors.

Brushed motors (common in smaller Rovers and Boats) require a high-frequency PWM signal to operate efficiently and quietly. If the frequency is too low, the motors will "sing" or "whine" audibly. If too high, the ESC may overheat due to switching losses.

- **Default:** 16 kHz.
- **Recommended: 16 kHz** to **20 kHz** for silent operation.

Tuning & Behavior

- **Default:** 16.
- **Note:** This parameter is only used if the vehicle is configured for Brushed motors. Standard brushless ESCs (using PWM/DShot) ignore this parameter.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





MOT_PWM_MAX (PWM output maximum)

DEFAULT: 2000	RANGE: 0 2000	UNITS: PWM

PWM output maximum (MOT_PWM_MAX)

Description

`MOT_PWM_MAX` defines the top of the throttle range sent to your ESCs.

- **For PWM ESCs:** This must match the calibrated maximum of your ESC. If set too low, you won't get full power. If set too high, you might hit the "Full Throttle" deadzone early.
- **For DShot:** Ignored for signal generation but used for scaling calculations.

Tuning & Behavior

- **Default Value:** 2000 µs.
- **Recommendation:** Perform an **ESC Calibration**.

MOT_PWM_MIN (PWM output minimum)

DEFAULT: 1000	RANGE: 0 2000	UNITS: PWM

PWM output minimum (MOT_PWM_MIN)

Description

`MOT_PWM_MIN` defines the bottom of the throttle range sent to your ESCs.

- **For PWM ESCs:** This must match the calibrated minimum of your ESC. If set too low, the motors might not start. If set too high, the motors might spin immediately upon arming (even if `MOT_SPIN_ARM` is 0).
- **For DShot:** This parameter is generally ignored for the signal itself (digital uses 0-2000 range internally), but ArduPilot still uses it for some internal scaling calculations.

Tuning & Behavior

- **Default Value:** 1000 µs.
- **Recommendation:** Perform an **ESC Calibration**. This will automatically detect the correct range.
- **Safety:** Ensure this is lower than `MOT_SPIN_ARM` + overhead to prevent accidental spins.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





MOT_PWM_TYPE (Output PWM type)

DEFAULT: 0	RANGE: 0 13

Output PWM type (MOT_PWM_TYPE)

Description

`MOT_PWM_TYPE` selects the communication protocol used to talk to the Electronic Speed Controllers (ESCs).

- **0: Normal (PWM).** Legacy analog protocol (1000-2000µs). Compatible with everything but slow.
- **4: DShot150.** Digital protocol. Robust, checksummed, no calibration needed.
- **5: DShot300.** Faster digital protocol. Good balance.
- **6: DShot600.** Fast digital protocol. Standard for modern racing/freestyle quads.
- **7: DShot1200.** Ultra-fast. Requires high-quality wiring and compatible ESCs.

The Mathematics

PWM: Analog pulse width modulation. Updates at ~400Hz.
DShot: Digital packet sending 16-bit commands (11-bit throttle + telemetry request + CRC).

- **DShot600:** 600,000 bits/sec. Frame time $\approx 26.7\mu s$.
- **DShot300:** 300,000 bits/sec. Frame time $\approx 53.4\mu s$.

The Engineer's View

Sets `_pwm_type` in `AP_Motors`.
Changing this usually requires a **Reboot** to reconfigure the timer hardware on the STM32.
If using DShot, you must also ensure your `SERVO_BLH_MASK` (or equivalent output mask) is configured correctly, although modern ArduPilot usually handles DShot on main outputs automatically if this param is set.

Tuning & Behavior

- **Default Value:** 0 (PWM)
- **Recommendation:**
  - **DShot600 (6):** Use this for almost all modern BLHeli_S, BLHeli_32, or AM32 ESCs. It enables features like "Turtle Mode", ESC Telemetry (RPM filtering), and digital precision.
  - **DShot300 (5):** Use if you have long signal wires or signal integrity issues with DShot600.
  - **Normal (0):** Use only for legacy SimonK/Melody ESCs or servos.

MOT_SAFE_DISARM (Motor PWM output disabled when disarmed)

DEFAULT: 0	RANGE: 0 1

Motor PWM output disabled when disarmed (MOT_SAFE_DISARM)

Description

`MOT_SAFE_DISARM` adds an extra layer of safety on the ground.

- **0 (Default):** The flight controller sends a "Min PWM" (e.g. 1000us) signal to the ESCs even when disarmed. This keeps the ESCs initialized and ready.
- **1 (Safe):** The flight controller stops sending *any* signal to the motors when disarmed.

The Engineer's View

Some ESCs will beep annoyingly if they don't receive a signal. However, stopping the signal ensures that a glitch (or a user bumping the throttle stick on a non-ArduPilot system) cannot spin the props.
For DShot, this parameter might stop the packet stream entirely.

Tuning & Behavior

- **Default Value:** 0.
- **Recommendation:** Leave at **0** for most ESCs to prevent them from entering "Programming Mode" or beeping.
- **Safety:** If your ESCs support it, setting this to **1** prevents accidental spin-ups.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





MOT_SAFE_TIME (Motor PWM Safety Transition Time)

DEFAULT: 0.1	RANGE: 0 5.0	UNITS: s

Motor PWM Safety Transition Time (MOT_SAFE_TIME)

Description

`MOT_SAFE_TIME` defines a short buffer period when the drone's output signal is transitioning between "Safe" (Disarmed) and "Live" (Armed) states.

This delay ensures that the ESCs and other hardware have a consistent signal before the drone attempts to apply flight power. It helps prevent "Twitches" or unexpected motor jumps the moment the arming switch is toggled.

Tuning & Behavior

- **Default:** 0.1 seconds.
- **Safety:** This is a low-level hardware safety parameter. It is recommended to leave it at the default setting unless you have specialized ESCs that require more time to synchronize.

MOT_SLEWRATE (Throttle Slew Rate)

DEFAULT: 100	RANGE: 0 1000	UNITS: %/s

Throttle Slew Rate (MOT_SLEWRATE)

Description

`MOT_SLEWRATE` acts as a "Smoothing" filter for your throttle.

It defines how fast the motor power can increase or decrease. A value of 100% means the motor can go from zero to full power in exactly 1 second. This is critical for heavy Rovers or vehicles with high-torque motors where a sudden "punch" of throttle could snap a drive belt, strip a gear, or cause a sudden voltage drop (brownout) that reboots the flight controller.

- **100:** (Default). 1 second to reach full power.
- **40:** 2.5 seconds to reach full power. Recommended for high-current vehicles with weak batteries.
- **0:** Disabled. No smoothing (instant response).

Tuning & Behavior

- **Default:** 100.
- **Mechanical Protection:** If your Rover "jerks" violently when you touch the throttle, reduce this value to **50** or **75**.
- **Electrical Stability:** If your flight controller reboots during hard acceleration, your battery cannot handle the current spike. Reducing this value will soften the ramp-up and may prevent the reboot.

MOT_SLEW_DN_TIME (Output slew time for decreasing throttle)

DEFAULT: 0	RANGE: 0 0.5	UNITS: s

Output slew time for decreasing throttle (MOT_SLEW_DN_TIME)

Description

`MOT_SLEW_DN_TIME` accounts for the "spool down" time of large props.

Active Braking (Damped Light) helps motors slow down fast, but large props still have momentum. This parameter prevents the PID loops from under-shooting when reducing throttle.

Tuning & Behavior

- **Default Value:** 0 (Disabled).
- **Recommendation:** Usually same as `MOT_SLEW_UP_TIME`.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





MOT_SLEW_UP_TIME (Output slew time for increasing throttle)

DEFAULT: 0	RANGE: 0 0.5	UNITS: s

Output slew time for increasing throttle (MOT_SLEW_UP_TIME)

Description

`MOT_SLEW_UP_TIME` prevents the flight controller from demanding impossible acceleration from the motors.

Large propellers (15"+) have significant inertia. If you command them to go from 10% to 90% throttle instantly, they physically can't do it. This causes the PID loop to "wind up" because it thinks the motor isn't responding. By telling ArduPilot how long the motor *actually* takes to spin up, the PID controller waits patiently, resulting in much better stability.

Tuning & Behavior

- **Default Value:** 0 (Disabled).
- **Recommendation:**
  - **5-10" Props:** 0.0 (Disabled).
  - **15-20" Props:** 0.15s.
  - **28"+ Props:** 0.3s.

MOT_SPD_SCA_BASE (Ground Speed Steering Scale Base)

DEFAULT: 1.0	RANGE: 0 10.0	UNITS: m/s

Ground Speed Steering Scale Base (MOT_SPD_SCA_BASE)

Description

`MOT_SPD_SCA_BASE` is a safety feature for Rovers and Boats that prevents high-speed rollovers or loss of control.

At low speeds (e.g., 1 m/s), you want 100% steering authority to make tight turns. At high speeds (e.g., 10 m/s), a full-lock steering command would cause the vehicle to flip or spin violently. This parameter defines the "Safe Speed" – above this speed, the autopilot automatically reduces the steering multiplier so that the vehicle remains stable.

Tuning & Behavior

- **Default:** 1.0 m/s.
- **Recommendation:** Set this to the maximum speed at which your vehicle can safely handle a full-lock turn.
- **Effect:** As speed increases above this base, the steering response becomes progressively "softer" or less sensitive.
- **0:** Disables speed-based steering scaling.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





MOT_SPIN_ARM (Motor Spin armed)

DEFAULT: 0.10	RANGE: 0.0 0.2

Motor Spin armed (MOT_SPIN_ARM)

Description

`MOT_SPIN_ARM` sets the idle speed of the motors when you arm the drone (even with the throttle stick at zero).

Crucial Function: This idle spin provides "Air Mode" authority. Even at zero throttle, the motors must spin fast enough so that if the autopilot needs to pitch or roll, the motors can respond instantly without stalling or desyncing.

The Mathematics

The throttle output sent to the ESC is:

$$\text{Idle Throttle} = \text{MOT_SPIN_ARM}$$

(Scaled by `MOT_PWM_MIN` and `MOT_PWM_MAX` if using PWM, or 0-1 range for DShot).

The Engineer's View

If set too low, motors may stop or stutter during zero-throttle maneuvers (like inverted hangs or rapid descents), causing loss of control ("Desync").
If set too high, the drone will be "floaty" and hard to land, or might even bounce on landing.

Tuning & Behavior

- **Default Value:** 0.10 (10%)
- **Range:** 0.0 - 0.2 (0% - 20%)
- **Recommendation:**
  - **5" Racing Quad:** 0.04 - 0.07 (4-7%). Small motors spin up easily.
  - **Large (15"+) Props:** 0.10 - 0.12 (10-12%). Heavy props need more torque to keep spinning reliably.
  - **Test:** Arm the drone (props OFF first to verify direction, then ON for idle check). Ensure all motors spin smoothly and don't stutter. Lower the value until they stutter, then add 0.02 or 0.03 safety margin.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY

MOT_SPIN_MAX (Motor Spin maximum)

DEFAULT: 0.95	RANGE: 0.9 1.0

Motor Spin maximum (MOT_SPIN_MAX)

Description

`MOT_SPIN_MAX` sets the upper limit for the throttle sent to the ESCs.

ArduPilot reserves the top few percent of the throttle range for the attitude controller. Even at full stick deflection, the average motor output is clamped to `MOT_SPIN_MAX` to ensure there is always "headroom" for the stabilization loop to increase the speed of one motor (to roll or pitch) without hitting the 100% saturation limit instantly.

The Mathematics

$$\text{Max Output} = \text{MOT_SPIN_MAX}$$

If `MOT_SPIN_MAX` is 0.95, the motors will never be commanded above 95% throttle for lift generation, leaving 5% headroom for yaw/roll/pitch authority at full throttle.

The Engineer's View

Used in `AP_MotorsMulticopter::output_to_motors()`.
Ideally, this matches the point where the thrust curve flattens out (diminishing returns).

Tuning & Behavior

- **Default Value:** 0.95 (95%)
- **Range:** 0.9 - 1.0
- **Recommendation:** Leave at **0.95**.
  - **0.90:** Safer for stability at full throttle, but slight loss of top speed.
  - **1.00:** Disables headroom. Not recommended, as the drone may lose yaw/roll authority during a full-throttle punch-out (unless `MOT_YAW_HEADROOM` saves it).

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY

MOT_SPIN_MIN (Motor Spin minimum)

DEFAULT: 0.15	RANGE: 0.0 0.3

Motor Spin minimum (MOT_SPIN_MIN)

Description

`MOT_SPIN_MIN` sets the lower limit for the throttle during active flight. Even if you cut the throttle stick to zero (while armed and not landed), the motors will not spin slower than this value.

Critical Safety Function: This prevents the motors from stalling or desyncing during aggressive maneuvers or rapid descents. If the props spin too slowly in the airflow, the ESC might lose synchronization, causing the motor to stop and the drone to tumble.

The Mathematics

The throttle output sent to the ESC is clamped:

$$\text{Output} \geq \text{MOT_SPIN_MIN}$$

(Scaled by PWM/DShot range).

The Engineer's View

Must be higher than `MOT_SPIN_ARM`.
Ideally, this is set to the lowest RPM where the motor still produces reliable, non-stuttering thrust under load (prop wash).

Tuning & Behavior

- **Default Value:** 0.15 (15%)
- **Range:** 0.0 - 0.3 (0% - 30%)
- **Recommendation:**
  - **5" Racing Quad:** 0.07 - 0.10 (7-10%).
  - **Large (15"+) Props:** 0.15 - 0.20 (15-20%). Large props have high inertia and stall easily in turbulent air.
  - **Rule of Thumb:** Set `MOT_SPIN_MIN` = `MOT_SPIN_ARM` + 0.03 (at least 3% higher).
- **Symptoms of Low Setting:** "Desyncs" or death-rolls at the bottom of a dive or flip.
- **Symptoms of High Setting:** Drone won't descend quickly (floaty).

MOT_SPOOL_TIME (Motor Spool-Up Time)

DEFAULT: 0.5	RANGE: 0.05 2.0	UNITS: s

Motor Spool-Up Time (MOT_SPOOL_TIME)

Description

`MOT_SPOOL_TIME` defines how "Soft" the initial motor start is.

When you arm the drone or first apply throttle, ArduPilot doesn't just "slam" the motors to speed. Instead, it ramps the power up gradually over this many seconds. This prevents sudden torque spikes that could flip a drone on the ground or put unnecessary stress on the propellers and frame.

- **Standard Setting:** 0.5 seconds.
- **Large Props:** May need 1.0s or more to ensure a smooth, synchronized start without stalling.

Tuning & Behavior

- **Default:** 0.5.
- **Recommendation:** If your drone "hops" or feels nervous the moment it arms, increase this value.
- **Note:** This parameter also affects the internal state machine that transitions the drone from "On the ground" to "In flight" logic.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





MOT_SPOOL_TIM_DN (Motor Spool-Down Time)

DEFAULT: 0	RANGE: 0 2.0	UNITS: s

Motor Spool-Down Time (MOT_SPOOL_TIM_DN)

Description

`MOT_SPOOL_TIM_DN` defines the duration of the motor ramp-down when the drone lands or is disarmed.

- **0 (Default):** Uses the same value as MOT_SPOOL_TIME.
- **Non-Zero:** Sets a specific time for the ramp-down.

Tuning & Behavior

- **Recommendation:** Keep at **0** to match the spool-up time.
- **Safety:** This ensures the motors don't stop instantly, which can be useful to prevent large props from unscrewing themselves due to sudden deceleration (on some older hardware) or simply to provide a more predictable disarm feel.

MOT_STR_THR_MIX (Steering vs Throttle Priority)

DEFAULT: 0.5	RANGE: 0.1 1.0

Steering vs Throttle Priority (MOT_STR_THR_MIX)

Description

`MOT_STR_THR_MIX` defines what happens when your Rover is "Maxed Out."

On a **Skid-Steer** rover (like a tank), the motors are used for both turning and driving forward. If you command full speed AND a hard turn, the autopilot may run out of motor power. This parameter tells the autopilot which one is more important:

- **Higher Value (0.8 - 1.0):** Prioritizes **Steering**. The rover will slow down to ensure it can make the turn accurately. (Recommended for precise autonomous missions).
- **Lower Value (0.2 - 0.5):** Prioritizes **Throttle**. The rover will keep its speed up, even if it has to widen the turn. (Recommended for high-speed racing).

Tuning & Behavior

- **Default:** 0.5 (Balanced).
- **Recommendation:** If your rover "Overshoots" corners while trying to maintain speed, increase this value to **0.7** or **0.8**.
- **Note:** This only applies to vehicles with skid-steering (tank mode) enabled.

MOT_THR_MAX (Throttle Maximum (Rover/Boat))

DEFAULT: 100	RANGE: 30 100	UNITS: %

Throttle Maximum (MOT_THR_MAX)

Description

`MOT_THR_MAX` sets the "Speed Governor" for your Rover or Boat.

Even if you push the transmitter stick to 100%, the autopilot will limit the actual power sent to the motors to this percentage. This is useful for high-powered vehicles to prevent overheating motors and ESCs, or to make a fast vehicle safer for beginners.

- **100:** (Default). Full power available.
- **50:** Half power limit.

Tuning & Behavior

- **Default:** 100.
- **Usage:** If your motors get too hot to touch after a short run, reduce this value to **70** or **80**.
- **Note:** This only applies to Rover and Boat. For Multicopters, use `MOT_SPIN_MAX`.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





MOT_THR_MIN (Throttle Minimum (Rover/Boat))

DEFAULT: 0	RANGE: 0 20	UNITS: %

Throttle Minimum (MOT_THR_MIN)

Description

`MOT_THR_MIN` sets the "Idle" limit for your Rover or Boat's powertrain.

This is particularly useful for vehicles with Internal Combustion Engines (Gasoline/Nitro), where setting a minimum throttle ensures the engine doesn't stall when the autopilot commands zero speed. It can also be used for electric vehicles to overcome static friction (stiction) in a drivetrain.

- **0:** (Default). No minimum limit.
- **10:** Autopilot will never command less than 10% throttle unless disarmed.

Tuning & Behavior

- **Default:** 0.
- **Usage:** For ICE vehicles, set this to the percentage that maintains a reliable idle. For electric rovers that won't start moving until 5% throttle, set this to **5** to improve responsiveness.
- **Note:** This only applies to Rover and Boat. For Multicopters, use `MOT_SPIN_ARM`.

MOT_THST_ASYM (Motor Thrust Asymmetry)

DEFAULT: 1.0	RANGE: 1.0 10.0

Motor Thrust Asymmetry (MOT_THST_ASYM)

Description

`MOT_THST_ASYM` is a calibration factor for **Skid-Steer** (tank-tread style) vehicles.

Most electric motors and propellers/tires are significantly more efficient when spinning forward than backward. If you apply the same PWM signal to both directions, the vehicle might move much slower in reverse than it does forward. This parameter allows the autopilot to "Boost" the reverse power so that the vehicle's speed and steering response remain consistent in both directions.

- **1.0 (Default):** Symmetric. Forward and reverse thrust are treated as equal.
- **2.0:** High Asymmetry. Autopilot will command twice as much signal to achieve the same thrust in reverse.

Tuning & Behavior

- **Calibration:** Drive your rover in a straight line at a constant speed forward, then reverse. If reverse is much slower, increase this value until the speeds match.
- **Note:** This only applies to Rover and Boat frames with skid-steering enabled.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





MOT_THST_EXPO (Thrust Curve Expo)

DEFAULT: 0.65	RANGE: -1.0 1.0

Thrust Curve Expo (MOT_THST_EXPO)

Description

`MOT_THST_EXPO` linearizes the relationship between the autopilot's requested thrust and the actual physical thrust produced by the motors.

Most brushless motors/propellers produce thrust roughly proportional to the **square** of RPM, and RPM is roughly proportional to throttle signal. This means 50% throttle signal might only produce 25% thrust. This nonlinearity confuses the PID controller (which assumes linear response). `MOT_THST_EXPO` corrects this.

The Mathematics

The conversion from desired thrust ($T_des \in [0, 1]$) to throttle signal ($Out \in [0, 1]$) is:

$$Out = \frac{(1 - k)T_des}{1 - kT_des}$$

where $k = \text{MOT_THST_EXPO}$.

If $k = 0$, $Out = T_des$ (Linear).
If $k \approx 1$, the curve corrects for the square-law thrust.

The Engineer's View

Used in `AP_MotorsMulticopter::output_to_motors()`.
Correct setting makes the PID gains effective across the entire throttle range.

- If set too **low**: Drones will oscillate at high throttle (over-gained) and feel sluggish at low throttle.
- If set too **high**: Drones will oscillate at low throttle and feel sluggish at high throttle.

Tuning & Behavior

- **Default Value:** 0.65
- **Range:** -1.0 to 1.0 (Typical range for multicopters is 0.5 to 0.8).
- **Recommendation:**
  - **5" Racing Quad:** 0.55 - 0.65
  - **10-15" Props:** 0.65 - 0.70
  - **Large (18"+) Props:** 0.70 - 0.75
  - **Flame/T-Motor ESCs:** These often have internal linearization. If so, set this to **0**. Check your ESC manual.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





MOT_THST_HOVER (Thrust Hover Value)

DEFAULT: 0.35	RANGE: 0.125 0.6875

Thrust Hover Value (MOT_THST_HOVER)

Description

`MOT_THST_HOVER` stores the throttle level (0.0 to 1.0) required to keep the drone stationary in the air.

This value is the foundation for the entire altitude controller. The autopilot assumes that `MOT_THST_HOVER` provides exactly 1G of lift. The PIDs then calculate adjustments *relative* to this baseline.

The Mathematics

$$\text{Throttle Output} = \text{MOT_THST_HOVER} + \text{PID}(\text{Altitude Error})$$

If this value is too low, the P-term has to work constantly to add thrust, causing the drone to sag when entering AltHold.
If too high, the drone jumps up when entering AltHold.

The Engineer's View

Used in `AP_MotorsMulticopter::set_throttle_hover()`.
Typically updated automatically if `MOT_HOVER_LEARN` is 2.
Range is clamped between `MOT_SPIN_MIN` and `MOT_SPIN_MAX`.

Tuning & Behavior

- **Default Value:** 0.35 (35%)
- **Range:** 0.125 - 0.6875
- **Recommendation:**
  - **Let it learn:** Set `MOT_HOVER_LEARN = 2`, fly in Loiter for 30 seconds, land, disarm. This value will update itself.
  - **Manual Setting:** Hover in Stabilize mode. Note the throttle stick position (e.g., 40%). Set `MOT_THST_HOVER` to 0.40.
  - **Optimal Range:** A well-designed multicopter hovers between **0.25 and 0.50**. If hovering > 0.65, the drone is underpowered. If hovering < 0.15, it is overpowered (and will be hard to land).

MOT_VEC_ANGLEMAX (Vector Thrust Angle Max)

DEFAULT: 0	RANGE: 0 90	UNITS: deg

Vector Thrust Angle Max (MOT_VEC_ANGLEMAX)

Description

`MOT_VEC_ANGLEMAX` defines the mechanical limit of your boat's steerable motor.

Unlike a rudder, which is a passive fin, **Vectored Thrust** means the entire motor or propeller unit rotates to steer the boat. This parameter tells the autopilot how many degrees of physical rotation your motor mount can achieve.

Tuning & Behavior

- **Default:** 0.
- **Action:** Measure the maximum angle your motor can physically pivot to the left or right. If the motor can turn 30 degrees in each direction, set this to **30**.
- **Note:** This is only used for Boat frames where the motor itself is the steering actuator.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





MOT_YAW_HEADROOM (Matrix Yaw Min)

DEFAULT: 200	RANGE: 0 500	UNITS: PWM

Matrix Yaw Min (MOT_YAW_HEADROOM)

Description

`MOT_YAW_HEADROOM` ensures you can always steer, even at full throttle.

When you punch the throttle to 100%, the motors are maxed out. If you try to yaw (spin) at that moment, the drone normally couldn't do it because it can't speed up the motors any further. This parameter forces the average throttle down slightly to keep a "reserve" for yaw.

Tuning & Behavior

- **Default Value:** 200 (approx 20% of range).
- **Recommendation:** Leave at **200**.
- **Race Quads:** You might reduce this to **50 or 100** to get more raw climb power, at the cost of yaw authority during punch-outs.

MOT_YAW_SV_ANGLE (Tricopter Yaw Servo Max Angle)

DEFAULT: 30	RANGE: 5 80	UNITS: deg

Tricopter Yaw Servo Max Angle (MOT_YAW_SV_ANGLE)

Description

`MOT_YAW_SV_ANGLE` defines the mechanical limit of the tilt mechanism on your Tricopter's tail motor.

Tricopters use a single tilting motor for yaw control. This parameter tells the autopilot how many degrees that motor can physically tilt to the left or right. This is used to calibrate the relationship between the requested "Yaw Force" and the physical servo movement.

Tuning & Behavior

- **Default:** 30 degrees.
- **Measurement:** Tilt your tail servo to its maximum physical limit. Use a protractor to measure the angle from the vertical. Enter that value here.
- **Significance:** If this value is incorrect, your Tricopter's yaw gains will be off, leading to sluggish or oscillating yaw behavior.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

Key Concepts

MOT10_DIRECTION (Motor normal or reverse)

DEFAULT: 1	RANGE: null

Motor normal or reverse

Note: This parameter functions identically to MOT1_DIRECTION.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

Key Concepts

MOT11_DIRECTION (Motor normal or reverse)

DEFAULT: 1	RANGE: null

Motor normal or reverse

Note: This parameter functions identically to MOT1_DIRECTION.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

Key Concepts

MOT12_DIRECTION (Motor normal or reverse)

DEFAULT: 1	RANGE: null

Motor normal or reverse

Note: This parameter functions identically to MOT1_DIRECTION.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **MSG** parameter group (specifically `MSG_TIME` or `OSD_MSG_TIME`) manages the visibility of status notifications.

Key Concepts

1. Message Persistence

When the autopilot triggers a status message (e.g., "Armed," "RTL Engaged," or "Low Battery"), it is displayed as a pop-up on the On-Screen Display (OSD).

- `MSG_TIME`: Sets how long (in seconds) the text remains on screen before fading away.
  - **Recommendation:** 5-10 seconds is standard. Longer values ensure you don't miss warnings, but may block other telemetry.

Developer Notes

- **Library:** `libraries/AP_OSD`.
- **Context:** Only relevant if an analog or digital OSD is active.

Message display duration in seconds ()

MSG_TIME: Message display duration in seconds

Description

Sets the duration (in seconds) that temporary status messages and warnings remain visible on the OSD.

Values

- **Range:** 1 to 20
- **Units:** s
- **Default:** 10
- **Increment:** 1

Description

This parameter controls the persistence of "pop-up" messages on the On-Screen Display.

- **Examples:** Flight mode changes ("STABILIZE"), Arm/Disarm notifications, Battery warnings ("LOW BATT"), and GCS text messages.
- **Tuning:**
  - **Too Short:** You might miss important warnings while flying.
  - **Too Long:** The message might obscure other critical flight data (like the horizon or battery voltage) for an annoying amount of time.

Source Code

ardupilot/libraries/AP_OSD/AP_OSD.cpp

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **MSP** parameter group configures the **Multiwii Serial Protocol** (MSP) interface. While originally developed for the MultiWii firmware, MSP has become the standard communication protocol for modern digital FPV systems (DJI, Avatar, HDZero) to receive OSD telemetry from the autopilot.

Key Concepts

1. Digital OSD Support

Digital VTX modules (like the DJI Air Unit) do not support the old analog "Overlay" method. Instead, they request data from the autopilot via MSP and render the OSD elements themselves within the goggles.

2. MSP DisplayPort

This is an extension of MSP used by HDZero and Avatar to provide high-definition, pixel-accurate OSD rendering that matches the look of Betaflight or ArduPilot's native OSD.

Parameter Breakdown

- `MSP_OPTIONS`: Bitmask for specific behaviors (e.g., enable DisplayPort mode).
- `MSP_OSD_NCELLS`: Manual cell count for digital OSDs that can't auto-calculate.

Integration Guide

1. **Hardware:** Connect the TX/RX pins of your Digital VTX to a UART on the flight controller.
2. **Serial:** Set `SERIALx_PROTOCOL = 33` (MSP).
3. **Config:** If using DJI (original V1/V2), no further config is needed. If using Avatar/HDZero, set `MSP_OPTIONS` to enable DisplayPort.

Developer Notes

- **Library:** `libraries/AP_MSP`.

MSP Serial Port ()

MSP_PORT: MSP Serial Port

Description

This is the serial port number where `SERIALx_PROTOCOL` will be set to MSP (MultiWii Serial Protocol).

Values

- **Range:** 0 to 10
- **Increment:** 1
- **Default:** 1

Description

This parameter automatically configures a specific hardware UART (Serial Port) to speak the **MSP** protocol.

- **Purpose:** Used for integrating with peripherals that primarily speak MSP, such as DJI FPV Goggles (for OSD data) or certain ESC telemetry systems.
- **Auto-Configuration:** Setting this parameter effectively overrides the manual `SERIALx_PROTOCOL` setting for the selected port, ensuring the correct driver is loaded for MSP communication.

Source Code

ardupilot/Tools/AP_Periph/Parameters.cpp

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





MSP_OPTIONS (MSP OSD Options)

DEFAULT: 0	RANGE: 0 15

MSP OSD Options (MSP_OPTIONS)

Description

`MSP_OPTIONS` provides compatibility settings for digital video systems (like DJI, FatShark Avatar, and HDZero) that use the **MSP** protocol for their On-Screen Display.

Since digital OSDs often have their own internal font sets and communication styles, this parameter ensures the flight controller sends data in a format the goggles can understand.

- **Bit 0 (1): EnableTelemetryMode.** Allows "push" mode telemetry if only the RX line is connected.
- **Bit 1 (2): EnableBTFLFonts.** Use character mapping compatible with Betaflight-style fonts.
- **Bit 2 (4): EnableINAVFonts.** Use character mapping compatible with INAV-style fonts.

Tuning & Behavior

- **Default:** 0.
- **Usage:** If your OSD icons (like the battery or GPS symbols) look like random letters or garbled blocks, try toggling the **BTFL** or **INAV** font bits to match your goggle's firmware.

MSP_OSD_NCELLS (MSP OSD Cell Count Override)

DEFAULT: 0	RANGE: 0 14

MSP OSD Cell Count Override (MSP_OSD_NCELLS)

Description

`MSP_OSD_NCELLS` allows you to manually set the cell count for the battery voltage display on digital OSDs.

While ArduPilot can usually detect cell count automatically, some digital goggles require an explicit cell count to calculate and display "Average Cell Voltage" correctly.

- **0:** Use global battery settings (Automatic detection).
- **1-14:** Forces the OSD to assume a specific number of cells (e.g. 4 for a 4S battery).

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **NAVL** parameter group configures the **L1 Navigation Controller**. This is the core horizontal path-following algorithm used by **ArduPlane** and **ArduRover**.

The L1 controller is responsible for steering the vehicle to follow a straight line between waypoints or to maintain a perfect orbit during a loiter. It is known for its ability to handle high-speed flight and varying ground speeds (wind) with high precision.

Key Concepts

1. The L1 Distance

The algorithm "looks ahead" along the path to a point at a specific distance (the L1 distance). It then calculates the lateral acceleration required to steer the vehicle toward that look-ahead point.

- `NAVL1_PERIOD`: This is the primary tuning parameter. It defines the "Time" it should take to correct a path error.
  - **Higher Period (e.g., 25s):** Softer, lazier tracking. Good for stable, slow planes.
  - **Lower Period (e.g., 15s):** Aggressive, sharp tracking. Required for fast planes or rovers that need to stick to a tight path.

2. Damping (`NAVL1_DAMPING`)

Adjusts the "settling" behavior of the controller.

- **0.75 (Default):** Standard critical damping.
- **Increase:** If the vehicle overshoots the line and then oscillates.

Parameter Breakdown

- `NAVL1_PERIOD`: Path following aggression (seconds).
- `NAVL1_XTRACK_I`: Integrator gain for correcting persistent crosstrack errors (e.g., in a strong crosswind).
- `NAVL1_LIM_BANK`: Max bank angle used for navigation.

Integration Guide

- **Rover Tuning:** Rovers typically require a much lower `NAVL1_PERIOD` (e.g., 5-8s) than planes to handle sharp corners.
- **Plane Tuning:** If your plane "S-turns" (weaves) back and forth across the mission line, **increase** the `NAVL1_PERIOD`.

Developer Notes

- **Library:** `libraries/AP_L1_Control`.
- **Algorithm:** Nonlinear guidance law that produces a centripetal acceleration command.

NAVL1_DAMPING (L1 Navigation Damping)

DEFAULT: 0.75	RANGE: 0.6 1.0

L1 Navigation Damping (NAVL1_DAMPING)

Description

`NAVL1_DAMPING` prevents the aircraft from overshooting the path when it tries to intercept a line.

It defines the "Sharpness" of the turn into the path. A value of **0.75** is standard for most aircraft. If the plane consistently "misses" the line and has to turn back to find it, increasing this value can help.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





NAVL1_LIM_BANK (Loiter Radius Bank Angle Limit)

DEFAULT: 0.0f	RANGE: 0 89	UNITS: deg

Loiter Radius Bank Angle Limit (NAVL1_LIM_BANK)

Description

This parameter enforces a structural or aerodynamic limit on the bank angle during loiters, normalized to sea-level air density. As altitude increases, True Airspeed (TAS) increases for the same Indicated Airspeed (IAS), which naturally increases the turn radius. This parameter ensures that the bank angle required to maintain a specific loiter radius does not exceed the airframe's loading limits.

The Mathematics

$$R_\text{turn} = \frac{V_TAS^2}{g \cdot \tan(\phi)}$$

If `NAVL1_LIM_BANK` is set > 0, the navigation controller calculates the maximum allowable bank angle based on air density and scales the loiter radius `WP_LOITER_RAD` upwards if necessary to stay within this bank limit.

The Engineer's View

Defined in `libraries/AP_L1_Control/AP_L1_Control.cpp`. It is a safety feature for high-altitude or high-speed operations.

- **0:** Disabled. The standard roll limits apply, and the vehicle attempts to fly the requested `WP_LOITER_RAD`.
- **>0:** The L1 controller dynamically increases the loiter radius to prevent banking more than this value (at equivalent sea-level loading).

Tuning & Behavior

- **Default Value:** 0 (Disabled)
- **Typical:** 0 for most users. 45-60 degrees for high-performance aircraft to prevent stall or structural failure in tight turns at altitude.

NAVL1_PERIOD (L1 Navigation Period)

DEFAULT: 17	RANGE: 1 60	UNITS: s

L1 Navigation Period (NAVL1_PERIOD)

Description

`NAVL1_PERIOD` is the primary tuning knob for how your plane or rover follows a line.

ArduPilot's **L1 controller** works by picking a point on the path ahead of the vehicle and steering toward it. This parameter effectively sets the distance to that point.

- **Lower Value (e.g. 10):** Very aggressive tracking. The drone will stick to the line tightly and make sharp turns.
- **Higher Value (e.g. 25):** Soft, smooth tracking. The drone will wander more from the line but the flight will be much more stable.

Tuning & Behavior

- **Default:** 17.
- **Recommendation:** For small agile planes, use **12 to 15**. For large, slow planes, use **20 to 25**.
- **Warning:** If you set this too low, the aircraft will oscillate (weaving) rapidly across the path.

NAVL1_XTRACK_I (L1 Cross-Track Integrator)

DEFAULT: 0.02	RANGE: 0 1.0

L1 Cross-Track Integrator (NAVL1_XTRACK_I)

Description

`NAVL1_XTRACK_I` corrects for "Crabbing" caused by wind.

If there is a strong crosswind, a plane might fly parallel to the path but a few meters to one side. This Integrator term slowly builds up over time to force the plane back onto the exact center of the line.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **NET** parameter group configures the **Ethernet** and **Networking** subsystem for high-end flight controllers (e.g., Pixhawk 5X, 6X, Orange Cube Plus with Ethernet Carrier).

This system allows the autopilot to communicate over a standard Local Area Network (LAN) using TCP or UDP, providing extremely high bandwidth compared to traditional serial telemetry.

Key Concepts

1. Static vs. Dynamic IP (`NET_DHCP`)

- **DHCP (1):** Autodetect IP address from a router.
- **Static (0):** Use the IP addresses manually defined in `NET_IPADDR0` ... `3`.

2. Network Ports (`NET_Pn_...`)

ArduPilot supports multiple concurrent network connections (Sockets).

- `NET_P1_TYPE`: 0=UDP Client, 1=UDP Server, 2=TCP Client, 3=TCP Server.
- `NET_P1_PORT`: The network port number (e.g., 14550 for MAVLink).
- `NET_P1_PROTOCOL`: Selects the protocol (1=MAVLink, 5=GPS, etc.).

3. MAC Address Configuration

Allows overriding the hardware MAC address for network environments with strict MAC filtering.

Parameter Breakdown

- `NET_ENABLE`: Master switch.
- `NET_IPADDRx`: Static IP configuration.
- `NET_NETMASK`: Network mask (usually 24 for 255.255.255.0).

Integration Guide

1. **Hardware:** Plug the autopilot into an Ethernet switch or companion computer.
2. **Enable:** Set `NET_ENABLE = 1`.
3. **Configure Port:** Set `NET_P1_TYPE = 1` (UDP Server) and `NET_P1_PORT = 14550`.
4. **Connect:** Open Mission Planner and select "UDP" with port 14550.

Developer Notes

- **Library:** `libraries/AP_Networking`.
- **Performance:** Ethernet can handle much higher logging and telemetry rates (100Hz+) without saturation.

NET_DHCP (DHCP Enable)

DEFAULT: 1	RANGE: 0 1

DHCP Enable (NET_DHCP)

Description

`NET_DHCP` tells the autopilot to ask your network router for an IP address automatically.

- **0 (Static):** You must manually set `NET_IPADDR`, `NET_NETMASK`, and `NET_GWADDR`. Use this for point-to-point connections (e.g. drone directly to laptop).
- **1 (DHCP):** The drone will get its settings from the router. Use this if the drone is plugged into a home/office network.

Tuning & Behavior

- **Default Value:** 1.
- **Recommendation:** Use **0 (Static)** for mission-critical telemetry to ensure the IP address never changes.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





NET_ENABLE (Networking Enable)

DEFAULT: 0	RANGE: 0 1

Networking Enable (NET_ENABLE)

Description

`NET_ENABLE` activates the IP (Internet Protocol) communications stack on the flight controller.

If your flight controller has an Ethernet port (like the Cube Orange with an ADS-B carrier board or the Holybro Pixhawk 6X), this parameter must be set to 1 to enable MAVLink over UDP or TCP.

Tuning & Behavior

- **0:** Disabled.
- **1:** Enabled.
- **Reboot Required:** Yes.

NET_IPADDR0 (IPv4 Address Byte 0)

DEFAULT: 0	RANGE: 0 255

IPv4 Address Byte 0 (NET_IPADDR0)

Description

`NET_IPADDR0` is the first part of your drone's static network address.

If you are using an Ethernet-capable flight controller (like the Cube Orange with an Ethernet carrier) and have disabled DHCP (`NET_DHCP = 0`), you must manually set the IP address using the `NET_IPADDR` parameters.

- **Example:** For an IP of **192**.168.1.50, set `NET_IPADDR0` to **192**.

Tuning & Behavior

- **Default Value:** 0.
- **Reboot Required:** Yes.
- **Constraint:** Only active if `NET_DHCP` is set to 0.

NET_IPADDR1 (IPv4 Address Byte 1)

DEFAULT: 0	RANGE: 0 255

IPv4 Address Byte 1 (NET_IPADDR1)

Description

`NET_IPADDR1` is the second part of your drone's static network address.

- **Example:** For an IP of 192.**168**.1.50, set `NET_IPADDR1` to **168**.

NET_IPADDR2 (IPv4 Address Byte 2)

DEFAULT: 0	RANGE: 0 255

IPv4 Address Byte 2 (NET_IPADDR2)

Description

`NET_IPADDR2` is the third part of your drone's static network address.

- **Example:** For an IP of 192.168.**1**.50, set `NET_IPADDR2` to **1**.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





NET_IPADDR3 (IPv4 Address Byte 3)

DEFAULT: 0	RANGE: 0 255

IPv4 Address Byte 3 (NET_IPADDR3)

Description

`NET_IPADDR3` is the fourth and final part of your drone's static network address.

- **Example:** For an IP of 192.168.1.**50**, set `NET_IPADDR3` to **50**.

NET_MACADDR0 (MAC Address Byte 0)

DEFAULT: 0	RANGE: 0 255

MAC Address Byte 0 (NET_MACADDR0)

Description

`NET_MACADDR0` is the first byte of your flight controller's unique physical network ID.

Normally, the flight controller generates a unique MAC address automatically based on its hardware Serial Number. You only need to change this if you have multiple drones on the same network that accidentally have the same MAC address, or if you need to bypass network security filters.

Tuning & Behavior

- **Default Value:** 0 (Automatically generated on boot).
- **Reboot Required:** Yes.

NET_MACADDR1 (MAC Address Byte 1)

DEFAULT: 0	RANGE: 0 255

MAC Address Byte 1 (NET_MACADDR1)

See NET_MACADDR0 for details.

NET_MACADDR2 (MAC Address Byte 2)

DEFAULT: 0	RANGE: 0 255

MAC Address Byte 2 (NET_MACADDR2)

See NET_MACADDR0 for details.

NET_MACADDR3 (MAC Address Byte 3)

DEFAULT: 0	RANGE: 0 255

MAC Address Byte 3 (NET_MACADDR3)

See NET_MACADDR0 for details.

NET_MACADDR4 (MAC Address Byte 4)

DEFAULT: 0	RANGE: 0 255

MAC Address Byte 4 (NET_MACADDR4)

See NET_MACADDR0 for details.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





NET_MACADDR5 (MAC Address Byte 5)

DEFAULT: 0	RANGE: 0 255

MAC Address Byte 5 (NET_MACADDR5)

See NET_MACADDR0 for details.

NET_NETMASK (Network Mask)

DEFAULT: 24	RANGE: 0 32	UNITS: bits

Network Mask (NET_NETMASK)

Description

`NET_NETMASK` defines the size of your local network. It uses CIDR notation (the number of bits set to 1).

- **24:** Equivalent to `255.255.255.0`. Standard for most home networks.
- **16:** Equivalent to `255.255.0.0`.

Tuning & Behavior

- **Default Value:** 24.

NET_OPTIONS (Networking Options)

DEFAULT: 0	RANGE: 0 127

Networking Options (NET_OPTIONS)

Description

`NET_OPTIONS` enables specialized networking "Bridges" and "Gateways."

- **Bit 0 (1):** Enable PPP Ethernet Gateway. Allows the drone to share its internet connection (e.g. from a cellular modem) with other devices on the Ethernet bus.
- **Bit 1 (2):** Enable CAN1 Multicast Endpoint. Forwards DroneCAN traffic to the network so you can use GCS tools over Wifi/Ethernet to tune CAN devices.
- **Bit 2 (4):** Enable CAN2 Multicast Endpoint.

Tuning & Behavior

- **Default Value:** 0.
- **Recommendation:** Enable **Bit 1** if you want to use the "DroneCAN GUI Tool" on your laptop without plugging in a USB cable to the flight controller.

NET_P1_PORT (Network Port 1 Number)

DEFAULT: 0	RANGE: 1 65535

Network Port 1 Number (NET_P1_PORT)

Description

`NET_P1_PORT` defines the "Door Number" for network traffic.

Tuning & Behavior

- **Default Value:** 0.
- **MAVLink Standard:** Set to **14550** or **14551** for standard ground station compatibility.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





NET_P1_PROTOCOL (Network Port 1 Protocol)

DEFAULT: 0	RANGE: 0 60

Network Port 1 Protocol (NET_P1_PROTOCOL)

Description

`NET_P1_PROTOCOL` identifies the "Language" spoken on the virtual network port. This uses the same values as the physical `SERIALx_PROTOCOL` parameters.

- **2:** MAVLink 2 (Default for most GCS).
- **5:** GPS.
- **20:** NMEA Output.

Tuning & Behavior

- **Default Value:** 0 (None).
- **Recommendation:** Set to **2** for GCS telemetry.

NET_P1_TYPE (Network Port 1 Type)

DEFAULT: 0	RANGE: 0 4

Network Port 1 Type (NET_P1_TYPE)

Description

`NET_P1_TYPE` creates a "Virtual Serial Port" over the network.

This allows you to send MAVLink data or GPS corrections over Ethernet or Wifi without using a physical UART cable.

- **0: Disabled.**
- **1: UDP Client.** The drone initiates a connection to a specific IP (defined in `NET_P1_IP`).
- **2: UDP Server.** The drone waits for a GCS to send data to it.
- **3: TCP Client.** Reliable connection initiated by the drone.
- **4: TCP Server.** Reliable connection where the drone acts as a host.

Tuning & Behavior

- **Default Value:** 0.
- **Recommendation:** Use **2 (UDP Server)** for standard GCS connections over Wifi. Use **3 (TCP Client)** if you are connecting to a fixed MAVLink routing server.

NET_P2_PORT (Control port)

DEFAULT: 0	RANGE: null	UNITS: null

Control port

Note: This parameter functions identically to NET_P1_PORT.

NET_P2_PROTOCOL (Enable use of specific protocol over virtual driver)

DEFAULT: float(AP_CAN::Protocol::DroneCAN	RANGE: null	UNITS: null

Enable use of specific protocol over virtual driver

Note: This parameter functions identically to NET_P1_PROTOCOL.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





NET_P2_TYPE (Tiltrotor type)

DEFAULT: TILT_TYPE_CONTINUOUS	RANGE: null	UNITS: null

Tiltrotor type

Note: This parameter functions identically to NET_P1_TYPE.

NET_P3_PORT (Control port)

DEFAULT: 0	RANGE: null	UNITS: null

Control port

Note: This parameter functions identically to NET_P1_PORT.

NET_P3_PROTOCOL (Enable use of specific protocol over virtual driver)

DEFAULT: float(AP_CAN::Protocol::DroneCAN	RANGE: null	UNITS: null

Enable use of specific protocol over virtual driver

Note: This parameter functions identically to NET_P1_PROTOCOL.

NET_P3_TYPE (Tiltrotor type)

DEFAULT: TILT_TYPE_CONTINUOUS	RANGE: null	UNITS: null

Tiltrotor type

Note: This parameter functions identically to NET_P1_TYPE.

NET_P4_PORT (Control port)

DEFAULT: 0	RANGE: null	UNITS: null

Control port

Note: This parameter functions identically to NET_P1_PORT.

NET_P4_PROTOCOL (Enable use of specific protocol over virtual driver)

DEFAULT: float(AP_CAN::Protocol::DroneCAN	RANGE: null	UNITS: null

Enable use of specific protocol over virtual driver

Note: This parameter functions identically to NET_P1_PROTOCOL.

NET_P4_TYPE (Tiltrotor type)

DEFAULT: TILT_TYPE_CONTINUOUS	RANGE: null	UNITS: null

Tiltrotor type

Note: This parameter functions identically to NET_P1_TYPE.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





NET_P5_PORT (Control port)

DEFAULT: 0	RANGE: null	UNITS: null

Control port

Note: This parameter functions identically to NET_P1_PORT.

NET_P5_PROTOCOL (Enable use of specific protocol over virtual driver)

DEFAULT: float(AP_CAN::Protocol::DroneCAN	RANGE: null	UNITS: null

Enable use of specific protocol over virtual driver

Note: This parameter functions identically to NET_P1_PROTOCOL.

NET_P5_TYPE (Tiltrotor type)

DEFAULT: TILT_TYPE_CONTINUOUS	RANGE: null	UNITS: null

Tiltrotor type

Note: This parameter functions identically to NET_P1_TYPE.

NET_P6_PORT (Control port)

DEFAULT: 0	RANGE: null	UNITS: null

Control port

Note: This parameter functions identically to NET_P1_PORT.

NET_P6_PROTOCOL (Enable use of specific protocol over virtual driver)

DEFAULT: float(AP_CAN::Protocol::DroneCAN	RANGE: null	UNITS: null

Enable use of specific protocol over virtual driver

Note: This parameter functions identically to NET_P1_PROTOCOL.

NET_P6_TYPE (Tiltrotor type)

DEFAULT: TILT_TYPE_CONTINUOUS	RANGE: null	UNITS: null

Tiltrotor type

Note: This parameter functions identically to NET_P1_TYPE.

NET_P7_PORT (Control port)

DEFAULT: 0	RANGE: null	UNITS: null

Control port

Note: This parameter functions identically to NET_P1_PORT.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





NET_P7_PROTOCOL (Enable use of specific protocol over virtual driver)

DEFAULT: float(AP_CAN::Protocol::DroneCAN	RANGE: null	UNITS: null

Enable use of specific protocol over virtual driver

Note: This parameter functions identically to NET_P1_PROTOCOL.

NET_P7_TYPE (Tiltrotor type)

DEFAULT: TILT_TYPE_CONTINUOUS	RANGE: null	UNITS: null

Tiltrotor type

Note: This parameter functions identically to NET_P1_TYPE.

NET_P8_PORT (Control port)

DEFAULT: 0	RANGE: null	UNITS: null

Control port

Note: This parameter functions identically to NET_P1_PORT.

NET_P8_PROTOCOL (Enable use of specific protocol over virtual driver)

DEFAULT: float(AP_CAN::Protocol::DroneCAN	RANGE: null	UNITS: null

Enable use of specific protocol over virtual driver

Note: This parameter functions identically to NET_P1_PROTOCOL.

NET_P8_TYPE (Tiltrotor type)

DEFAULT: TILT_TYPE_CONTINUOUS	RANGE: null	UNITS: null

Tiltrotor type

Note: This parameter functions identically to NET_P1_TYPE.

NET_P9_PORT (Control port)

DEFAULT: 0	RANGE: null	UNITS: null

Control port

Note: This parameter functions identically to NET_P1_PORT.

NET_P9_PROTOCOL (Enable use of specific protocol over virtual driver)

DEFAULT: float(AP_CAN::Protocol::DroneCAN	RANGE: null	UNITS: null

Enable use of specific protocol over virtual driver

Note: This parameter functions identically to NET_P1_PROTOCOL.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





NET_P9_TYPE (Tiltrotor type)

DEFAULT: TILT_TYPE_CONTINUOUS	RANGE: null	UNITS: null

Tiltrotor type

Note: This parameter functions identically to NET_P1_TYPE.

NET_PASS1_BAUD1 (Networking Passthrough BAUD 1)

DEFAULT: 115200	RANGE: 1 4000000	UNITS: bps

Networking Passthrough BAUD 1 (NET_PASS1_BAUD1)

Description

`NET_PASS1_BAUD1` sets the communication speed for the first endpoint in the passthrough bridge.

- **115200 (Default):** Standard for most telemetry and GPS units.

Tuning & Behavior

- **Constraint:** This only applies if `NET_PASS1_EP1` is set to a Serial UART (0-9).

NET_PASS1_BAUD2 (Networking Passthrough BAUD 2)

DEFAULT: 115200	RANGE: 1 4000000	UNITS: bps

Networking Passthrough BAUD 2 (NET_PASS1_BAUD2)

Description

`NET_PASS1_BAUD2` sets the communication speed for the second endpoint.

See NET_PASS1_BAUD1 for details.

NET_PASS1_ENABLE (Networking Passthrough Enable)

DEFAULT: 0	RANGE: 0 1

Networking Passthrough Enable (NET_PASS1_ENABLE)

Description

`NET_PASS1_ENABLE` activates a powerful "Digital Patch Bay" feature on peripheral nodes.

This allows you to take data coming in from one port (like an Ethernet socket) and pipe it directly out of another port (like a physical UART Serial port) on the same device. This is essential for transparently connecting specialized sensors or companion computers that are not natively CAN/DroneCAN compatible to the rest of the network.

Tuning & Behavior

- **Default Value:** 0 (Disabled).
- **Use Case:** Use this on an H7-based peripheral board to bridge a MAVLink telemetry stream from Ethernet to a Serial radio.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY

NET_PASS1_EP1 (Networking Passthrough Endpoint 1)

DEFAULT: -1	RANGE: -1 59

Networking Passthrough Endpoint 1 (NET_PASS1_EP1)

Description

`NET_PASS1_EP1` defines the first "Socket" in your data bridge.

You can select from a wide variety of physical and virtual ports. Data received on Endpoint 1 is sent to Endpoint 2, and vice versa.

Values

- **-1:** Disabled.
- **0-9:** Physical Serial Ports (Serial0 is usually USB).
- **21-29:** Network Ports (UDP/TCP sockets configured in `NET_Pn_`).
- **41-49:** CAN1 Virtual Ports.
- **51-59:** CAN2 Virtual Ports.

Tuning & Behavior

- **Reboot Required:** Yes.
- **Example:** To bridge a TCP socket (Network Port 1) to a Serial GPS (Serial 3), set `EP1 = 21` and `EP2 = 3`.

NET_PASS1_EP2 (Networking Passthrough Endpoint 2)

DEFAULT: -1	RANGE: -1 59

Networking Passthrough Endpoint 2 (NET_PASS1_EP2)

See NET_PASS1_EP1 for details.

NET_PASS1_OPT1 (Network Passthrough 1 Options 1)

DEFAULT: 0	RANGE: 0 255

Network Passthrough 1 Options 1 (NET_PASS1_OPT1)

Description

`NET_PASS1_OPT1` enables specific features for the first side of the bridge, such as RTS/CTS flow control or parity settings.

NET_PASS1_OPT2 (Network Passthrough 1 Options 2)

DEFAULT: 0	RANGE: 0 255

Network Passthrough 1 Options 2 (NET_PASS1_OPT2)

Description

`NET_PASS1_OPT2` enables specific features for the second side of the bridge.

NET_PASS2_BAUD1 (Endpoint 1 Baud Rate)

DEFAULT: 115200	RANGE: null	UNITS: null

Endpoint 1 Baud Rate

Note: This parameter functions identically to NET_PASS1_BAUD1.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





NET_PASS2_BAUD2 (Endpoint 2 Baud Rate)

DEFAULT: 115200	RANGE: null	UNITS: null

Endpoint 2 Baud Rate

Note: This parameter functions identically to NET_PASS1_BAUD2.

NET_PASS2_ENABLE (Enable pullup after altitude wait)

DEFAULT: 0	RANGE: null	UNITS: null

Enable pullup after altitude wait

Note: This parameter functions identically to NET_PASS1_ENABLE.

NET_PASS2_EP1 (Endpoint 1)

DEFAULT: -1	RANGE: null	UNITS: null

Endpoint 1

Note: This parameter functions identically to NET_PASS1_EP1.

NET_PASS2_EP2 (Endpoint 2)

DEFAULT: -1	RANGE: null	UNITS: null

Endpoint 2

Note: This parameter functions identically to NET_PASS1_EP2.

NET_PASS2_OPT1 (Serial Port Options EP1)

DEFAULT: 0	RANGE: null	UNITS: null

Serial Port Options EP1

Note: This parameter functions identically to NET_PASS1_OPT1.

NET_PASS2_OPT2 (Serial Port Options EP2)

DEFAULT: 0	RANGE: null	UNITS: null

Serial Port Options EP2

Note: This parameter functions identically to NET_PASS1_OPT2.

NET_PASS3_BAUD1 (Endpoint 1 Baud Rate)

DEFAULT: 115200	RANGE: null	UNITS: null

Endpoint 1 Baud Rate

Note: This parameter functions identically to NET_PASS1_BAUD1.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





NET_PASS3_BAUD2 (Endpoint 2 Baud Rate)

DEFAULT: 115200	RANGE: null	UNITS: null

Endpoint 2 Baud Rate

Note: This parameter functions identically to NET_PASS1_BAUD2.

NET_PASS3_ENABLE (Enable pullup after altitude wait)

DEFAULT: 0	RANGE: null	UNITS: null

Enable pullup after altitude wait

Note: This parameter functions identically to NET_PASS1_ENABLE.

NET_PASS3_EP1 (Endpoint 1)

DEFAULT: -1	RANGE: null	UNITS: null

Endpoint 1

Note: This parameter functions identically to NET_PASS1_EP1.

NET_PASS3_EP2 (Endpoint 2)

DEFAULT: -1	RANGE: null	UNITS: null

Endpoint 2

Note: This parameter functions identically to NET_PASS1_EP2.

NET_PASS3_OPT1 (Serial Port Options EP1)

DEFAULT: 0	RANGE: null	UNITS: null

Serial Port Options EP1

Note: This parameter functions identically to NET_PASS1_OPT1.

NET_PASS3_OPT2 (Serial Port Options EP2)

DEFAULT: 0	RANGE: null	UNITS: null

Serial Port Options EP2

Note: This parameter functions identically to NET_PASS1_OPT2.

NET_PASS4_BAUD1 (Endpoint 1 Baud Rate)

DEFAULT: 115200	RANGE: null	UNITS: null

Endpoint 1 Baud Rate

Note: This parameter functions identically to NET_PASS1_BAUD1.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





NET_PASS4_BAUD2 (Endpoint 2 Baud Rate)

DEFAULT: 115200	RANGE: null	UNITS: null

Endpoint 2 Baud Rate

Note: This parameter functions identically to NET_PASS1_BAUD2.

NET_PASS4_ENABLE (Enable pullup after altitude wait)

DEFAULT: 0	RANGE: null	UNITS: null

Enable pullup after altitude wait

Note: This parameter functions identically to NET_PASS1_ENABLE.

NET_PASS4_EP1 (Endpoint 1)

DEFAULT: -1	RANGE: null	UNITS: null

Endpoint 1

Note: This parameter functions identically to NET_PASS1_EP1.

NET_PASS4_EP2 (Endpoint 2)

DEFAULT: -1	RANGE: null	UNITS: null

Endpoint 2

Note: This parameter functions identically to NET_PASS1_EP2.

NET_PASS4_OPT1 (Serial Port Options EP1)

DEFAULT: 0	RANGE: null	UNITS: null

Serial Port Options EP1

Note: This parameter functions identically to NET_PASS1_OPT1.

NET_PASS4_OPT2 (Serial Port Options EP2)

DEFAULT: 0	RANGE: null	UNITS: null

Serial Port Options EP2

Note: This parameter functions identically to NET_PASS1_OPT2.

NET_PASS5_BAUD1 (Endpoint 1 Baud Rate)

DEFAULT: 115200	RANGE: null	UNITS: null

Endpoint 1 Baud Rate

Note: This parameter functions identically to NET_PASS1_BAUD1.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





NET_PASS5_BAUD2 (Endpoint 2 Baud Rate)

DEFAULT: 115200	RANGE: null	UNITS: null

Endpoint 2 Baud Rate

Note: This parameter functions identically to NET_PASS1_BAUD2.

NET_PASS5_ENABLE (Enable pullup after altitude wait)

DEFAULT: 0	RANGE: null	UNITS: null

Enable pullup after altitude wait

Note: This parameter functions identically to NET_PASS1_ENABLE.

NET_PASS5_EP1 (Endpoint 1)

DEFAULT: -1	RANGE: null	UNITS: null

Endpoint 1

Note: This parameter functions identically to NET_PASS1_EP1.

NET_PASS5_EP2 (Endpoint 2)

DEFAULT: -1	RANGE: null	UNITS: null

Endpoint 2

Note: This parameter functions identically to NET_PASS1_EP2.

NET_PASS5_OPT1 (Serial Port Options EP1)

DEFAULT: 0	RANGE: null	UNITS: null

Serial Port Options EP1

Note: This parameter functions identically to NET_PASS1_OPT1.

NET_PASS5_OPT2 (Serial Port Options EP2)

DEFAULT: 0	RANGE: null	UNITS: null

Serial Port Options EP2

Note: This parameter functions identically to NET_PASS1_OPT2.

NET_PASS6_BAUD1 (Endpoint 1 Baud Rate)

DEFAULT: 115200	RANGE: null	UNITS: null

Endpoint 1 Baud Rate

Note: This parameter functions identically to NET_PASS1_BAUD1.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





NET_PASS6_BAUD2 (Endpoint 2 Baud Rate)

DEFAULT: 115200	RANGE: null	UNITS: null

Endpoint 2 Baud Rate

Note: This parameter functions identically to NET_PASS1_BAUD2.

NET_PASS6_ENABLE (Enable pullup after altitude wait)

DEFAULT: 0	RANGE: null	UNITS: null

Enable pullup after altitude wait

Note: This parameter functions identically to NET_PASS1_ENABLE.

NET_PASS6_EP1 (Endpoint 1)

DEFAULT: -1	RANGE: null	UNITS: null

Endpoint 1

Note: This parameter functions identically to NET_PASS1_EP1.

NET_PASS6_EP2 (Endpoint 2)

DEFAULT: -1	RANGE: null	UNITS: null

Endpoint 2

Note: This parameter functions identically to NET_PASS1_EP2.

NET_PASS6_OPT1 (Serial Port Options EP1)

DEFAULT: 0	RANGE: null	UNITS: null

Serial Port Options EP1

Note: This parameter functions identically to NET_PASS1_OPT1.

NET_PASS6_OPT2 (Serial Port Options EP2)

DEFAULT: 0	RANGE: null	UNITS: null

Serial Port Options EP2

Note: This parameter functions identically to NET_PASS1_OPT2.

NET_PASS7_BAUD1 (Endpoint 1 Baud Rate)

DEFAULT: 115200	RANGE: null	UNITS: null

Endpoint 1 Baud Rate

Note: This parameter functions identically to NET_PASS1_BAUD1.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





NET_PASS7_BAUD2 (Endpoint 2 Baud Rate)

DEFAULT: 115200	RANGE: null	UNITS: null

Endpoint 2 Baud Rate

Note: This parameter functions identically to NET_PASS1_BAUD2.

NET_PASS7_ENABLE (Enable pullup after altitude wait)

DEFAULT: 0	RANGE: null	UNITS: null

Enable pullup after altitude wait

Note: This parameter functions identically to NET_PASS1_ENABLE.

NET_PASS7_EP1 (Endpoint 1)

DEFAULT: -1	RANGE: null	UNITS: null

Endpoint 1

Note: This parameter functions identically to NET_PASS1_EP1.

NET_PASS7_EP2 (Endpoint 2)

DEFAULT: -1	RANGE: null	UNITS: null

Endpoint 2

Note: This parameter functions identically to NET_PASS1_EP2.

NET_PASS7_OPT1 (Serial Port Options EP1)

DEFAULT: 0	RANGE: null	UNITS: null

Serial Port Options EP1

Note: This parameter functions identically to NET_PASS1_OPT1.

NET_PASS7_OPT2 (Serial Port Options EP2)

DEFAULT: 0	RANGE: null	UNITS: null

Serial Port Options EP2

Note: This parameter functions identically to NET_PASS1_OPT2.

NET_PASS8_BAUD1 (Endpoint 1 Baud Rate)

DEFAULT: 115200	RANGE: null	UNITS: null

Endpoint 1 Baud Rate

Note: This parameter functions identically to NET_PASS1_BAUD1.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





NET_PASS8_BAUD2 (Endpoint 2 Baud Rate)

DEFAULT: 115200	RANGE: null	UNITS: null

Endpoint 2 Baud Rate

Note: This parameter functions identically to NET_PASS1_BAUD2.

NET_PASS8_ENABLE (Enable pullup after altitude wait)

DEFAULT: 0	RANGE: null	UNITS: null

Enable pullup after altitude wait

Note: This parameter functions identically to NET_PASS1_ENABLE.

NET_PASS8_EP1 (Endpoint 1)

DEFAULT: -1	RANGE: null	UNITS: null

Endpoint 1

Note: This parameter functions identically to NET_PASS1_EP1.

NET_PASS8_EP2 (Endpoint 2)

DEFAULT: -1	RANGE: null	UNITS: null

Endpoint 2

Note: This parameter functions identically to NET_PASS1_EP2.

NET_PASS8_OPT1 (Serial Port Options EP1)

DEFAULT: 0	RANGE: null	UNITS: null

Serial Port Options EP1

Note: This parameter functions identically to NET_PASS1_OPT1.

NET_PASS8_OPT2 (Serial Port Options EP2)

DEFAULT: 0	RANGE: null	UNITS: null

Serial Port Options EP2

Note: This parameter functions identically to NET_PASS1_OPT2.

NET_PASS9_BAUD1 (Endpoint 1 Baud Rate)

DEFAULT: 115200	RANGE: null	UNITS: null

Endpoint 1 Baud Rate

Note: This parameter functions identically to NET_PASS1_BAUD1.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





NET_PASS9_BAUD2 (Endpoint 2 Baud Rate)

DEFAULT: 115200	RANGE: null	UNITS: null

Endpoint 2 Baud Rate

Note: This parameter functions identically to NET_PASS1_BAUD2.

NET_PASS9_ENABLE (Enable pullup after altitude wait)

DEFAULT: 0	RANGE: null	UNITS: null

Enable pullup after altitude wait

Note: This parameter functions identically to NET_PASS1_ENABLE.

NET_PASS9_EP1 (Endpoint 1)

DEFAULT: -1	RANGE: null	UNITS: null

Endpoint 1

Note: This parameter functions identically to NET_PASS1_EP1.

NET_PASS9_EP2 (Endpoint 2)

DEFAULT: -1	RANGE: null	UNITS: null

Endpoint 2

Note: This parameter functions identically to NET_PASS1_EP2.

NET_PASS9_OPT1 (Serial Port Options EP1)

DEFAULT: 0	RANGE: null	UNITS: null

Serial Port Options EP1

Note: This parameter functions identically to NET_PASS1_OPT1.

NET_PASS9_OPT2 (Serial Port Options EP2)

DEFAULT: 0	RANGE: null	UNITS: null

Serial Port Options EP2

Note: This parameter functions identically to NET_PASS1_OPT2.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





NET_PPP_BAUD (PPP Baud Rate)

DEFAULT: 57	RANGE: 1 2000

PPP Baud Rate (NET_PPP_BAUD)

Description

`NET_PPP_BAUD` sets the speed for the PPP connection.

This is used when a serial modem (like an LTE modem) is connected to a UART port and the autopilot is acting as a PPP client or gateway.

Tuning & Behavior

- **Default Value:** 57 (57600 baud).
- **Recommendation:** Set to match your modem's baud rate (often **115** or **921**).

NET_PPP_PORT (PPP Serial Port)

DEFAULT: 0	RANGE: 0 10

PPP Serial Port (NET_PPP_PORT)

Description

`NET_PPP_PORT` selects which physical UART (TELEM port) is connected to the PPP modem.

- **0:** Serial 0 (USB).
- **1:** Serial 1 (Telem 1).

NET_TESTS (Networking Tests)

DEFAULT: 0	RANGE: 0 15

Networking Tests (NET_TESTS)

Description

`NET_TESTS` enables standard debugging services on the drone's IP address.

- **Bit 0 (1):** UDP Echo (Port 7). The drone bounces back any UDP packet sent to port 7.
- **Bit 1 (2):** TCP Echo (Port 7).
- **Bit 2 (4):** TCP Discard (Port 9). Accepts connections but ignores data.
- **Bit 3 (8):** TCP Reflect (Port 19?).

Tuning & Behavior

- **Default Value:** 0.
- **Recommendation:** Leave **0** unless you are debugging the network stack itself.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **NMEA** parameter group configures the **NMEA 0183** data output from the autopilot. This is used to share the vehicle's position with external devices that expect a standard GPS-style serial stream (e.g., marine radars, handheld GPS units, or legacy data loggers).

Key Concepts

1. Message Selection (`NMEA_MSG_EN`)

Determines which standard NMEA sentences are included in the serial stream.

- **GGA:** Global Positioning System Fix Data (Time, Position, Fix Quality).
- **RMC:** Recommended Minimum Specific GNSS Data (Time, Position, Speed, Date).
- **VTG:** Course Over Ground and Ground Speed.

Parameter Breakdown

- `NMEA_MSG_EN`: Bitmask of enabled NMEA messages.

Integration Guide

1. **Port Setup:** Set `SERIALx_PROTOCOL = 15` (NMEA Output) on the desired UART.
2. **Baud Rate:** Set the baud rate to match the receiving device (usually 4800 or 9600).
3. **Config:** Enable the required messages in `NMEA_MSG_EN`.

Developer Notes

- **Library:** `libraries/AP_NMEA_Output`.

Messages Enable bitmask ()

NMEA_MSG_EN: Messages Enable bitmask

Description

This is a bitmask of enabled NMEA messages. All messages will be sent consecutively at the same rate interval.

Values

- **Bitmask:**
  - **Bit 0:** GPGGA (Global Positioning System Fix Data)
  - **Bit 1:** GPRMC (Recommended Minimum Specific GPS/Transit Data)
  - **Bit 2:** PASHR (Pitch, Roll, and Heading)

Description

This parameter controls which standard NMEA 0183 sentences are output by the autopilot when configured as an NMEA source.

- **Usage:** Typically used to feed position and attitude data to external payloads, cameras, or trackers that expect NMEA format (rather than MAVLink).
- **Default (3):** Enables **GGA** and **RMC** (3 = 1 + 2).

Source Code

ardupilot/libraries/AP_NMEA_Output/AP_NMEA_Output.cpp

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **NO** parameter group (specifically `NO_GO_ANGLE`) is a specialized configuration for **ArduRover Sailboats**.

In sailing, a boat cannot sail directly into the wind. The "No-Go Zone" is the range of angles relative to the wind direction where the sails cannot generate lift and the boat will stall.

Key Concepts

1. Tacking and Beating

When a target waypoint is directly upwind, the autopilot cannot sail a straight line to it. Instead, it must "Tack" (zig-zag) at an angle to the wind.

- `NO_GO_ANGLE`: Defines the angle relative to the wind (e.g., 45 degrees) that the boat will maintain when trying to go upwind.
- **Total Zone:** A `NO_GO_ANGLE` of 45 results in a 90-degree total arc centered on the wind direction where the boat will not enter.

Parameter Breakdown

- `NO_GO_ANGLE`: Upwind sailing limit (degrees).
  - **Typical:** 40 to 50 degrees depending on the efficiency of the boat's keel and sail.

Developer Notes

- **Library:** `Rover/sailboat.cpp`.
- **Logic:** Integrated with the L1 navigation controller to generate "Virtual Waypoints" that stay outside the no-go zone.

Sailing no go zone angle ()

NO_GO_ANGLE: Sailing no go zone angle

Description

The typical closest angle to the wind the vehicle will sail at. The vehicle will sail at this angle when going upwind.

Values

- **Range:** 0 to 90
- **Units:** deg
- **Increment:** 1
- **Default:** 45

Description

This parameter defines the "dead zone" into the wind where a sailboat cannot generate forward thrust.

- **Function:** When the autopilot needs to go to a waypoint that lies within this angle of the wind, it will tack (zig-zag) back and forth at this angle instead of trying to sail straight into the wind.
- **Tuning:**
  - **Performance Boats:** Can often sail closer to the wind (e.g., 35-40 degrees).
  - **Cruisers/Inefficient Rigs:** May need a wider angle (e.g., 50-60 degrees) to maintain speed.

Source Code

ardupilot/Rover/sailboat.cpp

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





NOTIFY CONFIGURATION (NTF)

Overview

The **NTF** parameter group configures the **Notification Subsystem**. This library handles user-facing feedback, including RGB status LEDs, buzzers, display screens, and OreoLEDs.

Effective notification setup is critical for safety, as it provides the pilot with visual and audible confirmation of arming state, GPS lock, and failsafe conditions.

Key Concepts

1. LED Types (`NTF_LED_TYPES`)

ArduPilot supports many LED hardware types.

- **Built-in:** The small LED on the flight controller.
- **External (I2C):** Common Toshiba or NeoPixel LEDs found on GPS modules.
- **Scripting:** LEDs controlled by Lua scripts.
- **OreoLED:** Specific to the 3DR Solo and similar high-power LED systems.

2. Buzzer Behavior (`NTF_BUZZ_...`)

- `NTF_BUZZ_VOLUME`: Global volume for the internal alarm buzzer.
- `NTF_BUZZ_TYPES`: Bitmask to enable/disable specific tones (e.g., quiet the "Startup" tone but keep the "Failsafe" tone).

3. Display Types (`NTF_DISPLAY_TYPE`)

Supports small I2C OLED screens (SSD1306) mounted to the drone to show battery voltage and status without a GCS.

Parameter Breakdown

- `NTF_LED_BRIGHT`: LED intensity (Low, Medium, High).
- `NTF_LED_OVERRIDE`: Allows selecting simplified color schemes (e.g., "Traffic Light").
- `NTF_OREO_THEME`: Color themes for OreoLED rings.

Integration Guide

- **NeoPixels:** To use NeoPixel strips for status, set `NTF_LED_TYPES` to include NeoPixel and ensure a servo output is mapped to `LED` (Function 120).

Developer Notes

- **Library:** `libraries/AP_Notify`.
- **Drivers:** Handles hardware abstraction for I2C and dedicated PWM notification devices.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Buzzer-on pin logic level ()

NTF_BUZZ_ON_LVL: Buzzer-on pin logic level

Description

Specifies pin level that indicates buzzer should play.

Values

Value	Meaning
0	LowIsOn (Active Low)
1	HighIsOn (Active High)

- **Default:** 1 (HighIsOn)

Description

This parameter defines the electrical polarity of the buzzer control signal.

- **Active High (1):** The autopilot outputs 3.3V (Logic High) to turn the buzzer ON. This is standard for most simple piezo buzzers driven by a transistor.
- **Active Low (0):** The autopilot outputs 0V (Logic Low) to turn the buzzer ON. This is common for some integrated buzzer modules or specific board designs.

Source Code

ardupilot/libraries/AP_Notify/AP_Notify.cpp

NTF_BUZZ_PIN (Buzzer Output Pin)

DEFAULT: -1	RANGE: -1 103

Buzzer Output Pin (NTF_BUZZ_PIN)

Description

`NTF_BUZZ_PIN` tells the autopilot which pin is connected to the piezo buzzer.

The buzzer is used for audible feedback, such as "Ready to Arm" beeps, low battery sirens, and SOS codes for sensor failures.

Tuning & Behavior

- **Default:** -1 (Use the board's default buzzer pin).
- **Usage:** If you are building a custom drone and want to move the buzzer to a different output (e.g., AUX 5), you can enter that GPIO number here.
- **Hardware Requirement:** Note that standard flight controller pins provide very little current. If you are using a large, loud buzzer, you may need a small MOSFET or transistor circuit between the FC and the buzzer.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY

NTF_BUZZ_TYPES (Buzzer Driver Types)

DEFAULT: 1	RANGE: 0 7

Buzzer Driver Types (NTF_BUZZ_TYPES)

Description

`NTF_BUZZ_TYPES` controls where the "Beeps" come from.

- **Bit 0 (1): Built-in buzzer.** The standard piezoelectric buzzer connected to the flight controller.
- **Bit 1 (2): DShot.** Uses the motors themselves to beep! The ESC vibrates the motor windings to create sound.
- **Bit 2 (4): DroneCAN.** Uses a buzzer on a CAN-bus peripheral (like a GPS puck).

The Engineer's View

Enabling **DShot (Bit 1)** is extremely useful for finding a lost drone in tall grass ("Lost Model Alarm"). Since the motors are much louder than the small piezo buzzer, this can be a lifesaver.

Tuning & Behavior

- **Default Value:** 1 (Built-in)
- **Recommendation:** Set to **3** (Built-in + DShot) to enable motor beeping.

NTF_BUZZ_VOLUME (Buzzer Volume)

DEFAULT: 100	RANGE: 0 100	UNITS: %

Buzzer Volume (NTF_BUZZ_VOLUME)

Description

`NTF_BUZZ_VOLUME` sets the loudness of the beeps and tunes played by the flight controller's buzzer (used for arming alerts, low battery warnings, and error codes).

- **100:** (Default). Full volume.
- **0:** Silent (Not recommended for safety reasons).

Tuning & Behavior

- **Note:** This only affects hardware buzzers that support volume control via PWM (common on modern Flight Controllers). Simple "Buzzer" pins on older hardware may only support ON/OFF and will ignore this setting.

NTF_DISPLAY_TYPE (I2C OLED Display Type)

DEFAULT: 0	RANGE: 0 1

I2C OLED Display Type (NTF_DISPLAY_TYPE)

Description

`NTF_DISPLAY_TYPE` enables support for tiny onboard OLED screens (usually 128×64 or 0.96").

These screens are mounted directly to the drone and provide at-a-glance information while the drone is on the ground, such as:

- Arming status.
- GPS lock quality (HDOP / Sat count).
- Battery voltage.
- Active flight mode.

Tuning & Behavior

- **0: Disabled (Default).**
- **1: SSD1306.** Standard I2C OLED driver used by most hobbyist screens.
- **Reboot Required:** Yes.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





NTF_LED_BRIGHT (Onboard LED Brightness)

DEFAULT: 3	RANGE: 1 3

Onboard LED Brightness (NTF_LED_BRIGHT)

Description

`NTF_LED_BRIGHT` controls how intense the status LEDs (e.g. the big blinking LED on a Pixhawk) appear.

- **1: Low.** Dim, good for night flight or bench testing.
- **2: Medium.**
- **3: High (Default).** Bright enough to be seen in direct sunlight.

Tuning & Behavior

- **Safety Feature:** When the flight controller is powered via **USB**, the brightness is automatically forced to **Low** to prevent over-current on the USB rail and to avoid blinding the developer. The high setting only takes effect when powered by a battery.

NTF_LED_LEN (Serial LED String Length)

DEFAULT: 1	RANGE: 1 32	UNITS: LEDs

Serial LED String Length (NTF_LED_LEN)

Description

`NTF_LED_LEN` defines how many NeoPixel-style LEDs are in your status light strip.

ArduPilot can control a chain of addressable RGB LEDs to provide super-bright, high-visibility status notifications (like green for "Home Set" or flashing yellow for "Radio Failsafe").

- **1 (Default):** Standard single status LED.
- **1-32:** Number of LEDs in your physical strip.

Tuning & Behavior

- **Usage:** Set this to match the number of LEDs you have physically wired to your serial LED port.
- **Power Warning:** Addressable LEDs consume significant current. If you use a long string (e.g., 20+ LEDs), ensure your flight controller's 5V regulator can handle the load, or use an external BEC.

NTF_LED_OVERRIDE (Specifies colour source for the RGBLed)

DEFAULT: 0	RANGE: 0 2

Specifies colour source for the RGBLed (NTF_LED_OVERRIDE)

Description

This parameter changes the color logic of the main Status LED. The standard ArduPilot blink codes are very detailed but complex. This option allows selecting simplified schemes used for specific competitions or safety standards.

The Mathematics

Enum mapping.

The Engineer's View

Defined in `libraries/AP_Notify/AP_Notify.cpp`.

- **0 (Standard):** Full ArduPilot blink codes (Yellow=Failsafe, Blue=GPS, Green=Armed/GPS, etc.).
- **1 (Traffic Light):** Red=Armed, Yellow=Disarmed (Unsafe/No GPS), Green=Disarmed (Safe).
- **2 (Medical Express):** Specific to the UAV Challenge rules.

Tuning & Behavior

- **Default Value:** 0
- **Usage:** Use "Traffic Light" (1) for simpler visual confirmation of safety state for ground personnel.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





NTF_LED_TYPES (LED Driver Types)

DEFAULT: 309	RANGE: 0 524287

LED Driver Types (NTF_LED_TYPES)

Description

`NTF_LED_TYPES` determines which lights on your drone will flash to indicate status (like GPS lock, Arming, or Low Battery).

ArduPilot supports many different LED systems simultaneously. You can have the built-in board LED flashing alongside an external NeoPixel strip and a DroneCAN high-brightness LED.

- **Bit 0 (1): Built-in LED.** The small RGB LED on the flight controller itself.
- **Bit 2 (4): External ToshibaLED.** The standard "I2C LED" found on many external GPS/Compass modules (e.g., Here2, Here3).
- **Bit 5 (32): DroneCAN.** For CAN-bus connected LEDs.
- **Bit 8 (256): NeoPixel.** For addressable LED strips (WS2812B) connected to a PWM output.
- **Bit 11 (2048): DShot.** Uses the LEDs built into BLHeli_32/AM32 ESCs.

The Engineer's View

Defined in `AP_Notify.cpp`.
During initialization, the `AP_Notify::add_backends()` function iterates through the set bits and instantiates the corresponding driver classes. Each driver is then added to the `_devices` array and updated at 50Hz.

Tuning & Behavior

- **Default Value:** 309 (Built-in + Internal Toshiba + External Toshiba + DroneCAN + NeoPixel).
- **Recommendation:**
  - **Standard Build:** Leave at default.
  - **ESC LEDs:** If you want your motor LEDs to flash status, set **Bit 11** (add 2048).
  - **Show Lights:** If you connect a NeoPixel strip for a light show, ensure **Bit 8** (256) is set.

NTF_OREO_THEME (OreoLED Theme Selection)

DEFAULT: 0	RANGE: 0 2

OreoLED Theme Selection (NTF_OREO_THEME)

Description

`NTF_OREO_THEME` enables the specialized status light driver for **OreoLED** hardware (found on the 3DR Solo and some companion peripherals).

- **0: Disabled (Default).**
- **1: Aircraft Theme.** Uses standard aviation colors (Red/Green/White) to indicate orientation and system status.
- **2: Rover Theme.** Uses specialized patterns for ground vehicles.

Tuning & Behavior

- **Default:** 0.
- **Usage:** Only enable this if you have physical OreoLED hardware connected to the I2C bus.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **NUM** parameter group (specifically `NUM_CELLS` or `BATT_BAL_NUM_CELLS`) configures the cell count for high-precision **Battery Balance** monitoring.

Key Concepts

1. Cell Count

When using a smart battery or a power module that monitors individual cell voltages (via a balance lead), the autopilot needs to know how many cells are in series.

- `NUM_CELLS`: The integer number of cells (e.g., 4 for a 4S LiPo).

Developer Notes

- **Library:** `libraries/AP_BattMonitor`.
- **Context:** Only relevant if your power module supports per-cell monitoring.

Number of battery cells ()

NUM_CELLS: Number of battery cells

Description

Number of battery cells to monitor.

Values

- **Range:** 0 to 64
- **Default:** 0

Description

This parameter configures the **AP_Periph Battery Balancer** feature.

- **Usage:** If you have an AP_Periph device wired to monitor individual cell voltages (e.g., via analog inputs), this sets how many cells are in series.
- **Function:** The peripheral will read `NUM_CELLS` sequential analog pins starting from `CELL1_PIN`, calculate the individual cell voltages (by subtracting the previous cell's total voltage), and broadcast this data over DroneCAN.
- **0:** Disabled.

Source Code

ardupilot/Tools/AP_Periph/batt_balance.cpp

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **OA** parameter group configures the **Object Avoidance** path-planning engine. This subsystem uses distance data from Proximity sensors (Lidar/Sonar) or Visual Odometry to autonomously steer the vehicle away from obstacles in real-time.

Unlike simple "BendyRuler" avoidance, the OA library can manage complex 3D environments and find optimal paths around large structures.

Key Concepts

1. Avoidance Types (`OA_TYPE`)

Selects the primary path-finding algorithm.

- **0:** Disabled.
- **1 (BendyRuler):** Simple and fast. Tries a few "rays" to see which path is clear and chooses the one closest to the target waypoint.
- **2 (Dijkstra):** Uses a pre-mapped database of the environment (`OA_DB_...`) to calculate the mathematically shortest path around an obstacle.
- **3 (SimpleGCS):** Offloads calculation to a Ground Station.

2. Database Management (`OA_DB_...`)

If using Dijkstra, the drone builds a "Database" of known obstacles.

- `OA_DB_SIZE`: The number of points held in the map.
- `OA_DB_EXPIRE`: How long a point stays in the map after it is no longer visible.

3. Lookahead (`OA_BR_LOOKAHEAD`)

Defines how far ahead of the vehicle the algorithm should search for obstacles.

Parameter Breakdown

- `OA_TYPE`: Algorithm selector.
- `OA_MARGIN_MAX`: The minimum distance the vehicle will attempt to maintain from any obstacle.
- `OA_OPTIONS`: Bitmask for specialized behaviors.

Integration Guide

1. **Sensors:** Requires a 360-degree Lidar (e.g., RPLidar) or multiple directional proximity sensors.
2. **Enable:** Set `OA_TYPE = 1` (BendyRuler) for general outdoor use.
3. **Margin:** Set `OA_MARGIN_MAX = 2.0` (meters) to ensure a safe buffer.

Developer Notes

- **Library:** `libraries/AC_Avoidance`.
- **Path Planning:** The OA engine calculates a "Virtual Waypoint" which is then fed to the L1 or Position Controller.

OA_BR_CONT_ANGLE (BendyRuler Continuity Angle)

DEFAULT: 75	RANGE: 0 180	UNITS: deg

BendyRuler Continuity Angle (OA_BR_CONT_ANGLE)

Description

`OA_BR_CONT_ANGLE` defines a "Trust Zone" for small maneuvers.

If the path planner suggests a turn smaller than this angle, the drone will take it immediately. If the suggested turn is larger than this angle (e.g. a sharp 90-degree turn), the drone will only commit if the new path is significantly clearer (as defined by OA_BR_CONT_RATIO). This keeps the drone's flight paths smooth and predictable.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY

OA_BR_CONT_RATIO (BendyRuler Continuity Ratio)

DEFAULT: 1.5	RANGE: 1.0 5.0

BendyRuler Continuity Ratio (OA_BR_CONT_RATIO)

Description

`OA_BR_CONT_RATIO` prevents the drone from "Indecisive" maneuvering.

When the BendyRuler algorithm finds an alternative path around an obstacle, it compares the "Clearance" (margin) of the new path to the current one. This parameter defines the required improvement. If set to 1.5, the new path must have 50% more clearance than the current path before the drone will turn toward it. This prevents the drone from constantly twitching back and forth between two similar-looking gaps.

OA_BR_LOOKAHEAD (BendyRuler Lookahead)

DEFAULT: 5.0	RANGE: 1 20	UNITS: m

BendyRuler Lookahead (OA_BR_LOOKAHEAD)

Description

`OA_BR_LOOKAHEAD` defines the "Vision Range" of the reactive path planner.

BendyRuler peeks ahead of the drone to see if the current path is blocked. If it sees an obstacle within this distance, it starts checking alternative "Probes" (slightly left, slightly right) to see if there's a clear gap.

- **Higher Value:** Earlier reactions to obstacles. Better for high-speed flight.
- **Lower Value:** Later reactions. Allows the drone to get closer to objects before maneuvering.

OA_BR_TYPE (BendyRuler Type)

DEFAULT: 1	RANGE: 0 2

BendyRuler Type (OA_BR_TYPE)

Description

`OA_BR_TYPE` restricts which way the drone can dodge.

- **1: Horizontal (Default).** The drone steers left or right to go around the obstacle. Standard behavior.
- **2: Vertical.** The drone climbs over or dives under the obstacle. (Risky if you don't have good ceiling/floor sensors).

Tuning & Behavior

- **Default Value:** 1.
- **Recommendation:** Stick to **1 (Horizontal)** for most multirotors unless you are flying in a narrow canyon where going up is the only option.

OA_DB_ALT_MIN (OADatabase Min Altitude)

DEFAULT: 0	RANGE: 0 5	UNITS: m

OADatabase Min Altitude (OA_DB_ALT_MIN)

Description

`OA_DB_ALT_MIN` prevents the drone from trying to "Avoid" the ground or grass during takeoff and landing.

If set to 1.0, the drone will ignore all sensor detections until it is more than 1 meter high. This prevents false detections from long grass or uneven terrain from triggering a panicked maneuver while the drone is still close to its launch point.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





OA_DB_BEAM_WIDTH (OADatabase Beam Width)

DEFAULT: 5.0	RANGE: 1 10	UNITS: deg

OADatabase Beam Width (OA_DB_BEAM_WIDTH)

Description

`OA_DB_BEAM_WIDTH` helps the drone estimate how "Wide" a detected object is.

Most Lidars have a divergent beam. If the sensor detects something 10 meters away, ArduPilot uses this angle to calculate the minimum radius of the obstacle. This ensures the path planner doesn't try to fly through a gap that is too small for the drone's actual size.

OA_DB_DIST_MAX (OADatabase Max Distance)

DEFAULT: 0	RANGE: 0 100	UNITS: m

OADatabase Max Distance (OA_DB_DIST_MAX)

Description

`OA_DB_DIST_MAX` sets the "Horizon" for obstacle avoidance.

Detection sensors often have long ranges, but their accuracy decreases with distance. By setting a max distance (e.g. 10m), you ensure that the autopilot only maneuvers for obstacles that are close enough to be accurately positioned and pose a real threat.

- **0 (Default):** No limit. All detected points are added to the database.

OA_DB_EXPIRE (OADatabase Item Timeout)

DEFAULT: 10	RANGE: 0 127	UNITS: s

OADatabase Item Timeout (OA_DB_EXPIRE)

Description

`OA_DB_EXPIRE` controls the "Forgetting Speed" of the drone.

In a dynamic environment (like people walking around), you want the drone to forget old obstacle locations quickly. In a static environment (like a building site), you want it to remember them longer.

- **0:** Never forget. Useful for pre-loaded static maps.
- **10 (Default):** Standard for mixed environments.

OA_DB_OUTPUT (OADatabase Output Level)

DEFAULT: 1	RANGE: 0 3

OADatabase Output Level (OA_DB_OUTPUT)

Description

`OA_DB_OUTPUT` determines what you see on your Mission Planner or QGC screen.

The drone always uses all known obstacles for its own avoidance, but sending every single point to the Ground Station can overwhelm the telemetry radio.

- **0: Disabled.** No obstacles are shown on the map.
- **1: High Importance (Default).** Only show obstacles that are very close.
- **3: All.** Show every detected point.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





OA_DB_QUEUE_SIZE (OADatabase Queue Size)

DEFAULT: 80	RANGE: 1 200

OADatabase Queue Size (OA_DB_QUEUE_SIZE)

Description

`OA_DB_QUEUE_SIZE` defines the size of the "Inbox" for sensor data.

Proximity sensors (like Lidars) send huge amounts of data. This queue holds the raw points until the flight controller has enough spare CPU time to process them into the obstacle database.

- **Higher Value:** Can handle larger bursts of data without losing points. Good for high-resolution Lidars.
- **Lower Value:** Saves RAM on small flight controllers.

OA_DB_RADIUS_MIN (OADatabase Minimum Radius)

DEFAULT: 0.01	RANGE: 0 10	UNITS: m

OADatabase Minimum Radius (OA_DB_RADIUS_MIN)

Description

`OA_DB_RADIUS_MIN` ensures that every detected point is treated as a solid object with at least some physical dimension. Even if the Lidar beam is perfectly narrow, ArduPilot will assume the obstacle has at least this much radius to provide a safety margin for the drone's frame.

OA_DB_SIZE (OADatabase Max Points)

DEFAULT: 100	RANGE: 0 10000

OADatabase Max Points (OA_DB_SIZE)

Description

`OA_DB_SIZE` sets the memory limit for the "Obstacle Memory."

When a drone uses a Lidar or Depth Camera, it sees thousands of points. The database filters these into a smaller list of distinct obstacles. This parameter determines how many of those obstacles the drone can remember at once.

- **Higher Value:** Better for complex environments (like dense forests), but uses more CPU and RAM.
- **Lower Value:** Safer for drones with limited processing power.

Tuning & Behavior

- **Default Value:** 100.
- **Reboot Required:** Yes.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





OA_MARGIN_MAX (Object Avoidance Margin)

DEFAULT: 2.0	RANGE: 0.1 10.0	UNITS: m

Object Avoidance Margin (OA_MARGIN_MAX)

Description

`OA_MARGIN_MAX` defines the "Safety Buffer" around obstacles.

When the path planner calculates a route around a tree or building, it will ensure that the drone never comes closer than this many meters to the object.

- **Higher Value:** Safer, but the drone might refuse to fly through narrow gaps (like a doorway).
- **Lower Value:** The drone will fly much closer to objects, allowing for tighter maneuvers.

Tuning & Behavior

- **Default Value:** 2.0m.
- **Recommendation:** Set this to at least **2 times** the radius of your drone (including propellers) for safety.

OA_OPTIONS (Object Avoidance Options)

DEFAULT: 0	RANGE: 0 7

Object Avoidance Options (OA_OPTIONS)

Description

`OA_OPTIONS` provides specialized toggles for the path planning system.

- **Bit 0 (1):** Enable Logging. (Useful for developers to see why a drone chose a specific path).
- **Bit 1 (2):** Use Proximity.
- **Bit 2 (4):** Use Fence.

OA_TYPE (Object Avoidance Type)

DEFAULT: 0	RANGE: 0 4

Object Avoidance Type (OA_TYPE)

Description

`OA_TYPE` activates the "Path Planner," which allows the drone to find its way around obstacles.

While standard avoidance (AVOID_ENABLE) just stops the drone, the Object Avoidance (OA) system tries to calculate a new route around the obstacle to reach the target.

- **0: Disabled.**
- **1: BendyRuler.** A fast, reactive algorithm that "Peeks" left and right to find a gap. (Best for trees and pillars).
- **2: Dijkstra.** A global planner that uses a pre-mapped "Fence" or "Database" of obstacles to find the shortest clear path. (Best for complex static environments).

Tuning & Behavior

- **Default Value:** 0.
- **Requirements:** Requires a functional proximity sensor (Lidar, Depth Camera) or a pre-loaded obstacle database.
- **Reboot Required:** Yes.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **OFF** parameter group (specifically `OFF_PCNT` or `ARSPD_OFF_PCNT`) configures the safety checks for **Airspeed Sensor Calibration**.

Key Concepts

1. Large Offset Warning

When you calibrate an airspeed sensor (zero the pressure), the autopilot compares the new zero-offset against the previous one.

- `OFF_PCNT`: If the difference would cause a change in reported airspeed greater than this percentage (relative to `AIRSPEED_MIN`), a warning is issued.
- **Purpose:** Detects if you accidentally calibrated while the pitot tube was uncovered in a breeze, which would lead to dangerous airspeed errors in flight.

Developer Notes

- **Library:** `libraries/AP_Airspeed`.

Maximum offset cal speed error ()

OFF_PCNT: Maximum offset cal speed error

Description

The maximum percentage speed change in airspeed reports that is allowed due to offset changes between calibrations before a warning is issued. This potential speed error is in percent of `AIRSPEED_MIN`.

Values

- **Range:** 0.0 to 10.0
- **Units:** %
- **Default:** 0

Description

This parameter is a safety check for airspeed sensor calibration.

- **Purpose:** It checks if a new calibration offset differs significantly from the previous one, which might indicate the pitot tube was uncovered during calibration (wind affecting the reading) or a sensor fault.
- **Mechanism:** If the new offset would cause a reported speed change greater than this percentage (relative to `AIRSPEED_MIN`), a warning is issued: "Arspd offset change large; cover and recal".
- **0:** Disabled.

Source Code

ardupilot/libraries/AP_Airspeed/AP_Airspeed.cpp

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **OFS** parameter group (specifically `OFS_TYPE` or `FOLL_OFS_TYPE`) configures the coordinate frame for **Follow Mode**.

Key Concepts

1. Frame Selection

- **0 (North-East-Down):** The offset from the leader is fixed to the world. If you set -10m X, the drone stays 10m South of the leader, regardless of which way the leader is facing.
- **1 (Relative to Heading):** The offset is fixed to the leader's nose. If you set -10m X, the drone stays 10m behind the leader's tail. If the leader turns, the drone orbits around to stay behind it.

Developer Notes

- **Library:** `libraries/AP_Follow`.
- **See Also:** FOLL for main follow configuration.

Follow offset type ()

OFS_TYPE: Follow offset type

Description

Follow offset type.

Values

Value	Meaning
0	North-East-Down (NED)
1	Relative to lead vehicle heading

- **Default:** 0 (NED)

Description

This parameter defines how the Follow Mode interprets the offset vector (X, Y, Z).

- **0 (NED):** The offsets are fixed to the compass. If X=10, the drone stays 10 meters North of the target, regardless of which way the target is facing.
- **1 (Relative):** The offsets rotate with the target. If X=10, the drone stays 10 meters *in front* of the target. If Y=10, it stays 10 meters to the *right*. This is usually the preferred mode for "Follow Me".

Source Code

ardupilot/libraries/AP_Follow/AP_Follow.cpp

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **ONESHOT** parameter group (specifically `ONESHOT_MASK`) configures the legacy **OneShot125** and **OneShot42** fast PWM ESC protocols.

OneShot was an intermediate step between standard 490Hz PWM and modern digital DShot. It uses shorter pulse widths to allow the autopilot to send motor updates faster (up to 8kHz) with less latency.

Key Concepts

1. OneShot Mask

Selects which output channels on the flight controller will use the OneShot protocol.

- **Recommendation:** Modern users should use **DShot** instead, as it is digital and more robust. OneShot is only recommended for legacy hardware that doesn't support DShot.

Developer Notes

- **Library:** `libraries/AP_HAL_ChibiOS`.
- **Selection:** This mask is often set automatically when `MOT_PWM_TYPE` is set to 1 (OneShot) or 2 (OneShot125).

Oneshot output mask ()

ONESHOT_MASK: Oneshot output mask

Description

Mask of output channels to use OneShot125 on.

Values

- **Bitmask:**
  - **Bit 0:** Servo 1
  - **Bit 1:** Servo 2
  - **...**
  - **Bit 15:** Servo 16
- **Default:** 0 (Disabled)

Description

This parameter enables the **OneShot125** ESC protocol on specific servo output channels.

- **OneShot125:** A faster analog PWM protocol (125-250µs pulse width) compared to standard PWM (1000-2000µs). It provides faster motor updates and is synchronized with the main loop.
- **Usage:** Set the bit corresponding to each motor channel connected to a OneShot-capable ESC.
- **Warning:** Do **not** enable this for standard servos or older ESCs, as the high refresh rate and short pulses will damage them.

Source Code

ardupilot/ArduPlane/Parameters.cpp

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **ONOFF** parameter group configures the behavior of **On/Off** (Bang-Bang) control for the **Antenna Tracker**.

While many trackers use high-precision servos (proportional control), some use simple DC motors or relays that can only be turned Fully On or Fully Off. This system estimates how long to run the motor to achieve the desired angle.

Key Concepts

1. Slew Rates (`ONOFF_..._RATE`)

Defines the speed (deg/s) of the tracker mechanism. Since there is no feedback from the motor itself, the autopilot assumes the tracker moves at exactly this rate.

2. Minimum Time (`ONOFF_..._MINT`)

Defines the shortest possible "pulse" or motor activation. This prevents the relays from clicking rapidly (chattering) when only a tiny correction is needed.

Parameter Breakdown

- `ONOFF_YAW_RATE`: Speed of the yaw rotation.
- `ONOFF_PITCH_RATE`: Speed of the pitch rotation.
- `ONOFF_YAW_MINT`: Minimum activation time for yaw.

Developer Notes

- **Library:** `AntennaTracker/Parameters.cpp`.
- **Context:** Only relevant if `SERVO_YAW_TYPE` or `SERVO_PITCH_TYPE` is set to 1 (On/Off).

ONOFF_PITCH_MINT (Pitch minimum movement time)

DEFAULT: 0.1f	RANGE: 0 2	UNITS: s

Pitch minimum movement time (ONOFF_PITCH_MINT)

Description

This parameter sets the minimum duration (pulse width) for activating the pitch motor in an On/Off tracker system. It prevents rapid switching or "chattering" that could damage relays or motors when small corrections are requested. If the calculated movement time is less than this value, the motor will not move at all (deadband effect) or will move for at least this long.

The Mathematics

$$\text{Pulse Width} = \max(\text{Calculated Time}, \text{ONOFF_PITCH_MINT})$$

(If correction < deadband, output is 0).

The Engineer's View

Defined in `AntennaTracker/Parameters.cpp`. Acts as a hysteresis or minimum actuation constraint.

Tuning & Behavior

- **Default Value:** 0.1 s
- **Increase:** If relays click too rapidly or motors heat up.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY

ONOFF_PITCH_RATE (Pitch rate for on/off servos)

DEFAULT: 1.0f	RANGE: 0 50	UNITS: deg/s

Pitch rate for on/off servos (ONOFF_PITCH_RATE)

Description

This parameter is used by the Antenna Tracker when controlling **On/Off** (Bang-Bang) type mechanisms, such as simple relay-driven DC motors without position feedback loops. It defines the estimated angular velocity of the pitch axis. The controller uses this to calculate how long to run the motor to achieve a desired angle change.

The Mathematics

$$\Delta t = \frac{|\Delta \text{Angle}|}{\text{ONOFF_PITCH_RATE}}$$

The Engineer's View

Defined in `AntennaTracker/Parameters.cpp`.

- **Constraint:** Requires the `SERVO_PITCH_TYPE` to be set to On/Off.
- **Accuracy:** Since this is an open-loop estimation, variations in battery voltage or wind load can affect the actual rate, potentially causing overshoot or undershoot.

Tuning & Behavior

- **Default Value:** 1.0 deg/s
- **Procedure:** Measure the time it takes for the tracker to pitch up by 90 degrees, then calculate the rate: $\text{Rate} = 90/\text{Time}$.

ONOFF_YAW_MINT (Yaw minimum movement time)

DEFAULT: 0.1f	RANGE: 0 2	UNITS: s

Yaw minimum movement time (ONOFF_YAW_MINT)

Description

This parameter sets the minimum duration for activating the yaw motor in an On/Off tracker system. See `ONOFF_PITCH_MINT` for details.

The Mathematics

$$\text{Pulse Width} = \max(\text{Calculated Time}, \text{ONOFF_YAW_MINT})$$

The Engineer's View

Defined in `AntennaTracker/Parameters.cpp`.

Tuning & Behavior

- **Default Value:** 0.1 s

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY

ONOFF_YAW_RATE (Yaw rate for on/off servos)

DEFAULT: 9.0f	RANGE: 0 50	UNITS: deg/s

Yaw rate for on/off servos (ONOFF_YAW_RATE)

Description

This parameter defines the estimated angular velocity of the Yaw axis for Antenna Trackers using **On/Off** (Bang-Bang) control mechanisms. It allows the tracker to estimate how long to activate the motor to achieve a specific heading change.

The Mathematics

$$\Delta t_\text{yaw} = \frac{|\Delta \text{Heading}|}{\text{ONOFF_YAW_RATE}}$$

The Engineer's View

Defined in `AntennaTracker/Parameters.cpp`. Matches `ONOFF_PITCH_RATE` but for the yaw axis.

Tuning & Behavior

- **Default Value:** 9.0 deg/s
- **Measurement:** Time a 360-degree rotation and calculate the rate: $360/\text{Time}$.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY

Overview

The **OPTIONS** parameter group is a generic container for bitmask settings that toggle specific features within a library or vehicle.

Because many ArduPilot features are "On/Off" toggles, the developers group them into `OPTIONS` bitmasks to save space in the parameter table.

Common Options Groups

- `GPS_DRV_OPTIONS`: Workarounds for specific GPS chips.
- `FENCE_OPTIONS`: Behavior of the geofence (e.g., enable on arming).
- `EK3_SRC_OPTIONS`: Sources for EKF3 (e.g., use compass or GPS for yaw).
- `FLIGHT_OPTIONS`: Global flight behavior overrides.

Parameter Breakdown

- `OPTIONS`: (The specific context depends on the parent folder).
  - Refer to the ArduPilot documentation for the specific meaning of each bit (Bit 0, Bit 1, etc.) for the subsystem you are investigating.

Developer Notes

- **Bitmasking:** To enable multiple options, you must add their values together (e.g., to enable Option 1 (value 1) and Option 3 (value 4), set the parameter to 5).

OPTIONS (Winch options)

DEFAULT: 0	RANGE: 0 3	UNITS: Bitmask

Winch options (OPTIONS)

Description

This parameter configures advanced behaviors for the Winch library.

The Mathematics

Bitmask.

The Engineer's View

Defined in `libraries/AP_Winch/AP_Winch.cpp`.

- **Bit 0:** Spin Free on Release. If set, the motor is disabled (coasts) when the winch is released, rather than actively driving out.
- **Bit 1:** Retry on Jam. If set, the winch will briefly reverse and try again if it detects a jam (current spike or stall).

Tuning & Behavior

- **Default Value:** 0
- **Tuning:** Enable Bit 1 if your winch mechanism is prone to sticking.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY

Overview

The **ORIENT** parameter group (specifically `ORIENT_YAW` or `FLOW_ORIENT_YAW`) defines the mechanical rotation of auxiliary sensors relative to the flight controller.

Key Concepts

1. Yaw Alignment

When mounting a sensor like an **Optical Flow** camera or a **Lidar**, the chip's internal X/Y axes must be aligned with the vehicle's Forward/Right axes.

- `ORIENT_YAW`: The rotation (in degrees) of the sensor around its vertical axis.
  - **0:** Sensor "Forward" points with the vehicle "Forward."
  - **90:** Sensor is rotated 90 degrees to the right.

Developer Notes

- **Library:** `libraries/AP_OpticalFlow`, `libraries/AP_RangeFinder`.
- **Criticality:** If this is wrong, the vehicle will attempt to correct horizontal drift in the wrong direction, leading to a "fly-away" or crash in Loiter modes.

Beacon systems rotation from north in degrees ()

ORIENT_YAW: Beacon systems rotation from north in degrees

Description

Beacon systems rotation from north in degrees.

Values

- **Range:** -180 to +180
- **Units:** deg
- **Increment:** 1
- **Default:** 0

Description

This parameter aligns the coordinate system of your Indoor Positioning System (Beacons) with the autopilot's heading.

- **Usage:** If your beacon system's "North" (Y-axis) is not aligned with True North, use this parameter to rotate the beacon frame so that the autopilot flies in the correct direction.
- **Example:** If your beacon system's X-axis points East, but you want it to point North, you would adjust this value.

Source Code

ardupilot/libraries/AP_Beacon/AP_Beacon.cpp

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **ORIGIN** parameter group (specifically `ORIGIN_LAT`, `LON`, and `ALT`) defines the global anchor point for vehicles operating in a **Local Coordinate Frame**.

This is primarily used when flying **Indoors** or in GPS-denied environments using **Beacons** (Pozyx/Marvelmind) or **Visual Odometry**.

Key Concepts

1. Anchoring the Local Map

Indoor positioning systems report position relative to a local origin (0,0,0). To allow the GCS to draw the drone on a world map, the autopilot must know where that (0,0,0) point is physically located on Earth.

- `ORIGIN_LAT` / `LON`: The Geodetic coordinates of the local origin.
- `ORIGIN_ALT`: The altitude (AMSL) of the local origin.

Developer Notes

- **Library:** `libraries/AP_Beacon`.
- **Context:** Without these parameters, the vehicle can still hover and fly relative to its starting point, but it cannot perform mission-based navigation using global waypoints.

Backup altitude (MSL) for EKF origin ()

ORIGIN_ALT: Backup altitude (MSL) for EKF origin

Description

Backup EKF origin altitude (Mean Sea Level) used when not using a positioning system.

Values

- **Units:** m
- **Default:** 0

Description

This parameter sets a fixed "Home" altitude for the EKF when no GPS or other absolute positioning system is available.

- **Usage:** Primarily used in **ArduSub** or indoor non-GPS environments where the vehicle needs a reference point to initialize its navigation filter.
- **Function:** If the EKF cannot find a global origin (e.g., via GPS lock), it will use this value (along with `ORIGIN_LAT` and `ORIGIN_LON`) to establish its local coordinate frame.

Source Code

ardupilot/ArduSub/Parameters.cpp

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





ORIGIN_LAT (EKF Backup Origin Latitude)

DEFAULT: 0	RANGE: -90 90	UNITS: deg

EKF Backup Origin Latitude (ORIGIN_LAT)

Description

`ORIGIN_LAT` (working with ORIGIN_LON) provides a "Start Map Here" coordinate for the drone's position estimator (EKF).

If you are flying indoors or using an **ArduSub** ROV without GPS, the drone doesn't know where it is on the planet. However, it still needs a global coordinate to satisfy internal math and GCS mapping. This parameter allows you to manually set that starting point.

Tuning & Behavior

- **Usage:** Set this to the coordinates of your dive site or indoor arena.
- **Significance:** Even if you only care about relative movement (e.g. "Move 5 meters left"), providing a valid origin ensures that logs and maps show your vehicle in the correct general area of the world.

ORIGIN_LON (EKF Backup Origin Longitude)

DEFAULT: 0	RANGE: -180 180	UNITS: deg

EKF Backup Origin Longitude (ORIGIN_LON)

Description

Provides the manual longitude coordinate for an EKF origin without GPS.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





ON-SCREEN DISPLAY CONFIGURATION (OSD)

Overview

The **OSD** parameter group configures the **On-Screen Display** subsystem. This library is responsible for rendering real-time flight data (HUD) onto the video feed, whether using an analog overlay chip (MAX7456) or digital telemetry (MSP).

ArduPilot supports up to 6 custom OSD screens, allowing the pilot to toggle between different layouts (e.g., a "Minimal" screen for racing and a "Full" screen for long-range).

Key Concepts

1. OSD Hardware (`OSD_TYPE`)

Defines the rendering method.

- **1 (MAX7456):** Standard analog OSD chip found on most classic flight controllers.
- **3 (MSP):** Digital OSD for DJI, HDZero, and Avatar systems.
- **5 (SITL):** Graphical OSD overlay in the simulator.

2. Screen Management

- `OSDn_ENABLE`: Toggles screens 1 through 6.
- `OSD_SW_METHOD`: Defines how to switch between screens (RC channel, push-button, or automatic based on flight stage).

3. Widget Placement (`OSD_W_...`)

Every element (Battery, Altitude, Speed) is a "Widget" that can be moved anywhere on the screen grid.

- `OSD_W_X` / `Y`: Coordinates (typically 0-29 columns, 0-15 rows).
- `OSD_W_EN`: Enables/Disables that specific element.

4. In-Flight Parameter Tuning

The OSD can display a "Tuning Menu" (Screen 5), allowing the pilot to change PID gains, rates, and limits using the RC sticks without needing a laptop.

Parameter Breakdown

- `OSD_UNITS`: Selects Metric, Imperial, or Aviation units.
- `OSD_FONT`: Selects the character set (Regular, Bold, Large).
- `OSD_MSG_TIME`: Duration of pop-up alerts.

Integration Guide

Setting up MSP (Digital) OSD

1. **Hardware:** Connect VTX to a UART.
2. **Serial:** Set `SERIALx_PROTOCOL = 33` (MSP).
3. **Config:** Set `OSD_TYPE = 3`.
4. **Reboot:** You should see the ArduPilot logo in your goggles.
5. **Layout:** Use the "OSD" tab in Mission Planner to drag-and-drop elements to your preferred positions.

Developer Notes

- **Library:** `libraries/AP_OSD`.
- **Extensibility:** New widgets can be added to the library to support specialized sensors (e.g., LiDAR distance, EFI status, or RPM).

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





OSD1_CHAN_MAX (OSD Screen Switch PWM Max (Instance 1))

DEFAULT: 2100	RANGE: 900 2100	UNITS: PWM

OSD Screen Switch PWM Max (OSD1_CHAN_MAX)

Description

Defines the upper PWM limit for selecting this OSD screen.

OSD1_CHAN_MIN (OSD Screen Switch PWM Min (Instance 1))

DEFAULT: 900	RANGE: 900 2100	UNITS: PWM

OSD Screen Switch PWM Min (OSD1_CHAN_MIN)

Description

`OSD1_CHAN_MIN` (and `OSD1_CHAN_MAX`) defines the PWM "Window" that activates this specific OSD screen.

If OSD_SW_METHOD is set to **1 (PWM_RANGE)**, the autopilot looks at the current value of the OSD_CHAN. If the value is between MIN and MAX for a screen, that screen is shown.

OSD1_ENABLE (OSD Screen Enable (Instance 1))

DEFAULT: 1	RANGE: 0 1

OSD Screen Enable (OSD1_ENABLE)

Description

`OSD1_ENABLE` (and its siblings OSD2..OSD6) determines if a specific "Screen Layout" is active.

ArduPilot supports up to 6 different OSD screens. You can switch between them using an RC channel (OSD_CHAN). This allows you to have one screen for takeoff (with flight timers and battery), another for long-range cruising (with GPS and efficiency stats), and a third "Minimalist" screen for unobstructed cinematic flight.

- **1: Enabled.** The screen is available and can be displayed.
- **0: Disabled.** The screen is ignored.

Tuning & Behavior

- **Default:** 1 for Screen 1, 0 for others.
- **Switching:** If you enable multiple screens, you must configure OSD_CHAN to toggle between them.

OSD1_ESC_IDX (OSD ESC Instance Selection (Instance 1))

DEFAULT: 0	RANGE: 0 32

OSD ESC Instance Selection (OSD1_ESC_IDX)

Description

`OSD1_ESC_IDX` determines which motor's data is shown if you have the ESCTEMP or ESCRPM widgets active.

- **0: Auto.** Shows the ESC with the highest value (e.g. the hottest motor or the fastest spin).
- **1-32:** Fixes the display to a specific motor index.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





OSD1_FONT (Sets the font index for this screen (MSP DisplayPort only))

DEFAULT: 0	RANGE: 0 21

Sets the font index for this screen (MSP DisplayPort only)

Note: This parameter functions identically to OSD_FONT.

OSD1_TXT_RES (OSD Screen Resolution (Instance 1))

DEFAULT: 0	RANGE: 0 2

OSD Screen Resolution (OSD1_TXT_RES)

Description

`OSD1_TXT_RES` defines the pixel/character density of the display.

While analog OSDs are fixed in resolution, digital systems like **DJI** or **Avatar** can support higher densities. This parameter allows you to choose between standard and high-definition layouts.

- **0: 30×16.** Standard Definition (Analog compatibility).
- **1: 50×18.** High Definition.
- **2: 60×22.** Ultra-High Definition.

Tuning & Behavior

- **Significance:** Only applicable to digital OSD backends (MSP DisplayPort). Changing this will rearrange your widgets because the coordinate system (X/Y) depends on the grid size.

OSD2_CHAN_MAX (Transmitter switch screen maximum pwm)

DEFAULT: 2100	RANGE: 900 2100

Transmitter switch screen maximum pwm

Note: This parameter functions identically to OSD1_CHAN_MAX.

OSD2_CHAN_MIN (Transmitter switch screen minimum pwm)

DEFAULT: 900	RANGE: 900 2100

Transmitter switch screen minimum pwm

Note: This parameter functions identically to OSD1_CHAN_MIN.

OSD2_ENABLE ()

DEFAULT: 0	RANGE: null

Note: This parameter functions identically to OSD1_ENABLE.

OSD2_ESC_IDX (ESC_IDX)

DEFAULT: 0	RANGE: 0 32

ESC_IDX

Note: This parameter functions identically to OSD1_ESC_IDX.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





OSD2_FONT (Sets the font index for this screen (MSP DisplayPort only))

DEFAULT: 0	RANGE: 0 21

Sets the font index for this screen (MSP DisplayPort only)

Note: This parameter functions identically to OSD_FONT.

OSD2_TXT_RES (Sets the overlay text resolution (MSP DisplayPort only))

DEFAULT: 0	RANGE: null

Sets the overlay text resolution (MSP DisplayPort only)

Note: This parameter functions identically to OSD1_TXT_RES.

OSD3_CHAN_MAX (Transmitter switch screen maximum pwm)

DEFAULT: 2100	RANGE: 900 2100

Transmitter switch screen maximum pwm

Note: This parameter functions identically to OSD1_CHAN_MAX.

OSD3_CHAN_MIN (Transmitter switch screen minimum pwm)

DEFAULT: 900	RANGE: 900 2100

Transmitter switch screen minimum pwm

Note: This parameter functions identically to OSD1_CHAN_MIN.

OSD3_ENABLE ()

DEFAULT: 0	RANGE: null

Note: This parameter functions identically to OSD1_ENABLE.

OSD3_ESC_IDX (ESC_IDX)

DEFAULT: 0	RANGE: 0 32

ESC_IDX

Note: This parameter functions identically to OSD1_ESC_IDX.

OSD3_FONT (Sets the font index for this screen (MSP DisplayPort only))

DEFAULT: 0	RANGE: 0 21

Sets the font index for this screen (MSP DisplayPort only)

Note: This parameter functions identically to OSD_FONT.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





OSD3_TXT_RES (Sets the overlay text resolution (MSP DisplayPort only))

DEFAULT: 0	RANGE: null

Sets the overlay text resolution (MSP DisplayPort only)

Note: This parameter functions identically to OSD1_TXT_RES.

OSD4_CHAN_MAX (Transmitter switch screen maximum pwm)

DEFAULT: 2100	RANGE: 900 2100

Transmitter switch screen maximum pwm

Note: This parameter functions identically to OSD1_CHAN_MAX.

OSD4_CHAN_MIN (Transmitter switch screen minimum pwm)

DEFAULT: 900	RANGE: 900 2100

Transmitter switch screen minimum pwm

Note: This parameter functions identically to OSD1_CHAN_MIN.

OSD4_ENABLE ()

DEFAULT: 0	RANGE: null

Note: This parameter functions identically to OSD1_ENABLE.

OSD4_ESC_IDX (ESC_IDX)

DEFAULT: 0	RANGE: 0 32

ESC_IDX

Note: This parameter functions identically to OSD1_ESC_IDX.

OSD4_FONT (Sets the font index for this screen (MSP DisplayPort only))

DEFAULT: 0	RANGE: 0 21

Sets the font index for this screen (MSP DisplayPort only)

Note: This parameter functions identically to OSD_FONT.

OSD4_TXT_RES (Sets the overlay text resolution (MSP DisplayPort only))

DEFAULT: 0	RANGE: null

Sets the overlay text resolution (MSP DisplayPort only)

Note: This parameter functions identically to OSD1_TXT_RES.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





OSD5_CHAN_MAX (Transmitter switch screen maximum pwm)

DEFAULT: 2100	RANGE: 900 2100

Transmitter switch screen maximum pwm

Note: This parameter functions identically to OSD1_CHAN_MAX.

OSD5_CHAN_MIN (Transmitter switch screen minimum pwm)

DEFAULT: 900	RANGE: 900 2100

Transmitter switch screen minimum pwm

Note: This parameter functions identically to OSD1_CHAN_MIN.

OSD5_ENABLE ()

DEFAULT: 0	RANGE: null

Note: This parameter functions identically to OSD1_ENABLE.

OSD5_SAVE_X (SAVE_X)

DEFAULT: 23	RANGE: 0 25

SAVE_X (OSD5_SAVE_X)

Description

This parameter defines the column number (X-coordinate) for the "SAVE" button on the OSD Parameter Tuning page (Screen 5).

The Mathematics

Grid coordinates (0-29 columns typically).

The Engineer's View

Defined in `libraries/AP_OSD/AP_OSD_ParamScreen.cpp`.

Tuning & Behavior

- **Default Value:** 23

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





OSD5_SAVE_Y (SAVE_Y)

DEFAULT: 11	RANGE: 0 15

SAVE_Y (OSD5_SAVE_Y)

Description

This parameter defines the row number (Y-coordinate) for the "SAVE" button on the OSD Parameter Tuning page (Screen 5).

The Mathematics

Grid coordinates (0-15 rows).

The Engineer's View

Defined in `libraries/AP_OSD/AP_OSD_ParamScreen.cpp`.

Tuning & Behavior

- **Default Value:** 11
- **Usage:** Move the button if it overlaps with other OSD elements.

OSD6_CHAN_MAX (Transmitter switch screen maximum pwm)

DEFAULT: 2100	RANGE: 900 2100

Transmitter switch screen maximum pwm

Note: This parameter functions identically to OSD1_CHAN_MAX.

OSD6_CHAN_MIN (Transmitter switch screen minimum pwm)

DEFAULT: 900	RANGE: 900 2100

Transmitter switch screen minimum pwm

Note: This parameter functions identically to OSD1_CHAN_MIN.

OSD6_ENABLE ()

DEFAULT: 0	RANGE: null

Note: This parameter functions identically to OSD1_ENABLE.

OSD_ARM_SCR (OSD Arm Screen)

DEFAULT: 0	RANGE: 0 4

OSD Arm Screen (OSD_ARM_SCR)

Description

`OSD_ARM_SCR` automatically switches your OSD view when you arm the drone.

Many pilots like to have a clean screen for setup but want full telemetry (Home distance, speed, time) as soon as they start flying.

- **0:** No change (Stay on current screen).
- **1-4:** Switch to Screen 1, 2, 3, or 4.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





OSD_BTN_DELAY (OSD Menu Button Delay)

DEFAULT: 300	RANGE: 0 3000	UNITS: ms

OSD Menu Button Delay (OSD_BTN_DELAY)

Description

`OSD_BTN_DELAY` determines the "Key Repeat" speed when navigating ArduPilot's On-Screen Display menus using your transmitter sticks.

If this value is too short, the menu might "skip" entries or move too fast to control. If it is too long, the menu will feel sluggish and unresponsive.

Tuning & Behavior

- **Default:** 300 ms.
- **Recommendation:** Leave at **300ms**. If you find it difficult to select specific items in the OSD setup menu, increase to **500ms**.

OSD_CELL_COUNT (OSD Battery Cell Count)

DEFAULT: -1	RANGE: -1 14

OSD Battery Cell Count (OSD_CELL_COUNT)

Description

`OSD_CELL_COUNT` tells the OSD how many cells are in your battery.

This is critical for the "Average Cell Voltage" display and warning. If you set this incorrectly, the OSD will report the wrong voltage per cell and might flash "Low Battery" warnings when the battery is actually full.

- **-1 (Default):** Auto-detect based on voltage at boot. (Assumes you plug in a fully charged battery).
- **1-14:** Manually specify the cell count (e.g. 4 for a 4S pack).

Tuning & Behavior

- **Recommendation:** If you sometimes fly with partially discharged batteries, **manually set this value** to your battery's cell count to avoid detection errors.

OSD_CHAN (OSD Screen Switch Channel)

DEFAULT: 0	RANGE: 0 16

OSD Screen Switch Channel (OSD_CHAN)

Description

`OSD_CHAN` allows you to change what information you see on your goggles using a switch on your transmitter.

ArduPilot supports up to 4 different "Screens" (e.g., Screen 1 for takeoff, Screen 2 for cruising, Screen 3 for racing). By assigning a switch to this channel, you can cycle through them while flying.

- **0:** Disabled. Only Screen 1 is shown.
- **1-16:** RC Channel number.

OSD_DSARM_SCR (OSD Disarm Screen)

DEFAULT: 0	RANGE: 0 4

OSD Disarm Screen (OSD_DSARM_SCR)

Description

`OSD_DSARM_SCR` switches the OSD view when you land and disarm. This is often used to show a "Flight Summary" screen with max speed, total distance, and battery used.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY

OSD_FONT (OSD Font)

DEFAULT: 0	RANGE: 0 5

OSD Font (OSD_FONT)

Description

`OSD_FONT` allows you to change the visual style of the text and icons on your video feed.

Most flight controllers store multiple font sheets in their flash memory.

- **0:** Default (Clarity / Betaflight style).
- **1:** Bold.
- **2:** Digital.
- **3:** Mavlink (Data heavy).

Tuning & Behavior

- **Default Value:** 0.
- **Note:** If you see garbled characters or Chinese symbols instead of battery icons, your font sheet might be corrupted or incompatible with the firmware version. Upload a new font via the GCS.

OSD_FS_SCR (OSD Failsafe Screen)

DEFAULT: 0	RANGE: 0 4

OSD Failsafe Screen (OSD_FS_SCR)

Description

`OSD_FS_SCR` switches the OSD view when an emergency (Failsafe) occurs. This allows you to have a dedicated "Emergency" screen that only shows critical info like GPS home direction and battery voltage in a high-visibility layout.

OSD_H_OFFSET (OSD Horizontal Offset)

DEFAULT: 32	RANGE: 0 63

OSD Horizontal Offset (OSD_H_OFFSET)

Description

`OSD_H_OFFSET` centers the OSD horizontally.

- **Adjustment:** Increase the value to shift the OSD **Right**. Decrease to shift it **Left**.

OSD_MSG_TIME (OSD Message Duration)

DEFAULT: 10	RANGE: 0 60	UNITS: s

OSD Message Duration (OSD_MSG_TIME)

Description

`OSD_MSG_TIME` controls how long text notifications (like "Flight Mode: Loiter" or "Bad Compass Health") stay on screen.

- **10 (Default):** 10 seconds. Good for ensuring you see the message.
- **3:** Short duration. Good for experts who want a clean feed.

Tuning & Behavior

- **Default Value:** 10s.
- **Recommendation:** Reduce to **5** if you find the text covers up important video details for too long.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





OSD_OPTIONS (OSD Options)

DEFAULT: 0	RANGE: 0 1024

OSD Options (OSD_OPTIONS)

Description

OSD_OPTIONS toggles advanced display features.

Tuning & Behavior

- **Bit 0:** Decimal pack voltage (Show 12.6V instead of 12V).
- **Bit 1:** Invert pitch ladder.
- **Bit 2:** Use alternative crosshair.

OSD_PARAM_EN (OSD Parameter Entry Enable)

DEFAULT: 0	RANGE: 0 1

OSD Parameter Entry Enable (OSD_PARAM_EN)

Description

OSD_PARAM_EN toggles the visibility of a specific parameter on the OSD's in-flight tuning menu.

ArduPilot supports special "Parameter Screens" where you can adjust PIDs or other settings while flying using your transmitter sticks. This parameter determines if a specific slot in that menu is active.

- **0: Disabled.**
- **1: Enabled.**

OSD_PARAM_GRP (OSD Parameter Group)

DEFAULT: -1	RANGE: -1 1000

OSD Parameter Group (OSD_PARAM_GRP)

Description

OSD_PARAM_GRP is a technical identifier that helps the OSD system locate the correct group of parameters (e.g. all ATC_ parameters) to which the tuned setting belongs.

OSD_PARAM_IDX (OSD Parameter Index)

DEFAULT: -1	RANGE: -1 255

OSD Parameter Index (OSD_PARAM_IDX)

Description

OSD_PARAM_IDX is a technical identifier used by the OSD system to find the correct parameter in the flight controller's memory. It is usually set automatically when you configure the OSD using a tool like Mission Planner.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





OSD_PARAM_KEY (OSD Parameter Key)

DEFAULT: -1	RANGE: -1 65535

OSD Parameter Key (OSD_PARAM_KEY)

Description

`OSD_PARAM_KEY` is the unique database identifier for a parameter. This key ensures that even if parameters are added or removed in future firmware versions, the OSD tuning menu continues to point at the correct setting.

OSD_PARAM_MAX (OSD Parameter Maximum Value)

DEFAULT: 100	RANGE: -10000 10000

OSD Parameter Maximum Value (OSD_PARAM_MAX)

Description

`OSD_PARAM_MAX` sets a safety ceiling for in-flight tuning. It prevents you from accidentally increasing a gain to a level that causes violent oscillations or mechanical failure.

OSD_PARAM_MIN (OSD Parameter Minimum Value)

DEFAULT: 0	RANGE: -10000 10000

OSD Parameter Minimum Value (OSD_PARAM_MIN)

Description

`OSD_PARAM_MIN` sets a safety floor for in-flight tuning.

When you are adjusting a parameter (like a PID gain) using your transmitter sticks and the OSD menu, this value prevents you from accidentally setting the parameter too low, which could make the drone unstable or unable to fly.

OSD_PARAM_TYPE (OSD Parameter Type)

DEFAULT: 0	RANGE: 0 7

OSD Parameter Type (OSD_PARAM_TYPE)

Description

`OSD_PARAM_TYPE` tells the OSD how to format the value of a tuned parameter.

Normally, parameters are just numbers. However, some parameters (like `SERIAL_PROTOCOL`) are easier to understand if shown as text (e.g., "MAV" instead of "1").

- **0: None.** Show the raw numerical value.
- **1: Serial Protocol.** Show names of serial protocols.
- **2: Servo Function.** Show servo function names.
- **4: Flight Mode.** Show flight mode names.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





OSD_PARAM_X (OSD Parameter X Position)

DEFAULT: 0	RANGE: 0 59

OSD Parameter X Position (OSD_PARAM_X)

Description

`OSD_PARAM_X` sets the horizontal coordinate for a parameter editing element on the OSD display.

This is part of the in-flight parameter tuning system, allowing you to move where specific PID gains or settings appear when you enter the OSD menu.

Tuning & Behavior

- **0:** Left edge.
- **High Value:** Right edge.

OSD_PARAM_Y (OSD Parameter Y Position)

DEFAULT: 0	RANGE: 0 31

OSD Parameter Y Position (OSD_PARAM_Y)

Description

`OSD_PARAM_Y` sets the vertical coordinate for a parameter editing element on the OSD display.

OSD_P_EN (OSD Parameter Slot Enable)

DEFAULT: 0	RANGE: 0 1

OSD Parameter Slot Enable (OSD_P_EN)

Description

`OSD_P_EN` (usually appearing in groups like `OSD5_PARAM1_EN`) enables a specific slot in the in-flight tuning menu.

ArduPilot allows you to configure special OSD screens where you can adjust PIDs, rates, or other parameters while flying using your transmitter sticks. Each screen has multiple slots (typically 9). This parameter controls if a particular slot is active and visible.

- **0: Disabled.** The slot is empty.
- **1: Enabled.** The slot displays the configured parameter for adjustment.

Tuning & Behavior

- **Setup:** Once enabled, you must also set the parameter group (`OSD_P_GRP`) and index (`OSD_P_IDX`) to tell the autopilot which parameter you want to tune in this slot.
- **Safety:** Only enable the parameters you actually intend to tune during the flight to avoid accidental changes to critical settings.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





OSD_P_GRP (OSD Parameter Slot Group)

DEFAULT: 0	RANGE: 0 32767

OSD Parameter Slot Group (OSD_P_GRP)

Description

`OSD_P_GRP` identifies the "Class" or "Category" of the parameter you want to tune in an in-flight menu.

Every parameter in ArduPilot has a group ID. This is typically configured using a GCS (like Mission Planner's OSD setup screen) rather than entered manually, but this parameter stores that selection.

- **0:** Root/Global parameters.
- **Other Values:** Internal IDs for specific libraries (e.g. 1 for Attitude Control).

Tuning & Behavior

- **Setup:** This should be used in combination with OSD_P_IDX to target a specific parameter.
- **Ease of Use:** It is highly recommended to use the graphical OSD editor in your GCS to set these values.

OSD_P_IDX (OSD Parameter Slot Index)

DEFAULT: 0	RANGE: 0 127

OSD Parameter Slot Index (OSD_P_IDX)

Description

`OSD_P_IDX` specifies which individual parameter inside a group (set by OSD_P_GRP) is being controlled.

For example, if the group is "Attitude Control," the index determines if you are adjusting the Roll P, Pitch I, or Yaw D term.

Tuning & Behavior

- **Setup:** Use the graphical OSD editor in your GCS to find the correct index for the parameter you wish to tune.

OSD_P_INCR (OSD Parameter Slot Increment)

DEFAULT: 0.001	RANGE: 0 100

OSD Parameter Slot Increment (OSD_P_INCR)

Description

`OSD_P_INCR` defines the "Sensitivity" of the in-flight tuning.

When you push the transmitter stick to change a parameter, this is the amount added to or subtracted from the current value.

- **Small Value (e.g. 0.001):** Precision tuning for delicate PIDs.
- **Large Value (e.g. 1.0):** Quick adjustments for broad settings like MAVLink stream rates.

Tuning & Behavior

- **Recommendation:** Match the increment to the scale of the parameter. For a P-gain that is usually around 0.1, an increment of 0.005 is appropriate. For an altitude limit of 100m, an increment of 5.0 is better.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





OSD_P_KEY (OSD Parameter Slot Key)

DEFAULT: 0	RANGE: 0 32767

OSD Parameter Slot Key (OSD_P_KEY)

Description

`OSD_P_KEY` is a unique identifier used by ArduPilot to locate a parameter in its non-volatile memory (EEPROM).

This is part of the triple-identifier system (Group, Key, Index) used by the OSD tuning menu to ensure that when you change a value while flying, it is saved to the correct location and persists even after the drone is powered off.

Tuning & Behavior

- **Setup:** This value is usually calculated and set automatically by your Ground Control Station when you choose a parameter for the OSD. You should not need to edit it manually.

OSD_P_MAX (OSD Parameter Slot Maximum)

DEFAULT: 0	RANGE: -1000000 1000000

OSD Parameter Slot Maximum (OSD_P_MAX)

Description

`OSD_P_MAX` defines the upper safety limit for in-flight tuning.

This prevents you from increasing a parameter (like a P-gain) to a point where the drone becomes unstable or the servos oscillate to the point of failure.

Tuning & Behavior

- **Recommendation:** Set this to a value that is slightly above your target but still within the known safe operating limits for your specific vehicle.
- **Safety:** Always test your tuning limits in an open area with plenty of altitude.

OSD_P_MIN (OSD Parameter Slot Minimum)

DEFAULT: 0	RANGE: -1000000 1000000

OSD Parameter Slot Minimum (OSD_P_MIN)

Description

`OSD_P_MIN` defines the lower safety limit for in-flight tuning.

When you use your sticks to decrease a parameter value on the OSD, the autopilot will stop at this value. This is critical for safety to ensure you don't accidentally set a PID to zero or a negative value that could crash the vehicle.

Tuning & Behavior

- **Recommendation:** Set this to the lowest "Safe" value for the parameter you are tuning. For PIDs, this is usually a small positive number.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY

OSD_P_TYPE (OSD Parameter Tuning Type)

DEFAULT: 0	RANGE: 0 7

OSD Parameter Tuning Type (OSD_P_TYPE)

Description

`OSD_P_TYPE` tells the OSD how to format and display the parameter assigned to this tuning slot.

This is used to provide user-friendly names for certain common settings.

- **0: None.** Uses the raw parameter value.
- **1: Serial Protocol.** Displays protocol names (MAVLink, GPS, etc.).
- **2: Servo Function.** Displays function names (Motor 1, Aileron, etc.).
- **4: Flight Mode.**

Tuning & Behavior

- **Recommendation:** Set this to match the category of the parameter you are tuning to make the OSD menu easier to read while in flight.

OSD_P_X (OSD Parameter Slot X Position)

DEFAULT: 2	RANGE: 0 59

OSD Parameter Slot X Position (OSD_P_X)

Description

`OSD_P_X` defines the horizontal position of an in-flight tuning slot.

By default, these slots are aligned on the left side of the screen (`X=2`) to allow space for the parameter name and its current value to be displayed.

Tuning & Behavior

- **PAL/NTSC Width:** ~30 columns.
- **Digital Width:** ~60 columns.
- **Recommendation:** Keep these aligned in a column for readability.

OSD_P_Y (OSD Parameter Slot Y Position)

DEFAULT: 0	RANGE: 0 21

OSD Parameter Slot Y Position (OSD_P_Y)

Description

`OSD_P_Y` defines the vertical position of an in-flight tuning slot.

Since each tuning screen typically displays a list of parameters, these values are usually staggered (e.g., Slot 1 at `Y=2`, Slot 2 at `Y=4`) to create a clear vertical menu.

Tuning & Behavior

- **Analog Height:** ~16 rows.
- **Digital Height:** ~22 rows.
- **Note:** Ensure you don't overlap your parameter slots with permanent flight widgets like the Artificial Horizon or Battery voltage.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY

OSD_SB_H_OFS (OSD Sidebar Horizontal Offset)

DEFAULT: 0	RANGE: -10 10

OSD Sidebar Horizontal Offset (OSD_SB_H_OFS)

Description

`OSD_SB_H_OFS` moves the airspeed and altitude "Ladders" closer together or further apart.

This is useful for optimizing the center of your screen for the camera view while keeping vital data visible in your peripheral vision.

OSD_SB_V_EXT (OSD Sidebar Vertical Extension)

DEFAULT: 0	RANGE: 0 5

OSD Sidebar Vertical Extension (OSD_SB_V_EXT)

Description

`OSD_SB_V_EXT` makes the OSD sidebars (the tapes on the left and right) taller.

A higher value increases the number of tick marks visible above and below the center, helping you see upcoming altitude or speed changes.

OSD_SW_METHOD (OSD Switch Method)

DEFAULT: 0	RANGE: 0 2

OSD Switch Method (OSD_SW_METHOD)

Description

`OSD_SW_METHOD` controls how you cycle through the 4 available OSD screens using your RC transmitter.

You can set up different layouts (e.g., one for flight, one for landing, one blank) and switch between them in flight.

- **0: Toggle (Default).** Assign a switch to `RCx_OPTION = 224` (OSD Toggle). Each toggle moves to the next screen (1 → 2 → 3 → 4 → 1).
- **1: PWM Range.** Assign an RC channel to `OSD_CHAN`. The position of the switch/knob determines the screen (Low=1, Mid-Low=2, Mid-High=3, High=4).
- **2: Auto Switch.** The OSD automatically switches screens based on flight mode or events (requires Lua scripting or specific backend support).

Tuning & Behavior

- **Default Value:** 0 (Toggle).
- **Recommendation:** Use **0** if you have a momentary switch. Use **1** if you have a multi-position switch or a knob.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY

OSD_TYPE (OSD Type)

DEFAULT: 0	RANGE: 0 5

OSD Type (OSD_TYPE)

Description

`OSD_TYPE` tells the flight controller what kind of "Video Overlay" chip is being used.

The OSD chip is what draws the text (Altitude, Speed, Battery) over your camera's video feed.

- **0: None.** OSD is disabled.
- **1: MAX7456.** The standard analog OSD chip used on most flight controllers (e.g., Matek, Holybro).
- **3: MSP.** For digital video systems like DJI, FatShark Avatar, or HDZero.
- **5: Screen.** Integrated graphics for specialized boards.

Tuning & Behavior

- **Reboot Required:** Yes.
- **Note:** If using a digital system (DJI/Avatar), you must also set a Serial port to `SERIALx_PROTOCOL = 33` (MSP).

OSD_TYPE2 (OSD type 2)

DEFAULT: 0	RANGE: null	UNITS: null

OSD type 2

Note: This parameter functions identically to OSD_TYPE.

OSD_UNITS (OSD Display Units)

DEFAULT: 0	RANGE: 0 3

OSD Display Units (OSD_UNITS)

Description

`OSD_UNITS` sets the measurement system for your HUD.

- **0: Metric (Default).** Meters (m), Kilometers per hour (km/h).
- **1: Imperial.** Feet (ft), Miles per hour (mph).
- **2: SI.** Meters (m), Meters per second (m/s).
- **3: Aviation.** Feet (ft), Knots (kn).

Tuning & Behavior

- **Default Value:** 0.
- **Recommendation:**
  - **Science/Engineering:** Use **2 (SI)**.
  - **General Aviation:** Use **3 (Aviation)**.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





OSD_V_OFFSET (OSD Vertical Offset)

DEFAULT: 16	RANGE: 0 31

OSD Vertical Offset (OSD_V_OFFSET)

Description

`OSD_V_OFFSET` centers the OSD on your screen.

Due to differences in analog video formats (PAL vs NTSC) and camera processing, the OSD might appear too high or too low, cutting off text at the edges. This parameter shifts the entire layout vertically.

Tuning & Behavior

- **Adjustment:** Increase the value to shift the OSD **Down**. Decrease to shift it **Up**.

OSD_W_ACRVOLT (OSD Avg Cell Resting Voltage Warning)

DEFAULT: 3.6	RANGE: 0 100	UNITS: V

OSD Avg Cell Resting Voltage Warning (OSD_W_ACRVOLT)

Description

`OSD_W_ACRVOLT` combines two of ArduPilot's smartest battery monitoring features: **Average Cell Voltage** and **Resting Voltage Estimation**.

It monitors the estimated resting voltage divided by the number of cells. This is the ultimate "True Energy" gauge. It tells you the state of the battery without the confusion of voltage sag or varying cell counts. When this value drops below the threshold, the cell voltage element on your OSD will flash.

Tuning & Behavior

- **Default:** 3.6V.
- **Recommendation:** Set to **3.5V** or **3.6V** for LiPo. This provides a very consistent warning regardless of whether you are hovering or performing a full-throttle climb.
- **Prerequisite:** Requires OSD_CELL_COUNT to be set correctly.

OSD_W_AVGCELLV (OSD Average Cell Warning)

DEFAULT: 3.6	RANGE: 2.5 4.2	UNITS: V

OSD Average Cell Warning (OSD_W_AVGCELLV)

Description

`OSD_W_AVGCELLV` is a much better way to monitor battery life than total voltage.

Different drones use different batteries (4S, 6S, etc.). Total voltage changes, but cell voltage is universal.

- **3.6V (Default):** The voltage element will flash if any individual cell drops below 3.6V. This is a very safe "Land Now" voltage for LiPo batteries.

Tuning & Behavior

- **LiPo:** Set to **3.5V** or **3.6V**.
- **Li-Ion:** Set to **3.0V** or **3.1V** (they can handle lower voltages safely).
- **Requirement:** Requires OSD_CELL_COUNT to be set correctly.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY

OSD_W_BATVOLT (OSD Battery Voltage Warning)

DEFAULT: 10.0	RANGE: 0 100	UNITS: V

OSD Battery Voltage Warning (OSD_W_BATVOLT)

Description

`OSD_W_BATVOLT` is your "Fuel Low" light.

When your battery drops to this voltage, the battery readout on your screen will flash to tell you it's time to land.

Tuning & Behavior

- **Recommendation:** Set this to roughly **3.4V or 3.5V per cell** for LiPo batteries. For a 4S battery, you would set this to **14.0**.

OSD_W_EN (OSD Element Enable (General))

DEFAULT: 0	RANGE: 0 1

OSD Element Enable (OSD_W_EN)

Description

`OSD_W_EN` (usually appearing as `OSDx_WIDGET_EN`) is the master toggle for showing or hiding information on your video feed.

Each piece of data (Altitude, Voltage, Sats) has its own enable parameter. If set to 1, the element is drawn on the screen at its configured X and Y coordinates. If set to 0, that space on the screen remains clear.

- **0: Disabled.**
- **1: Enabled.**

Tuning & Behavior

- **Recommendation:** Enable only the parameters you need for your current mission to keep the video feed uncluttered.
- **Instant Update:** Changes made via MAVLink (GCS) typically appear on the screen immediately without requiring a reboot.

OSD_W_LQ (OSD Link Quality Warning)

DEFAULT: 50	RANGE: 0 100	UNITS: %

OSD Link Quality Warning (OSD_W_LQ)

Description

`OSD_W_LQ` sets the "Alarm Trigger" for your radio signal quality.

When your link quality (LQ) percentage drops below this value, the LQ readout on your goggles/screen will begin to flash. This gives you a clear visual warning that you are nearing the edge of your radio range or that you are experiencing high interference.

Tuning & Behavior

- **Default:** 50%.
- **Recommendation:** Set to **70** if you want an early warning. Set to **30** if you are comfortable flying on the edge of the link capability.
- **Note:** This is based on the **Link Quality** (percentage of successful packets), which is generally a more reliable indicator of link health than simple RSSI (signal strength).

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY

OSD_W_NSAT (OSD Satellites Warning)

DEFAULT: 9	RANGE: 0 30

OSD Satellites Warning (OSD_W_NSAT)

Description

`OSD_W_NSAT` warns you if your GPS signal is getting weak.

If the number of satellites drops below this value, the satellite icon on your OSD will flash. This is a sign that you might be flying near tall buildings or trees that are blocking the sky.

OSD_W_RESTVOLT (OSD Resting Voltage Warning)

DEFAULT: 10.0	RANGE: 0 100	UNITS: V

OSD Resting Voltage Warning (OSD_W_RESTVOLT)

Description

`OSD_W_RESTVOLT` provides a "Smart" low-battery alarm.

Unlike a simple voltage alarm that might flash every time you punch the throttle (causing "voltage sag"), ArduPilot's EKF estimates the **Resting Voltage** – what the battery would show if the motors were off. This is a much more accurate representation of the remaining energy. When this resting estimate falls below this value, the voltage readout will flash.

Tuning & Behavior

- **Default:** 10.0V.
- **Recommendation:** Set this to the same value as your OSD_W_BATVOLT. This ensures that you get a steady warning when the battery is truly empty, rather than a flickering one that only triggers under load.
- **Benefit:** Prevents "Phantom" low-battery alarms during high-current maneuvers.

OSD_W_RSSI (OSD RSSI Warning)

DEFAULT: 30	RANGE: 0 100	UNITS: %

OSD RSSI Warning (OSD_W_RSSI)

Description

`OSD_W_RSSI` is a "Low Signal" alert.

When your radio connection drops below this percentage, the signal bar on your screen will start flashing to warn you that you are nearing the limit of your control range.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





OSD_W_SNR (OSD RC Link SNR Warning)

DEFAULT: 0	RANGE: -20 10	UNITS: dB

OSD RC Link SNR Warning (OSD_W_SNR)

Description

`OSD_W_SNR` sets the "Noise Alarm" for your radio signal.

SNR (Signal-to-Noise Ratio) measures how much stronger your transmitter's signal is compared to the background electrical noise. Even if your signal is strong (high RSSI), if the noise is also high, the link will be poor. A higher SNR is better. If the SNR drops below this value, the readout on your OSD will flash.

Tuning & Behavior

- **Default:** 0 dB.
- **Context:** This is primarily used by modern digital radio links (like ELRS or Crossfire) that report SNR data.
- **Recommendation:** Consult your radio manufacturer's documentation for the minimum reliable SNR for your specific frequency and protocol (e.g., ELRS at 250Hz might need > -5 dB).
- **Note:** If your SNR is consistently low, check for noisy electronics (VTX, ESCs) mounted too close to your RC receiver.

OSD_W_TERR (OSD Terrain Height Warning)

DEFAULT: -1	RANGE: -1 1000	UNITS: m

OSD Terrain Height Warning (OSD_W_TERR)

Description

`OSD_W_TERR` is a "Low Altitude" alarm based on terrain data.

If the drone is equipped with terrain data (on the SD card) or a long-range distance sensor, it knows its height above the ground. If that height falls below this value, the terrain height readout on your screen will flash to warn you of a potential ground collision.

- **-1:** Disabled (Default).
- **Value (m):** Height in meters at which the warning triggers.

Tuning & Behavior

- **Recommendation:** Set to **10** or **20** meters if you are flying in hilly terrain to provide an early warning of rising ground.
- **Prerequisite:** Requires a valid terrain database or a functional rangefinder.

OSD_W_X (OSD Element X Position)

DEFAULT: 0	RANGE: 0 59

OSD Element X Position (OSD_W_X)

Description

`OSD_W_X` defines the horizontal position of an OSD widget.

- **For Analog (PAL/NTSC):** The screen is typically 30 columns wide.
- **For Digital (MSP):** The screen can be up to 60 columns wide depending on the resolution.
- **0:** Left edge.
- **Max:** Right edge.

Tuning & Behavior

- **Instant Update:** Changes appear immediately on the screen.
- **Validation:** If an element disappears, check that the X value is not higher than your OSD's maximum width.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





OSD_W_Y (OSD Element Y Position)

DEFAULT: 0	RANGE: 0 21

OSD Element Y Position (OSD_W_Y)

Description

`OSD_W_Y` defines the vertical position of an OSD widget.

- **For Analog (PAL/NTSC):** The screen is typically 16 rows high.
- **For Digital (MSP):** The screen can be up to 22 rows high.
- **0:** Top edge.
- **Max:** Bottom edge.

Tuning & Behavior

- **Instant Update:** Changes appear immediately on the screen.
- **Validation:** If an element disappears, check that the Y value is not higher than your OSD's maximum height.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **OUT** parameter group (specifically within **AP_Periph** firmware) configures the motor and servo **Output Protocols**.

When using a DroneCAN peripheral to drive servos or ESCs, these parameters define how those electrical signals are generated and managed.

Key Concepts

1. ESC Protocols (`OUT_BLH_...`)

Configures **BLHeli_32** and **BLHeli_S** support for ESCs connected to the peripheral node.

- `OUT_BLH_MASK`: Bitmask of which outputs should use BLHeli-compatible PWM.
- `OUT_BLH_BDMASK`: Enables Bidirectional DShot (RPM telemetry).

2. DShot Configuration (`OUT_DSHOT_RATE`)

Sets the digital speed of the motor protocol (e.g., 150, 300, 600).

3. GPIO Conversion (`OUT_GPIO_MASK`)

Allows some PWM pins on the peripheral to be converted to digital High/Low pins.

Developer Notes

- **Library:** `Tools/AP_Periph`.
- **Performance:** DShot rates are limited by the MCU speed of the peripheral node (typically F4 or G4).

OUT1_FUNCTION (Relay function)

DEFAULT: 0	RANGE: 0 100

Relay function (OUT1_FUNCTION)

Description

This parameter defines the function of the **Relay 1** output. Relays are simple On/Off (GPIO) switches used for things like camera triggers, parachutes, or powering lights.

The Mathematics

N/A. Function Mapping.

The Engineer's View

Defined in `libraries/AP_Relay/AP_Relay_Params.cpp`. Matches the `Relay::function` enum.

- **0:** None
- **1:** Relay (General purpose)
- **...**

Tuning & Behavior

- **Default Value:** 0 (None)
- **Set to:** 1 (Relay) to control via `RELAY_PIN`.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





OUT_32_ENABLE (Enable pullup after altitude wait)

DEFAULT: 0	RANGE: null	UNITS: null

Enable pullup after altitude wait

Note: This parameter configures instance 32_ENABLE. It functions identically to SERVO_32_ENABLE.

OUT_BLH_3DMASK (BLHeli 3D Mode Mask)

DEFAULT: 0	RANGE: 0 4294967295

BLHeli 3D Mode Mask (OUT_BLH_3DMASK)

Description

`OUT_BLH_3DMASK` enables "3D Mode" (reversible throttle) for specific motors.

In standard flight, 0 throttle is stop, 100% is full speed.
In 3D mode, 50% throttle is stop, 100% is full forward, and 0% is full reverse.

Tuning & Behavior

- **Default Value:** 0 (Disabled).
- **Usage:** Used for 3D aerobatic quads (flying upside down) or rovers/boats with reversible ESCs.
- **Safety:** Ensure your props are tight! Reversing direction in flight exerts huge torque on the prop nut.

OUT_BLH_AUTO (BLHeli pass-thru auto-enable for multicopter motors)

DEFAULT: 0	RANGE: null

BLHeli pass-thru auto-enable for multicopter motors

Note: This parameter configures instance BLH_AUTO. It functions identically to SERVO_BLH_AUTO.

OUT_BLH_BDMASK (BLHeli bitmask of bi-directional dshot channels)

DEFAULT: 0	RANGE: null

BLHeli bitmask of bi-directional dshot channels

Note: This parameter configures instance BLH_BDMASK. It functions identically to SERVO_BLH_BDMASK.

OUT_BLH_DEBUG (BLHeli debug level)

DEFAULT: 0	RANGE: null	UNITS: null

BLHeli debug level

Note: This parameter configures instance BLH_DEBUG. It functions identically to SERVO_BLH_DEBUG.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





OUT_BLH_MASK (BLHeli Output Mask)

DEFAULT: 0	RANGE: 0 4294967295

BLHeli Output Mask (OUT_BLH_MASK)

Description

`OUT_BLH_MASK` tells the autopilot which motors are "Smart ESCs" (BLHeli_S, BLHeli_32, or AM32).

Enabling these bits allows:

1. **DShot Pass-through:** Change ESC settings using the BLHeliSuite configurator.
2. **Telemetry:** Read RPM, Temperature, and Voltage from the ESCs.

- **Bit 0 (1):** Servo 1
- **Bit 1 (2):** Servo 2

Tuning & Behavior

- **Default Value:** 0
- **Recommendation:** Set this bitmask to match your motor output channels (e.g., **15** for a standard Quad).
- **Reboot Required:** Yes.

OUT_BLH_OTYPE (BLHeli Output Type)

DEFAULT: 0	RANGE: 0 4

BLHeli Output Type (OUT_BLH_OTYPE)

Description

`OUT_BLH_OTYPE` selects the "Language" the autopilot uses to talk to your ESCs for telemetry.

- **0: None.** No telemetry.
- **1: BLHeli_S / BLHeli_32.** Standard DShot telemetry.
- **4: AM32.** For AM32 based ESCs.

Tuning & Behavior

- **Default Value:** 0
- **Recommendation:** Match your ESC hardware. Most modern ESCs use **1**.

OUT_BLH_POLES (Nr. electrical poles)

DEFAULT: 14	RANGE: 2 50

Nr. electrical poles

Note: This parameter configures instance BLH_POLES. It functions identically to SERVO_BLH_POLES.

OUT_BLH_PORT (Control port)

DEFAULT: 0	RANGE: null	UNITS: null

Control port

Note: This parameter configures instance BLH_PORT. It functions identically to SERVO_BLH_PORT.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





OUT_BLH_RVMASK (BLHeli Reversed Mask)

DEFAULT: 0	RANGE: 0 4294967295

BLHeli Reversed Mask (OUT_BLH_RVMASK)

Description

`OUT_BLH_RVMASK` allows you to change motor direction without soldering.

If Motor 1 is spinning clockwise but needs to spin counter-clockwise, you can simply check the box for Bit 0 in this mask. ArduPilot sends a DShot command to the ESC to invert its rotation.

Tuning & Behavior

- **Default Value:** 0.
- **Recommendation:** Use this during setup to fix motor directions.
- **Note:** Only works with DShot-capable ESCs (BLHeli_S, BLHeli_32, AM32).

OUT_BLH_TEST (BLHeli internal interface test)

DEFAULT: 0	RANGE: null	UNITS: null

BLHeli internal interface test

Note: This parameter configures instance BLH_TEST. It functions identically to SERVO_BLH_TEST.

OUT_BLH_TMOUT (BLHeli protocol timeout)

DEFAULT: 0	RANGE: 0 300	UNITS: s

BLHeli protocol timeout

Note: This parameter configures instance BLH_TMOUT. It functions identically to SERVO_BLH_TMOUT.

OUT_BLH_TRATE (BLHeli telemetry rate)

DEFAULT: 10	RANGE: 0 500	UNITS: Hz

BLHeli telemetry rate

Note: This parameter configures instance BLH_TRATE. It functions identically to SERVO_BLH_TRATE.

OUT_DSHOT_RATE ()

OUT_DSHOT_RATE

Note: This parameter configures instance DSHOT_RATE. It functions identically to SERVO_DSHOT_RATE.

OUT_FTW_MASK (Tiltrotor mask)

DEFAULT: 0	RANGE: null	UNITS: null

Tiltrotor mask

Note: This parameter configures instance FTW_MASK. It functions identically to SERVO_FTW_MASK.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





OUT_FTW_POLES (BLHeli Motor Poles)

DEFAULT: 14	RANGE: 1 127	UNITS: null

BLHeli Motor Poles

Note: This parameter configures instance FTW_POLES. It functions identically to SERVO_FTW_POLES.

OUT_FTW_RVMASK (BLHeli bitmask of reversed channels)

DEFAULT: 0	RANGE: null	UNITS: null

BLHeli bitmask of reversed channels

Note: This parameter configures instance FTW_RVMASK. It functions identically to SERVO_FTW_RVMASK.

OUT_GPIO_MASK (Tiltrotor mask)

DEFAULT: 0	RANGE: null	UNITS: null

Tiltrotor mask

Note: This parameter configures instance GPIO_MASK. It functions identically to SERVO_GPIO_MASK.

OUT_RATE (Output Rate)

DEFAULT: 50	RANGE: 25 400	UNITS: Hz

Output Rate (OUT_RATE)

Description

`OUT_RATE` defines how many times per second the autopilot sends a new PWM signal to your servos.

- **50Hz (Standard):** Standard for analog servos.
- **400Hz (High-Speed):** Standard for digital servos.

Using a high rate with an analog servo can burn it out!

The Engineer's View

Sets the global default timer period for the PWM peripheral. Note that if you use DShot or certain specific functions (like ESC telemetry), this global rate might be overridden for specific groups of pins sharing a timer.

Tuning & Behavior

- **Default Value:** 50 Hz.
- **Digital Servos:** Set to **400** for significantly better flight performance and faster response times.
- **Analog Servos:** Keep at **50**.

OUT_ROB_POSMAX (Robotis servo position max)

DEFAULT: 4095	RANGE: 0 4095

Robotis servo position max

Note: This parameter configures instance ROB_POSMAX. It functions identically to SERVO_ROB_POSMAX.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





OUT_ROB_POSMIN (Robotis servo position min)

DEFAULT: 0	RANGE: 0 4095

Robotis servo position min

Note: This parameter configures instance ROB_POSMIN. It functions identically to SERVO_ROB_POSMIN.

OUT_SBUS_RATE (Maximum Controlled Rate)

DEFAULT: default_rate	RANGE: 1 360	UNITS: deg/s

Maximum Controlled Rate

Note: This parameter configures instance SBUS_RATE. It functions identically to SERVO_SBUS_RATE.

OUT_VOLZ_MASK (Volz Servo Mask (Alias))

DEFAULT: 0	RANGE: 0 4294967295

Volz Servo Mask (OUT_VOLZ_MASK)

Note: This parameter is an alias for SERVO_VOLZ_MASK.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **OVERRIDE** parameter group (specifically `OVERRIDE_CHAN`) configures the manual override safety feature for **ArduCopter**.

Key Concepts

1. The Override Channel

In some complex setups (like using a Joystick or external script for navigation), it is important to have a physical switch that "Forces" the autopilot to ignore the external inputs and return control to the pilot's RC transmitter.

- `OVERRIDE_CHAN`: Specifies the RC channel (1-16) mapped to an "External Input Enable" switch.
  - **Low:** Autopilot uses external MAVLink control.
  - **High:** Autopilot ignores external control and follows the RC stick positions.

Developer Notes

- **Library:** `libraries/AP_RCProtocol`.
- **Safety:** Critical for testing new companion computer code; allows the pilot to "take back the drone" instantly if the software malfunctions.

IO override channel ()

OVERRIDE_CHAN: IO override channel

Description

If set to a non-zero value then this is an RC input channel number to use for giving IO manual control in case the main FMU microcontroller on a board with a IO co-processor fails.

Values

- **Range:** 0 to 16
- **Increment:** 1
- **Default:** 0

Description

This parameter configures a safety feature for flight controllers with a separate I/O processor (IOMCU).

- **Function:** If you assign an RC channel to `OVERRIDE_CHAN`, switching that channel high (>1750 PWM) will bypass the main flight processor (FMU) and give you direct manual control of the servos via the I/O processor.
- **Use Case:** Critical for recovering a plane if the main flight code crashes or freezes. It allows you to fly manually ("stick-to-servo") using the backup I/O chip.
- **0:** Disabled.

Source Code

ardupilot/ArduPlane/Parameters.cpp

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **PARAM** parameter group (specifically `PARAM1` ... `PARAM3`) provides generic placeholders for **Custom Control** backends.

Key Concepts

1. Extensibility

To support developers who are writing custom C++ algorithms or Lua scripts, ArduPilot provides these generic floating-point parameters. This allows a developer to expose tunable gains or thresholds to the Ground Control Station without having to modify the core parameter definitions or recompile the entire firmware.

Developer Notes

- **Library:** `libraries/AC_CustomControl`.
- **Usage:** Only relevant if a custom backend (`CC_TYPE > 0`) is active.

Empty param2 ()

PARAM2: Empty param2

Description

Dummy parameter for empty custom controller backend.

Values

- **Default:** 0.0

Description

This parameter is a placeholder reserved for use by the **Empty Custom Controller** backend.

- **Purpose:** ArduPilot allows developers to implement custom attitude control logic in C++ or Lua. This parameter provides a configurable float value that can be read by that custom logic (e.g., as a gain or threshold) without needing to recompile the main code to add new parameters.
- **Normal Operation:** If you are not using a custom controller backend, this parameter has no effect.

Empty param3 ()

PARAM3: Empty param3

Description

Dummy parameter for empty custom controller backend.

Values

- **Default:** 0.0

Description

This parameter is a placeholder reserved for use by the **Empty Custom Controller** backend.

- **Purpose:** ArduPilot allows developers to implement custom attitude control logic in C++ or Lua. This parameter provides a configurable float value that can be read by that custom logic (e.g., as a gain or threshold) without needing to recompile the main code to add new parameters.
- **Normal Operation:** If you are not using a custom controller backend, this parameter has no effect.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **PHLD** parameter group configures the **PosHold** flight mode. PosHold is a hybrid mode for multicopters that combines the stick-feel of "AltHold" with the automated position-holding of "Loiter."

Key Concepts

1. Hybrid Control

- **Active Sticks:** The drone feels like it is in a stabilized/manual mode. When you push the stick, it banks and moves directly proportional to your input.
- **Released Sticks:** When you center the sticks, the autopilot automatically applies a brake maneuver and then "locks" the current position using GPS.

2. Braking (`PHLD_BRAKE_...`)

Configures the aggression of the stop when the pilot lets go of the sticks.

- `PHLD_BRAKE_RATE`: How fast (deg/s) the drone can tilt back to stop its momentum.
- `PHLD_BRAKE_ANGLE`: The maximum lean angle allowed during the braking phase.

Parameter Breakdown

- `PHLD_BRAKE_RATE`: Braking speed.
- `PHLD_BRAKE_ANGLE`: Max braking tilt.

Integration Guide

- **Feel:** If PosHold feels too "slidey" when you release the sticks, increase `PHLD_BRAKE_RATE`.
- **Safe Ops:** PosHold is excellent for FPV pilots who want a manual feel but need the drone to "stay put" if they take their hands off the controls.

Developer Notes

- **Library:** `ArduCopter/mode_poshold.cpp`.

PosHold braking angle max ()

PHLD_BRAKE_ANGLE: PosHold braking angle max

Description

PosHold flight mode's max lean angle during braking in centi-degrees.

Values

- **Range:** 2000 to 4500
- **Units:** cdeg
- **Default:** 3000 (30 degrees)

Description

This parameter sets the maximum tilt angle the drone will use to stop when you release the sticks in **PosHold** mode.

- **PosHold Mode:** Unlike Loiter (where the stick controls velocity), PosHold feels like Stabilize (stick controls angle) but stops and holds position when you let go.
- **Higher Value:** Faster stopping, but more aggressive pitch-back motion.
- **Lower Value:** Gentler, smoother stops, but takes more distance to come to a halt.

Source Code

ardupilot/ArduCopter/Parameters.cpp

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





`PosHold braking rate ()`

PHLD_BRAKE_RATE: PosHold braking rate

Description

PosHold flight mode's rotation rate during braking in deg/sec.

Values

- **Range:** 4 to 12
- **Units:** deg/s
- **Default:** 8

Description

This parameter controls how quickly the drone rotates to the braking angle when stopping in **PosHold** mode.

- **Function:** When you release the sticks, the drone needs to pitch backwards to stop. This parameter limits how fast that pitch rotation happens.
- **Higher Value:** Snappier braking action. The drone rotates quickly to the braking angle.
- **Lower Value:** Smoother onset of braking. The drone rotates slowly to the braking angle.

Source Code

ardupilot/ArduCopter/Parameters.cpp

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **PID** parameter group (specifically `PID_DZ`) configures the **Controller Deadzone**.

Key Concepts

1. Deadzone (`PID_DZ`)

In highly sensitive systems (like an indoor Blimp), the PID controllers might constantly "hunt" or jitter around the target even if the error is extremely small.

- `PID_DZ`: Defines the minimum error threshold (e.g., in meters for position) below which the controller will output zero command.
- **Purpose:** Saves battery and reduces motor wear by ignoring negligible errors.

Developer Notes

- **Vehicle:** Primarily used in **ArduBlimp**.

PID_DZ (Deadzone for the position PIDs)

DEFAULT: 0	RANGE: 0.1 1	UNITS: m

Deadzone for the position PIDs (PID_DZ)

Description

This parameter defines a spherical "Rest Zone" around the target loiter point for the Blimp. If the vehicle is within this distance of the target, the position controller outputs zero thrust. This prevents the blimp from constantly making tiny motor adjustments to correct negligible errors, which saves battery and reduces noise.

The Mathematics

$$ \text{If } |\vec{P}*{|\text{err}}| < |\text{PID}_DZ} \implies \vec{T}*{\text{cmd}} = 0 $$

The Engineer's View

Defined in `Blimp/Parameters.cpp`. Affects the output of the Loiter controller.

Tuning & Behavior

- **Default Value:** 0 (Disabled)
- **Large Blimps:** Set to 0.5m or 1.0m to allow for drift without constant actuation.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY

PILOT INPUT CONFIGURATION (PILOT)

Overview

The **PILOT** parameter group defines how the pilot's stick inputs are interpreted in assisted and autonomous flight modes.

These parameters do not affect Manual/Stabilize modes. Instead, they control the "feel" of modes like AltHold, Loiter, and Auto when the pilot uses the sticks to override or guide the autopilot.

Key Concepts

1. Vertical Speed (`PILOT_SPEED_UP` / `DN`)

In modes with automated altitude control (AltHold, Loiter), the throttle stick no longer controls raw motor power. Instead, it commands a **Climb** or **Descent** rate.

- `PILOT_SPEED_UP`: Maximum climb rate at full throttle (cm/s).
- `PILOT_SPEED_DN`: Maximum descent rate at zero throttle (cm/s).

2. Vertical Smoothness (`PILOT_ACCEL_Z`)

Limits the vertical acceleration. A lower value makes the drone feel more "floaty" and smooth when starting or stopping a climb.

3. Takeoff Behavior (`PILOT_TKOFF_ALT`)

Defines the altitude (centimeters) the vehicle will automatically climb to when an automated takeoff is triggered via the pilot's stick.

Parameter Breakdown

- `PILOT_SPEED_UP`: Climb rate (e.g., 250 = 2.5 m/s).
- `PILOT_ACCEL_Z`: Vertical acceleration limit (cm/s/s).
- `PILOT_THR_BHV`: Bitmask for throttle behaviors (e.g., feedback or spring-centered).

Integration Guide

- **Heavy Payloads:** If your drone struggles to stop a descent, you may need to reduce `PILOT_SPEED_DN` or increase `PILOT_ACCEL_Z`.
- **Precision:** If you find it hard to hover at a precise altitude, increase `PILOT_THR_FILT` to smooth out noisy RC sticks.

Developer Notes

- **Library:** `libraries/AC_AttitudeControl` and vehicle main loops.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





PILOT_ACCEL_Z (Pilot vertical acceleration)

DEFAULT: 250	RANGE: 50 500	UNITS: cm/s/s

Pilot vertical acceleration (PILOT_ACCEL_Z)

Description

`PILOT_ACCEL_Z` determines the "Snappiness" of the altitude control. It sets the limit for how fast the drone is allowed to change its vertical speed.

- **Low Value (e.g., 100):** The drone will feel heavy and slow to respond. It will take several seconds to reach its full climb speed.
- **High Value (e.g., 500):** The drone will feel very responsive and "locked-in," jumping immediately when you move the stick.

The Mathematics

This parameter limits the derivative of the vertical velocity command $\dot{V}_z$:

$$\left| \frac{dV_z}{dt} \right| \leq \text{PILOT_ACCEL_Z}$$

Tuning & Behavior

- **Default Value:** 250 cm/s/s
- **Recommendation:**
  - **Smooth Cinematic Flight:** Reduce to **100 or 150**.
  - **Aggressive Acro/Racing:** Increase to **400 or 500**.
- **Vibration:** If your drone has high vibration, setting this too high can cause "twitches" in altitude as the controller reacts too aggressively to sensor noise.

PILOT_SPEED_DN (Pilot maximum vertical speed descending)

DEFAULT: 0	RANGE: 0 500	UNITS: cm/s

Pilot maximum vertical speed descending (PILOT_SPEED_DN)

Description

`PILOT_SPEED_DN` determines how fast your drone descends when you pull the throttle stick down in modes like **AltHold** or **Loiter**.

- **0 (Default):** Use the same value as `PILOT_SPEED_UP` (usually 250 cm/s).
- **Custom Value:** Sets a specific limit for descent.

Warning: Rapid descents can be dangerous for multicopters. If you descend too fast through your own propeller wash, the drone can enter **Vortex Ring State (VRS)**, causing it to wobble violently and potentially flip or crash.

Tuning & Behavior

- **Default Value:** 0
- **Recommendation:** Set to **150 or 200** if you want safer, slower descents. Never set this higher than 500 unless you have a high-speed racing drone and know how to recover from a stall.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





PILOT_SPEED_UP (Pilot maximum vertical speed ascending)

DEFAULT: 250	RANGE: 50 500	UNITS: cm/s

Pilot maximum vertical speed ascending (PILOT_SPEED_UP)

Description

`PILOT_SPEED_UP` determines how fast your drone climbs when you push the throttle stick all the way up in modes like **AltHold**, **Loiter**, and **PosHold**.

- **250 (Default):** The drone will climb at 2.5 meters per second (m/s).

Unlike **Stabilize** mode (where the stick directly controls motor power), in autonomous modes, the stick controls the "Requested Vertical Velocity." This parameter acts as the speed limiter for that request.

The Engineer's View

Defined in `ArduCopter/Parameters.cpp`.
The vertical velocity controller in `AC_PosControl` takes the pilot's normalized stick input ($S \in [-1, 1]$) and maps it to a target velocity V_z:

$$V_z_target = S_throttle \cdot \text{PILOT_SPEED_UP}$$

Tuning & Behavior

- **Default Value:** 250 cm/s
- **Recommendation:**
  - **Cinematography:** Reduce to **150 or 200** for smoother, more cinematic ascents.
  - **Sport / Inspection:** Increase to **400 or 500** for rapid altitude changes.
- **Note:** If your drone is underpowered, setting this too high will cause it to lose stability during a climb as the motors hit their maximum output and can no longer maintain attitude (Tilt).

PILOT_STEER_TYPE (Pilot Steering Type)

DEFAULT: 0	RANGE: 0 1

Pilot Steering Type (PILOT_STEER_TYPE)

Description

`PILOT_STEER_TYPE` defines how your transmitter sticks move the Rover or Boat.

- **0: Default.** The steering stick controls the vehicle's turn rate or wheel angle.
- **1: Two-Paddle.** (Used for skid-steer). Left stick controls the left motor, right stick controls the right motor. This provides a direct, "old-school" tank control feel.

Tuning & Behavior

- **Default:** 0.
- **Recommendation:** Use **0** for most vehicles, as it allows the autopilot to perform stabilization and speed control more effectively. Only use **1** if you are specifically trained on dual-throttle tank controls.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





PILOT_THR_BHV (Pilot Throttle Behavior)

DEFAULT: 0	RANGE: 0 1

Pilot Throttle Behavior (PILOT_THR_BHV)

Description

`PILOT_THR_BHV` allows you to customize how the autopilot interprets your throttle stick, particularly useful for "Self-Centering" transmitters.

- **Bit 0 (1): Center-Thrust.** Use this if your transmitter's throttle stick has a spring and always returns to the center (common on DJI-style or consumer controllers). When enabled, the drone assumes "Neutral" is in the middle of the stick's travel. This allows for easier altitude hold and prevents the drone from accidentally landing or jumping if you let go of the stick.

Tuning & Behavior

- **Default:** 0 (Standard throttle - 0% at bottom).
- **Recommendation:** Only enable if you have a physical spring on your throttle stick.

PILOT_THR_FILT (Pilot Throttle Filter)

DEFAULT: 0	RANGE: 0 10.0	UNITS: Hz

Pilot Throttle Filter (PILOT_THR_FILT)

Description

`PILOT_THR_FILT` smooths your manual throttle movements.

If you have "shaky" hands or a transmitter with noisy potentiometers, this filter prevents the drone's motors from reacting to every tiny, unintended vibration in the stick.

- **0: Disabled (Default).**
- **Value (Hz):** Cutoff frequency. A value of **2.0 Hz** provides good smoothing without making the throttle feel "laggy."

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





PILOT_TKOFF_ALT (Pilot Takeoff Altitude)

DEFAULT: 0	RANGE: 0 1000	UNITS: cm

Pilot Takeoff Altitude (PILOT_TKOFF_ALT)

Description

`PILOT_TKOFF_ALT` enables the "Automatic Launch" feature for manual flight modes.

Normally, in modes like **Loiter** or **AltHold**, you have to manually push the throttle stick up to lift the drone off the ground. With this parameter, you can simply arm the drone and push the throttle stick slightly above the center (or use a GCS button). The drone will then automatically climb to this specific height and wait for your next command.

- **Units:** Centimeters.
- **0 (Default):** Feature is disabled. Standard manual throttle control for takeoff.
- **200:** The drone will automatically climb to 2 meters (6.5 ft) above the ground.

The Engineer's View

Defined in `ArduCopter/Parameters.cpp`.
When the drone is armed and landed in a semi-autonomous mode (Loiter, AltHold, PosHold, FlowHold), the position controller monitors the throttle stick. If the stick is moved above the `THR_DZ` (Deadzone) and `PILOT_TKOFF_ALT` is non-zero, the `takeoff.start()` function is called.

```
// ArduCopter/mode_loiter.cpp
if (ap.land_complete && throttle_stick > center_stick) {
    takeoff.start(constrain_float(g.pilot_takeoff_alt, 0.0f, 1000.0f));
}
```

The drone then executes a kinematically shaped vertical path to the target altitude, using the `PSC_JERK_Z` and `PSC_ACCEL_Z` limits.

Tuning & Behavior

- **Default Value:** 0 (Manual Takeoff).
- **Recommendation:** Set to **200 (2 meters)**. This is a safe "Head Height" that ensures the drone is clear of ground-effect turbulence and obstacles before the pilot takes over full manual control.
- **Safety:** Always ensure the area above the drone is clear. If you trigger an automatic takeoff indoors, the drone will hit the ceiling if `PILOT_TKOFF_ALT` is set too high!

Troubleshooting

- **Scenario:** My drone "jumps" into the air as soon as I move the throttle stick just a little bit.
  - **Diagnosis:** `PILOT_TKOFF_ALT` is set to a non-zero value, and you are triggering the auto-launch.
  - **Fix:** Set to **0** if you prefer manual throttle control during takeoff.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **PITCH** parameter group handles configuration for the **Pitch (Tilt) Axis** across various vehicle and peripheral types.

Key Concepts

1. Tracker Speed (`PITCH_SLEW_TIME`)

(Antenna Tracker) This parameter limits the maximum angular velocity of the tilt mechanism.

- **Purpose:** Prevents mechanical damage to heavy high-gain antennas by ensuring the servos don't jerk the assembly too violently.
- **Units:** Time (seconds) to traverse 180 degrees.

2. Trim (`PITCH_TRIM`)

(Helicopter) Defines the collective pitch trim value required to maintain hover.

Developer Notes

- **Context:** `PITCH_SLEW_TIME` is specific to **AntennaTracker**. `PITCH_TRIM` is common in **ArduCopter Heli**.

`PITCH_SLEW_TIME` (Antenna Tracker Pitch Slew Time)

DEFAULT: 2	RANGE: 0 20	UNITS: s

Antenna Tracker Pitch Slew Time (PITCH_SLEW_TIME)

Description

`PITCH_SLEW_TIME` acts as a speed governor for the Antenna Tracker's tilt axis.

`PITCH_TRIM` (Antenna Tracker Pitch Trim)

DEFAULT: 0	RANGE: -10 10	UNITS: deg

Antenna Tracker Pitch Trim (PITCH_TRIM)

Description

`PITCH_TRIM` allows you to tilt the Antenna Tracker's beam up or down manually.

This is useful if your tracker mount is not perfectly level, or if there is a consistent error in the altitude reported by the drone or the tracker's own sensors. By adding a few degrees of trim, you can center the antenna's signal beam exactly on the drone.

Tuning & Behavior

- **Default:** 0.
- **Usage:** If you find the signal is strongest when you manually tilt the antenna slightly, enter that offset here so the software can handle it automatically.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **PLDP** parameter group configures the **Autonomous Payload Placement** (Payload Place) feature. This mode allows the drone to descend over a target, detect when the payload has touched the ground (e.g., via slack in the winch or a weight-off-hook sensor), and autonomously release it.

Key Concepts

1. Touchdown Detection (`PLDP_THRESH`)

The system monitors the motor thrust required to hover. When the payload touches the ground, the drone becomes "lighter," and the required thrust drops.

- `PLDP_THRESH`: The percentage drop in thrust required to trigger the "Touchdown" state.

2. Descent Speed (`PLDP_SPEED_DN`)

Defines the slow, controlled speed used for the final drop-off.

Parameter Breakdown

- `PLDP_THRESH`: Thrust change threshold.
- `PLDP_DELAY`: Time to wait after touchdown before releasing the gripper.
- `PLDP_RNG_MAX`: Maximum altitude allowed to start the placement procedure.

Developer Notes

- **Library:** `libraries/AC_PayloadPlace`.
- **Mission Item:** Triggered by the `NAV_PAYLOAD_PLACE` waypoint.

PLDP_RNG_MAX (Payload-Place Max Range)

DEFAULT: 0	RANGE: 0 100	UNITS: m

Payload-Place Max Range (PLDP_RNG_MAX)

Description

`PLDP_RNG_MAX` is a configuration setting for the **Payload Place** autonomous mission command.

In "Payload Place" mode, the drone descends until it detects that the package has touched the ground. This detection relies on a rangefinder (Lidar) to see how far away the ground is. This parameter tells the autopilot: "Only start looking for the ground when the rangefinder says we are closer than this height."

- **0:** Disabled.
- **Value (m):** The height threshold (e.g. 5.0m) to begin the final placement phase.

Tuning & Behavior

- **Safety:** This prevents the drone from misinterpreting rangefinder "noise" at high altitudes as a landing event.

PLDP_SPEED_DN (Payload-Place Descent Speed)

DEFAULT: 50	RANGE: 10 500	UNITS: cm/s

Payload-Place Descent Speed (PLDP_SPEED_DN)

Description

`PLDP_SPEED_DN` sets the vertical speed of the drone as it lowers the package to the ground.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY

`Payload Place climb delay` ()

PLDP_DELAY: Payload Place climb delay

Description

Delay after release, in seconds, before aircraft starts to climb back to starting altitude.

Values

- **Range:** 0 to 120
- **Units:** s
- **Default:** 0.0

Description

This parameter is used during the "Payload Place" mission command.

- **Function:** After the drone detects the payload has touched down (via thrust loss or rangefinder check) and the gripper has released, the drone will wait for this many seconds on the ground (or hovering low) before ascending to return to the mission altitude.
- **Usage:** Useful to ensure the load is fully detached and the hook is clear before lifting off.

`Payload Place thrust ratio threshold` ()

PLDP_THRESH: Payload Place thrust ratio threshold

Description

Ratio of vertical thrust during descent below which payload touchdown will trigger.

Values

- **Range:** 0.5 to 0.9
- **Default:** 0.9

Description

This parameter sets the sensitivity for detecting when the payload has touched the ground.

- **Mechanism:** As the drone lowers the payload, the weight on the motors decreases. When the throttle output drops to this percentage of the average hover throttle (while descending), the autopilot assumes the load is now supported by the ground.
- **Example:** If hover throttle is 50% and `PLDP_THRESH` is 0.9, detection triggers when throttle drops to 45% (0.9 * 50%).
- **Lower Value:** Harder to trigger. Requires more weight off-loading (slacker line) before release.
- **Higher Value:** Easier to trigger. Risk of premature release if the drone descends quickly.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





PRECISION LANDING CONFIGURATION (PLND)

Overview

The **PLND** parameter group configures the **Precision Landing** system. This library uses visual sensors (IR Beacons, AprilTags, or companion computer vision) to land the vehicle on a specific target with centimeter-level accuracy, regardless of GPS drift.

Key Concepts

1. Landing Target Sensors (`PLND_TYPE`)

Defines the hardware used to see the target.

- **1 (IR Lock):** Uses a specialized IR camera (IRLock) to track a pulsed infrared beacon.
- **2 (Companion):** Receives target position updates from a companion computer (e.g., Raspberry Pi running OpenMV or AprilTag detection).
- **3 (SITL):** Simulated landing target.

2. Estimator and Filter (`PLND_EST_TYPE`)

Determines how the target's relative position is fused.

- **0 (Raw):** Direct sensor-to-motor correction (only recommended for simple IR lock).
- **1 (Kalman Filter):** Fuses vehicle velocity and sensor data to predict target position even if the target is briefly obscured.

3. Safety Thresholds

- `PLND_XY_DIST_MAX`: If the target is further away than this, the drone will ignore it and land normally (prevents chasing a false positive).
- `PLND_ALT_MAX`: Maximum altitude to start looking for the target.

Parameter Breakdown

- `PLND_ENABLED`: Master switch.
- `PLND_YAW_ALIGN`: Rotates the vehicle to match the orientation of the target (if using oriented tags).

Integration Guide

1. **Hardware:** Mount the sensor (e.g., IRLock) facing straight down.
2. **Enable:** Set `PLND_ENABLED = 1` and `PLND_TYPE`.
3. **Mode:** Set `LAND_TYPE = 1` (Precision) if you want it to trigger every time you switch to Land mode.

Developer Notes

- **Library:** `libraries/AC_PrecLand`.
- **Accuracy:** Typically achieves < 10cm accuracy. Essential for automated charging pads or shipboard landings.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





PLND_ACC_P_NSE (Kalman Filter Accelerometer Noise)

DEFAULT: 2.5	RANGE: 0.5 5.0

Kalman Filter Accelerometer Noise (PLND_ACC_P_NSE)

Description

`PLND_ACC_P_NSE` defines the "Trust Balance" for the precision landing system.

When the drone is landing on a target, it uses two pieces of information:

1. **The Sensor:** Where the camera/IR sensor sees the target right now.
2. **The IMU:** Where the drone thinks it has moved based on its internal accelerometers.

- **Higher Value:** The drone assumes the IMU is noisy. It will react faster to new data from the landing sensor. Good for high-vibration drones.
- **Lower Value:** The drone trusts its IMU more. It will ignore small jitters in the landing sensor and move more smoothly.

The Engineer's View

This parameter sets the process noise covariance (Q) for the internal precision landing Kalman filter. It specifically weights the accelerometer inputs against the measurement updates from the vision system.

Tuning & Behavior

- **Default Value:** 2.5.
- **Oscillations:** If the drone "Twitches" or over-corrects as it approaches the landing pad, decrease this value to **1.5**.
- **Lag:** If the drone is slow to react to target movements, increase to **3.5**.

PLND_ALT_MAX (PrecLand maximum alt for retry)

DEFAULT: 8.0	RANGE: 0 50	UNITS: m

PrecLand maximum alt for retry (PLND_ALT_MAX)

Description

This parameter defines the "Active Search Zone" upper boundary for Precision Landing.

When the vehicle is descending in a precision landing mode, it expects to see the landing target (e.g., an IR beacon or AprilTag). If the vehicle is above `PLND_ALT_MAX`, it will continue to descend vertically even if it doesn't see the target. Once it drops **below** this altitude, the "Retry Logic" becomes active. If the target is lost while the vehicle is below this height, it will trigger a retry (climb back up to look for it) or a failsafe, depending on `PLND_STRICT`.

The Mathematics

This parameter acts as a height-based logic gate (H_gate):

$$\text{RetryLogicActive} = \text{IF } (Alt_RangeFinder < PLND_ALT_MAX)$$

It requires a valid downward-facing rangefinder to function. If the rangefinder is not present or unhealthy, this altitude check is bypassed or fails safe depending on `PLND_OPTIONS`.

The Engineer's View

In `AC_PrecLand::check_if_sensor_in_range()` (`libraries/AC_PrecLand/AC_PrecLand.cpp`):

1. The code checks if `_sensor_max_alt` is non-zero.
2. It verifies that the rangefinder altitude is valid.
3. If `rangefinder_alt_m > _sensor_max_alt`, it returns `false`, meaning the "Retry/Lost" logic should not yet be triggered (the vehicle is too high to reasonably expect a solid lock).
4. Once below this height, the `TargetState` can transition to `TARGET_RECENTLY_LOST` or `TARGET_OUT_OF_RANGE`.

Tuning & Behavior

- **Default Value:** 8 meters.
- **Agile Aircraft:** Can be set higher (e.g., 15m) if the camera/sensor has a very wide field of view.
- **Limited Sensors:** Should be set to the maximum reliable range of your landing sensor (e.g., IR-Lock is usually effective up to ~10-15m).
- **0:** Disables the maximum altitude check; retry logic is always active if the target is lost.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY

PLND_ALT_MIN (PrecLand minimum alt for retry)

DEFAULT: 0.75	RANGE: 0 5	UNITS: m

PrecLand minimum alt for retry (PLND_ALT_MIN)

Description

This parameter defines the "Commit Zone" lower boundary for Precision Landing.

When a vehicle is very close to the ground (e.g., during the final flare or touchdown), it is often dangerous to attempt a "Retry" (climbing back up) if the landing sensor loses the target (which often happens due to dust, ground effect, or camera focal limits).

If the vehicle is **below** `PLND_ALT_MIN`, it will ignore the fact that it lost the target and simply continue to land vertically. This ensures a safe touchdown rather than an unexpected and potentially unstable climb-out at very low altitude.

The Mathematics

This parameter acts as a low-altitude safety gate:

$$\text{InCommitZone} = \text{IF } (Alt_RangeFinder < PLND_ALT_MIN)$$

If `InCommitZone` is true, the `TargetState` remains `TARGET_FOUND` (or behaves as such for the landing logic) even if the sensor reporting fails.

The Engineer's View

In `AC_PrecLand::check_if_sensor_in_range()` (`libraries/AC_PrecLand/AC_PrecLand.cpp`):

1. The code verifies that the rangefinder altitude is valid.
2. If `rangefinder_alt_m < _sensor_min_alt`, it returns `false`, preventing the transition to a "Lost" state that would trigger a retry.
3. This is a robust way to handle the "Blind Spot" that many optical and IR sensors have when they get within a few dozen centimeters of the target.

Tuning & Behavior

- **Default Value:** 0.75 meters.
- **Tall Gear:** If your aircraft has very tall landing gear or a underslung camera, you may need to increase this slightly (e.g., 1.0m).
- **Small Drones:** Can often leave this at the default or slightly lower.
- **0:** Disables the minimum altitude check; the vehicle may attempt to retry even a few centimeters from the ground. **Not recommended.**

PLND_BUS (Sensor Bus)

DEFAULT: -1	RANGE: -1 3

Sensor Bus (PLND_BUS)

Description

`PLND_BUS` specifies where you plugged in your I2C-based landing sensor (like an IR-Lock).

- **-1 (Default):** The autopilot probes all external buses for a compatible sensor.
- **0:** Forces the internal bus.
- **1:** Forces the external bus (standard GPS/Compass port).

Tuning & Behavior

- **Default Value:** -1.
- **Recommendation:** Leave at **-1** unless you have multiple identical sensors on different buses or a specialized hardware conflict.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





PLND_CAM_POS_X (Camera X Position Offset)

DEFAULT: 0	RANGE: -5 5	UNITS: m

Camera X Position Offset (PLND_CAM_POS_X)

Description

`PLND_CAM_POS_X` tells the autopilot exactly where the camera is mounted on the drone.

For precision landing to work perfectly, the flight controller needs to know the "Lever Arm" effect. If the camera is 10cm forward of the center of gravity, and the drone tilts forward, the camera moves. Without this offset, the drone might "hunt" or circle the target because it thinks it has moved more (or less) than it actually has.

The Engineer's View

This parameter defines the longitudinal offset in the body frame.

- **Positive X:** Camera is in front of the center.
- **Negative X:** Camera is behind the center.

The values are used to transform the camera's raw sight-line vector into a global position relative to the vehicle's navigation origin.

Tuning & Behavior

- **Default Value:** 0 m.
- **Recommendation:** Measure the distance from the lens center to the center of your flight controller's IMU. Enter the value in meters (e.g. 0.1 for 10cm).

PLND_CAM_POS_Y (Camera Y Position Offset)

DEFAULT: 0	RANGE: -5 5	UNITS: m

Camera Y Position Offset (PLND_CAM_POS_Y)

Description

`PLND_CAM_POS_Y` defines the lateral offset of the landing sensor.

- **Positive Y:** Camera is to the right of center.
- **Negative Y:** Camera is to the left of center.

See PLND_CAM_POS_X for detailed explanation.

PLND_ENABLED (Precision Landing Enable)

DEFAULT: 0	RANGE: 0 1

Precision Landing Enable (PLND_ENABLED)

Description

`PLND_ENABLED` activates the "Visual Landing" system.

Standard GPS landing is accurate to about 1-2 meters. Precision Landing (PrecLand) uses a sensor (like an IRLock camera or an onboard computer) to see a target on the ground and guide the drone onto it with centimeter-level accuracy.

- **0: Disabled.**
- **1: Enabled.**

Tuning & Behavior

- **Reboot Required:** Yes.
- **Trigger:** Precision landing is typically used in the `LAND` flight mode. Once enabled, the drone will look for the target as it descends.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





PLND_EST_TYPE (Precision Land Estimator Type)

DEFAULT: 1	RANGE: 0 1

Precision Land Estimator Type (PLND_EST_TYPE)

Description

`PLND_EST_TYPE` defines how the drone interprets the data from its landing sensor.

- **0: RawSensor.** The drone flies directly toward where the sensor "sees" the target right now. (Simple, but can be twitchy).
- **1: KalmanFilter (Default).** The drone uses a mathematical model to predict where the target is, filtering out sensor noise and accounting for the drone's own movement. (Smooth and much more reliable).

The Engineer's View

Used in `AC_PrecLand::run_estimator()`.
The **Kalman Filter (1)** mode uses the `_ekf_x` and `_ekf_y` internal state machines to track the target's relative velocity and position. This allows the drone to continue "knowing" where the target is for a few seconds even if the sensor momentarily loses sight of it (e.g., due to glare).

Tuning & Behavior

- **Default Value:** 1
- **Recommendation:** Always use **1 (KalmanFilter)**. It provides much better performance and handles sensor dropouts more gracefully.

PLND_LAG (Precision Landing Sensor Lag)

DEFAULT: 0.02	RANGE: 0.02 0.250	UNITS: s

Precision Landing Sensor Lag (PLND_LAG)

Description

`PLND_LAG` compensates for the processing time of your landing camera or sensor.

Computer vision takes time. By the time a Raspberry Pi or OpenMV cam tells the flight controller "The target is at X,Y," the drone has already moved several centimeters. This parameter allows the EKF to "Rewind" its internal state to match the moment the photo was actually taken, resulting in much smoother and more accurate centering.

- **Default Value:** 0.02 seconds (20ms).
- **Too High:** The drone will "overshoot" its corrections.
- **Too Low:** The drone will feel sluggish and take longer to center.

Tuning & Behavior

- **Default Value:** 0.02s.
- **Recommendation:**
  - **IR-Lock:** 0.02s - 0.04s.
  - **Companion Computer (OpenCV):** 0.05s - 0.10s.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





PLND_LAND_OFS_X (Land offset forward)

DEFAULT: 0	RANGE: -20 20	UNITS: cm

Land offset forward (PLND_LAND_OFS_X)

Description

This parameter allows you to intentionally "offset" the final touchdown position of the vehicle along its forward/back (X) axis. While precision landing usually aims to put the camera directly over the center of the target, you might want the vehicle to land slightly forward or back from that center point.

This is particularly useful if your landing gear requires a specific alignment on a platform or if you want to avoid landing directly on top of a physical beacon.

The Mathematics

The offset is applied as a static translation in the vehicle's body-frame before being rotated into the global navigation frame.

$$ \text{Pos}*{target_final} = \text{Pos}*{target_detected} + \text{Rotation}_{BodyToNED}(\text{LAND_OFS_X}, \text{LAND_OFS_Y}, 0) $$

A **positive** value shifts the landing point **forward** (the vehicle will land with the camera ahead of the target). A **negative** value shifts it **backward**.

The Engineer's View

In `AC_PrecLand::run_output_prediction()` (`libraries/AC_PrecLand/AC_PrecLand.cpp`), the `_land_ofs_cm_x` and `_land_ofs_cm_y` values are converted to meters and added to the relative target position estimate.
This translation is performed *after* the camera's physical position offset (`CAM_POS`) has been accounted for, ensuring that `LAND_OFS` is purely a bias on the final desired landing spot.

Tuning & Behavior

- **Default Value:** 0 cm.
- **Range:** -20 to 20 cm.
- **Tuning:** If your drone consistently lands 5cm too far back from the center of the pad, set this to **5** to shift the landing point forward.
- **Note:** This is an intentional bias. If the vehicle is missing the target due to wind or poor tuning, do not use this parameter to "fix" it; instead, tune the position controller or L1 parameters.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





PLND_LAND_OFS_Y (Land offset right)

DEFAULT: 0	RANGE: -20 20	UNITS: cm

Land offset right (PLND_LAND_OFS_Y)

Description

This parameter allows you to intentionally "offset" the final touchdown position of the vehicle along its left/right (Y) axis. While precision landing usually aims to put the camera directly over the center of the target, you might want the vehicle to land slightly to the right or left of that center point.

This is useful for aligning the drone's landing gear with specific markings on a landing pad or for offset-mounted hardware.

The Mathematics

The offset is applied as a static translation in the vehicle's body-frame before being rotated into the global navigation frame.

$$ \text{Pos}*{target_final} = \text{Pos}*{target_detected} + \text{Rotation}_{BodyToNED}(0, \text{LAND_OFS_Y}, 0) $$

A **positive** value shifts the landing point to the **right**. A **negative** value shifts it to the **left**.

The Engineer's View

In `AC_PrecLand::run_output_prediction()` (`libraries/AC_PrecLand/AC_PrecLand.cpp`), the `_land_ofs_cm_y` value is converted to meters and added to the relative target position estimate.
This translation is performed *after* the camera's physical position offset (`CAM_POS`) has been accounted for, ensuring that `LAND_OFS` is purely a bias on the final desired landing spot.

Tuning & Behavior

- **Default Value:** 0 cm.
- **Range:** -20 to 20 cm.
- **Tuning:** If your drone consistently lands 5cm to the left of the beacon, set this to **5** (positive) to shift the landing point to the right.
- **Note:** This is an intentional bias. If the vehicle is missing the target due to wind or poor tuning, do not use this parameter to "fix" it; instead, tune the position controller or L1 parameters.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY

PLND_OPTIONS (Precision Landing Extra Options)

DEFAULT: 0	RANGE: null	UNITS: Bitmask

Precision Landing Extra Options (PLND_OPTIONS)

Description

This parameter is a bitmask that enables specific advanced features for the Precision Landing sub-system. It allows the pilot to customize how the vehicle handles dynamic landing scenarios, such as landing on a moving deck or responding to pilot interventions.

The Mathematics

The bits are processed as individual boolean flags:

- **Bit 0 (Value 1): Moving Landing Target.**
  Enables velocity estimation for the target. If the landing sensor detects the target is moving relative to the ground, the autopilot will attempt to match its speed for a more accurate touchdown.
- **Bit 1 (Value 2): Allow Precision Landing after manual reposition.**
  Normally, if a pilot nudges the sticks during an autonomous landing, precision landing might be disabled. Setting this bit allows the autopilot to resume tracking the target even after the pilot has manually moved the vehicle.
- **Bit 2 (Value 4): Maintain high speed in final descent.**
  Prevents the vehicle from slowing down to a very cautious descent rate as it nears the target, which is often required for landing on moving ships where "sticking the landing" quickly is safer than hovering close to a moving deck.

The Engineer's View

In `AC_PrecLand.cpp`, this maps to the `_options` member.

- **Moving Target Logic:** Checked in `AC_PrecLand::get_target_velocity()`. If bit 0 is not set, the target velocity is always assumed to be zero, which simplifies the Kalman filter state transitions.
- **Manual Reposition:** This bit is typically checked in the vehicle-specific `mode_land.cpp`.
- **High Speed Descent:** Affects the velocity constraints in the final stages of the land mission.

Tuning & Behavior

- **Bit 0 (1): Highly Recommended** if landing on a boat or a moving ground vehicle. Requires an EKF-based estimator (`PLND_EST_TYPE=1`).
- **Bit 1 (2):** Useful for search-and-rescue or research missions where a pilot might want to move the drone slightly but still use the automated target centering.
- **Bit 2 (4):** Use with caution. Only recommended for expert users landing on highly dynamic platforms.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





PLND_ORIENT (Camera Orientation)

DEFAULT: 25	RANGE: null	UNITS: Rotation

Camera Orientation (PLND_ORIENT)

Description

This parameter defines the physical mounting orientation of the camera or IR sensor used for precision landing. It tells the autopilot which direction the sensor is "looking" relative to the front of the aircraft.

Correct orientation is vital for the navigation math to work. If the sensor is pointing down but the software thinks it's pointing forward, the drone will attempt to fly in the wrong direction to center the target.

The Mathematics

The autopilot uses this rotation matrix (R_orient) to transform the raw vector reported by the sensor (V_raw) into the body-frame coordinate system (V_body).

$$V_body = R_orient \times V_raw$$

This ensures that regardless of how the sensor is mounted, the "Error X" and "Error Y" inputs to the landing controller are always relative to the vehicle's Forward and Right axes.

The Engineer's View

In `AC_PrecLand::retrieve_los_meas()` (`libraries/AC_PrecLand/AC_PrecLand.cpp`):

1. The code retrieves the unit vector from the sensor backend.
2. If the orientation is not the default `PITCH_270` (Down), it performs a multi-step rotation:
    - First, it rotates by `PITCH_90` to bring the vector from a default "downward" assumption to "forward."
    - Then, it applies the user-configured `_orient` rotation.
3. This internal normalization allows the backend drivers to report vectors in a consistent frame while the user has the flexibility to mount hardware in various configurations.

Tuning & Behavior

- **Default Value:** 25 (Rotation_Pitch_270 - Looking Down).
- **Rover Default:** 0 (Rotation_None - Looking Forward).
- **Common Values:**
    - **25:** Downward (Standard for Copter/QuadPlane).
    - **0:** Forward (Looking ahead, common for Rover or Plane-approach).
    - **4:** Backward (Looking behind).
- **Reboot Required:** Yes.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





PLND_RET_BEHAVE (PrecLand retry behaviour)

DEFAULT: 0	RANGE: 0 1	UNITS: Option

PrecLand retry behaviour (PLND_RET_BEHAVE)

Description

This parameter defines the "Targeting Strategy" used during a Precision Landing retry. If the vehicle loses sight of the landing target and `PLND_STRICT` is set to retry, the drone will climb and reposition itself to re-acquire the signal. This parameter tells the drone *where* to go during that climb.

- **0: Go to Target Location.** The drone moves to the last estimated GPS coordinate of the landing target itself.
- **1: Go to Last Vehicle Location.** The drone moves back to the GPS coordinate where the vehicle was at the moment it last had a solid lock on the target.

The Mathematics

Regardless of the strategy chosen, the retry logic adds a hardcoded vertical climb offset (H_retry) to ensure the sensor has a wider field of view for re-acquisition.

$$Z_target = Z_last_valid - 1.5\text{m}$$

(Note: In NED coordinates, negative Z is up).

The horizontal (XY) coordinates are determined by the `PLND_RET_BEHAVE` selection:

- **Mode 0:** $XY_retry = XY_target_last_seen$
- **Mode 1:** $XY_retry = XY_vehicle_at_last_seen$

The Engineer's View

In `AC_PrecLand_StateMachine::retry_landing()` (`libraries/AC_PrecLand/AC_PrecLand_StateMachine.cpp`):

1. The code reads the behavior preference.
2. If set to `GO_TO_TARGET_LOC` (0), it calls `_precland->get_last_detected_landing_pos()`.
3. If set to `GO_TO_LAST_LOC` (1), it calls `_precland->get_last_vehicle_pos_when_target_detected()`.
4. The state machine then enters the `RETRYING` status, commanding the position controller to converge on this 3D point. Once the drone is within 0.75m of the goal, it attempts to descend again.

Tuning & Behavior

- **0 (Default):** Usually best for stationary targets. If the drone was slightly off-center, it will move directly over the last known pad location.
- **1:** Better for moving targets (like a boat) or if the landing sensor has a narrow field of view. Moving the drone back to where it *knows* it could see the target is often more reliable than moving to where it *thinks* the target was.

PLND_RET_MAX (Precision Land Maximum Retries)

DEFAULT: 4	RANGE: 0 10

Precision Land Maximum Retries (PLND_RET_MAX)

Description

`PLND_RET_MAX` limits how many "Bail-Outs" the drone is allowed to perform.

If the drone loses the target while landing (due to glare, blockage, etc.), and `PLND_STRICT` is set to **1 (Retry)**, the drone will climb back up to its staging altitude and try again. This parameter ensures the drone doesn't do this forever and waste its battery.

- **Default Value:** 4 retries.
- **0:** Disable retries. The drone will follow the `PLND_STRICT` fallback on the first loss.

Tuning & Behavior

- **Default Value:** 4.
- **Recommendation:** Leave at **4**. If the drone hasn't found the target after 4 tries, it's statistically unlikely to find it on the 5th, and it's better to land or RTL.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





PLND_STRICT (Precision Landing Strictness)

DEFAULT: 1	RANGE: 0 2

Precision Landing Strictness (PLND_STRICT)

Description

`PLND_STRICT` decides the "Plan B" if the drone loses sight of its landing target.

Sometimes the target is obscured by grass, glare, or a temporary sensor glitch.

- **0: Land Vertically (Not strict).** If the target is lost, the drone just lands wherever it is.
- **1: Retry Landing (Normal).** If the target is lost, the drone will stop descending, potentially climb back up, and try to re-acquire the target.
- **2: Just Hover (Very Strict).** The drone will never land unless it can see the target. If the target is lost, it will hover indefinitely (or until failsafe triggers).

Tuning & Behavior

- **Default Value:** 1.
- **Recommendation:** Use **1** for most autonomous missions. Use **2** for high-value payloads where landing on bare earth (instead of the pad) is unacceptable.

PLND_TIMEOUT (PrecLand retry timeout)

DEFAULT: 4	RANGE: 0 20	UNITS: s

PrecLand retry timeout (PLND_TIMEOUT)

Description

This parameter sets the "Grace Period" for target loss during a precision landing. It defines how many seconds the vehicle is allowed to continue its descent after it stops seeing the landing target.

If the target reappears within this window, the landing continues normally. If the timeout expires and the target is still missing, the vehicle will take action based on `PLND_STRICT` (usually climbing back up for a retry or switching to a standard land).

The Mathematics

The system maintains a timer (t_lost) that resets whenever a valid target measurement is received.

$$\text{IF } (t_now - t_last_seen > PLND_TIMEOUT) \rightarrow \text{Trigger Retry Action}$$

This duration allows the vehicle to "fly through" brief sensor outages caused by sun glare, dust, or momentary signal blockage.

The Engineer's View

In `AC_PrecLand.cpp`, this maps to `_retry_timeout_sec`.
The `TargetState` transitions from `TARGET_FOUND` to `TARGET_RECENTLY_LOST` as soon as the sensor data stops. The landing state machine (often implemented in the vehicle's `mode_land.cpp`) checks this timeout.

- Note that `AC_PrecLand` also has a hardcoded `LANDING_TARGET_TIMEOUT_MS` (2000ms) which controls the "Healthy" status of the sensor itself, but `PLND_TIMEOUT` is the user-tunable delay for the physical landing behavior.

Tuning & Behavior

- **Default Value:** 4 seconds.
- **Agile Aircraft:** Can use a shorter timeout (e.g., 2s) to initiate retries faster.
- **High Dust/Shadow Environments:** Increase to **5-8 seconds** to allow the vehicle to drop through the obscuration and hope the target reappears at a lower altitude.
- **0:** Disables the grace period; a retry is triggered as soon as the sensor misses a frame.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





PLND_TYPE (Precision Landing Type)

DEFAULT: 0	RANGE: 0 4

Precision Landing Type (PLND_TYPE)

Description

`PLND_TYPE` tells the autopilot what kind of eye it's using to find the landing target.

- **0: None.**
- **1: MAVLink.** The drone receives target coordinates from an external computer or camera (e.g. OpenMV) via a telemetry port.
- **2: IRLock.** Specifically for the MarkOne IRLock sensor (IR Beacon).
- **4: SITL.** Simulation mode for testing.

Tuning & Behavior

- **Reboot Required:** Yes.

PLND_XY_DIST_MAX (Precision Land Maximum Start Distance)

DEFAULT: 2.5	RANGE: 0 10.0	UNITS: m

Precision Land Maximum Start Distance (PLND_XY_DIST_MAX)

Description

`PLND_XY_DIST_MAX` is a "Descent Lock."

When the drone arrives at the landing site and sees the target, it must first align itself horizontally. If the drone is too far to the side (e.g. 5 meters away), starting the descent would cause it to land at an angle or miss the pad.

This parameter tells the drone: "Do not start moving down until you are within **X meters** of the center of the target."

- **Default Value:** 2.5 meters.
- **Safety:** Setting this to **0** disables the check, and the drone will descend immediately while still trying to fix its horizontal position.

The Engineer's View

Used in the Precision Landing state machine. While the horizontal error is greater than this value, the vertical velocity demand is held at 0 (or a very low "Wait" value). Once the error drops below the threshold, the `Land` stage is permitted to proceed with its standard vertical descent profile.

Tuning & Behavior

- **Default Value:** 2.5m.
- **Precision:** If you have a very small landing pad (e.g. 1 meter wide), decrease this to **1.0m** to ensure a perfectly centered approach.
- **Windy Sites:** If the wind is strong and the drone is struggling to get perfectly centered, increase this to **4.0m** to prevent the drone from "hanging" indefinitely at height.

PLND_YAW_ALIGN (Precision Landing Yaw Alignment)

DEFAULT: 0	RANGE: 0 36000	UNITS: cdeg

Precision Landing Yaw Alignment (PLND_YAW_ALIGN)

Description

`PLND_YAW_ALIGN` is used if your landing camera isn't pointing perfectly straight forward.

If you mounted your sensor rotated (e.g., 90 degrees sideways to fit the frame), the "Forward" direction in the camera view would actually be the drone's "Left" direction. This parameter allows you to correct that rotation in software.

- **Units:** Centi-degrees (100 = 1 degree).

Tuning & Behavior

- **Default Value:** 0.
- **Tuning:** If you command the drone to move "Forward" toward the target, but it moves "Sideways" instead, your alignment is wrong.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **POS** parameter group (often appearing as `PSC_POSXY_P` or `PSC_POSZ_P`) configures the **Position Loop Gains**.

Key Concepts

1. Position Error to Velocity

The position controller is the outer-most loop in the navigation hierarchy. It compares the vehicle's actual position (GPS/EKF) against the target position (Waypoint).

- `POS_P`: The Proportional gain. It defines the "Stiffness" of the position hold.
  - **Higher values:** The drone will react more aggressively to stay exactly on the point.
  - **Lower values:** The drone will drift more but movements will be smoother.

Developer Notes

- **Library:** `libraries/AC_PosControl`.
- **Hierarchy:** `Pos_Error * POS_P` → `Target_Velocity`.

Winch control position error P gain ()

POS_P: Winch control position error P gain

Description

Winch control position error P gain.

Values

- **Range:** 0.01 to 10.0
- **Default:** 1.0

Description

This parameter controls the responsiveness of the winch when moving to a specific cable length.

- **Function:** It converts the error between the desired cable length and the actual cable length into a desired deployment or retraction speed.
- **Formula:** $TargetRate = Error \times P_gain$
- **Tuning:**
  - **Higher Value:** Faster correction of length errors, but may cause oscillation or jerking if too high.
  - **Lower Value:** Slower, smoother approach to the target length.

Source Code

ardupilot/libraries/AP_Winch/AP_Winch.cpp

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **PROBE** parameter group (specifically `PROBE_EXT` or `BARO_PROBE_EXT`) configures the automatic hardware discovery for **External Sensors**.

Key Concepts

1. I2C Probing

ArduPilot supports many different barometer and compass chips. Rather than forcing the user to select the exact chip model, the firmware can "probe" the I2C bus to see which chips respond.

- `PROBE_EXT`: Defines which external I2C buses or specific addresses to scan during the boot sequence.

Developer Notes

- **Library:** `libraries/AP_Baro`, `libraries/AP_Compass`.
- **Usage:** Usually left at default (Probe all) unless you have a specific hardware conflict.

External barometers to probe ()

PROBE_EXT: External barometers to probe

Description

This sets which types of external i2c barometer to look for. It is a bitmask of barometer types.

Values

- **Bitmask:**
  - **Bit 0:** BMP085
  - **Bit 1:** BMP280
  - **Bit 2:** MS5611
  - **Bit 3:** MS5607
  - **Bit 4:** MS5637
  - **Bit 5:** FBM320
  - **Bit 6:** DPS280
  - **Bit 7:** LPS25H
  - **Bit 8:** KELLER
  - **Bit 9:** MS5837
  - **Bit 10:** BMP388
  - **Bit 11:** ICP10100
  - **Bit 12:** ICP20100
  - **Bit 13:** SPL06
- **Default:** Varies by board (HAL_BARO_PROBE_EXT_DEFAULT)

Description

This parameter controls which barometer drivers are loaded when searching the external I2C bus.

- **Usage:** If you connect an external I2C barometer (e.g., on a GPS module), this parameter determines which chips the autopilot will try to find.
- **Optimization:** You can disable unused bits to speed up the boot process and prevent false detection of other I2C devices.

Source Code

ardupilot/libraries/AP_Baro/AP_Baro.cpp

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **PRX** parameter group configures the **Proximity Sensors** used for obstacle detection and avoidance. This library supports 360-degree scanners (Lidar), simple rangefinders used as proximity sensors, and virtual proximity data from companion computers.

The data from these sensors is fed into the **AVOID** and **OA** (Object Avoidance) libraries to prevent collisions.

Key Concepts

1. Sensor Types (`PRXn_TYPE`)

ArduPilot supports up to 5 proximity sensor instances.

- **1 (LightWareSF40):** 360-degree scanning Lidar.
- **2 (RPLidarA2/A3):** Popular scanning Lidars.
- **4 (DroneCAN):** CAN-based proximity sensors.
- **15 (RangeFinder):** Converts a standard downward or forward-facing Lidar (`RNGFND_`) into a proximity source.

2. Ignore Zones (`PRXn_IGN_...`)

Often, the drone's own frame, landing gear, or antennas might block the sensor's field of view, causing false obstacle detections.

- `PRXn_IGN_ANG`: The center angle of the zone to ignore.
- `PRXn_IGN_WID`: The width (degrees) of the zone to ignore.

3. Vertical Limits (`PRX_ALT_MIN`)

Defines the minimum altitude required for the sensors to be active, preventing the ground from being seen as an obstacle during takeoff.

Parameter Breakdown

- `PRX1_TYPE`: Driver selection for first sensor.
- `PRX1_MAX`: Maximum effective range (meters).
- `PRX_FILT`: Low-pass filter for proximity data (Hz).

Integration Guide

1. **Hardware:** Mount the Lidar on top or bottom with a clear 360-degree view.
2. **Enable:** Set `PRX1_TYPE`.
3. **Clean View:** Look at the "Proximity" window in Mission Planner. If you see "phantom" obstacles that move with the drone, use `IGN_ANG` to mask out the propeller arms or frame.

Developer Notes

- **Library:** `libraries/AP_Proximity`.
- **Output:** Generates the "Proximity" MAVLink message and logs to the `PRX` dataflash message.

PRX1_ADDR (Proximity Sensor Bus Address (Sensor 1))

DEFAULT: 0	RANGE: 0 127

Proximity Sensor Bus Address (PRX1_ADDR)

Description

`PRX1_ADDR` identifies the unique "Hardware Address" of your proximity sensor on the shared I2C or DroneCAN bus.

This allows you to connect multiple sensors of the same type to the same bus without conflict.

- **0:** Auto-detection.
- **1-127:** Specific hardware address.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





PRX1_IGN_ANG1 (Proximity Ignore Angle (Sector 1))

DEFAULT: 0	RANGE: 0 360	UNITS: deg

Proximity Ignore Angle (PRX1_IGN_ANG1)

Description

`PRX1_IGN_ANG1` (and ANG2-ANG4) defines the center of a "Blind Zone" for your proximity sensor.

This is used to prevent the drone from reacting to its own structure (like landing gear, arms, or GPS masts) that might be in the sensor's line of sight. By telling the autopilot to "ignore" a specific angle, you avoid false obstacle detections.

- **Units:** Degrees (0 to 360).
- **Reference:** 0 is forward, 90 is right, 180 is back, 270 is left.

Tuning & Behavior

- **Default:** 0.
- **Integration:** Use this in combination with PRX1_IGN_WID1 to define the width of the excluded sector.

PRX1_IGN_ANG2 (Proximity sensor ignore angle 2)

DEFAULT: 0	RANGE: 0 360	UNITS: deg

Proximity sensor ignore angle 2

Note: This parameter functions identically to PRX1_IGN_ANG1.

PRX1_IGN_ANG3 (Proximity sensor ignore angle 3)

DEFAULT: 0	RANGE: 0 360	UNITS: deg

Proximity sensor ignore angle 3

Note: This parameter functions identically to PRX1_IGN_ANG1.

PRX1_IGN_ANG4 (Proximity sensor ignore angle 4)

DEFAULT: 0	RANGE: 0 360	UNITS: deg

Proximity sensor ignore angle 4

Note: This parameter functions identically to PRX1_IGN_ANG1.

PRX1_IGN_WID1 (Proximity Ignore Width (Sector 1))

DEFAULT: 0	RANGE: 0 127	UNITS: deg

Proximity Ignore Width (PRX1_IGN_WID1)

Description

`PRX1_IGN_WID1` defines the size of the "Blind Zone" sector centered on PRX1_IGN_ANG1.

If you have a 10-degree wide arm in the sensor's path, you would set the center angle to the arm's position and this width to 10. The autopilot will then ignore any data reported within 5 degrees on either side of the center.

Tuning & Behavior

- **Default:** 0.
- **Safety:** Do not set the width unnecessarily large, as you might hide real obstacles from the avoidance system.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





PRX1_IGN_WID2 (Proximity sensor ignore width 2)

DEFAULT: 0	RANGE: 0 127	UNITS: deg

Proximity sensor ignore width 2

Note: This parameter functions identically to PRX1_IGN_WID1.

PRX1_IGN_WID3 (Proximity sensor ignore width 3)

DEFAULT: 0	RANGE: 0 127	UNITS: deg

Proximity sensor ignore width 3

Note: This parameter functions identically to PRX1_IGN_WID1.

PRX1_IGN_WID4 (Proximity sensor ignore width 4)

DEFAULT: 0	RANGE: 0 127	UNITS: deg

Proximity sensor ignore width 4

Note: This parameter functions identically to PRX1_IGN_WID1.

PRX1_MAX (Proximity Maximum Range (Sensor 1))

DEFAULT: 0	RANGE: 0 500	UNITS: m

Proximity Maximum Range (PRX1_MAX)

Description

`PRX1_MAX` defines the "Horizon" for your drone's obstacle avoidance system.

If the sensor reports an object farther away than this value, the autopilot ignores it. This is important to filter out noise at the edge of the sensor's capability or to ignore distant objects that aren't a threat.

- **0: Use sensor default.** (Recommended for most digital sensors).
- **1-500: Range in meters.**

Tuning & Behavior

- **Default:** 0.
- **Recommendation:** If you notice your drone "jerking" or reacting to "ghost" obstacles in the distance, set this to **80%** of the sensor's physical max range (e.g., set to 16 for a 20m Lidar).

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





PRX1_MIN (Proximity Minimum Range (Sensor 1))

DEFAULT: 0	RANGE: 0 500	UNITS: m

Proximity Minimum Range (PRX1_MIN)

Description

`PRX1_MIN` defines the "Blind Spot" around your drone.

If the sensor reports an object closer than this value, the autopilot ignores it. This is critical to prevent the drone from being "scared" of its own frame, propellers, or mounting brackets if they are within the sensor's field of view.

- **0: Use sensor default.** (Recommended).
- **Value (m):** Set this to slightly more than the distance from the sensor to the edge of your drone's props.

Tuning & Behavior

- **Default:** 0.
- **Safety:** Setting this too low may cause the drone to detect its own frame as an obstacle, preventing it from moving.

PRX1_ORIENT (Proximity Orientation (Sensor 1))

DEFAULT: 0	RANGE: 0 1

Proximity Orientation (PRX1_ORIENT)

Description

`PRX1_ORIENT` tells ArduPilot if your 360-degree Lidar is mounted normally or upside down.

- **0: Default (Right-side up).**
- **1: Upside Down.**

Tuning & Behavior

- **Default:** 0.
- **Action:** If your obstacle avoidance seems to react to objects on the wrong side (left instead of right), your sensor might be mounted upside down without this parameter being set correctly.

PRX1_RECV_ID (Proximity MAVLink System ID)

DEFAULT: 0	RANGE: 0 255

Proximity MAVLink System ID (PRX1_RECV_ID)

Description

`PRX1_RECV_ID` specifies the source of the proximity data when using a MAVLink-based sensor (like an OpenMV or a specialized companion computer script).

It tells the autopilot: "Only listen to `DISTANCE_SENSOR` messages from this specific MAVLink system ID."

- **0 (Default):** Listen to all MAVLink proximity sources.
- **1-255:** Listen only to this specific System ID.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





PRX1_TYPE (Proximity type (Sensor 1))

DEFAULT: 0	RANGE: 0 18

Proximity type (PRX1_TYPE)

Description

`PRX1_TYPE` enables a 360-degree Lidar or Radar for Object Avoidance.

Unlike a Rangefinder (which is usually a 1D single-point distance sensor), a Proximity sensor scans a sector or a full circle to build a map of obstacles around the vehicle. This data is used by the `AVOID` (Object Avoidance) library to stop before hitting objects or path-plan around them.

- **0: None.**
- **2: MAVLink.** Data sent from a Companion Computer.
- **4: RangeFinder.** Uses an existing `RNGFND` instance (turned into a proximity sensor).
- **5: RPLidarA2.** Common 360 Lidar.
- **6: TeraRangerTowerEvo.** 8-zone ToF array.
- **14: DroneCAN.**
- **16: LD06.** Inexpensive 360 Lidar.

The Engineer's View

Defined in `AP_Proximity`.
Enabling this allocates the driver. The driver then populates the `AP_Proximity_Boundary_3D` structure, which the Avoidance library (`AC_Avoid`) queries during flight.

Tuning & Behavior

- **Default Value:** 0 (Disabled).
- **Reboot Required:** Yes.
- **Orientation:** 360-degree Lidars must be mounted flat. Check `PRX1_ORIENT` if mounted upside down.

PRX1_YAW_CORR (Proximity Sensor Yaw Correction)

DEFAULT: 0	RANGE: -180 180	UNITS: deg

Proximity Sensor Yaw Correction (PRX1_YAW_CORR)

Description

`PRX1_YAW_CORR` allows you to digitally "Rotate" your proximity sensor if it wasn't mounted perfectly straight.

For obstacle avoidance to work, the autopilot must know exactly which direction each sensor reading corresponds to. If your Lidar is mounted 90 degrees to the right, you set this to 90 so the autopilot knows to shift all incoming data accordingly.

Tuning & Behavior

- **Default:** 0.
- **Calibration:** Point your drone at a known obstacle (like a wall) and check the proximity radar in your GCS. If the wall appears to the side instead of directly in front, adjust this parameter until they match.
- **Significance:** Critical for accurate avoidance maneuvers; if this is off, the drone might steer *into* an obstacle while trying to avoid it.

PRX2_ADDR (Bus address of sensor)

DEFAULT: 0	RANGE: 0 127

Bus address of sensor

Note: This parameter functions identically to PRX1_ADDR.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





PRX2_IGN_ANG1 (Proximity sensor ignore angle 1)

DEFAULT: 0	RANGE: 0 360	UNITS: deg

Proximity sensor ignore angle 1

Note: This parameter functions identically to PRX1_IGN_ANG1.

PRX2_IGN_ANG2 (Proximity sensor ignore angle 2)

DEFAULT: 0	RANGE: 0 360	UNITS: deg

Proximity sensor ignore angle 2

Note: This parameter functions identically to PRX1_IGN_ANG1.

PRX2_IGN_ANG3 (Proximity sensor ignore angle 3)

DEFAULT: 0	RANGE: 0 360	UNITS: deg

Proximity sensor ignore angle 3

Note: This parameter functions identically to PRX1_IGN_ANG1.

PRX2_IGN_ANG4 (Proximity sensor ignore angle 4)

DEFAULT: 0	RANGE: 0 360	UNITS: deg

Proximity sensor ignore angle 4

Note: This parameter functions identically to PRX1_IGN_ANG1.

PRX2_IGN_WID1 (Proximity sensor ignore width 1)

DEFAULT: 0	RANGE: 0 127	UNITS: deg

Proximity sensor ignore width 1

Note: This parameter functions identically to PRX1_IGN_WID1.

PRX2_IGN_WID2 (Proximity sensor ignore width 2)

DEFAULT: 0	RANGE: 0 127	UNITS: deg

Proximity sensor ignore width 2

Note: This parameter functions identically to PRX1_IGN_WID1.

PRX2_IGN_WID3 (Proximity sensor ignore width 3)

DEFAULT: 0	RANGE: 0 127	UNITS: deg

Proximity sensor ignore width 3

Note: This parameter functions identically to PRX1_IGN_WID1.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





PRX2_IGN_WID4 (Proximity sensor ignore width 4)

DEFAULT: 0	RANGE: 0 127	UNITS: deg

Proximity sensor ignore width 4

Note: This parameter functions identically to PRX1_IGN_WID1.

PRX2_MAX (Proximity maximum range)

DEFAULT: 0.0f	RANGE: 0 500	UNITS: m

Proximity maximum range

Note: This parameter functions identically to PRX1_MAX.

PRX2_MIN (Proximity minimum range)

DEFAULT: 0.0f	RANGE: 0 500	UNITS: m

Proximity minimum range

Note: This parameter functions identically to PRX1_MIN.

PRX2_ORIENT (Proximity sensor orientation)

DEFAULT: 0	RANGE: null

Proximity sensor orientation

Note: This parameter functions identically to PRX1_ORIENT.

PRX2_TYPE (Proximity type)

DEFAULT: 0	RANGE: null

Proximity type

Note: This parameter functions identically to PRX1_TYPE.

PRX2_YAW_CORR (Proximity sensor yaw correction)

DEFAULT: 0	RANGE: -180 180	UNITS: deg

Proximity sensor yaw correction

Note: This parameter functions identically to PRX1_YAW_CORR.

PRX3_ADDR (Bus address of sensor)

DEFAULT: 0	RANGE: 0 127

Bus address of sensor

Note: This parameter functions identically to PRX1_ADDR.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





PRX3_IGN_ANG1 (Proximity sensor ignore angle 1)

DEFAULT: 0	RANGE: 0 360	UNITS: deg

Proximity sensor ignore angle 1

Note: This parameter functions identically to PRX1_IGN_ANG1.

PRX3_IGN_ANG2 (Proximity sensor ignore angle 2)

DEFAULT: 0	RANGE: 0 360	UNITS: deg

Proximity sensor ignore angle 2

Note: This parameter functions identically to PRX1_IGN_ANG1.

PRX3_IGN_ANG3 (Proximity sensor ignore angle 3)

DEFAULT: 0	RANGE: 0 360	UNITS: deg

Proximity sensor ignore angle 3

Note: This parameter functions identically to PRX1_IGN_ANG1.

PRX3_IGN_ANG4 (Proximity sensor ignore angle 4)

DEFAULT: 0	RANGE: 0 360	UNITS: deg

Proximity sensor ignore angle 4

Note: This parameter functions identically to PRX1_IGN_ANG1.

PRX3_IGN_WID1 (Proximity sensor ignore width 1)

DEFAULT: 0	RANGE: 0 127	UNITS: deg

Proximity sensor ignore width 1

Note: This parameter functions identically to PRX1_IGN_WID1.

PRX3_IGN_WID2 (Proximity sensor ignore width 2)

DEFAULT: 0	RANGE: 0 127	UNITS: deg

Proximity sensor ignore width 2

Note: This parameter functions identically to PRX1_IGN_WID1.

PRX3_IGN_WID3 (Proximity sensor ignore width 3)

DEFAULT: 0	RANGE: 0 127	UNITS: deg

Proximity sensor ignore width 3

Note: This parameter functions identically to PRX1_IGN_WID1.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





PRX3_IGN_WID4 (Proximity sensor ignore width 4)

DEFAULT: 0	RANGE: 0 127	UNITS: deg

Proximity sensor ignore width 4

Note: This parameter functions identically to PRX1_IGN_WID1.

PRX3_MAX (Proximity maximum range)

DEFAULT: 0.0f	RANGE: 0 500	UNITS: m

Proximity maximum range

Note: This parameter functions identically to PRX1_MAX.

PRX3_MIN (Proximity minimum range)

DEFAULT: 0.0f	RANGE: 0 500	UNITS: m

Proximity minimum range

Note: This parameter functions identically to PRX1_MIN.

PRX3_ORIENT (Proximity sensor orientation)

DEFAULT: 0	RANGE: null

Proximity sensor orientation

Note: This parameter functions identically to PRX1_ORIENT.

PRX3_TYPE (Proximity type)

DEFAULT: 0	RANGE: null

Proximity type

Note: This parameter functions identically to PRX1_TYPE.

PRX3_YAW_CORR (Proximity sensor yaw correction)

DEFAULT: 0	RANGE: -180 180	UNITS: deg

Proximity sensor yaw correction

Note: This parameter functions identically to PRX1_YAW_CORR.

PRX4_ADDR (Bus address of sensor)

DEFAULT: 0	RANGE: 0 127

Bus address of sensor

Note: This parameter functions identically to PRX1_ADDR.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





PRX4_IGN_ANG1 (Proximity sensor ignore angle 1)

DEFAULT: 0	RANGE: 0 360	UNITS: deg

Proximity sensor ignore angle 1

Note: This parameter functions identically to PRX1_IGN_ANG1.

PRX4_IGN_ANG2 (Proximity sensor ignore angle 2)

DEFAULT: 0	RANGE: 0 360	UNITS: deg

Proximity sensor ignore angle 2

Note: This parameter functions identically to PRX1_IGN_ANG1.

PRX4_IGN_ANG3 (Proximity sensor ignore angle 3)

DEFAULT: 0	RANGE: 0 360	UNITS: deg

Proximity sensor ignore angle 3

Note: This parameter functions identically to PRX1_IGN_ANG1.

PRX4_IGN_ANG4 (Proximity sensor ignore angle 4)

DEFAULT: 0	RANGE: 0 360	UNITS: deg

Proximity sensor ignore angle 4

Note: This parameter functions identically to PRX1_IGN_ANG1.

PRX4_IGN_WID1 (Proximity sensor ignore width 1)

DEFAULT: 0	RANGE: 0 127	UNITS: deg

Proximity sensor ignore width 1

Note: This parameter functions identically to PRX1_IGN_WID1.

PRX4_IGN_WID2 (Proximity sensor ignore width 2)

DEFAULT: 0	RANGE: 0 127	UNITS: deg

Proximity sensor ignore width 2

Note: This parameter functions identically to PRX1_IGN_WID1.

PRX4_IGN_WID3 (Proximity sensor ignore width 3)

DEFAULT: 0	RANGE: 0 127	UNITS: deg

Proximity sensor ignore width 3

Note: This parameter functions identically to PRX1_IGN_WID1.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





PRX4_IGN_WID4 (Proximity sensor ignore width 4)

DEFAULT: 0	RANGE: 0 127	UNITS: deg

Proximity sensor ignore width 4

Note: This parameter functions identically to PRX1_IGN_WID1.

PRX4_MAX (Proximity maximum range)

DEFAULT: 0.0f	RANGE: 0 500	UNITS: m

Proximity maximum range

Note: This parameter functions identically to PRX1_MAX.

PRX4_MIN (Proximity minimum range)

DEFAULT: 0.0f	RANGE: 0 500	UNITS: m

Proximity minimum range

Note: This parameter functions identically to PRX1_MIN.

PRX4_ORIENT (Proximity sensor orientation)

DEFAULT: 0	RANGE: null

Proximity sensor orientation

Note: This parameter functions identically to PRX1_ORIENT.

PRX4_TYPE (Proximity type)

DEFAULT: 0	RANGE: null

Proximity type

Note: This parameter functions identically to PRX1_TYPE.

PRX4_YAW_CORR (Proximity sensor yaw correction)

DEFAULT: 0	RANGE: -180 180	UNITS: deg

Proximity sensor yaw correction

Note: This parameter functions identically to PRX1_YAW_CORR.

PRX5_ADDR (Bus address of sensor)

DEFAULT: 0	RANGE: 0 127

Bus address of sensor

Note: This parameter functions identically to PRX1_ADDR.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





PRX5_IGN_ANG1 (Proximity sensor ignore angle 1)

DEFAULT: 0	RANGE: 0 360	UNITS: deg

Proximity sensor ignore angle 1

Note: This parameter functions identically to PRX1_IGN_ANG1.

PRX5_IGN_ANG2 (Proximity sensor ignore angle 2)

DEFAULT: 0	RANGE: 0 360	UNITS: deg

Proximity sensor ignore angle 2

Note: This parameter functions identically to PRX1_IGN_ANG1.

PRX5_IGN_ANG3 (Proximity sensor ignore angle 3)

DEFAULT: 0	RANGE: 0 360	UNITS: deg

Proximity sensor ignore angle 3

Note: This parameter functions identically to PRX1_IGN_ANG1.

PRX5_IGN_ANG4 (Proximity sensor ignore angle 4)

DEFAULT: 0	RANGE: 0 360	UNITS: deg

Proximity sensor ignore angle 4

Note: This parameter functions identically to PRX1_IGN_ANG1.

PRX5_IGN_WID1 (Proximity sensor ignore width 1)

DEFAULT: 0	RANGE: 0 127	UNITS: deg

Proximity sensor ignore width 1

Note: This parameter functions identically to PRX1_IGN_WID1.

PRX5_IGN_WID2 (Proximity sensor ignore width 2)

DEFAULT: 0	RANGE: 0 127	UNITS: deg

Proximity sensor ignore width 2

Note: This parameter functions identically to PRX1_IGN_WID1.

PRX5_IGN_WID3 (Proximity sensor ignore width 3)

DEFAULT: 0	RANGE: 0 127	UNITS: deg

Proximity sensor ignore width 3

Note: This parameter functions identically to PRX1_IGN_WID1.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





PRX5_IGN_WID4 (Proximity sensor ignore width 4)

DEFAULT: 0	RANGE: 0 127	UNITS: deg

Proximity sensor ignore width 4

Note: This parameter functions identically to PRX1_IGN_WID1.

PRX5_MAX (Proximity maximum range)

DEFAULT: 0.0f	RANGE: 0 500	UNITS: m

Proximity maximum range

Note: This parameter functions identically to PRX1_MAX.

PRX5_MIN (Proximity minimum range)

DEFAULT: 0.0f	RANGE: 0 500	UNITS: m

Proximity minimum range

Note: This parameter functions identically to PRX1_MIN.

PRX5_ORIENT (Proximity sensor orientation)

DEFAULT: 0	RANGE: null

Proximity sensor orientation

Note: This parameter functions identically to PRX1_ORIENT.

PRX5_TYPE (Proximity type)

DEFAULT: 0	RANGE: null

Proximity type

Note: This parameter functions identically to PRX1_TYPE.

PRX5_YAW_CORR (Proximity sensor yaw correction)

DEFAULT: 0	RANGE: -180 180	UNITS: deg

Proximity sensor yaw correction

Note: This parameter functions identically to PRX1_YAW_CORR.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





PRX_ALT_MIN (Proximity Minimum Altitude)

DEFAULT: 1.0	RANGE: 0 10	UNITS: m

Proximity Minimum Altitude (PRX_ALT_MIN)

Description

`PRX_ALT_MIN` defines the "Activation Floor" for object avoidance.

Proximity sensors (like 360-degree Lidars) can sometimes "see" the ground when the drone is very low or during takeoff/landing, causing false obstacle detections. This parameter tells the autopilot to ignore all proximity data until the drone has climbed above this altitude.

- **Default Value:** 1.0 meter.
- **Safety:** This ensures the drone doesn't try to "dodge" the grass or its own landing pad during the critical phases of flight.

Tuning & Behavior

- **Default:** 1.0.
- **Note:** This requires a valid downward-facing Rangefinder (Lidar/Sonar) to know the altitude accurately. If no rangefinder is present, this parameter may not function as expected.

PRX_BAUDRATE (Proximity Sensor Baud Rate)

DEFAULT: 0	RANGE: 1 1000000

Proximity Sensor Baud Rate (PRX_BAUDRATE)

Description

`PRX_BAUDRATE` sets the communication speed for proximity sensors (like Lidars) that connect via a Serial (UART) port.

This parameter is primarily found on **AP_Periph** nodes that are being used as CAN bridges for serial sensors.

Tuning & Behavior

- **Default:** 0 (Use driver default).
- **Common Values:** 115200 (Most common), 921600 (High speed).
- **Reboot Required:** Yes.

PRX_FILT (Proximity Data Low-Pass Filter)

DEFAULT: 0.25	RANGE: 0 20	UNITS: Hz

Proximity Data Low-Pass Filter (PRX_FILT)

Description

`PRX_FILT` smooths out the "noisy" data from proximity sensors (like Lidars).

Lidars often report small, random fluctuations in distance. Without filtering, these fluctuations can cause the drone to twitch or make small steering corrections even when stationary. This filter removes that high-frequency noise, providing a stable "Boundary" for the obstacle avoidance algorithm to work with.

- **Higher Frequency:** Faster reaction to obstacles, but more twitchy.
- **Lower Frequency (Default):** Very smooth movement, but may react slightly slower to a fast-moving object.

Tuning & Behavior

- **Default:** 0.25 Hz.
- **Recommendation:** Leave at **0.25 Hz** for most setups. If you find your drone is too slow to stop for obstacles, try increasing to **1.0 Hz**.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





PRX_IGN_GND (Proximity Ignore Ground)

DEFAULT: 0	RANGE: 0 1

Proximity Ignore Ground (PRX_IGN_GND)

Description

`PRX_IGN_GND` prevents the drone from being "Scared" of the floor.

When a drone is close to the ground, its Lidar often sees reflections from the dirt, grass, or landing pad as "Obstacles." This can prevent the drone from landing or cause it to jump away during takeoff. When enabled, this feature uses a downward rangefinder to determine exactly where the ground is and ignores any proximity data that is too close to that surface.

PRX_LOG_RAW (Proximity Raw Data Logging)

DEFAULT: 0	RANGE: 0 1

Proximity Raw Data Logging (PRX_LOG_RAW)

Description

`PRX_LOG_RAW` is a developer and troubleshooting tool.

When enabled, the autopilot records every individual measurement from the proximity sensor (e.g. all 360 degrees of data from a Lidar) into the internal DataFlash log.

- **0: Disabled (Default).** Only the processed "Boundary" data is logged.
- **1: Enabled.** Logs the full raw data stream.

Tuning & Behavior

- **Warning:** This creates extremely large log files.
- **Usage:** Only enable this if you are debugging a sensor issue or verifying that your ignore sectors (PRX1_IGN_ANG1) are correctly configured.

PRX_PORT (Proximity Sensor Port (AP_Periph))

DEFAULT: 0	RANGE: 0 10

Proximity Sensor Port (PRX_PORT)

Description

`PRX_PORT` is used on **AP_Periph** nodes to map a physical UART port to the proximity sensor driver.

If you are using a CAN-based peripheral node to bridge a serial Lidar to the main flight controller, this parameter tells the peripheral node which of its own serial ports is connected to the Lidar.

Tuning & Behavior

- **Default:** 0.
- **Setup:** Refer to the wiring diagram for your specific AP_Periph hardware (e.g., Matek or CUAV CAN nodes) to find the correct port number.
- **Reboot Required:** Yes.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





PRX_MAX_RATE: Proximity Sensor max rate

Description

This is the maximum rate we send Proximity Sensor data in Hz. Zero means no limit.

Values

- **Range:** 0 to 200
- **Units:** Hz
- **Default:** 50

Description

This parameter limits the frequency of proximity data messages sent by the AP_Periph device.

- **Purpose:** Prevents high-speed Lidar sensors from flooding the CAN bus or telemetry link with too much data.
- **50Hz (Default):** A good balance for obstacle avoidance.
- **0:** Sends data as fast as the sensor can read (can be >1000Hz for some Lidars, which may choke the bus).

Source Code

ardupilot/Tools/AP_Periph/Parameters.cpp

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





POSITION CONTROLLER CONFIGURATION (PSC)

Overview

The **PSC** parameter group contains the **PID Tuning** parameters for the **Position Controller**. This library is the "brains" of the navigation system for multicopters and blimps.

While the `ATC` group handles how the drone rotates (Attitude), the `PSC` group handles how it moves through 3D space (Position and Velocity). It translates a waypoint target into the specific tilt angles and throttle levels needed to get there.

Key Concepts: The Cascaded Loop

Like attitude control, position control uses a cascaded structure:

1. Position Loop (Outer)

- **Input:** Target Position (Lat/Lon/Alt) vs. Actual Position.
- **Output:** Target Velocity (m/s).
- **Parameters:** `PSC_POSXY_P`, `PSC_POSZ_P`.

2. Velocity Loop (Middle)

- **Input:** Target Velocity vs. Actual Velocity (from EKF).
- **Output:** Target Acceleration (m/s^2).
- **Parameters:** `PSC_VELXY_P/I/D`, `PSC_VELZ_P/I/D`.

3. Acceleration Loop (Inner)

- **Input:** Target Acceleration vs. Actual Acceleration (from IMU).
- **Output:** Desired Lean Angle (to `ATC`) and Throttle level.
- **Parameters:** `PSC_ACCZ_P/I/D` (specifically for vertical control).

Key Parameters

- `PSC_POSXY_P`: How aggressively the drone tries to close a horizontal position gap.
- `PSC_VELXY_P`: Horizontal velocity responsiveness.
- `PSC_ACCZ_P/I/D`: Vertical acceleration tuning. Critical for stable hover and smooth altitude changes.
- `PSC_JERK_XY`: Smoothness of horizontal transitions (m/s/s/s).

Integration Guide

Tuning Altitude Hold

1. If the drone "bounces" or "hunts" vertically while trying to hold altitude, your `PSC_ACCZ_` gains are likely too high.
2. If it feels "sloppy" and drops significantly when moving forward, increase `PSC_ACCZ_I` and `PSC_POSZ_P`.

Tuning Loiter

1. If the drone overshoots its target and circles back (toilet-bowling), first check the compass. If compass is perfect, then reduce `PSC_POSXY_P`.

Developer Notes

- **Library:** `libraries/AC_PosControl`.
- **Coordinates:** Uses the NED (North-East-Down) frame for all internal calculations.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





PSC_ACCZ_D (Position Control Acceleration (Vertical) D Gain)

DEFAULT: 0.0	RANGE: 0.0 0.4

Position Control Acceleration (Vertical) D Gain (PSC_ACCZ_D)

Description

`PSC_ACCZ_D` is the vertical "shock absorber." It provides damping to the vertical movement by reacting to how fast the vertical acceleration is changing. It is primarily used to stop the drone from "ringing" or bouncing vertically when it hits its target altitude.

In many standard builds, this parameter is set to **0.0** because the air itself provides enough natural damping for the vertical axis. However, for high-performance or very large vehicles, a small amount of D-gain can significantly clean up vertical tracking.

- **Low Value (0.0):** Standard vertical behavior. Relies on physics and P-gain for stabilization.
- **High Value:** Stiffens the vertical response. Makes the drone feel "locked" to its height even in turbulent air.
- **Too High:** Amplifies IMU noise and motor vibrations, causing the motors to "chirp" or heat up rapidly without any visible movement of the drone.

The Mathematics

This parameter scales the derivative of the acceleration error.

$$ \text{Throttle}_{out} += k_D \cdot \frac{d(\vec{A}_{z_error})}{dt} $$

Where:

- $\vec{A}_{z_error} = \vec{A}_{z_target} - \vec{A}_{z_measured}$.
- k_D is `PSC_ACCZ_D`.

Filter dependency: Just like the horizontal D-term, the vertical D-term is extremely sensitive to noise. It is heavily dependent on the `PSC_ACCZ_FLTD` filter to prevent motor damage.

The Engineer's View

This parameter is the `_kd` member of the `_pid_accel_z` object inside `AC_PosControl`.

It is executed in `AC_PID::update_all()`:

```
// AC_PID.cpp
if (is_positive(dt)) {
    float derivative = (error - _last_error) / dt;
    _derivative = _last_derivative + (dt / (_filt_D_hz + dt)) * (derivative - _last_error) / dt;
}
return ... + _derivative * _kd;
```

Note that the derivative is filtered using a low-pass filter defined by `PSC_ACCZ_FLTD`. Without this filter, the D-term would be unusable due to the high vibration levels found in multirotors.

Tuning & Behavior

- **Default Value:** 0.0
- **Range:** 0.0 - 0.4
- **Effect of Increasing:** Reduces vertical "bounce" at the end of a climb or descent. Better stability in ground effect.
- **Effect of Decreasing:** Smoother, quieter motor operation.

Use Case Recommendations

- **Small High-Performance Drones: Increase to 0.05 - 0.1.** Helps the drone stop exactly on its altitude target after aggressive maneuvers.
- **Heavy Lift / Cinematic: Keep at 0.0.** Large props are slow to react; adding D-gain here usually just creates heat without improving tracking.
- **High Turbulence / High Wind: Increase to 0.1.** Provides additional authority to fight vertical gusts.

Troubleshooting

- **Scenario:** Vertical height hold is perfect, but motors sound "scratchy" or "grind-y" in Loiter.
  - **Diagnosis:** `PSC_ACCZ_D` is too high, amplifying IMU noise into the ESCs.
  - **Fix:** Set `PSC_ACCZ_D` to 0.0.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





PSC_ACCZ_D_FF (Position Control Acceleration (Vertical) Derivative Feed-Forward)

DEFAULT: 0.0	RANGE: 0 0.02

Position Control Acceleration (Vertical) Derivative Feed-Forward (PSC_ACCZ_D_FF)

Description

`PSC_ACCZ_D_FF` is an advanced "predictive" term for vertical flight. While standard Feed-Forward (`PSC_ACCZ_FF`) looks at the target acceleration itself, the **Derivative Feed-Forward (D_FF)** looks at how fast the *target* is changing.

Think of it as the "lookahead" for a vertical punch. If the software suddenly requests a massive increase in acceleration, the D_FF term instantly adds a burst of throttle to help the motors overcome the inertia of the propellers. It helps the drone "lead" into a maneuver before any error has even occurred.

- **Low Value (0.0):** Standard behavior. The drone accelerates as commanded by the shaper.
- **High Value:** Creates a very aggressive vertical response. The drone "snaps" into climbs and descents.
- **Too High:** Can cause "kickback" or jerky movements, as the drone over-predicts the needed force.

The Mathematics

The D_FF term is proportional to the derivative of the target signal ($\vec{A}_z_target$):

$$ \text{Throttle}_{D_FF} = \frac{d(\vec{A}_z_target)}{dt} \cdot k_D_FF $$

Where:

- k_D_FF is `PSC_ACCZ_D_FF`.
- $\frac{d(\vec{A}_z_target)}{dt}$ is the rate of change of the acceleration setpoint.

This is added to the final throttle sum alongside the P, I, D, and base FF terms.

The Engineer's View

This parameter maps to `_kdff` in the `_pid_accel_z` object.

It is executed in `AC_PID::update_all()`:

```
// AC_PID.cpp
if (is_positive(dt)) {
    float target_derivative = (target - _last_target) / dt;
    _pid_info.D_FF = target_derivative * _kdff;
}
```

Because it looks at the *target derivative*, it is immune to IMU vibration noise (unlike the standard D-term). However, it is very sensitive to "steps" in the high-level path planner, which is why it's usually used with a well-shaped S-curve.

Tuning & Behavior

- **Default Value:** 0.0
- **Range:** 0.0 - 0.02
- **Effect of Increasing:** Snappier, more instantaneous vertical transitions. Reduces altitude sag at the start of a climb.
- **Effect of Decreasing:** Smoother, more organic vertical movement.

Use Case Recommendations

- **Acrobatic / FPV Autonomous Flight: Increase to 0.005.** Helps the drone track aggressive 3D flight paths where altitude changes are frequent and sharp.
- **Standard Multirotors: Keep at 0.0.** The standard shaper and P-gain are usually sufficient.
- **High-Speed Vertical Search: Increase to 0.01.** If the drone is rapidly oscillating its height to search for an object, D_FF improves the accuracy of those transitions.

Troubleshooting

- **Scenario:** Drone makes a distinct "pop" or "click" sound in the motors when starting a vertical movement.
  - **Diagnosis:** `PSC_ACCZ_D_FF` is likely too high, commanding an impossible step-change in motor torque.
  - **Fix:** Set back to 0.0.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





PSC_ACCZ_FF (Position Control Acceleration (Vertical) Feed-Forward)

DEFAULT: 0.0	RANGE: 0 0.5

Position Control Acceleration (Vertical) Feed-Forward (PSC_ACCZ_FF)

Description

`PSC_ACCZ_FF` is the "prediction" component of the altitude controller. While the P, I, and D terms react to **errors** (mistakes the drone has already made), the Feed-Forward (FF) term reacts to the **command** itself.

If you tell the drone to climb at 2 m/s, the FF term instantly calculates a baseline throttle increase *before* the drone even starts to sink or slow down. It provides "pre-emptive authority," making the vertical control feel significantly more responsive and connected to the pilot's sticks.

- **Low Value (0.0):** The drone relies entirely on error-correction. It will always lag slightly behind your climb/descent commands.
- **High Value:** The drone "jumps" instantly when you command a climb.
- **Too High:** The drone will "kick" violently and potentially overshoot its climb rate because it's adding too much power before the sensors can even measure the movement.

The Mathematics

The Feed-Forward term is multiplied directly by the target acceleration:

$$ \text{Throttle}_{FF} = |\vec{A}|z_target} \cdot k_{FF} $$

Where:

- $\vec{A}_z_target$ is the requested acceleration.
- k_FF is `PSC_ACCZ_FF`.

The total output becomes:
$$ \text{Throttle}_{total} = |\text{Throttle}_{FF} + \text{Throttle}_{P} + |\text{Throttle}_{I} + \text{Throttle}_{D} $$

The Engineer's View

This parameter maps to `_kff` in the `_pid_accel_z` object.

It is applied in `AC_PosControl::update_z_controller()`:

```
// AC_PosControl.cpp
thr_out = _pid_accel_z.update_all(_accel_target.z, z_accel_meas, _dt, ...) * 0.001f;
thr_out += _pid_accel_z.get_ff() * 0.001f;
```

In ArduPilot, Feed-Forward is often preferred over high P-gains because it does not create the same high-frequency oscillations. It is a "clean" way to get performance.

Tuning & Behavior

- **Default Value:** 0.0 (Standard)
- **Range:** 0.0 - 0.5
- **Effect of Increasing:** Faster response to altitude commands. Reduces the "initial drop" when switching into AltHold/Loiter while moving fast.
- **Effect of Decreasing:** Smoother, more damped vertical response.

Use Case Recommendations

- **Standard Build: Keep at 0.0.** The vertical loop is usually strong enough without it.
- **High-Dynamics / Racing: Increase to 0.1 - 0.2.** Gives the "vertical punch" needed to clear obstacles instantly.
- **Helicopters (Traditional): Keep at 0.0.** Collective pitch response is different from multirotor RPM response; FF can cause over-speeding of the head.

Troubleshooting

- **Scenario:** When you push the throttle stick up, the drone "twitches" or "kicks" up violently for a split second, then settles into a smooth climb.
  - **Diagnosis:** `PSC_ACCZ_FF` is too high. The initial "kick" from the command is overpowering the drone's inertia.
  - **Fix:** Reduce `PSC_ACCZ_FF` by 0.05.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





PSC_ACCZ_FLTD (Position Control Acceleration (Vertical) D-Term Filter)

DEFAULT: 0.0	RANGE: 0 100	UNITS: Hz

Position Control Acceleration (Vertical) D-Term Filter (PSC_ACCZ_FLTD)

Description

`PSC_ACCZ_FLTD` is the specific filter for the vertical **Derivative (D) gain**.

In the altitude controller, the D-gain (`PSC_ACCZ_D`) acts as a damper to stop the drone from bouncing. However, the derivative calculation (which looks at the rate of change) is extremely sensitive to motor vibration. Without a filter, even a tiny D-gain would amplify IMU noise so much that the motors would oscillate at hundreds of Hertz, potentially causing fire or mechanical failure.

- **Standard Build:** If `PSC_ACCZ_D` is 0, this parameter does nothing.
- **High Performance:** If you use D-gain to stabilize a heavy or high-power vehicle, you *must* set this filter to a sensible value (e.g. 10-20Hz).
- **Default (0):** The filter is effectively disabled or follows the main error filter.

The Mathematics

The D-term calculates the derivative of the error, then passes it through this low-pass filter:

$$ \vec{D}{*filtered*} = \vec{D}{prev} + \alpha \cdot \left(\frac{d(\vec{A}{*err*})}{dt} - \vec{D}{prev} \right) $$

Where the filter coefficient α is calculated from `PSC_ACCZ_FLTD` (f_c):

$$\alpha = \frac{dt}{dt + \frac{1}{2\pi f_c}}$$

The Engineer's View

This parameter maps to `_filt_D_hz` in the `_pid_accel_z` object.

It is applied in `AC_PID::update_all()`:

```
// AC_PID.cpp
if (is_positive(dt)) {
    float derivative = (error - _last_error) / dt;
    _derivative = _last_derivative + (dt / (_filt_D_hz + dt)) * (derivative - _last_derivative);
}
```

The frequency is typically set lower than the main error filter because the derivative calculation inherently "creates" noise by looking at infinitesimal changes between samples.

Tuning & Behavior

- **Default Value:** 0.0 Hz (Disabled/Auto)
- **Range:** 0.0 - 100.0 Hz
- **Effect of Increasing:** Allows the D-term to react faster to "thumps" or turbulence, but increases motor heating.
- **Effect of Decreasing:** Smoother vertical behavior, but the D-term becomes less effective at stopping vertical bounces.

Use Case Recommendations

- **Most Multirotors: Keep at 0.0.** Unless you are manually tuning `PSC_ACCZ_D`, this filter is not needed.
- **Heavy Industrial Drones: Set to 10.0 Hz (if using D-gain).** Provides a clean signal for the damper to work without excites frame resonances.
- **Autonomous Landing on Ships/Moving Targets: Set to 20.0 Hz.** Allows for the high-authority damping needed when the drone must "snap" to a height target as the platform moves beneath it.

Troubleshooting

- **Scenario:** You increased `PSC_ACCZ_D` and now the motors are making a high-pitched "singing" noise.
  - **Diagnosis:** High-frequency derivative noise.
  - **Fix:** Decrease `PSC_ACCZ_FLTD` to 5.0 or 7.0 Hz.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





PSC_ACCZ_FLTE (Position Control Acceleration (Vertical) Error Filter)

DEFAULT: 20.0	RANGE: 1 100	UNITS: Hz

Position Control Acceleration (Vertical) Error Filter (PSC_ACCZ_FLTE)

Description

`PSC_ACCZ_FLTE` is the frontline defense against vertical vibrations. It filters the difference between the drone's *target* vertical acceleration and the *actual* acceleration measured by the IMU.

On most multirotors, the "Z-axis" is the noisiest. Propeller wash and motor imbalance create constant vertical vibration. If the innermost throttle controller (Acceleration loop) tried to track these vibrations, the motors would work overtime, wasting battery and potentially burning out. This filter ensures that only the "real" movements of the drone are passed into the throttle logic.

- **Higher Frequency (e.g. 40Hz):** Faster throttle response to turbulence, but high risk of passing motor noise through.
- **Lower Frequency (e.g. 10Hz):** Very smooth vertical hold, but the drone may feel "mushy" or slow to respond to heavy gusts.
- **Default (20Hz):** A robust balance for 99% of multirotors.

The Mathematics

This is a single-pole low-pass filter:

$$ \vec{A}_{err_filtered} = \vec{A}_{err_prev} + \alpha \cdot (\vec{A}_{err_raw} - \vec{A}_{err_prev}) $$

Where α is derived from `PSC_ACCZ_FLTE`:

$$\alpha = \frac{dt}{dt + \frac{1}{2\pi f_c}}$$

Note: ArduPilot's innermost filters often run at the full loop rate (400Hz+), making them very effective at rejecting high-frequency noise.

The Engineer's View

This parameter maps to `_filt_E_hz` in the `_pid_accel_z` object.

It is applied in `AC_PID::update_all()`:

```
// AC_PID.cpp
if (_reset_filter) {
    _reset_filter = false;
    _error = target - measurement;
} else {
    _error += ((target - measurement) - _error) * get_filt_E_alpha(dt);
}
```

The filtered `_error` is then used for the P, I, and D calculations.

Tuning & Behavior

- **Default Value:** 20.0 Hz
- **Range:** 1.0 - 100.0 Hz
- **Effect of Increasing:** Faster vertical "clamping" to the target. Better for racing and aggressive maneuvering.
- **Effect of Decreasing:** Quieter motors, less vibration-induced heating, but altitude hold may "sag" slightly during fast turns.

Use Case Recommendations

- **Small, High-Vibration Racers: Decrease to 10.0 - 15.0 Hz.** Helps isolate the flight controller from the extreme noise generated by small, high-RPM motors.
- **Heavy Lift / Smooth Cinematic: Keep at 20.0 Hz.** This provides plenty of responsiveness for smooth flight.
- **Large-Prop High-Inertia Drones (>22 inch): Decrease to 10.0 Hz.** Slow props can't react to high frequencies anyway, so there is no benefit to a high filter cutoff.

Troubleshooting

- **Scenario:** Drone makes a "grinding" or "growling" sound when hovering in AltHold, but is quiet in manual Stabilize mode.
  - **Diagnosis:** High vertical vibration is being amplified by the Acceleration loop.
  - **Fix:** Reduce `PSC_ACCZ_FLTE` to 10.0 Hz and check if the noise disappears.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





PSC_ACCZ_FLTT (Position Control Acceleration (Vertical) Target Filter)

DEFAULT: 0.0	RANGE: 0 50	UNITS: Hz

Position Control Acceleration (Vertical) Target Filter (PSC_ACCZ_FLTT)

Description

`PSC_ACCZ_FLTT` smooths the "command" signal before it reaches the vertical acceleration loop.

While other filters (like `PSC_ACCZ_FLTE`) deal with *noise* from the sensors, this filter deals with *jumps* in the code. Because the altitude controller is a cascade of loops, a sudden change in the target altitude (a "step" input) can result in a sudden jump in the requested acceleration. This filter rounds off those sharp edges, ensuring the motors increase their RPM smoothly rather than with a violent "kick."

- **Low Frequency (e.g. 5Hz):** Very smooth vertical transitions. The drone "wafts" into its movements.
- **High Frequency (e.g. 40Hz):** Immediate response to target changes. The drone reacts "instantly" to stick inputs.
- **Default (0):** The filter is typically disabled or follows the main loop rate.

The Mathematics

This is a low-pass filter applied to the target acceleration ($\vec{A}_target$):

$$ \vec{A}*{target_filtered} = \vec{A}*{target_prev} + \alpha \cdot (\vec{A}*{target_raw} - \vec{A}*{target_prev}) $$

Where α is derived from `PSC_ACCZ_FLTT`:

$$\alpha = \frac{dt}{dt + \frac{1}{2\pi f_c}}$$

Effect: This effectively acts as a "Time Constant" for the vertical command. A 10Hz filter means the command will reach 63% of its target value in about 16ms.

The Engineer's View

This parameter maps to `_filt_T_hz` in the `_pid_accel_z` object.

It is applied in `AC_PID::update_all()`:

```
// AC_PID.cpp
_target += (target - _target) * get_filt_T_alpha(dt);
```

By filtering the target *at the entry point* of the innermost loop, ArduPilot ensures that the entire acceleration/throttle PID is protected from discontinuous jumps in the software's setpoints.

Tuning & Behavior

- **Default Value:** 0.0 Hz (Disabled)
- **Range:** 0.0 - 50.0 Hz
- **Effect of Increasing:** Snappier, more "connected" feel to the throttle stick.
- **Effect of Decreasing:** Smoother, more professional cinematic vertical movement.

Use Case Recommendations

- **Standard FPV / General Flying: Keep at 0.0.** The existing kinematic shapers (`PSC_JERK_Z`) already provide excellent smoothing.
- **Smooth Cinematic Reveleals: Set to 5.0 Hz.** Adds an extra layer of "organic" smoothness to any automated altitude changes.
- **Rescue / High-Speed Obstacle Avoidance: Keep at 0.0 or set >30Hz.** You want the absolute minimum lag when the avoidance system commands a climb.

Troubleshooting

- **Scenario:** Drone feels "disconnected" vertically; there is a noticeable lag between moving the throttle stick and the drone starting to climb.
  - **Diagnosis:** `PSC_ACCZ_FLTT` is likely set too low.
  - **Fix:** Increase to 20.0 Hz or set to 0.0.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





PSC_ACCZ_I (Position Control Acceleration (Vertical) I Gain)

DEFAULT: 1.0	RANGE: 0.0 3.0

Position Control Acceleration (Vertical) I Gain (PSC_ACCZ_I)

Description

`PSC_ACCZ_I` is the "gravity memory" of the drone. It works in the innermost vertical loop to ensure that the drone actually generates the required lift to stay level.

Its most important role is compensating for **Hover Throttle**. If you add a heavy camera to your drone, it now requires 60% throttle to hover instead of 50%. The I-term "notices" the persistent downward acceleration error and increases the baseline throttle until the drone stops sinking.

- **Low Value:** The drone will sink significantly when you first switch to AltHold or Loiter. It may also sink during the end of a battery's life as voltage drops.
- **High Value:** The drone adapts quickly to changes in weight or voltage.
- **Too High:** The drone will "bounce" or "hunt" for the hover point, creating a slow vertical surge.

The Mathematics

This parameter scales the accumulation of vertical acceleration error.

$$ \text{Throttle}_{out} += \int_0^t (\vec{A}_{z_target} - \vec{A}_{z_measured}) \cdot k_I \cdot dt $$

Where:

- $\vec{A}_{z_target} - \vec{A}_{z_measured}$ is the instantaneous error in vertical force.
- k_I is `PSC_ACCZ_I`.
- **Dimensionality:** This gain converts Acceleration-Time into Throttle.

Anti-Windup: ArduPilot heavily limits the maximum value of this integrator via `PSC_ACCZ_IMAX` to ensure the drone doesn't full-throttle itself into space if it's held down on a table.

The Engineer's View

This parameter is the `_ki` member of the `_pid_accel_z` object inside `AC_PosControl`.

It is executed in `AC_PID::update_all()`:

```
// AC_PID.cpp
_integrator += (_error * _ki) * dt;
```

A unique feature in ArduPilot is that this I-term is often **pre-seeded** (initialised) with the known `MOT_THST_HOVER` value during take-off to prevent the drone from "dropping" when it first transitions into an autonomous vertical mode.

Tuning & Behavior

- **Default Value:** 1.0
- **Range:** 0.0 - 3.0
- **Effect of Increasing:** Faster adaptation to weight changes. More consistent hover height as the battery drains.
- **Effect of Decreasing:** Smoother vertical behavior, but the drone will "sag" more under load.

Use Case Recommendations

- **Standard Build: Keep Default (1.0).** This is robust for almost all vehicles.
- **Heavy Lift / Cargo (Variable Weight): Increase to 1.5.** If you are dropping a payload, you want the drone to instantly "learn" its new lighter weight and not rocket upwards.
- **Low Power-to-Weight (Underpowered): Decrease to 0.5.** Prevents the integrator from pushing the motors into saturation too quickly.

Troubleshooting

- **Scenario:** Drone sinks for 2-3 seconds every time you switch to Loiter, then slowly climbs back to the right height.
  - **Diagnosis:** `PSC_ACCZ_I` is too low. It's taking too long to "learn" the gravity offset.
  - **Fix:** Increase `PSC_ACCZ_I` by 0.5.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





PSC_ACCZ_IMAX (Position Control Acceleration (Vertical) I-Term Maximum)

DEFAULT: 800	RANGE: 0 1000	UNITS: d%

Position Control Acceleration (Vertical) I-Term Maximum (PSC_ACCZ_IMAX)

Description

`PSC_ACCZ_IMAX` is a critical safety limit for the vertical controller. It defines the maximum "authority" given to the vertical integrator (`PSC_ACCZ_I`).

The integrator's job is to learn the hover throttle. However, if the drone is physically prevented from climbing (e.g., it is tangled in a tree, or you are holding it down during a test), the integrator will keep increasing the throttle command to try and fix the error. Without a limit, the drone would eventually reach 100% throttle, leading to a dangerous situation if it suddenly breaks free.

- **Low Value:** Limits the drone's ability to compensate for very heavy payloads or extremely low battery voltages.
- **High Value:** Allows the drone to compensate for massive weight changes, but increases the risk of "runaway" throttle if the drone is stuck.
- **Units:** Expressed in "Deci-percent" of throttle (e.g., 800 = 80%).

The Mathematics

The integrator ($\vec{I}$) is calculated over time, but is clamped by this parameter:

$$ \vec{I}_{new} = \text{constrain}(\vec{I}_{prev} + \text{error} \cdot k_I \cdot dt, -\text{IMAX}, \text{IMAX}) $$

Where:

- IMAX is `PSC_ACCZ_IMAX`.
- The output is typically scaled so that 1000 = full motor authority.

The Engineer's View

This parameter maps to `_kimax` in the `_pid_accel_z` object.

It is applied in `AC_PID::update_all()`:

```
// AC_PID.cpp
_integrator = constrain_float(_integrator + (error * _ki * dt), -_kimax, _kimax);
```

ArduPilot also includes a **Secondary Safety**: the integrator is not allowed to increase if the motors are already at their physical limits (`_motors.limit.throttle_upper`), which further prevents windup.

Tuning & Behavior

- **Default Value:** 800 (80% throttle authority)
- **Range:** 0 - 1000
- **Effect of Increasing:** Allows the drone to handle much heavier payloads than its "natural" hover point.
- **Effect of Decreasing:** Safer for indoor testing; ensures the drone can't "power out" of a situation with more than a certain amount of force.

Use Case Recommendations

- **Standard Quad: Keep Default (800).** This is safe and effective.
- **Heavy Lift / Delivery Drone: Increase to 1000 (if needed).** If your payload is so heavy that you hover at 75% throttle, you need the full 1000 range to ensure the drone can still climb effectively.
- **Tethered Drone: Decrease to 400.** If the drone is on a power-tether, you want to strictly limit how much it can pull on the cable if the sensors fail.

Troubleshooting

- **Scenario:** Drone hovers perfectly at the start of the flight, but starts to sink slowly as the battery gets low, and never recovers.
  - **Diagnosis:** `PSC_ACCZ_IMAX` is too low. The drone needs more than 80% authority to stay level at low voltage, but the integrator is hitting the cap.
  - **Fix:** Increase `PSC_ACCZ_IMAX` to 1000.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY

PSC_ACCZ_NEF (Position Control Acceleration (Vertical) Error Notch Filter Index)

DEFAULT: 0	RANGE: 1 8

Position Control Acceleration (Vertical) Error Notch Filter Index (PSC_ACCZ_NEF)

Description

`PSC_ACCZ_NEF` is an advanced "surgical" filter for altitude control. While standard filters (like `PSC_ACCZ_FLTE`) remove *all* high-frequency noise, a **Notch Filter** removes noise at one specific, problematic frequency (and its harmonics).

On many drones, the frame or the vertical column of air under the propellers has a "resonant frequency"—a specific speed where the whole drone starts to vibrate vertically. If the altitude controller "hears" this vibration and tries to correct it, it can lead to a feedback loop that destroys the drone. By setting this index, you tell the vertical controller to use one of your pre-configured Harmonic Notch filters (usually linked to motor RPM or an FFT) to specifically ignore that resonant vertical frequency.

- **Set to 0:** No notch filter is applied to the vertical error.
- **Set to 1-8:** Uses the corresponding Harmonic Notch filter (configured via `INS_HNTCH_*` parameters).

The Mathematics

The notch filter (H_notch) is applied to the error signal (E) before it enters the PID logic:

$$ \vec{E}*{clean} = H*{notch}(\vec{A}*{z_target} - |*vec{A}*{z_measured}) $$

Where H_notch is a dynamic filter whose center frequency is typically driven by motor RPM ($f \propto RPM$):

$$f_center = \text{INS_HNTCH_FREQ} \cdot \frac{\text{RPM}}{\text{RPM}_ref}$$

The Engineer's View

This parameter maps to `_notch_E_filter` in the `_pid_accel_z` object.

It is applied in `AC_PID::update_all()`:

```
// AC_PID.cpp
if (_notch_E_filter > 0) {
    // Note: The actual filtering happens via a pointer to the
    // INS harmonic notch library.
    error = _notch_E_filter_ptr->apply(error);
}
```

This integration allows the innermost altitude loop to benefit from the same advanced vibration rejection techniques used by the rate controllers (the main flight loops).

Tuning & Behavior

- **Default Value:** 0 (Disabled)
- **Range:** 1 - 8 (Filter Index)
- **Effect of Enabling:** Significantly cleaner motor signals in Loiter. Allows for higher vertical P-gains (`PSC_ACCZ_P`) without causing vertical "chirp."
- **Effect of Disabling:** More susceptible to vertical resonance and Z-axis motor noise.

Use Case Recommendations

- **Large Professional Drones: Set to 1 (usually).** If you have already configured a Harmonic Notch for your main flight loops, applying it to the vertical loop is highly recommended to improve altitude hold performance in wind.
- **Small Racers: Keep at 0.** Smaller drones usually have resonant frequencies high enough that a standard 20Hz low-pass filter is sufficient.
- **Coaxial / X8 Drones: Set to 1.** Overlapping prop wash creates complex vertical resonances; the notch filter is often mandatory for a stable hover.

Troubleshooting

- **Scenario:** Drone hovers well, but you see constant high-frequency oscillations in your `PSCZ.Acc` logs that match your motor RPM.
  - **Diagnosis:** Vertical resonance is passing into the altitude controller.
  - **Fix:** Ensure a Harmonic Notch is configured for the frame, and set `PSC_ACCZ_NEF` to that notch's index.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





PSC_ACCZ_NTF (Position Control Acceleration (Vertical) Target Notch Filter Index)

DEFAULT: 0	RANGE: 1 8

Position Control Acceleration (Vertical) Target Notch Filter Index (PSC_ACCZ_NTF)

Description

`PSC_ACCZ_NTF` is the sister filter to `PSC_ACCZ_NEF`. While the "Error Notch" filter cleans up noise from the sensors (IMU), the **Target Notch** filter cleans up noise from the commands (Software).

In a complex vertical control system, the "target acceleration" is often the result of many calculations—path planning, terrain following, and obstacle avoidance. Sometimes, these software processes can introduce rhythmic "stepping" or high-frequency ripples into the command signal. If these ripples happen to match the physical resonance of the frame or the motors, they can cause the drone to vibrate. This parameter allows the drone to specifically "ignore" those command-side frequencies.

- **Set to 0:** No notch filter is applied to the vertical target.
- **Set to 1-8:** Uses the corresponding Harmonic Notch filter (configured via `INS_HNTCH_*`).

The Mathematics

The notch filter (H_notch) is applied to the target acceleration signal (T) before it enters the PID loop:

$$ \vec{T}*{clean} = H*{notch}(\vec{A}_{z_target_raw}) $$

The clean target $\vec{T}_clean$ is then compared against the actual acceleration to compute the error.

The Engineer's View

This parameter maps to `_notch_T_filter` in the `_pid_accel_z` object.

It is applied in `AC_PID::update_all()`:

```
// AC_PID.cpp
if (_notch_T_filter > 0) {
    target = _notch_T_filter_ptr->apply(target);
}
```

This is rarely used on standard multirotors but is critical for large, flexible frames or vehicles where the path planner (e.g., Terrain Following) might be updating at a lower rate than the PID loops, creating high-frequency "stairs" in the input signal.

Tuning & Behavior

- **Default Value:** 0 (Disabled)
- **Range:** 1 - 8 (Filter Index)
- **Effect of Enabling:** Smoother motor sound during vertical maneuvers. Reduces "ringing" in the motor output when following complex vertical paths.
- **Effect of Disabling:** Maximum responsiveness to command changes.

Use Case Recommendations

- **Terrain Following / Lidar-Based flight: Consider setting to 1.** Lidar and terrain sensors often update at low frequencies (10-50Hz). This notch filter can round off the "jumps" in target altitude as the drone crosses uneven terrain.
- **Standard Build: Keep at 0.** Standard low-pass filters (`PSC_ACCZ_FLTT`) are usually sufficient for command smoothing.
- **Very Large VTOL (Flexible Wings): Set to 1.** Flexible wings can act like giant vertical springs. A command-side notch filter ensures the software never "kicks" the wing at its resonant frequency.

Troubleshooting

- **Scenario:** Drone makes a distinct "thumping" or "humming" sound specifically when following terrain or a mission path, but is quiet when hovering still.
  - **Diagnosis:** Command-side resonance.
  - **Fix:** Apply a notch filter at the resonant frequency of the frame via `PSC_ACCZ_NTF`.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





PSC_ACCZ_P (Position Control Acceleration (Vertical) P Gain)

DEFAULT: 0.5	RANGE: 0.2 1.5

Position Control Acceleration (Vertical) P Gain (PSC_ACCZ_P)

Description

`PSC_ACCZ_P` is the innermost "knob" of the altitude control system. It determines how hard the flight controller pushes the motors to correct an error in **Vertical Acceleration**.

While `PSC_POSZ_P` cares about height and `PSC_VELZ_P` cares about climb speed, `PSC_ACCZ_P` cares about the *instantaneous force* required to counteract gravity and change speed. It effectively translates "I need more lift" into a specific motor PWM signal.

- **Low Value:** The drone will feel "heavy" and slow to recover from drops. It may feel like it is struggling to stay airborne when carrying a heavy battery.
- **High Value:** The drone will feel very "light" and responsive.
- **Too High:** The drone will oscillate rapidly in the vertical axis, creating a "buzzing" sound as the motors rapidly oscillate their RPM.

The Mathematics

This parameter is the Proportional (k_P) term in the Vertical Acceleration PID loop.

$$ \text{Throttle}_{out} = k_P \cdot (\vec{A}_z_target} - \vec{A}_z_measured}) + \dots + \text{Hover_Throttle} $$

Where:

- $\vec{A}_z_target$ is the acceleration required to reach the target climb rate.
- $\vec{A}_z_measured$ is the vertical acceleration from the IMU (with gravity removed).
- k_P is `PSC_ACCZ_P`.
- **Dimensionality:** This gain converts Acceleration (LT^{-2}) into a dimensionless Throttle percentage (0.0 to 1.0).

The Engineer's View

This parameter is the `_kp` member of the `_pid_accel_z` object (class `AC_PID`) inside `AC_PosControl`.

It is executed in `AC_PosControl::update_z_controller()`:

```
// AC_PosControl.cpp
thr_out = _pid_accel_z.update_all(_accel_target.z, z_accel_meas, _dt, ...) * 0.001f;
thr_out += _motors.get_throttle_hover();
```

Note the `* 0.001f` scaling. This is because the internal PID library often calculates in larger units (e.g., centi-acceleration) and then scales back to the $0..1$ throttle range.

Tuning & Behavior

- **Default Value:** 0.5
- **Range:** 0.2 - 1.5
- **Effect of Increasing:** Snappier recovery from vertical drops. Better tracking of the terrain in terrain-following modes.
- **Effect of Decreasing:** Smoother, "softer" vertical behavior. Reduces vertical jitters in high-vibration environments.

Use Case Recommendations

- **Racing / Acro-Loiter: Increase to 0.7 - 0.9.** Provides the high-authority lift needed for aggressive maneuvers.
- **Heavy Cargo Drone: Decrease to 0.3 - 0.4.** Heavy vehicles have slow motor response times; pushing them too hard vertically with a high P-gain will lead to dangerous "vertical porpoising."
- **High Vibration Frames: Decrease to 0.3.** High vibration creates noise in the IMU's Z-acceleration measurement. Reducing the P-gain prevents this noise from being amplified into the motors.

Troubleshooting

- **Scenario:** Drone "bobs" up and down rapidly after a vertical punch.
  - **Diagnosis:** `PSC_ACCZ_P` is too high (P-gain oscillation).
  - **Fix:** Reduce `PSC_ACCZ_P` by 0.1 until the bobbing stops.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





PSC_ACCZ_PDMX (Position Control Acceleration (Vertical) PD Maximum)

DEFAULT: 0	RANGE: 0 1000	UNITS: d%

Position Control Acceleration (Vertical) PD Maximum (PSC_ACCZ_PDMX)

Description

`PSC_ACCZ_PDMX` is a safety clamp for the "active" parts of the vertical controller. In a PID loop, the **P** and **D** terms are responsible for reacting to noise and sudden changes. If the drone is hit by a massive vertical gust, the P and D terms might compute a requested throttle change of 200%—which is physically impossible and can cause the control loop to behave erratically.

This parameter sets a ceiling on the combined power of P and D. It ensures that no matter how big the vertical error is, the *immediate* correction requested by the drone is kept within a reasonable range.

- **Low Value:** Limits the drone's "snap" or ability to fight hard against turbulence.
- **High Value:** Allows the drone to use its full motor power for immediate corrections.
- **Disabled (0):** The PD sum is unlimited (limited only by the physical constraints of the motors).

The Mathematics

The P and D terms are summed and then constrained by this value:

$$\text{PD_}output = \text{constrain}(P + D, -\text{PDMX}, \text{PDMX})$$

Where:

- P is the proportional contribution ($k_P \cdot \text{error}$).
- D is the derivative contribution ($k_D \cdot \frac{d}{dt}\text{error}$).
- PDMX is `PSC_ACCZ_PDMX`.

Note: This limit is applied *before* the Integral (I) term and Feed-Forward (FF) are added. This ensures that even if P and D are clamped, the drone can still achieve its full lift via the I-term.

The Engineer's View

This parameter maps to `_kpdmax` in the `_pid_accel_z` object.

It is applied in `AC_PID::update_all()`:

```
// AC_PID.cpp
if (is_positive(_kpdmax)) {
    float pd_sum = P + D;
    pd_sum = constrain_float(pd_sum, -_kpdmax, _kpdmax);
    // ... use pd_sum ...
}
```

This is particularly useful in high-vibration environments where the D-term might otherwise produce extreme, unnecessary throttle spikes.

Tuning & Behavior

- **Default Value:** 0 (Unlimited)
- **Range:** 0 - 1000 (0 to 100% throttle authority)
- **Effect of Increasing:** Increases the drone's ability to "punch" out of a vertical drop.
- **Effect of Decreasing:** Makes the altitude hold safer and smoother, but less capable of handling extreme vertical wind.

Use Case Recommendations

- **Standard Drone: Keep at 0.** Most modern builds don't need this clamp.
- **Large-Scale VTOL / Passenger Drones: Set to 500 (50%).** Ensures that no software glitch or sensor spike can command more than half-throttle correction instantly, providing a smoother ride.
- **Indoor / Research Drones: Set to 300 (30%).** A safe setting for testing new code where you want to limit how much the drone can "jump" if a sensor fails.

Troubleshooting

- **Scenario:** Drone is very slow to recover from a vertical drop, even though `PSC_ACCZ_P` is high.
  - **Diagnosis:** `PSC_ACCZ_PDMX` might be set too low, capping the controller's authority.
  - **Fix:** Increase `PSC_ACCZ_PDMX` or set it to 0.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





PSC_ACCZ_SMAX (Position Control Acceleration (Vertical) Slew Rate Limit)

DEFAULT: 0.0	RANGE: 0 200

Position Control Acceleration (Vertical) Slew Rate Limit (PSC_ACCZ_SMAX)

Description

`PSC_ACCZ_SMAX` is a protection mechanism for your motors and frame. A **Slew Rate** is a limit on how fast a signal can change over time.

If the vertical acceleration loop commands 10% throttle now and 90% throttle in the very next millisecond (due to noise or a sudden correction), the motors will try to accelerate violently. This causes massive current spikes and can actually shake the drone's frame apart. This parameter rounds off those sudden spikes, ensuring that the motor command always changes at a physically safe rate.

- **Low Value:** Flight feels extremely soft and "buttery." The motors will sound very quiet.
- **High Value:** Allows the motors to reach their target power as fast as they physically can.
- **Too High:** Risk of "motor shrieking" and ESC damage due to extreme current transients.
- **Disabled (0):** The software does not limit the rate of change of the motor command.

The Mathematics

The slew limit acts as a "speed limit" for the change in the PID output (ΔO):

$$|O_new - O_prev| \leq SMAX \cdot dt$$

If the calculated change in output exceeds $SMAX$, ArduPilot automatically reduces the P and D gains of the acceleration loop for that specific cycle to keep the output within the limit:

$$\text{Scale Factor} = \frac{SMAX}{|P + D|_rate}$$

The Engineer's View

This parameter maps to `_slew_rate_max` in the `_pid_accel_z` object.

It is executed in `AC_PID::update_all()`:

```
// AC_PID.cpp
if (is_positive(_slew_rate_max)) {
    _pid_info.slew_rate = (P + D - _last_out_pd) / dt;
    if (fabsf(_pid_info.slew_rate) > _slew_rate_max) {
        // Reduce gains to meet the slew rate
        float gain_reduction = _slew_rate_max / fabsf(_pid_info.slew_rate);
        // ... apply reduction ...
    }
}
```

This is a **dynamic gain reduction** strategy. Instead of just clipping the output (which creates lag), it softens the controller until the "jerkiness" is gone.

Tuning & Behavior

- **Default Value:** 0.0 (Disabled)
- **Range:** 0 - 200
- **Effect of Increasing:** Snappier, more immediate vertical control authority.
- **Effect of Decreasing:** Smoother motor response, longer motor life, and potentially less "jello" in the camera feed.

Use Case Recommendations

- **Standard Build: Keep at 0.0.** Modern ESCs and frames are usually strong enough to handle raw PID output.
- **Large Professional Drones (>10kg): Set to 50 - 100.** Large drones have massive rotating inertia; trying to change their motor speed instantly is inefficient and stresses the ESCs.
- **Passenger-Carrying / Large VTOL: Set to 20.** Safety and comfort are paramount; violent motor changes must be prevented by the software.

Troubleshooting

- **Scenario:** Motors are making a "crunchy" or "scratchy" sound during fast descents.
  - **Diagnosis:** High-frequency PID chatter is creating excessive slew rates.
  - **Fix:** Set `PSC_ACCZ_SMAX` to 100 and check if the sound cleans up.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY

PSC_ANGLE_MAX (Position Control Angle Max)

DEFAULT: 0	RANGE: 0 4500	UNITS: cdeg

Position Control Angle Max (PSC_ANGLE_MAX)

Description

The `PSC_ANGLE_MAX` parameter serves as a safety governor for autonomous flight. It strictly limits how far the drone can tilt (lean) when it is flying itself (Loiter, Auto, Guided, RTL).

By default (`0`), it inherits the global `ANGLE_MAX` setting (typically 30-45 degrees), which applies to manual flight (Stabilize/AltHold). However, you often want autonomous modes to be gentler than manual modes. This parameter allows you to cap the auto-flight tilt to a lower value (e.g., 20 degrees) for smoother video or passenger comfort, while keeping the full range available for manual recovery.

- **Set to 0:** Disabled. Uses `ANGLE_MAX`.
- **Set > 0:** Overrides `ANGLE_MAX` for Position Control modes.

The Mathematics

The logic uses a ternary selector:

$$\theta_limit = \begin{cases} \text{PSC_ANGLE_MAX}, & \text{if PSC_ANGLE_MAX} > 0 \\ \text{ANGLE_MAX}, & \text{otherwise} \end{cases}$$

Where θ_limit is the maximum centi-degrees of roll or pitch allowed.

The Engineer's View

This parameter corresponds to `_lean_angle_max` in `AC_PosControl`.

The logic is explicitly defined in `AC_PosControl::get_lean_angle_max_cd()`:

```
// AC_PosControl.cpp
float AC_PosControl::get_lean_angle_max_cd() const
{
    if (is_positive(_angle_max_override_cd)) {
        return _angle_max_override_cd;
    }
    if (!is_positive(_lean_angle_max)) {
        return _attitude_control.lean_angle_max_cd(); // Returns global ANGLE_MAX
    }
    return _lean_angle_max * 100.0f;
}
```

This limit is applied at the very end of the `update_xy_controller()` loop to clip the requested acceleration.

Tuning & Behavior

- **Default Value:** 0 (Inherit Global)
- **Range:** 0 - 4500 (0 to 45 degrees)
- **Effect of Increasing:** Allows the drone to fly faster in Auto/Loiter (since speed comes from tilt).
- **Effect of Decreasing:** Limits top speed and acceleration. Makes flight smoother but reduces wind resistance.

Use Case Recommendations

- **Cinematography: Set to 1500 - 2000 (15-20 deg).**
  - *Why:* Ensures the drone never tilts aggressively, keeping the landing gear out of the shot and the horizon stable.
- **High Wind Operations: Set to 0 (or >3000).**
  - *Why:* If you cap this too low (e.g., 10 degrees) and the wind requires a 12-degree lean to hold position, the drone will be blown away (Fly-away risk).
- **General: Keep Default (0).**

PSC_D_ACC_D (PID Derivative Gain)

DEFAULT: 0	RANGE: null	UNITS: null

PID Derivative Gain

Note: This parameter functions identically to PSC_ACCZ_D.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





PSC_D_ACC_D_FF (FF FeedForward Gain)

DEFAULT: 0	RANGE: null	UNITS: null

FF FeedForward Gain

Note: This parameter functions identically to PSC_ACCZ_D_FF.

PSC_D_ACC_FF (FF FeedForward Gain)

DEFAULT: 0	RANGE: null	UNITS: null

FF FeedForward Gain

Note: This parameter functions identically to PSC_ACCZ_FF.

PSC_D_ACC_FLTD (PID D term filter frequency in Hz)

DEFAULT: AC_PID_DFILT_HZ_DEFAULT	RANGE: null	UNITS: Hz

PID D term filter frequency in Hz

Note: This parameter functions identically to PSC_ACCZ_FLTD.

PSC_D_ACC_FLTE (PID Error filter frequency in Hz)

DEFAULT: AC_PID_EFILT_HZ_DEFAULT	RANGE: null	UNITS: Hz

PID Error filter frequency in Hz

Note: This parameter functions identically to PSC_ACCZ_FLTE.

PSC_D_ACC_FLTT (PID Target filter frequency in Hz)

DEFAULT: AC_PID_TFILT_HZ_DEFAULT	RANGE: null	UNITS: Hz

PID Target filter frequency in Hz

Note: This parameter functions identically to PSC_ACCZ_FLTT.

PSC_D_ACC_I (PID Integral Gain)

DEFAULT: 0	RANGE: null	UNITS: null

PID Integral Gain

Note: This parameter functions identically to PSC_ACCZ_I.

PSC_D_ACC_IMAX (PID Integral Maximum)

DEFAULT: 0	RANGE: null	UNITS: null

PID Integral Maximum

Note: This parameter functions identically to PSC_ACCZ_IMAX.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





PSC_D_ACC_NEF (PID Error notch filter index)

DEFAULT: 0	RANGE: 1 8	UNITS: null

PID Error notch filter index

Note: This parameter functions identically to PSC_ACCZ_NEF.

PSC_D_ACC_NTF (PID Target notch filter index)

DEFAULT: 0	RANGE: 1 8	UNITS: null

PID Target notch filter index

Note: This parameter functions identically to PSC_ACCZ_NTF.

PSC_D_ACC_P (PID Proportional Gain)

DEFAULT: 0	RANGE: null	UNITS: null

PID Proportional Gain

Note: This parameter functions identically to PSC_ACCZ_P.

PSC_D_ACC_PDMX (PD sum maximum)

DEFAULT: 0	RANGE: null	UNITS: null

PD sum maximum

Note: This parameter functions identically to PSC_ACCZ_PDMX.

PSC_D_ACC_SMAX (Slew rate limit)

DEFAULT: 0	RANGE: 0 200	UNITS: null

Slew rate limit (PSC_D_ACC_SMAX)

Note: This parameter functions identically to PSC_ACCZ_SMAX.

PSC_D_POS_P (PID Proportional Gain)

DEFAULT: 0	RANGE: null	UNITS: null

PID Proportional Gain

Note: This parameter functions identically to PSC_POSZ_P.

PSC_D_VEL_D (PID Derivative Gain)

DEFAULT: 0	RANGE: null	UNITS: null

PID Derivative Gain

Note: This parameter functions identically to PSC_VELZ_D.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





PSC_D_VEL_FF (FF FeedForward Gain)

DEFAULT: 0	RANGE: null	UNITS: null

FF FeedForward Gain

Note: This parameter functions identically to PSC_VELZ_FF.

PSC_D_VEL_FLTD (PID D term filter frequency in Hz)

DEFAULT: AC_PID_DFILT_HZ_DEFAULT	RANGE: null	UNITS: Hz

PID D term filter frequency in Hz

Note: This parameter functions identically to PSC_VELZ_FLTD.

PSC_D_VEL_FLTE (PID Error filter frequency in Hz)

DEFAULT: AC_PID_EFILT_HZ_DEFAULT	RANGE: null	UNITS: Hz

PID Error filter frequency in Hz

Note: This parameter functions identically to PSC_VELZ_FLTE.

PSC_D_VEL_I (PID Integral Gain)

DEFAULT: 0	RANGE: null	UNITS: null

PID Integral Gain

Note: This parameter functions identically to PSC_VELZ_I.

PSC_D_VEL_IMAX (PID Integral Maximum)

DEFAULT: 0	RANGE: null	UNITS: null

PID Integral Maximum

Note: This parameter functions identically to PSC_VELZ_IMAX.

PSC_D_VEL_P (PID Proportional Gain)

DEFAULT: 0	RANGE: null	UNITS: null

PID Proportional Gain

Note: This parameter functions identically to PSC_VELZ_P.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





PSC_JERK_D (Position Control Jerk (Down/Vertical))

DEFAULT: 5.0	RANGE: 5.0 50.0	UNITS: m/s^3

Position Control Jerk (Down/Vertical) (PSC_JERK_D)

Description

`PSC_JERK_D` is a common alias (often used in documentation and older Ground Control Stations) for the **PSC_JERK_Z** parameter. It controls the "smoothness" of vertical transitions (takeoff, landing, and climb/descent starts).

For full details on how this parameter affects flight dynamics and for tuning advice, please refer to the primary documentation page:

Click here for PSC_JERK_Z Documentation

The Mathematics

Identical to `PSC_JERK_Z`. It defines the third derivative of vertical position:

$$j_z = \frac{d^3z}{dt^3}$$

The Engineer's View

In modern ArduPilot C++ source code, this maps to `_shaping_jerk_z` within the `AC_PosControl` class. While older parameter systems may have used the "D" (Down) suffix for the NEU (North East Up) coordinate frame's vertical axis, modern versions standardize on "Z".

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





PSC_JERK_XY (Position Control Jerk XY)

DEFAULT: 5.0	RANGE: 1.0 20.0	UNITS: m/s^3

Position Control Jerk XY (PSC_JERK_XY)

Description

`PSC_JERK_XY` defines the "smoothness" of the drone's horizontal movements. In physics, **Jerk** is the rate at which acceleration changes. High jerk means the drone snaps into a lean instantly; low jerk means it rolls into the lean gracefully.

ArduPilot uses **S-Curve Kinematics** to generate paths. Instead of jerky, "bang-bang" movements, the controller shapes the velocity profile so that acceleration and deceleration are applied gradually, respecting this Jerk limit.

- **Low Value:** Flight feels very "fluid" and "soft." Ideal for cinematography.
- **High Value:** Flight feels "robotic" and "snappy." The drone stops and starts with minimal lag.
- **Too High:** Can cause mechanical stress or "ringing" (oscillation) as the motors try to change the vehicle's attitude faster than is physically possible.

The Mathematics

Jerk (j) is the third derivative of position (p):

$$j(t) = \frac{d\vec{A}}{dt} = \frac{d^3\vec{P}}{dt^3}$$

In the ArduPilot shaper, this parameter limits the slope of the acceleration curve:

$$ |\vec{A}_{target}(t) - \vec{A}_{target}(t-dt)| \leq j_{limit} \cdot dt $$

Where j_{limit} is `PSC_JERK_XY`.

The Engineer's View

This parameter is the `_shaping_jerk_xy` member in `AC_PosControl`.

It is used in `AC_PosControl::set_max_speed_accel_xy()` to initialize the kinematic path generators:

```
// AC_PosControl.cpp
_jerk_max_xy_cmsss = _shaping_jerk_xy * 100.0;
```

Crucially, the code includes a **Safety Override**: it will automatically lower your jerk limit if the Attitude Controller's maximum angular rates (`ATC_RAT_RLL_P` etc.) are too low to support the requested snap. It ensures the position controller doesn't "ask" for a lean faster than the drone can rotate.

Tuning & Behavior

- **Default Value:** 5.0 m/s³
- **Range:** 1.0 - 20.0 m/s³
- **Effect of Increasing:** Snappier response to stick inputs in Loiter. Tighter cornering in Auto missions.
- **Effect of Decreasing:** Smoother, "lazier" movement. Great for reducing "jello" in video.

Use Case Recommendations

- **Cinematic Drone: Decrease to 1.0 - 2.5.** Creates beautiful, smooth transitions between hover and flight.
- **Industrial / Delivery: Keep Default (5.0).** Good balance of speed and mechanical preservation.
- **Racing / Aggressive Auto: Increase to 10.0 - 15.0.** Necessary for high-speed obstacle avoidance where every millisecond of acceleration counts.

Troubleshooting

- **Scenario:** Drone "bobs" or "overshoots" slightly when stopping, despite high D-gain.
  - **Diagnosis:** `PSC_JERK_XY` might be too low, preventing the drone from applying brakes fast enough.
  - **Fix:** Increase `PSC_JERK_XY` in increments of 1.0.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY

PSC_JERK_Z (Position Control Jerk Z (Vertical))

DEFAULT: 5.0	RANGE: 5.0 50.0	UNITS: m/s^3

Position Control Jerk Z (Vertical) (PSC_JERK_Z)

Description

`PSC_JERK_Z` defines the vertical "acceleration ramp." It limits how quickly the drone can change its vertical acceleration.

In simple terms, it prevents the drone from "punching" the throttle or "slamming" the brakes when changing altitude. A high jerk value allows the drone to reach its target climb rate instantly, while a low value makes it "float" into the climb.

- **Low Value:** Altitude changes feel very smooth and gentle. The drone "glides" into a stop. Ideal for cinematography or passenger-carrying vehicles.
- **High Value:** The drone is very responsive to height changes. It stops exactly on the target line with no hesitation.
- **Too High:** Can cause mechanical stress on the frame and battery, and potentially lead to vertical "ringing" as the motors try to achieve physically impossible acceleration rates.

The Mathematics

Vertical Jerk (j_z) is the rate of change of vertical acceleration (a_z):

$$j_z(t) = \frac{da_z}{dt} = \frac{d^3z}{dt^3}$$

The ArduPilot shaper ensures that:

$$ |\vec{A}{z_target}(t) - |\vec{A}{z_target}(t-dt)| \leq j_{z_limit} \cdot dt $$

Where j_z_limit is `PSC_JERK_Z`.

The Engineer's View

This parameter is the `_shaping_jerk_z` member in `AC_PosControl`.

It is used in `AC_PosControl::set_max_speed_accel_z()` to initialize the vertical shaper:

```
// AC_PosControl.cpp
_jerk_max_z_cmsss = _shaping_jerk_z * 100.0;
```

The shaper is applied in `AC_PosControl::input_pos_vel_accel_z()`, where the commanded altitude is transformed into a kinematically consistent target that the drone's motors can actually follow.

Tuning & Behavior

- **Default Value:** 5.0 m/s³
- **Range:** 5.0 - 50.0 m/s³
- **Effect of Increasing:** Snappier takeoff and landing. More aggressive response to altitude stick (throttle).
- **Effect of Decreasing:** Smoother, "softer" vertical movement. Great for reducing camera gimbal "bob" during climb-out.

Use Case Recommendations

- **Cinematography: Keep Default (5.0) or decrease to 2.5.** Standard 5.0 is already quite smooth, but for very slow cinematic reveals, lower is better.
- **Autonomous Landing (Precision): Increase to 10.0 - 15.0.** When landing on a small target or a moving platform, you want the altitude controller to be extremely responsive to avoid "drifting" during the touchdown phase.
- **Industrial Inspection: Increase to 10.0.** Allows the drone to maintain a consistent distance from a structure (e.g., wind turbine blade) even in gusty vertical air.

Troubleshooting

- **Scenario:** Drone "over-accelerates" vertically when the throttle stick is moved small amounts, making fine altitude adjustments difficult.
  - **Diagnosis:** `PSC_JERK_Z` might be too high for the vehicle's thrust-to-weight ratio.
  - **Fix:** Reduce `PSC_JERK_Z` to 5.0 or 2.5.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY

PSC_NE_POS_P (PID Proportional Gain)

DEFAULT: 0	RANGE: null	UNITS: null

PID Proportional Gain

Note: This parameter functions identically to PSC_POSXY_P.

PSC_NE_VEL_D (PID Derivative Gain)

DEFAULT: 0	RANGE: null	UNITS: null

PID Derivative Gain

Note: This parameter functions identically to PSC_VELXY_D.

PSC_NE_VEL_FF (FF FeedForward Gain)

DEFAULT: 0	RANGE: null	UNITS: null

FF FeedForward Gain

Note: This parameter functions identically to PSC_VELXY_FF.

PSC_NE_VEL_FLTD (PID D term filter frequency in Hz)

DEFAULT: AC_PID_DFILT_HZ_DEFAULT	RANGE: null	UNITS: Hz

PID D term filter frequency in Hz

Note: This parameter functions identically to PSC_VELXY_FLTD.

PSC_NE_VEL_FLTE (PID Error filter frequency in Hz)

DEFAULT: AC_PID_EFILT_HZ_DEFAULT	RANGE: null	UNITS: Hz

PID Error filter frequency in Hz

Note: This parameter functions identically to PSC_VELXY_FLTE.

PSC_NE_VEL_I (PID Integral Gain)

DEFAULT: 0	RANGE: null	UNITS: null

PID Integral Gain

Note: This parameter functions identically to PSC_VELXY_I.

PSC_NE_VEL_IMAX (PID Integral Maximum)

DEFAULT: 0	RANGE: null	UNITS: null

PID Integral Maximum

Note: This parameter functions identically to PSC_VELXY_IMAX.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





PSC_NE_VEL_P (PID Proportional Gain)

DEFAULT: 0	RANGE: null	UNITS: null

PID Proportional Gain

Note: This parameter functions identically to PSC_VELXY_P.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





PSC_POSXY_P (Position Control (XY) P Gain)

DEFAULT: 1.0	RANGE: 0.5 2.0

Position Control (XY) P Gain (PSC_POSXY_P)

Description

The `PSC_POSXY_P` parameter determines the "stiffness" of the vehicle's position hold in the horizontal (XY) plane. It acts as the primary driver for converting position errors (how far you are from the target) into requested velocities (how fast you should fly to get back).

- **Low Value:** The drone feels "loose" and will drift further from the target before correcting. It may feel sluggish to stop.
- **High Value:** The drone fights aggressively to hold position. It stops sharply.
- **Too High:** The drone will overshoot the target and oscillate (jitter) around the position hold, often observed as a rapid "twitching" in Loiter mode.

The Mathematics

This parameter is the Proportional (k_P) term in the outer loop of the cascaded position controller.

$$ \vec{V}*{target} = k_P |cdot (|vec{P}*{target} - \vec{P}_{current}) $$

Where:

- $\vec{V}_target$ is the requested velocity (in cm/s) sent to the inner Velocity Loop.
- $\vec{P}_target$ is the desired location (e.g., where the sticks are commanded or the loiter point).
- $\vec{P}_current$ is the EKF estimated position.
- k_P is the `PSC_POSXY_P` gain.
- **Dimensionality:** Since it converts Position (L) to Velocity (LT^{-1}), the unit of k_P is $1/s$ (Hertz).

The Engineer's View

This parameter is an instance of the `AC_P_2D` class, instantiated as `_p_pos_xy` within the `AC_PosControl` library.

It is executed on every main loop cycle (typically 400Hz) inside `AC_PosControl::update_xy_controller()`:

```
// AC_PosControl.cpp
Vector2f vel_target = _p_pos_xy.update_all(_pos_target.x, _pos_target.y, comb_pos);
```

The output of this P-controller becomes the input target for the **Velocity PID** loop (`PSC_VELXY_*`), meaning any noise or oscillation here will cascade down into the velocity and acceleration controllers.

Tuning & Behavior

- **Default Value:** 1.0 (Standard for most quadcopters)
- **Range:** 0.5 - 2.0
- **Effect of Increasing:** Sharper position holding, faster braking when sticks are released.
- **Effect of Decreasing:** Smoother braking, "softer" loiter, less aggressive correction against wind gusts.
- **Dependencies:** If you change this significantly, you may need to re-tune `PSC_VELXY_P` and `PSC_VELXY_D` to handle the more aggressive velocity demands.

Use Case Recommendations

- **Cinematography: Decrease to 0.7 - 0.8.**
  - *Why:* Prevents the "hard stop" when you release the sticks at a waypoint. It creates a gentle, coasting deceleration that looks more organic on camera.
- **Precision Inspection / Survey: Increase to 1.2 - 1.5.**
  - *Why:* When inspecting a wind turbine or bridge, you want the drone to hold its coordinate exactly against wind gusts, even if the motor response is aggressive.
- **General Flying: Keep Default (1.0).**
  - *Why:* Provides a balanced feel where the drone is responsive but not twitchy.

Practical Tuning Tip

For large, heavy cinematic drones, reduce this to **0.8** or **0.7** to prevent sudden jerky stops that ruin camera footage. For FPV racing or aggressive autonomous missions, increase to **1.2 - 1.5** for tight cornering accuracy.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





PSC_POSZ_P (Position Control (Vertical) P Gain)

DEFAULT: 1.0	RANGE: 1.0 3.0

Position Control (Vertical) P Gain (PSC_POSZ_P)

Description

The `PSC_POSZ_P` parameter controls the "springiness" of the drone's altitude hold. It defines how aggressively the drone attempts to correct its height when it notices it is above or below its target altitude.

- **Low Value:** The drone will feel "lazy" vertically. If you push it down, it will drift back up very slowly. It may sag when flying forward at speed.
- **High Value:** The drone fights hard to maintain its altitude. It reacts instantly to small changes in height.
- **Too High:** The drone will start to bounce or oscillate vertically (up and down) as it over-corrects small pressure sensor errors.

The Mathematics

This parameter is the Proportional (k_P) term in the outer Altitude loop.

$$V_z_target = k_P \cdot (Z_target - Z_current)$$

Where:

- V_z_target is the requested climb/descent rate (in cm/s) passed to the inner Velocity Loop.
- Z_target is the desired altitude (Barometer or Rangefinder).
- $Z_current$ is the EKF estimated altitude.
- k_P is `PSC_POSZ_P`.
- **Dimensionality:** Since it converts Altitude (L) to Velocity (LT^{-1}), the unit of k_P is $1/s$ (Hertz).

Example: If the drone is 1 meter (100cm) below its target and `PSC_POSZ_P` is 1.0, it will command a 100 cm/s climb rate. If you increase it to 2.0, it will command 200 cm/s for the same error.

The Engineer's View

This parameter is an instance of `AC_P_1D` (member `_p_pos_z`) within `AC_PosControl`.

It is executed in `AC_PosControl::update_z_controller()`:

```
// AC_PosControl.cpp
_vel_target.z = _p_pos_z.update_all(pos_target_zf, _inav.get_position_z_up_cm());
```

The climb rate output is then capped by `PILOT_SPEED_UP` and `PILOT_SPEED_DN` before being sent to the Velocity PID loop.

Tuning & Behavior

- **Default Value:** 1.0 (Standard)
- **Range:** 1.0 - 3.0
- **Effect of Increasing:** Faster return to target altitude. "Locked-in" feel.
- **Effect of Decreasing:** Smoother altitude transitions. More forgiving of sensor noise (Baro drift).

Use Case Recommendations

- **Cinematography: Keep Default (1.0).** Standard 1.0 provides a natural, smooth altitude hold that isn't robotic or jerky.
- **Terrain Following (Low Altitude): Increase to 1.5 - 2.0.** When flying close to the ground using a rangefinder, you want the drone to follow the contours of the earth precisely. A higher P-gain ensures it doesn't "crash" into a hill because it was too slow to climb.
- **Heavy Lift / Tethered: Decrease to 0.8.** Reduces the chance of vertical oscillations if the drone is physically constrained by a cable or heavy payload.

Troubleshooting

- **Scenario:** Drone "bounces" up and down slowly in Loiter.
  - **Diagnosis:** `PSC_POSZ_P` is too high, creating a resonant oscillation with the landing throttle.
  - **Fix:** Reduce `PSC_POSZ_P` in steps of 0.2.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





PSC_VELXY_D (Position Control Velocity (XY) D Gain)

DEFAULT: 0.5	RANGE: 0.0 1.0

Position Control Velocity (XY) D Gain (PSC_VELXY_D)

Description

The `PSC_VELXY_D` parameter provides **Damping** to the velocity controller. It reacts to the *rate of change* of the error. If the drone is rapidly approaching the target velocity, the D-term counter-acts the P-term to slow down the reaction, preventing it from shooting past the target.

Think of it as the shock absorbers on a car. Without them (`D=0`), the car bounces (oscillates) after a bump. With them, it settles quickly.

- **Too Low:** The drone may overshoot the target velocity and "wobble" before settling.
- **Too High:** The drone amplifies sensor noise (vibrations), causing rapid, high-frequency twitching in the motors and potentially overheating them.

The Mathematics

This parameter scales the filtered derivative of the error.

$$ \vec{A}_{target} += k_D \cdot \text{Filter}\left(\frac{d(\vec{V}_{error})}{dt} \right) $$

Where:

- $\frac{d(\vec{V}_{error})}{dt}$ is the rate of change of the velocity error.
- $\text{Filter}(\dots)$ is a low-pass filter (set by `PSC_VELXY_FLTD`) to remove high-frequency noise.
- k_D is `PSC_VELXY_D`.

Significance: The D-term opposes rapid changes. If the error is decreasing quickly (you are fixing the problem), the derivative is negative, so the D-term subtracts from the output, "braking" the correction to prevent overshoot.

The Engineer's View

This parameter is the `_kd` member of the `_pid_vel_xy` object in `AC_PosControl`.

It is executed in `AC_PID_2D::update_all()`:

```
// AC_PID_2D.cpp
if (is_positive(dt)) {
    const Vector2f derivative{(_error - error_last) / dt};
    _derivative += (derivative - _derivative) * get_filt_D_alpha(dt);
}
// ...
return ... + _derivative * _kd ...;
```

The derivative calculation relies on a clean `dt` (time step) and is heavily dependent on the input filter frequency (`PSC_VELXY_FLTD`) to be usable on a real flying vehicle.

Tuning & Behavior

- **Default Value:** 0.5 (Standard Copter)
- **Range:** 0.0 - 1.0
- **Effect of Increasing:** Reduces overshoot, makes stops look "crisper" and more robotic. Increases susceptibility to vibration noise.
- **Effect of Decreasing:** Softer stops, potential for "bounce-back" or low-frequency wobble.

Use Case Recommendations

- **Large Propellers (>15 inch): Decrease to 0.1 - 0.3.**
  - *Why:* Large props have high inertia and slow response times. A high D-term tries to force rapid changes they can't physically achieve, leading to motor overheating.
- **Small Racers (<5 inch): Increase to 0.6 - 0.8.**
  - *Why:* Low inertia allows for rapid response; high D-term locks the velocity in "on rails".
- **General: Keep Default (0.5).**

Practical Tuning Tip

Tune `P` first until you see oscillation, then increase `D` to dampen it. If you hear a high-pitched grinding sound from the motors (noise amplification), reduce `D` immediately.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





PSC_VELXY_FF (Position Control Velocity (XY) Feed-Forward)

DEFAULT: 0.0	RANGE: 0 6.0

Position Control Velocity (XY) Feed-Forward (PSC_VELXY_FF)

Description

`PSC_VELXY_FF` is the "predictive throttle" for horizontal movement. While the P, I, and D terms react to **mistakes** (e.g., "I'm not flying as fast as I should be"), the Feed-Forward (FF) term reacts to the **goal** (e.g., "The pilot wants to fly at 5 m/s, so I need to lean by 10 degrees now").

By bypassing the error-correction loop, the FF term eliminates the "lag" associated with PID controllers. It allows the drone to start leaning the moment the stick is moved, rather than waiting for a velocity error to build up. This is the difference between a drone that "oozes" into movement and one that "snaps" into movement.

- **Low Value (0.0):** Standard behavior. The drone relies on error-correction. There will be a slight delay between stick movement and the drone reaching target speed.
- **High Value:** The drone leans instantly and aggressively.
- **Too High:** The drone will "over-lean" when you first move the stick, causing it to shoot past the target speed and then jerk back.

The Mathematics

The Feed-Forward term is a direct multiplier of the desired velocity:

$$ \vec{A}{*target*_FF} = |*vec{V}*{desired} \cdot k_{FF} $$

Where:

- $\vec{V}_desired$ is the speed requested by the pilot or the Position loop.
- k_FF is `PSC_VELXY_FF`.

The output is an acceleration ($\mathrm{cm/s^2}$), which is directly converted into a target Roll and Pitch angle.

The Engineer's View

This parameter maps to `_kff` in the `_pid_vel_xy` object.

It is applied in `AC_PID_2D::update_all()`:

```
// AC_PID_2D.cpp
_pid_info_x.FF = _target.x * _kff;
_pid_info_y.FF = _target.y * _kff;
return ... + _target * _kff;
```

In modern ArduPilot versions, this is often set to **0.0** by default because the kinematic shaper and high P-gains handle most use cases. However, for specialized flight profiles like high-speed object tracking, it is an essential tuning tool.

Tuning & Behavior

- **Default Value:** 0.0 (Standard)
- **Range:** 0.0 - 6.0
- **Effect of Increasing:** Snappier starts and stops. Better "lead" in autonomous missions.
- **Effect of Decreasing:** Smoother, more organic-feeling movement.

Use Case Recommendations

- **Racing / FPV Chase: Increase to 0.5 - 1.0.** Allows the drone to mimic the pilot's inputs with zero perceived lag, which is critical when chasing a moving subject.
- **Standard GPS Photography: Keep at 0.0.** You want the drone to be smooth and predictable; any "jumpiness" from high FF gains will ruin shots.
- **Precision Docking: Increase to 0.2.** Helps the drone make tiny, instantaneous adjustments when closing the final few centimeters to a target.

Troubleshooting

- **Scenario:** When you push the stick forward, the drone "nods" its nose down too far, then lifts it back up slightly as it reaches speed.
  - **Diagnosis:** `PSC_VELXY_FF` is too high (Initial overshoot).
  - **Fix:** Reduce `PSC_VELXY_FF` or set to 0.0.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





PSC_VELXY_FLTD (Position Control Velocity (XY) D-Term Filter)

DEFAULT: 5.0	RANGE: 0 100	UNITS: Hz

Position Control Velocity (XY) D-Term Filter (PSC_VELXY_FLTD)

Description

`PSC_VELXY_FLTD` is the "heat shield" for your motors. In a PID loop, the **Derivative (D) term** is extremely sensitive to noise. Because the D-term looks at the *rate of change* of error, small, high-frequency vibrations in the IMU (noise) look like massive velocity changes to the math, leading to "noisy" motor outputs.

This filter specifically targets the D-term calculation, stripping away the high-frequency vibration before it gets amplified by `PSC_VELXY_D`.

- **Low Frequency (e.g. 2Hz):** Very smooth, but makes the D-term laggy. This may lead to overshoot because the "damping" effect happens too late.
- **High Frequency (e.g. 20Hz):** Instant damping, but carries a high risk of "motor chirping" or overheating as vibration noise passes through.
- **Disabled (0):** The D-term will likely be unusable on any real aircraft due to vibration.

The Mathematics

The filter is a single-pole low-pass filter applied specifically to the derivative calculation:

$$ \vec{D}*{filtered} = \vec{D}{prev} + \alpha \cdot \left(\frac{\vec{V}*{err} - \vec{V}{err_prev}}{dt} - \vec{D}_{prev} \right) $$

Where the filter coefficient α is derived from `PSC_VELXY_FLTD` (f_c):

$$\alpha = \frac{dt}{dt + \frac{1}{2\pi f_c}}$$

The Engineer's View

This parameter maps to `_filt_D_hz` in the `_pid_vel_xy` object.

It is applied in `AC_PID_2D::update_all()`:

```
// AC_PID_2D.cpp
if (is_positive(dt)) {
    const Vector2f derivative{(_error - error_last) / dt};
    _derivative += (derivative - _derivative) * get_filt_D_alpha(dt);
}
```

The output of this filtered derivative is then multiplied by the D-gain: `_derivative * _kd`.

Tuning & Behavior

- **Default Value:** 5.0 Hz
- **Range:** 0 - 100 Hz
- **Effect of Increasing:** crisper "stops" and tighter damping, at the cost of motor heat/vibration.
- **Effect of Decreasing:** smoother vertical/horizontal behavior, but may introduce vertical bobbing or "bounce" when stopping.

Use Case Recommendations

- **Clean, Low-Vibration Frames: Increase to 10.0 Hz.** If you have high-quality motors and excellent balancing, a higher D-filter allows for a more effective D-gain, leading to a "locked-in" feel.
- **Large, Noisy Vehicles: Keep at 5.0 Hz or decrease to 3.0 Hz.** Large props generate significant low-frequency noise that can easily saturate the D-term.
- **Racing Drones: Increase to 15.0 - 20.0 Hz.** Speed is everything; racer pilots accept some motor noise for the sake of instant damping.

Troubleshooting

- **Scenario:** Motors are coming down hot after a flight in Loiter, but the tune feels good.
  - **Diagnosis:** High-frequency vibration is leaking through the D-term.
  - **Fix:** Decrease `PSC_VELXY_FLTD` to 4.0 or 3.0.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY

PSC_VELXY_FLTE (Position Control Velocity (XY) Error Filter)

DEFAULT: 5.0	RANGE: 0 100	UNITS: Hz

Position Control Velocity (XY) Error Filter (PSC_VELXY_FLTE)

Description

`PSC_VELXY_FLTE` is a "noise cleaner" for the velocity controller. It applies a low-pass filter to the velocity error (the difference between where you want to fly and where the EKF thinks you are flying).

GPS and EKF velocity estimates are never perfectly smooth; they contain high-frequency noise and "jitter." Without this filter, the PID controller would try to react to every tiny jitter, causing the motors to vibrate and the drone to feel "nervous."

- **Higher Frequency (e.g., 20Hz):** The controller reacts faster to real velocity changes, but is more likely to pass noise to the motors.
- **Lower Frequency (e.g., 2Hz):** The drone feels much smoother and "heavy," but there is more lag in the control response.
- **Too Low:** Introduces so much "phase lag" that the drone may become unstable and start to orbit its target (toilet bowling).

The Mathematics

This parameter defines the Cutoff Frequency (f_c) of a single-pole low-pass filter. The filter coefficient (α) is calculated as:

$$\alpha = \frac{dt}{dt + RC} \quad \text{where } RC = \frac{1}{2\pi f_c}$$

The filtered error is then updated every time step:

$$ \vec{V}{*err_filtered*} = \vec{V}{err_prev} + \alpha \cdot (\vec{V}{*err_raw*} - \vec{V}{err_prev}) $$

Where:

- f_c is `PSC_VELXY_FLTE`.
- $\vec{V}{*err_raw*} is the unfiltered error (V{target} - V_{current}$).

The Engineer's View

This parameter maps to `_filt_E_hz` in the `_pid_vel_xy` object.

It is applied in `AC_PID_2D::update_all()`:

```
// AC_PID_2D.cpp
_error += ((_target - measurement) - _error) * get_filt_E_alpha(dt);
```

By filtering the error *before* the P, I, and D calculations, ArduPilot ensures that the entire PID loop operates on a consistent, smooth signal.

Tuning & Behavior

- **Default Value:** 5.0 Hz
- **Range:** 0 - 100 Hz
- **Effect of Increasing:** Sharper control, better wind rejection, but increases motor noise and heat.
- **Effect of Decreasing:** Smoother flight, quieter motors, but increases control lag and "drifting" tendency.

Use Case Recommendations

- **Standard GPS Drones: Keep Default (5.0 Hz).** This is the sweet spot for typical GPS-based navigation.
- **Optical Flow / Indoor: Increase to 10.0 Hz.** Optical flow often provides a cleaner, higher-rate velocity estimate than GPS, allowing for a higher filter cutoff and tighter control.
- **Large, Flexible Frames: Decrease to 3.0 Hz.** If the drone's arms or frame are flexible, they can vibrate at 10-15Hz. Lowering the filter frequency prevents the position controller from exciting these mechanical resonances.

Troubleshooting

- **Scenario:** Drone oscillates slowly (1-2 Hz) in Loiter, drawing circles in the air.
  - **Diagnosis:** `PSC_VELXY_FLTE` is likely too low (Phase Lag instability).
  - **Fix:** Increase `PSC_VELXY_FLTE` to 7.0 or 10.0.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





PSC_VELXY_I (Position Control Velocity (XY) I Gain)

DEFAULT: 1.0	RANGE: 0.02 1.00

Position Control Velocity (XY) I Gain (PSC_VELXY_I)

Description

The `PSC_VELXY_I` parameter is the **Integral** term of the velocity controller. Its primary job is to "learn" and compensate for persistent external forces—most commonly **Wind**.

If `PSC_VELXY_P` is the "muscle" that reacts to immediate error, `PSC_VELXY_I` is the "memory" that notices if the P-term isn't doing enough. If the drone is commanding a 0 m/s velocity (hover) but the wind keeps pushing it backwards, the I-term builds up over time, leaning the drone further and further into the wind until the drift stops.

- **Too Low:** The drone will drift significantly downwind before stopping. In fast forward flight, it may lag behind the desired speed.
- **Too High:** The drone will "overshoot" when the wind stops or when stopping after a fast flight, taking a long time to level out (known as "I-term windup").

The Mathematics

This parameter scales the accumulated error over time.

$$ \vec{A}_{target} += \int_0^t (\vec{V}_{target} - \vec{V}_{current}) \cdot k_I \cdot dt $$

Where:

- $\vec{A}_{target}$ is the requested acceleration (lean angle).
- The integral $\int$ accumulates the velocity error every time step.
- k_I is `PSC_VELXY_I`.

Anti-Windup: The code explicitly limits this accumulation via `PSC_VELXY_IMAX` to prevent the drone from building up a dangerous lean angle if it is physically stuck or held.

The Engineer's View

This parameter is the `_ki` member of the `_pid_vel_xy` object in `AC_PosControl`.

It is executed in `AC_PID_2D::update_i()`:

```
// AC_PID_2D.cpp
Vector2f delta_integrator = (_error * _ki) * dt;
_integrator += delta_integrator;
```

Crucially, the integrator has **Leakiness** or **Reset** logic handled in `AC_PosControl::soften_for_landing_xy()`, which prevents the drone from tipping over if it is on the ground but thinks it is drifting.

Tuning & Behavior

- **Default Value:** 1.0 (Standard Copter)
- **Range:** 0.02 - 1.00
- **Effect of Increasing:** stronger wind resistance, better holding of exact coordinates over long periods.
- **Effect of Decreasing:** Less "slop" or overshoot when stopping, but potential for "toilet bowling" (orbiting the target) if wind is strong and P-gain is low.

Use Case Recommendations

- **Windy Environments / Maritime: Increase to 1.5 - 2.0.**
  - *Why:* The drone needs to "lean into" the wind aggressively to hold position above a boat or shoreline.
- **Indoor / GPS-Denied: Decrease to 0.5.**
  - *Why:* Without wind, a high I-term can accumulate noise from Optical Flow sensors, causing a slow, wobbling drift.
- **General: Keep Default (1.0).**

Troubleshooting

- **Scenario:** Drone flies to a waypoint, stops, but then slowly overshoots and has to back up.
  - **Diagnosis:** `PSC_VELXY_I` is too high (I-term windup). The drone built up a "lean forward" memory during the flight that it didn't dump fast enough when stopping.
  - **Fix:** Reduce `PSC_VELXY_I` or increase `PSC_VELXY_D` to dampen the stop.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





PSC_VELXY_IMAX (Position Control Velocity (XY) I-Term Maximum)

DEFAULT: 1000	RANGE: 0 4500	UNITS: cm/s/s

Position Control Velocity (XY) I-Term Maximum (PSC_VELXY_IMAX)

Description

`PSC_VELXY_IMAX` defines the maximum "tilt authority" given to the velocity integrator (`PSC_VELXY_I`).

The integrator's primary job is to fight constant wind. If a 20-knot wind is pushing your drone, the integrator builds up a permanent lean into the wind to hold position. This parameter prevents that lean from becoming too extreme. It ensures that even if the drone is struggling against a hurricane, the software won't command a tilt so steep that the drone loses lift and falls out of the sky.

- **Low Value:** Limits the drone's ability to hold position in strong winds.
- **High Value:** Allows the drone to lean very steeply to fight wind, but increases the risk of "windup" (where the drone stays tilted for several seconds after the wind stops).
- **Default (1000):** Equivalent to 1000 cm/s² (about 10% of gravity), or roughly 6 degrees of "I-term lean."

The Mathematics

The velocity integrator ($\vec{I}$) is clamped by this value in the XY plane:

$$ \text{Length}(\vec{I}*{new}) = |\text{constrain}(|\text{Length}(|\vec{I}*{prev} + \vec{V}_{err} \cdot k_I \cdot dt), 0, \text{IMAX}) $$

Where:

- IMAX is `PSC_VELXY_IMAX`.
- The output is a 2D vector (North/East), so the *magnitude* of the vector is capped.

The Engineer's View

This parameter maps to `_kimax` in the `_pid_vel_xy` object.

It is applied in `AC_PID_2D::update_i()`:

```
// AC_PID_2D.cpp
_integrator += delta_integrator;
_integrator.limit_length(_kimax);
```

Note that `AC_PID_2D` uses a circular (vector) limit, which is mathematically superior to capping North and East independently, as it prevents the drone from having more authority on the diagonals.

Tuning & Behavior

- **Default Value:** 1000 (10.0 m/s² authority)
- **Range:** 0 - 4500 (0 to 45 m/s² authority)
- **Effect of Increasing:** Better position holding in extreme wind.
- **Effect of Decreasing:** Safer flight in gusty conditions; reduces the chance of the drone "darting" after a windup.

Use Case Recommendations

- **Extreme High-Wind Ops (Offshore): Increase to 2000.** This allows for about 12 degrees of permanent tilt to fight sustained ocean winds.
- **Precision Indoor Mapping: Decrease to 500.** Indoors there is no wind; a lower IMAX prevents the integrator from learning "fake" offsets from sensor drift.
- **Racing / Acro: Keep Default.** Standard values are usually sufficient as racers rely more on P-gain than I-term authority.

Troubleshooting

- **Scenario:** Drone is holding position against wind, but when the wind stops, the drone "shoots" forward for a few meters before leveling out.
  - **Diagnosis:** Integrator Windup. `PSC_VELXY_IMAX` is too high, allowing too much "memory" to build up.
  - **Fix:** Reduce `PSC_VELXY_IMAX` to 800 or 500.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





PSC_VELXY_P (Position Control Velocity (XY) P Gain)

DEFAULT: 2.0	RANGE: 0.1 6.0

Position Control Velocity (XY) P Gain (PSC_VELXY_P)

Description

The `PSC_VELXY_P` parameter controls how aggressively the drone adjusts its acceleration to match the desired velocity. It is the "muscle" of the position controller. While `PSC_POSXY_P` determines the *speed* you want to fly at to fix a position error, `PSC_VELXY_P` determines the *force* (lean angle) applied to achieve that speed.

- **Low Value:** The drone feels "slushy" or "drifty." It takes a long time to reach the desired speed and may overshoot the stopping point because it doesn't brake hard enough.
- **High Value:** The drone reacts instantly to velocity errors with sharp leans.
- **Too High:** The drone will oscillate rapidly (jitter) as it over-corrects velocity noise, especially in wind.

The Mathematics

This parameter is the Proportional (k_P) term of the velocity PID loop.

$$ \vec{A}*{target} = k_P \cdot (\vec{V}*{target} - \vec{V}_{current}) + \dots $$

Where:

- $\vec{A}_target$ is the requested acceleration (cm/s²).
- $\vec{V}_target$ is the desired velocity (cm/s).
- k_P is `PSC_VELXY_P`.
- **Dimensionality:** Since it converts Velocity (LT^{-1}) to Acceleration (LT^{-2}), the unit of k_P is technically $1/s$ (Hertz), though it is treated as a unitless scalar in the GCS.

The Engineer's View

This parameter is part of the `AC_PID_2D` object `_pid_vel_xy` inside `AC_PosControl`.

It is executed in `AC_PosControl::update_xy_controller()`:

```
// AC_PosControl.cpp
Vector2f accel_target = _pid_vel_xy.update_all(_vel_target.xy(), comb_vel, _dt, _limit_vector.xy());
```

The output `accel_target` is then converted to lean angles (`_roll_target`, `_pitch_target`) which are sent to the Attitude Controller.

Tuning & Behavior

- **Default Value:** 2.0 (Standard Copter)
- **Range:** 0.1 - 6.0
- **Effect of Increasing:** Sharper tracking of velocity requests. Better wind rejection.
- **Effect of Decreasing:** Smoother flight, less "twitchiness," but reduced position holding accuracy.
- **Dependencies:** Must be tuned in conjunction with `PSC_VELXY_D` (Damping). High P usually requires some D to prevent overshoot.

Use Case Recommendations

- **Cinematography / Heavy Lift: Decrease to 1.0 - 1.5.**
  - *Why:* Smoothes out the "twitch" caused by GPS noise or wind gusts. Prevents the drone from jerking the camera gimbal when it corrects small position errors.
- **FPV Racing / Aggressive Flight: Increase to 2.5 - 3.0.**
  - *Why:* Ensures the drone "locks" onto velocity commands instantly. Essential for sharp cornering where any lag results in a wide turn radius.
- **Long-Range Mapping: Keep Default (2.0).**
  - *Why:* Standard tuning provides the best compromise between track adherence (straight lines) and efficiency.

Troubleshooting

- **Scenario:** Drone "twitches" rapidly while hovering in wind.
  - **Diagnosis:** `PSC_VELXY_P` is too high. The controller is reacting to every tiny gust with a hard lean.
  - **Fix:** Reduce `PSC_VELXY_P` (try 1.5).
- **Scenario:** Drone drifts away from target in wind and takes too long to correct.
  - **Diagnosis:** `PSC_VELXY_P` (or `I`) is too low.
  - **Fix:** Increase `PSC_VELXY_P`.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





PSC_VELZ_D (Position Control Velocity (Vertical) D Gain)

DEFAULT: 0.0	RANGE: 0.0 1.0

Position Control Velocity (Vertical) D Gain (PSC_VELZ_D)

Description

`PSC_VELZ_D` acts as a vertical damper for the drone's climb and descent rates. It works in the middle loop of the altitude controller to smooth out the transition between climbing and hovering.

By looking at how fast the vertical velocity error is changing, the D-term can "predict" when the drone is about to reach its target speed and begin braking early. This prevents the drone from "launching" past its target altitude and then having to correct back down.

- **Note:** Just like the vertical I-term, this is often set to **0.0** in standard configurations. The natural damping of the propellers and the high-frequency response of the innermost acceleration loop are usually sufficient for vertical stability.
- **Too Low:** The drone may overshoot its climb rate setpoint, leading to slightly jerky vertical movement.
- **Too High:** Can introduce high-frequency motor noise and vibration, as the controller over-reacts to tiny vertical speed fluctuations (e.g., from Baro noise).

The Mathematics

This parameter scales the derivative of the vertical velocity error:

$$ \vec{A}{*z_target} += k_D \cdot \text{Filter}\left(\frac{d(V{z_target} - V_{z_current})}{dt} \right) $$

Where:

- k_D is `PSC_VELZ_D`.
- $extFilter(...)$ is a low-pass filter (set by `PSC_VELZ_FLTD`) to clean up the derivative signal.

The Engineer's View

This parameter is the `_kd` member of the `_pid_vel_z` object inside `AC_PosControl`.

It is executed in `AC_PID_Basic::update_all()`:

```
// AC_PID_Basic.cpp
if (is_positive(dt)) {
    float derivative = (error - _last_error) / dt;
    _derivative = _last_derivative + (dt / (_filt_D_hz + dt)) * (derivative - _last_derivative);
}
return ... + _derivative * _kd;
```

Vertical damping is typically handled by the innermost `PSC_ACCZ_D` gain first, with `PSC_VELZ_D` acting as a secondary, slower damping layer.

Tuning & Behavior

- **Default Value:** 0.0 (Standard)
- **Range:** 0.0 - 1.0
- **Effect of Increasing:** Smoother vertical starts and stops. Less "vertical bobbing" during aggressive climbs.
- **Effect of Decreasing:** Maximum vertical responsiveness, but potential for "bounce" when stopping a climb.

Use Case Recommendations

- **Standard Build: Keep at 0.0.** Vertical aerodynamics usually provide enough damping.
- **Large, Low-RPM Propellers: Consider 0.05 - 0.1.** Large props have high vertical "surge" and slow response; a small amount of velocity damping can help level them out.
- **Precision Lifting / Construction: Increase to 0.1.** Helps the drone move vertically in a very deliberate, smooth manner.

Troubleshooting

- **Scenario:** Drone oscillates vertically at a moderate speed (once every 0.5 seconds) specifically when you let go of the throttle stick.
  - **Diagnosis:** Inadequate vertical damping.
  - **Fix:** Increment `PSC_VELZ_D` by 0.02 and see if the settle time improves.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





PSC_VELZ_FF (Position Control Velocity (Vertical) Feed-Forward)

DEFAULT: 0.0	RANGE: 0 1.0

Position Control Velocity (Vertical) Feed-Forward (PSC_VELZ_FF)

Description

`PSC_VELZ_FF` provides "pre-emptive" vertical force. While the P and I terms in the vertical velocity loop wait for the drone to *miss* its target speed before they act, the Feed-Forward (FF) term acts the moment you move the throttle stick.

If you command a 1 m/s climb, the FF term instantly calculates a requested acceleration to start the drone moving upward. This significantly reduces the vertical "sag" or "delay" felt in altitude-controlled modes (AltHold, Loiter).

- **Standard Build:** Often set to **0.0** because the shaper and high vertical P-gain handle most situations.
- **Performance Build:** A small amount of FF makes altitude changes feel "punchier" and more professional.

The Mathematics

The FF term is proportional to the vertical speed target:

$$ \vec{A}{z_target_FF} = |vec{V}{z_target} \cdot k_{FF} $$

Where:

- $\vec{V}_z_target$ is the commanded climb/descent rate.
- k_FF is `PSC_VELZ_FF`.

The Engineer's View

This parameter maps to `_kff` in the `_pid_vel_z` object.

It is applied in `AC_PosControl::update_z_controller()`:

```
// AC_PosControl.cpp
_vel_target.z += _vel_desired.z + _vel_offset.z + _vel_terrain;
// ...
_accel_target.z = _pid_vel_z.update_all(_vel_target.z, curr_vel_z, ...);
```

The `AC_PID_Basic` library includes the FF term in its final output calculation. It effectively "leads" the acceleration target, reducing the work the P-term has to do.

Tuning & Behavior

- **Default Value:** 0.0 (Standard)
- **Range:** 0.0 - 1.0
- **Effect of Increasing:** Snappier response to throttle stick changes. Less vertical altitude loss at the start of a climb.
- **Effect of Decreasing:** Smoother, more damped vertical movement.

Use Case Recommendations

- **Heavy Cargo VTOL: Set to 0.1.** Helps the drone overcome the huge vertical inertia of a heavy frame to start moving the moment the command is given.
- **FPV Freestyle in Loiter: Set to 0.2.** Makes vertical "pops" feel more like a manual Acro quad.
- **Standard GPS Photography: Keep at 0.0.** Smoothness is priority; raw stick-to-acceleration mapping can sometimes be too "jumpy."

Troubleshooting

- **Scenario:** When you push the throttle stick up, the drone "jumps" up suddenly and then settles into the climb rate.
  - **Diagnosis:** `PSC_VELZ_FF` is too high (initial jerk overshoot).
  - **Fix:** Reduce `PSC_VELZ_FF` or set to 0.0.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





PSC_VELZ_FLTD (Position Control Velocity (Vertical) D-Term Filter)

DEFAULT: 5.0	RANGE: 0 100	UNITS: Hz

Position Control Velocity (Vertical) D-Term Filter (PSC_VELZ_FLTD)

Description

`PSC_VELZ_FLTD` is the specific filter for the vertical **Derivative (D) gain** in the velocity loop.

Because the D-term (`PSC_VELZ_D`) reacts to the *acceleration* of vertical errors, it is extremely sensitive to sensor noise and air turbulence. Without this filter, the D-term would amplify tiny "jitters" in the barometer signal into large, rapid throttle changes. This filter strips away those high-frequency jitters, ensuring that the damping effect only applies to the real vertical movements of the drone.

- **Note:** If `PSC_VELZ_D` is 0 (the default), this parameter does nothing.
- **Safety:** If you do enable vertical velocity damping, you *must* have this filter set to a reasonable value (5-10Hz) to protect your ESCs from noise spikes.

The Mathematics

The filter is a single-pole low-pass filter applied specifically to the vertical velocity derivative calculation:

$$ \vec{D}*{z_filtered} = \vec{D}*{z_prev} + \alpha \cdot \left(\frac{d(V_{z_err})}{dt} - \vec{D}_{z_prev} \right) $$

Where the filter coefficient α is derived from `PSC_VELZ_FLTD` (f_c):

$$\alpha = \frac{dt}{dt + \frac{1}{2\pi f_c}}$$

The Engineer's View

This parameter maps to `_filt_D_hz` in the `_pid_vel_z` object.

It is applied in `AC_PID_Basic::update_all()`:

```
// AC_PID_Basic.cpp
if (is_positive(dt)) {
    float derivative = (error - _last_error) / dt;
    _derivative = _last_derivative + (dt / (_filt_D_hz + dt)) * (derivative - _last_derivative);
}
```

This filter provides a "cushion" for the vertical damping logic, preventing the drone from having an aggressive vertical "bark" when it encounters turbulence.

Tuning & Behavior

- **Default Value:** 5.0 Hz
- **Range:** 0 - 100 Hz
- **Effect of Increasing:** Allows for higher vertical D-gains with less lag, but risk of motor heat.
- **Effect of Decreasing:** Smoother vertical damping, but the D-term may become too laggy to stop altitude bobbing.

Use Case Recommendations

- **Standard Build: Keep at 5.0 Hz.** (If `PSC_VELZ_D` is used).
- **Large, Low-Frequency Propellers: Decrease to 3.0 Hz.** Large props create low-frequency "chuffing" that can easily trigger a noisy D-term.
- **High-End Autonomy: Increase to 10.0 Hz.** Provides the fast vertical damping needed for precision tasks like lidar-based power line inspection.

Troubleshooting

- **Scenario:** Motors are getting warm and making a rhythmic "pulsing" sound during vertical descents.
  - **Diagnosis:** D-term noise from the vertical velocity loop.
  - **Fix:** Decrease `PSC_VELZ_FLTD` to 4.0 or 3.0 Hz, or reduce `PSC_VELZ_D`.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





PSC_VELZ_FLTE (Position Control Velocity (Vertical) Error Filter)

DEFAULT: 5.0	RANGE: 0 100	UNITS: Hz

Position Control Velocity (Vertical) Error Filter (PSC_VELZ_FLTE)

Description

`PSC_VELZ_FLTE` smooths the drone's vertical performance by cleaning up noise in the climb/descent speed measurement.

Vertical velocity is primarily estimated using a combination of the Barometer and Accelerometers. Barometers are notoriously noisy—they are affected by wind gusts blowing into the drone's frame and even by the drone's own propeller wash. Without this filter, the vertical PID controller would try to react to every tiny "pop" in the pressure sensor, causing the motors to stutter and the drone to sound uneven.

- **Higher Frequency (e.g. 10Hz):** Faster response to real climb rate changes, but more likely to make the motors sound "jittery."
- **Lower Frequency (e.g. 2Hz):** Extremely smooth vertical flight, but makes the drone slow to react to changes in terrain.
- **Default (5Hz):** Optimal for most Baro-based altitude hold systems.

The Mathematics

This parameter defines the Cutoff Frequency (f_c) for a low-pass filter:

$$ \vec{V}{z_err_filtered} = \vec{V}{z_err_prev} + \alpha \cdot (\vec{V}{z_err_raw} - \vec{V}{z_err_prev}) $$

Where the smoothing factor α is:

$$\alpha = \frac{dt}{dt + \frac{1}{2\pi f_c}}$$

A lower `PSC_VELZ_FLTE` value creates a smaller α, meaning the drone trusts its previous state more than the new (potentially noisy) measurement.

The Engineer's View

This parameter maps to `_filt_E_hz` in the `_pid_vel_z` object.

It is applied in `AC_PID_Basic::update_all()`:

```
// AC_PID_Basic.cpp
_error += ((target - measurement) - _error) * get_filt_E_alpha(dt);
```

By filtering the vertical error here, ArduPilot ensures that the "Velocity P-Gain" (`PSC_VELZ_P`) acts on a smooth signal, preventing high-frequency throttle oscillations while maintaining a precise hover.

Tuning & Behavior

- **Default Value:** 5.0 Hz
- **Range:** 0.0 - 100.0 Hz
- **Effect of Increasing:** Snappier throttle response, drone stops its vertical movement more accurately at the cost of motor noise.
- **Effect of Decreasing:** Quieter, smoother flight; great for cinematography.

Use Case Recommendations

- **Standard Build: Keep at 5.0 Hz.** This provides excellent results for most users.
- **Rangefinder-Only Flight (Indoor): Increase to 10.0 Hz.** Rangefinders are much cleaner than barometers; you can afford a higher filter frequency to get tighter vertical control.
- **Large-Scale Survey Drones: Keep Default.** Smoothness is more important than millisecond-level vertical accuracy for mapping.

Troubleshooting

- **Scenario:** Drone hovers at the right height, but the motors sound like they are "pulsing" or "surging" rapidly.
  - **Diagnosis:** Baro noise is passing through the filter.
  - **Fix:** Decrease `PSC_VELZ_FLTE` to 3.0 Hz.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





PSC_VELZ_I (Position Control Velocity (Vertical) I Gain)

DEFAULT: 0.0	RANGE: 0.02 1.00

Position Control Velocity (Vertical) I Gain (PSC_VELZ_I)

Description

`PSC_VELZ_I` is the **Integral** term for the vertical velocity controller. It works to eliminate persistent errors in climb or descent rates.

While the innermost acceleration loop (`PSC_ACCZ_I`) handles gravity and hover throttle, the vertical velocity loop ensures that if you command a 1 m/s climb, you actually get exactly 1 m/s, even if there is an updraft or downdraft.

- **Note:** In many multirotor configurations, this is set to **0.0** by default. This is because the innermost Acceleration loop is so effective at maintaining the vertical state that an additional integrator in the velocity loop often adds unnecessary complexity and potential for oscillation.
- **Too Low:** The drone may not reach its target climb/descent speed in high-drag situations.
- **Too High:** Can cause vertical "surging" or low-frequency bouncing during altitude changes.

The Mathematics

This parameter scales the accumulation of vertical velocity error:

$$ \vec{A}{z_target} += \int_0^t (V{z_target} - V_{z_current}) \cdot k_I \cdot dt $$

Where:

- k_I is `PSC_VELZ_I`.
- The integral accumulates the error between commanded and actual vertical speed.

The Engineer's View

This parameter is the `_ki` member of the `_pid_vel_z` object (class `AC_PID_Basic`) inside `AC_PosControl`.

It is executed in `AC_PosControl::update_z_controller()`:

```
// AC_PosControl.cpp
_accel_target.z = _pid_vel_z.update_all(_vel_target.z, curr_vel_z, _dt, ...);
```

By default, the `AC_PID_Basic` vertical velocity PID is initialised with `kI = 0` in the `AC_PosControl` constructor.

Tuning & Behavior

- **Default Value:** 0.0 (Standard Multirotor)
- **Range:** 0.02 - 1.00
- **Effect of Increasing:** Better adherence to target climb rates under load or in heavy vertical wind.
- **Effect of Decreasing:** Smoother vertical transitions, less vertical "hunting."

Use Case Recommendations

- **Heavy Lift / Tethered: Consider setting to 0.1.** Helps the drone overcome the constant downward pull of a heavy cable to maintain a precise climb rate.
- **Standard GPS Drones: Keep at 0.0.** The default cascaded architecture is designed to work without this integrator.
- **High Performance / Racing: Keep at 0.0.** Vertical speed is handled by the high-frequency P-gain (`PSC_VELZ_P`) and the Acceleration loop.

Troubleshooting

- **Scenario:** Drone oscillates up and down slowly (1 second per bounce) only when climbing at high speeds.
  - **Diagnosis:** `PSC_VELZ_I` is too high, causing vertical phase lag.
  - **Fix:** Reduce `PSC_VELZ_I` to 0.0 or 0.05.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





PSC_VELZ_IMAX (Position Control Velocity (Vertical) I-Term Maximum)

DEFAULT: 1000	RANGE: 1.0 8.0

Position Control Velocity (Vertical) I-Term Maximum (PSC_VELZ_IMAX)

Description

`PSC_VELZ_IMAX` is a safety clamp for the vertical velocity loop. It limits the maximum "correction authority" that the integrator (`PSC_VELZ_I`) can accumulate over time.

While the innermost loop has its own limit (`PSC_ACCZ_IMAX`) for hover throttle, this middle loop limit ensures that the software doesn't "try too hard" to maintain a specific climb rate if the vehicle is physically unable to do so (e.g., if it is carrying an overweight payload or is stuck). It prevents the drone from building up a massive "power surge" that would cause it to rocket upwards if the physical resistance suddenly disappeared.

- **Standard Build:** Often set to **1000** (full authority), but since the vertical velocity integrator is usually disabled (`I=0`), this parameter often has no practical effect.
- **Safety:** Always keep this at a reasonable value to prevent runaway vertical commands.

The Mathematics

The vertical velocity integrator is constrained to this maximum value:

$$ \vec{I}*{new} = |text{constrain}(|vec{I}*{prev} + \text{error} \cdot k_I \cdot dt, -\text{IMAX}, \text{IMAX}) $$

Where:

- $IMAX$ is `PSC_VELZ_IMAX`.
- The output is typically scaled to match the units of vertical acceleration.

The Engineer's View

This parameter maps to `_kimax` in the `_pid_vel_z` object.

It is applied in `AC_PID_Basic::update_all()`:

```
// AC_PID_Basic.cpp
_integrator = constrain_float(_integrator + (error * _ki * dt), -_kimax, _kimax);
```

Just like the other PID loops in ArduPilot, this integrator is also protected by "Saturation Checking"—it will stop increasing if the output throttle has already reached 100%, regardless of the IMAX setting.

Tuning & Behavior

- **Default Value:** 1000 (Standard)
- **Range:** 1.0 - 8.0 (Internal units vary)
- **Effect of Increasing:** Allows the drone to maintain more consistent climb rates under extreme loads.
- **Effect of Decreasing:** Safer; limits the maximum "punch" the velocity controller can add over long periods.

Use Case Recommendations

- **Tethered Operations: Decrease to 300.** Limits the tension the drone can put on the cable if the vertical path planner fails.
- **Standard Flight: Keep at Default (1000).** This is a safe ceiling for normal operation.

Troubleshooting

- **Scenario:** Drone holds climb rate perfectly for the first 10 seconds of a heavy lift, then starts to slowly lose speed.
  - **Diagnosis:** `PSC_VELZ_IMAX` is likely being hit. The drone needs more integrator authority to maintain the speed against the weight.
  - **Fix:** Increase `PSC_VELZ_IMAX` or increase `PSC_VELZ_P` to reduce reliance on the integrator.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY

PSC_VELZ_P (Position Control Velocity (Vertical) P Gain)

DEFAULT: 5.0	RANGE: 1.0 8.0

Position Control Velocity (Vertical) P Gain (PSC_VELZ_P)

Description

`PSC_VELZ_P` is the "throttle response" of the altitude controller. While `PSC_POSZ_P` determines *what* climb rate we want, `PSC_VELZ_P` determines how much *force* (vertical acceleration) the drone applies to achieve that rate.

- **Low Value:** The drone feels "underpowered" vertically. It takes a long time to reach its target climb rate. If you increase the throttle stick, the drone will accelerate slowly.
- **High Value:** The drone reacts instantly to vertical commands.
- **Too High:** The drone will oscillate vertically at a high frequency, sounding like a "vibration" in the motors, especially during rapid descents.

The Mathematics

This parameter is the Proportional (k_P) term in the inner Vertical Velocity loop.

$$A_z_target = k_P \cdot (V_z_target - V_z_current) + \dots$$

Where:

- A_z_target is the requested vertical acceleration (cm/s²) sent to the Accel controller.
- V_z_target is the desired climb/descent rate.
- $V_z_current$ is the current EKF vertical velocity.
- k_P is `PSC_VELZ_P`.
- **Dimensionality:** Since it converts Velocity (LT^{-1}) to Acceleration (LT^{-2}), the unit of k_P is $1/s$ (Hertz).

The Engineer's View

This parameter is the `_kp` member of the `_pid_vel_z` object (class `AC_PID_Basic`) inside `AC_PosControl`.

It is executed in `AC_PosControl::update_z_controller()`:

```
// AC_PosControl.cpp
_accel_target.z = _pid_vel_z.update_all(_vel_target.z, curr_vel_z, _dt, _motors.limit.throttle_lower, _motors.limit.throttle_upper);
```

The output `_accel_target.z` is then combined with the feed-forward and gravity compensation to determine the final throttle output.

Tuning & Behavior

- **Default Value:** 5.0
- **Range:** 1.0 - 8.0
- **Effect of Increasing:** Faster vertical acceleration. Drone maintains climb rate more accurately under varying loads.
- **Effect of Decreasing:** Smoother, softer vertical movement.

Use Case Recommendations

- **Standard Multirotors: Keep Default (5.0).** ArduPilot defaults are very well tuned for vertical response on 4-10 inch props.
- **Large / Heavy Vehicles: Decrease to 3.0 - 4.0.** Heavy vehicles cannot accelerate vertically as fast as small ones; a high P-gain will lead to "ringing" in the throttle.
- **Precision Docking / Precision Landing: Increase to 6.0.** Ensures the vertical velocity is perfectly matched to the descent profile to avoid hard landings.

Troubleshooting

- **Scenario:** Drone "jitters" or makes a high-pitched oscillation sound only when climbing or descending.
  - **Diagnosis:** `PSC_VELZ_P` is too high for the motor/prop combination.
  - **Fix:** Reduce `PSC_VELZ_P` by 0.5 until the noise stops.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **PTCH** parameter group (often appearing as `PTCH_` or `PTCH2SRV_`) configures the **Pitch Attitude Controller** for **ArduPlane**.

This library is responsible for maintaining the requested pitch angle (e.g., "Nose up 5 degrees") and rotation rates of the aircraft by driving the elevator servos.

Key Concepts

1. Pitch-to-Servo (`PTCH2SRV_...`)

This is the primary PID controller for fixed-wing pitch.

- `PTCH2SRV_P/I/D`: Standard PID gains.
- `PTCH2SRV_TCONST`: The time constant (seconds) for the pitch response. Lower values make the plane more responsive but require higher gains.

2. Rate Limits (`PTCH2SRV_RMAX_UP / DN`)

Defines the maximum physical rotation rate (deg/s) of the airframe. This prevents the autopilot from requesting a maneuver so sharp that it stalls the wing or damages the structure.

3. Trim (`PTCH_TRIM_DEG`)

Allows for a software-defined pitch offset. If the plane requires a few degrees of up-elevator to fly level at cruise speed, you can enter that offset here to avoid "fighting" the PID integrator.

Parameter Breakdown

- `PTCH2SRV_P`: Proportional gain (immediate elevator response).
- `PTCH2SRV_TCONST`: Speed of response (typically 0.4s to 0.7s).
- `PTCH2SRV_RLL`: Pitch compensation in turns (automatically adds up-elevator when banking to maintain altitude).

Integration Guide

- **Tuning:** Use **AUTOTUNE** in ArduPlane to set these gains. It is much safer and more accurate than manual tuning for fixed-wing aircraft.

Developer Notes

- **Library:** `libraries/APM_Control/AP_PitchController.cpp`.

PTCH2SRV_RLL (Roll Compensation)

DEFAULT: 1.0	RANGE: 0.7 1.5

Roll Compensation (PTCH2SRV_RLL)

Description

`PTCH2SRV_RLL` is the "Turn Coordination" elevator boost.

When a plane banks, it loses vertical lift component. To stay level, you must pull back on the stick. This parameter does that automatically.

- **1.0 (Default):** Calculated compensation based on physics.
- **> 1.0:** Pulls harder in turns (gains altitude).
- **< 1.0:** Pulls less (loses altitude).

Tuning & Behavior

- **Default Value:** 1.0.
- **Recommendation:** Tune this by flying continuous circles in FBWA mode. If the nose drops, increase this value. If the nose rises, decrease it.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





PTCH2SRV_RMAX_DN (Pitch Down Max Rate)

DEFAULT: 0	RANGE: 0 100	UNITS: deg/s

Pitch Down Max Rate (PTCH2SRV_RMAX_DN)

Description

`PTCH2SRV_RMAX_DN` limits how fast the nose can push down.

This is critical for preventing negative-G structural failures or engine starvation (if your fuel tank pickup doesn't like negative Gs).

PTCH2SRV_RMAX_UP (Pitch Up Max Rate)

DEFAULT: 0	RANGE: 0 100	UNITS: deg/s

Pitch Up Max Rate (PTCH2SRV_RMAX_UP)

Description

`PTCH2SRV_RMAX_UP` limits the G-force of a pull-up maneuver.

If you set this to a reasonable value (e.g. 60 deg/s), the autopilot will never try to pull the nose up faster than that, even if you yank the stick back or the navigation controller demands a steep climb. This protects the wings from snapping.

Tuning & Behavior

- **Default Value:** 0 (Unlimited).
- **Recommendation:** Set to match your airframe's structural limits or your desired flight style.

PTCH2SRV_TCONST (Pitch Time Constant)

DEFAULT: 0.5	RANGE: 0.4 1.0	UNITS: s

Pitch Time Constant (PTCH2SRV_TCONST)

Description

`PTCH2SRV_TCONST` defines how fast the plane responds to a pitch command.

If you command a 10-degree pitch up, this parameter determines how quickly the rate controller is asked to get there.

- **Low Value (0.4):** Snappy response.
- **High Value (0.8):** Smooth, relaxed response.

Tuning & Behavior

- **Default Value:** 0.5 seconds.
- **Recommendation:** Match this to the physical capabilities of your plane. A heavy cargo plane physically cannot pitch up as fast as a fighter jet, so it needs a larger time constant.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





PTCH_RATE_D (Pitch Rate D Gain)

DEFAULT: 0.004	RANGE: 0.001 0.03

Pitch Rate D Gain (PTCH_RATE_D)

Description

`PTCH_RATE_D` stops the nose from bouncing.

When you release the elevator stick, the plane should stop pitching instantly. If it bounces up and down before settling, you need more D.

Tuning & Behavior

- **Default Value:** 0.004.
- **Recommendation:** Tune carefully. Too much D causes rapid vibration and hot servos.

PTCH_RATE_D_FF (FF FeedForward Gain)

DEFAULT: 0	RANGE: null	UNITS: null

FF FeedForward Gain

Note: This parameter configures instance D_FF. It functions identically to RLL_RATE_D_FF.

PTCH_RATE_FF (Pitch Rate Feed-Forward)

DEFAULT: 0.345	RANGE: 0 3.0

Pitch Rate Feed-Forward (PTCH_RATE_FF)

Description

`PTCH_RATE_FF` is the primary gain for pitch control.

It predicts how much elevator is needed to achieve a certain pitch rate based on the physics of the airframe.

Tuning & Behavior

- **Default Value:** 0.345.
- **Recommendation:** This is the most critical parameter to tune. Use Autotune or increase slowly until the pitch rate matches the stick input perfectly.

PTCH_RATE_FLTD (PID D term filter frequency in Hz)

DEFAULT: AC_PID_DFILT_HZ_DEFAULT	RANGE: null	UNITS: Hz

PID D term filter frequency in Hz

Note: This parameter configures instance FLTD. It functions identically to RLL_RATE_FLTD.

PTCH_RATE_FLTE (PID Error filter frequency in Hz)

DEFAULT: AC_PID_EFILT_HZ_DEFAULT	RANGE: null	UNITS: Hz

PID Error filter frequency in Hz

Note: This parameter configures instance FLTE. It functions identically to RLL_RATE_FLTE.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





PTCH_RATE_FLTT (PID Target filter frequency in Hz)

DEFAULT: AC_PID_TFILT_HZ_DEFAULT	RANGE: null	UNITS: Hz

PID Target filter frequency in Hz

Note: This parameter configures instance FLTT. It functions identically to RLL_RATE_FLTT.

PTCH_RATE_I (Pitch Rate I Gain)

DEFAULT: 0.15	RANGE: 0.01 0.6

Pitch Rate I Gain (PTCH_RATE_I)

Description

`PTCH_RATE_I` compensates for Center of Gravity (CG) errors.

If your plane is nose-heavy, the I-term will learn to hold a little bit of "Up Elevator" to keep the nose level without you having to hold the stick back.

Tuning & Behavior

- **Default Value:** 0.15.
- **Recommendation:** Usually set equal to `PTCH_RATE_FF`.

PTCH_RATE_IMAX (Pitch Rate I Max)

DEFAULT: 0.666	RANGE: 0 1

Pitch Rate I Max (PTCH_RATE_IMAX)

Description

`PTCH_RATE_IMAX` limits the authority of the I-term.

If your CG is dangerously far back or forward, the I-term will try to fix it by moving the elevator. This limit prevents it from using up *all* the elevator travel, ensuring you still have some control left for maneuvering.

Tuning & Behavior

- **Default Value:** 0.666.

PTCH_RATE_P (Pitch Rate P Gain)

DEFAULT: 0.15	RANGE: 0.08 0.35

Pitch Rate P Gain (PTCH_RATE_P)

Description

`PTCH_RATE_P` is the "Instant Reaction" gain for the elevator.

When you pull back on the stick, you command a pitch rate. This gain forces the elevator to move proportionally to the error between your command and the gyro's measurement.

Tuning & Behavior

- **Default Value:** 0.15.
- **Too Low:** Mushy, slow pitch response.
- **Too High:** Rapid "Dolphin" oscillations or flutter.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY

PTCH_RATE_PDMX (PD sum maximum)

DEFAULT: 0	RANGE: null	UNITS: null

PD sum maximum

Note: This parameter configures instance PDMX. It functions identically to RLL_RATE_PDMX.

PTCH_RATE_SMAX (Pitch Slew Rate Limit)

DEFAULT: 0	RANGE: 0 200

Pitch Slew Rate Limit (PTCH_RATE_SMAX)

Description

`PTCH_RATE_SMAX` limits how fast the elevator servo can move.

- **0:** Disabled (Unlimited speed).

Tuning & Behavior

- **Use Case:** Enable this if you have large control surfaces that might cause structural damage if moved too quickly at high speed.

PTCH_TRIM_DEG (Pitch angle offset)

DEFAULT: 0.0f	RANGE: -45 45	UNITS: deg

Pitch angle offset (PTCH_TRIM_DEG)

Description

This parameter trims the aircraft's pitch attitude definition of "Level Flight". If your plane flies level but the HUD shows it is pitched up (or down), you can adjust this parameter to align the autopilot's horizon with reality. It effectively rotates the AHRS solution.

The Mathematics

$$\theta_\text{nav} = \theta_\text{imu} - \text{PTCH_TRIM_DEG}$$

The Engineer's View

Defined in `ArduPlane/Parameters.cpp`.

- **Usage:** Alternative to performing a "Level Horizon" calibration on the bench. Useful for tuning in the field.

Tuning & Behavior

- **Default Value:** 0.0 deg
- **Adjustment:** If the plane loses altitude in "Level" flight modes (FBWA), increase this value (pitch up).

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

Key Concepts

PTCH2SRV_RLL (Pitch Compensation in Roll)

DEFAULT: 1.0	RANGE: 0 2.0

Pitch Compensation in Roll (PTCH2SRV_RLL)

Description

`PTCH2SRV_RLL` is an "Elevator-to-Aileron" mix.

When a fixed-wing plane banks (rolls) into a turn, it loses vertical lift. To maintain its altitude while turning, it must pull slightly back on the elevator. This parameter automates that process.

- **1.0 (Default):** Standard compensation.
- **Higher Value:** Adds more back-elevator during turns (useful for "draggy" or low-lift planes).
- **0:** No compensation (the plane will lose altitude in every turn unless the pilot manually pulls back).

PTCH2SRV_RMAX_DN (Pitch Max Rate Down)

DEFAULT: 0	RANGE: 0 180	UNITS: deg/s

Pitch Max Rate Down (PTCH2SRV_RMAX_DN)

Description

`PTCH2SRV_RMAX_DN` limits how fast the autopilot is allowed to push the nose down.

This is used to prevent aggressive "nose dives" that could lead to over-speed conditions or excessive negative G-loads.

PTCH2SRV_RMAX_UP (Pitch Max Rate Up)

DEFAULT: 0	RANGE: 0 180	UNITS: deg/s

Pitch Max Rate Up (PTCH2SRV_RMAX_UP)

Description

`PTCH2SRV_RMAX_UP` limits how fast the autopilot is allowed to pull the nose up.

This is an important safety limit to prevent the drone from performing a maneuver so aggressive that it stalls or exceeds the structural limits of the airframe.

- **0:** Disabled (No limit).
- **Value:** Rate in degrees per second.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY

PTCH2SRV_TCONST (Pitch Controller Time Constant)

DEFAULT: 0.5	RANGE: 0.4 1.0	UNITS: s

Pitch Controller Time Constant (PTCH2SRV_TCONST)

Description

`PTCH2SRV_TCONST` defines the "Crispness" of the pitch response for fixed-wing aircraft.

It represents the time constant of the commanded pitch. A lower value makes the plane pitch up or down more aggressively to reach its target, while a higher value makes the movement smoother and more gradual.

Tuning & Behavior

- **Default:** 0.5 seconds.
- **Recommendation:** Most models fly well with **0.5**. For high-performance racing planes, you can reduce this toward **0.4**. For large, heavy drones, you may need to increase it toward **0.7** or **1.0**.
- **Note:** If set too low, the plane may oscillate or "hunt" for the target pitch angle.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **PUP** parameter group configures the **Pullup** library for **ArduPlane**. This is an advanced safety feature designed for high-altitude aircraft or specialized drop-launch missions (where the plane is dropped from another vehicle).

The pullup logic handles the transition from a vertical freefall or steep dive into stable level flight without exceeding the structural G-load limits of the airframe.

Key Concepts

1. Freefall Recovery

When a plane is dropped, it has no initial airspeed and the control surfaces are ineffective.

- `PUP_ENABLE`: Master switch.
- `PUP_ARSPD_START`: The airspeed (m/s) the autopilot must reach before it attempts to pull the nose up.

2. G-Load Protection (`PUP_NG_LIM`)

Pulling up too sharply at high speed can snap the wings.

- `PUP_NG_LIM`: Sets the maximum vertical G-load (in Gs) allowed during the recovery maneuver. The autopilot will modulate the elevator to stay within this limit.

Parameter Breakdown

- `PUP_PITCH_START`: The pitch angle at which the system starts the pullup logic.
- `PUP_ELEV_OFS`: A constant elevator offset used to initiate the nose-up movement.

Integration Guide

- **Special Missions:** This is primarily used for atmospheric research drones or high-altitude balloon launches.
- **Simulation: Test in SITL** drop-launch scenarios before attempting a real drop mission.

Developer Notes

- **Library:** `ArduPlane/pullup.cpp`.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Enable pullup after altitude wait ()

PUP_ENABLE: Enable pullup after altitude wait

Description

Enable pullup after altitude wait.

Values

Value	Meaning
0	Disabled
1	Enabled

- **Default:** 0

Description

This parameter activates the special "Pull-Up" sequence for high-altitude glider drops (e.g., from a balloon).

- **Function:** When the mission command `NAV_ALTITUDE_WAIT` completes (meaning the balloon has burst or the release altitude is reached), the plane will execute a controlled pull-up maneuver to transition from a vertical dive to level flight without over-stressing the airframe.
- **Safety:** Prevents structural failure by limiting the G-force (`PUP_NG_LIM`) during the recovery.

Source Code

ardupilot/ArduPlane/pullup.cpp

PUP_ARSPD_START (Pullup target airspeed)

DEFAULT: 30	RANGE: 0 100	UNITS: m/s

PUP_ARSPD_START: Pullup target airspeed

Description

Target airspeed for initial airspeed wait.

Values

- **Range:** 0 to 100
- **Units:** m/s
- **Default:** 30

Description

This parameter sets the **Minimum Airspeed Trigger** for the pull-up maneuver.

- **Function:** If an airspeed sensor is available, the autopilot will wait until the vehicle reaches this speed during the freefall before attempting to pull up.
- **Purpose:** Prevents stalling the control surfaces. The control surfaces need airflow to be effective; pulling up at 0 airspeed does nothing but stall the servos.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





PUP_ELEV_OFS (Elevator deflection used before starting pullup)

DEFAULT: 0	RANGE: -1.0 1.0	UNITS: %

PUP_ELEV_OFS: Elevator deflection used before starting pullup

Description

Elevator deflection offset from -1 to 1 while waiting for airspeed to rise before starting close loop control of the pullup.

Values

- **Range:** -1.0 to 1.0
- **Units:** % (Normalized)
- **Default:** 0

Description

This parameter applies a fixed elevator offset during the "Freefall" phase.

- **Function:** While the aircraft is diving to gain speed (before the high-G pull-up begins), this offset is applied to the elevator.
- **Usage:** Typically set to 0. Can be used to trim the aircraft for a specific dive angle if it has a tendency to pitch up or down naturally.

PUP_NG_JERK_LIM (Maximum normal load factor rate of change during pullup)

DEFAULT: 5.0	RANGE: 1.0 10.0	UNITS: 1/s

PUP_NG_JERK_LIM: Maximum normal load factor rate of change during pullup

Description

The maximum rate of change of normal load factor (Jerk) allowed during the pullup maneuver.

Values

- **Range:** 1.0 to 10.0
- **Units:** 1/s
- **Default:** 5.0

Description

This parameter controls how smoothly the G-force is applied.

- **Function:** Instead of snapping instantly from 1G to 4G (which could snap the wing spar), this parameter ramps the G-load up gradually.
- **Physics:** It limits the "Jerk" (the derivative of acceleration).
- **Effect:** A lower value makes the onset of the pull-up smoother but takes longer to reach the maximum turn rate, requiring more altitude. A higher value initiates the turn faster but puts a sudden shock load on the airframe.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





PUP_NG_LIM (Maximum normal load factor during pullup)

DEFAULT: 2.0	RANGE: 1.0 4.0	UNITS: G

PUP_NG_LIM: Maximum normal load factor during pullup

Description

This is the nominal maximum value of normal load factor used during the closed loop pitch rate control of the pullup.

Values

- **Range:** 1.0 to 4.0
- **Units:** G (Gravities)
- **Default:** 2.0

Description

This parameter sets the G-force limit for the autonomous pull-up.

- **Function:** During the recovery from a vertical dive, the autopilot calculates a pitch rate that will produce this specific Normal Acceleration (G-load).
- **Safety:** It prevents the autopilot from pulling up so hard that it snaps the wings off the glider.
- **Tuning:** Set this according to your airframe's structural limits (e.g., 2.0 G is generally safe for foamies, 4.0 G requires composite reinforcement).

PUP_PITCH (Custom pitch)

DEFAULT: 0	RANGE: -180 180	UNITS: deg

Custom pitch (PUP_PITCH)

Description

This parameter defines the **Pitch** component of a custom user-defined rotation. It is part of the `CUSTOM_ORIENTATION` system, allowing you to define arbitrary sensor mounting angles that do not fit the standard 90-degree increments (e.g., a GPS mounted on a 15-degree slope).

The Mathematics

The rotation is applied as an intrinsic Euler sequence (Yaw → Pitch → Roll).

$$R_\text{total} = R_z(\text{PUP_YAW}) \cdot R_y(\text{PUP_PITCH}) \cdot R_x(\text{PUP_ROLL})$$

The Engineer's View

Defined in `libraries/AP_CustomRotations/AP_CustomRotations_params.cpp`.

Tuning & Behavior

- **Default Value:** 0 deg
- **Usage:** Set `AHRS_ORIENTATION` (or sensor specific orientation) to `100` (Custom) to use these values.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





PUP_PITCH_START (Pullup target pitch)

DEFAULT: -60	RANGE: -80 0	UNITS: deg

PUP_PITCH_START: Pullup target pitch

Description

Target pitch for initial pullup.

Values

- **Range:** -80 to 0
- **Units:** deg
- **Default:** -60

Description

This parameter defines the **Pitch Angle Trigger** for the pull-up maneuver.

- **Function:** After the altitude wait finishes (e.g., balloon burst), the autopilot assumes the vehicle is tumbling or diving vertically. It waits for the nose to stabilize at this pitch angle (e.g., -60 degrees, nose down) before initiating the high-G pull-up.
- **Purpose:** Ensures the aircraft has gained sufficient airspeed and aerodynamic authority before attempting to pull out of the dive.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **PWM** parameter group (specifically `PWM_FREQ` or `MOT_PWM_FREQ`) configures the physical output frequency of the **Pulse Width Modulation** signals.

Key Concepts

1. Output Rate

- `PWM_FREQ`: Defines how many times per second (Hz) the motor or servo position is updated.
  - **50Hz:** Standard for legacy analog servos.
  - **490Hz:** High-speed standard for most digital ESCs.
  - **16,000Hz+:** Ultra-high frequency used for brushed motors to prevent the audible "whine."

Integration Guide

- **Warning:** Do not set a high frequency (e.g., 400Hz) if you are using old analog servos; they will overheat and burn out. Digital servos can usually handle 333Hz or more.

Developer Notes

- **Library:** `libraries/AP_HAL/RCOutput`.
- **Scope:** This is often a global setting, but many boards allow setting different frequencies for different "Groups" of pins.

PWM_FREQ (Motor Output PWM freq for brushed motors)

DEFAULT: 16	RANGE: 1 20	UNITS: kHz

PWM_FREQ: Motor Output PWM freq for brushed motors

Description

Motor Output PWM freq for brushed motors.

Values

- **Range:** 1 to 20
- **Units:** kHz
- **Increment:** 1
- **Default:** 16

Description

This parameter sets the Pulse Width Modulation (PWM) frequency used to drive brushed motors on a Rover or Boat.

- **Usage:** Only applies when `MOT_PWM_TYPE` (or `PWM_TYPE` on Rover) is set to a "Brushed" mode (e.g., `3` for BrushedWithRelay or `4` for BrushedBiPolar).
- **Default (16kHz):** A high enough frequency to be outside the audible range of human hearing (silent operation).
- **Lower Values:** May be required for some high-current motor drivers or to reduce switching losses (heat), but the motors may "whine" audibly.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





QUADPLANE CONFIGURATION (Q)

Overview

The **Q** parameter group is the primary configuration set for **QuadPlanes** (VTOL Fixed-Wing aircraft). A QuadPlane is a hybrid vehicle that uses multicopter motors for vertical takeoff and landing (VTOL) and a standard forward motor for efficient winged flight.

Because QuadPlanes are essentially two vehicles in one, the `Q` group mirrors many parameters found in the `ATC_`, `PSC_`, `MOT_`, and `WPNAV_` groups, but specifically for the VTOL phases of flight.

Key Concepts

1. The VTOL "Shadow" Parameters

Almost every multicopter parameter has a `Q_` equivalent:

- `Q_A_RAT_...`: Attitude PID gains for VTOL mode (equivalent to `ATC_RAT_`).
- `Q_P_POS...`: Position control gains for VTOL (equivalent to `PSC_POS...`).
- `Q_M_...`: Motor and mixer settings for the lift motors (equivalent to `MOT_`).

2. Transitions

The most complex part of QuadPlane flight is the transition between VTOL and Fixed-Wing.

- **Forward Transition:** The VTOL motors hold altitude while the forward motor pushes the plane to flying speed. Once `ARSPD_FBW_MIN` is reached, the VTOL motors shut off.
- **Back Transition:** The plane slows down, and as it approaches stall speed, the VTOL motors spin up to take over the weight of the aircraft.
- `Q_TRANSITION_MS`: The duration of the transition phase.

3. Assisted Flight (`Q_ASSIST_...`)

A safety feature where the VTOL motors automatically spin up to "save" the plane if it stalls or loses attitude control during fixed-wing flight.

4. Tailsitters and Tilt-Rotors

The `Q` group also handles specialized VTOL mechanics:

- `Q_TAILSIT_...`: For aircraft that sit on their tails and rotate 90 degrees for forward flight.
- `Q_TILT_...`: For vehicles that tilt their motors forward (like the Osprey).

Parameter Breakdown

- `Q_ENABLE`: Master switch to enable VTOL support in ArduPlane.
- `Q_FRAME_CLASS`: Lift motor layout (Quad, Hexa, etc.).
- `Q_TRANS_FAIL`: Action to take if the transition takes too long (e.g., Q_RTL).

Integration Guide

1. **Frame Setup:** Set `Q_ENABLE = 1` and configure `Q_FRAME_CLASS` / `TYPE`.
2. **Tuning:** Use **Q_AUTOTUNE** to tune the VTOL attitude controllers.
3. **Transition Test:** Perform high-altitude transitions to verify the aircraft reaches flying speed before the lift motors disarm.

Developer Notes

- **Library:** `ArduPlane/QuadPlane.cpp`.
- **Architecture:** The QuadPlane code acts as a bridge, handing control between the `AP_Motors` multicopter library and the standard fixed-wing flight code.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY



Q_ACRO_PIT_RATE (QACRO mode pitch rate)

DEFAULT: 180	RANGE: 10 500	UNITS: deg/s

QACRO mode pitch rate

QuadPlane Note: This parameter configures the VTOL motors/behavior. It functions identically to the standard Copter parameter ACRO_RP_RATE.

Please refer to the ACRO_RP_RATE documentation for detailed tuning instructions.

Q_ACRO_RLL_RATE (QACRO mode roll rate)

DEFAULT: 360	RANGE: 10 500	UNITS: deg/s

QACRO mode roll rate

QuadPlane Note: This parameter configures the VTOL motors/behavior. It functions identically to the standard Copter parameter ACRO_RP_RATE.

Please refer to the ACRO_RP_RATE documentation for detailed tuning instructions.

Q_ACRO_YAW_RATE (QACRO mode yaw rate)

DEFAULT: 90	RANGE: 10 500	UNITS: deg/s

QACRO mode yaw rate

QuadPlane Note: This parameter configures the VTOL motors/behavior. It functions identically to the standard Copter parameter ACRO_Y_RATE.

Please refer to the ACRO_Y_RATE documentation for detailed tuning instructions.

Q_ANGLE_MAX (QuadPlane Maximum Tilt Angle)

DEFAULT: 3000	RANGE: 1000 8000	UNITS: cdeg

QuadPlane Maximum Tilt Angle (Q_ANGLE_MAX)

Description

`Q_ANGLE_MAX` sets the "Lean Limit" of your QuadPlane while it is in a hover mode.

If you push the stick all the way to the side, the drone will tilt over until it hits this limit and then stop. This prevents the drone from flipping over or losing too much vertical thrust.

The Engineer's View

Defined in `quadplane.cpp`.
This parameter is used by the `AC_AttitudeControl` library specifically for VTOL modes. It is separate from the fixed-wing `LIM_PITCH_MAX/MIN` and `LIM_ROLL_CD` parameters.

Tuning & Behavior

- **Default Value:** 3000 (30 degrees).
- **Recommendation:** Set to **30-45 degrees**.
- **High Performance:** If you need to move fast in QLOITER or buck high winds, increase this to **4500 (45 degrees)**.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Q_APPROACH_DIST (QuadPlane Approach Distance)

DEFAULT: 0	RANGE: 0 1000	UNITS: m

QuadPlane Approach Distance (Q_APPROACH_DIST)

Description

`Q_APPROACH_DIST` sets the "Braking Zone" for autonomous VTOL landings.

As the plane flies towards the landing spot at cruise speed, it needs to slow down and start the VTOL motors. This parameter tells the plane how far out it should start that process.

- **0 (Default):** The plane uses a calculated distance based on its current speed and `Q_TRANSITION_MS`.

Tuning & Behavior

- **Default Value:** 0.
- **Recommendation:** Set to **50-100m** if you want consistent, repeatable landing paths.

Q_ASSIST_ALT (QuadPlane Assist Altitude)

DEFAULT: 0	RANGE: 0 100	UNITS: m

QuadPlane Assist Altitude (Q_ASSIST_ALT)

Description

`Q_ASSIST_ALT` is a safety net for landing.

If the plane drops below this altitude (e.g., 30 meters), the VTOL motors will "wake up" and assist if the airspeed drops too low. This prevents the plane from stalling near the ground during a fixed-wing approach.

Tuning & Behavior

- **Default Value:** 0 (Disabled).
- **Recommendation:** Set to **30m to 50m** if you are flying manual approaches and want a safety backup.

Q_ASSIST_ANGLE (QuadPlane Assist Angle)

DEFAULT: 0	RANGE: 0 90	UNITS: deg

QuadPlane Assist Angle (Q_ASSIST_ANGLE)

Description

`Q_ASSIST_ANGLE` is an emergency "Uprighting" mechanism.

If the plane tilts past this angle (e.g., 60 degrees) while in a fixed-wing mode, the VTOL motors will immediately engage to level the aircraft. This is useful for recovering from a stall or a pilot mistake.

Tuning & Behavior

- **Default Value:** 0 (Disabled).
- **Recommendation:** Set to **60** for large, slow-rolling planes. Set to **0** for aerobatic planes that are intended to fly at extreme angles.
- **Safety:** This can save an airframe during a high-speed stall where the control surfaces have lost authority.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Q_ASSIST_DELAY (QuadPlane Assist Delay)

DEFAULT: 0.5	RANGE: 0 5	UNITS: s

QuadPlane Assist Delay (Q_ASSIST_DELAY)

Description

`Q_ASSIST_DELAY` filters out momentary noise.

If the airspeed drops below `Q_ASSIST_SPEED` for just a split second (e.g., due to a sensor glitch or a gust of wind), we don't want the loud VTOL motors to scream to life immediately. This parameter ensures the low speed condition is real.

Tuning & Behavior

- **Default Value:** 0.5 seconds.
- **Recommendation:** Leave at **0.5**.

Q_ASSIST_OPTIONS (QuadPlane Assist Options)

DEFAULT: 0	RANGE: 0 4294967295

QuadPlane Assist Options (Q_ASSIST_OPTIONS)

Description

`Q_ASSIST_OPTIONS` toggles advanced behaviors for the VTOL safety net.

Tuning & Behavior

- **Default Value:** 0.
- **Bit 0:** Disable assistance during fixed-wing takeoff.
- **Bit 1:** Disable assistance during fixed-wing landing.
- **Bit 2:** Enable assistance even in Manual/Stabilize modes (Standard is only for assisted modes like FBWA).

Q_ASSIST_SPEED (QuadPlane Assist Speed)

DEFAULT: 0	RANGE: 0 30	UNITS: m/s

QuadPlane Assist Speed (Q_ASSIST_SPEED)

Description

`Q_ASSIST_SPEED` is the primary stall prevention mechanism for QuadPlanes.

If the airspeed drops below this value while in a fixed-wing mode (like FBWA or AUTO), the VTOL motors will spin up to provide lift and thrust, preventing a stall.

Tuning & Behavior

- **Default Value:** 0 (Disabled).
- **Recommendation:** Set to **1-2 m/s above your stall speed**.
- **Example:** If your plane stalls at 10 m/s, set `Q_ASSIST_SPEED` to **12**.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Q_AUTOTUNE_AGGR (QuadPlane Autotune Aggressiveness)

DEFAULT: 0.1	RANGE: 0.05 0.15

QuadPlane Autotune Aggressiveness (Q_AUTOTUNE_AGGR)

Description

`Q_AUTOTUNE_AGGR` tells the autotuner how "twitchy" you want the final tune to be.

- **0.05 (Soft):** Very smooth, slow response. Good for aerial photography and large, flexible frames.
- **0.1 (Medium):** Good all-around performance.
- **0.15 (Sharp):** Very aggressive. Good for small, rigid frames. Risk of oscillation if the frame isn't stiff enough.

Tuning & Behavior

- **Default Value:** 0.1
- **Recommendation:** Start at **0.075 or 0.1**.

Q_AUTOTUNE_AXES (QuadPlane Autotune Axes)

DEFAULT: 7	RANGE: 1 7

QuadPlane Autotune Axes (Q_AUTOTUNE_AXES)

Description

`Q_AUTOTUNE_AXES` selects which parts of the drone the Autotuner will wiggle and measure.

- **Bit 0 (1):** Roll
- **Bit 1 (2):** Pitch
- **Bit 2 (4):** Yaw

Tuning & Behavior

- **Default Value:** 7 (All Axes).
- **Recommendation:** If you have a fragile airframe or limited battery, tune one axis at a time.
  - Set to **1** (Roll only), fly, land, save.
  - Set to **2** (Pitch only), fly, land, save.
  - Set to **4** (Yaw only), fly, land, save.

Q_AUTOTUNE_MIN_D (QuadPlane Autotune Minimum D)

DEFAULT: 0.001	RANGE: 0 0.005

QuadPlane Autotune Minimum D (Q_AUTOTUNE_MIN_D)

Description

`Q_AUTOTUNE_MIN_D` prevents the autotuner from creating a sloppy, undamped tune.

Sometimes, a noisy frame tricks the autotuner into thinking the D-gain (Damping) is causing vibration, so it reduces D to near zero. This results in a drone that wobbles uncontrollably. This parameter sets a "floor" that the D-gain cannot drop below.

Tuning & Behavior

- **Default Value:** 0.001
- **Recommendation:** Leave at default unless you have extremely noisy motors.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY

Q_A_ACCEL_P_MAX (Tiltrotor maximum VTOL angle)

DEFAULT: 45	RANGE: 20 80	UNITS: deg

Tiltrotor maximum VTOL angle

QuadPlane Note: This parameter configures the VTOL motors/behavior. It functions identically to the standard Copter parameter ATC_ACCEL_P_MAX.

Please refer to the ATC_ACCEL_P_MAX documentation for detailed tuning instructions.

Q_A_ACCEL_R_MAX (Tiltrotor maximum VTOL angle)

DEFAULT: 45	RANGE: 20 80	UNITS: deg

Tiltrotor maximum VTOL angle

QuadPlane Note: This parameter configures the VTOL motors/behavior. It functions identically to the standard Copter parameter ATC_ACCEL_R_MAX.

Please refer to the ATC_ACCEL_R_MAX documentation for detailed tuning instructions.

Q_A_ACCEL_Y_MAX (Tiltrotor maximum VTOL angle)

DEFAULT: 45	RANGE: 20 80	UNITS: deg

Tiltrotor maximum VTOL angle

QuadPlane Note: This parameter configures the VTOL motors/behavior. It functions identically to the standard Copter parameter ATC_ACCEL_Y_MAX.

Please refer to the ATC_ACCEL_Y_MAX documentation for detailed tuning instructions.

Q_A_ANGLE_BOOST (Angle Boost)

DEFAULT: 1	RANGE: null	UNITS: null

Angle Boost

QuadPlane Note: This parameter configures the VTOL motors/behavior. It functions identically to the standard Copter parameter ATC_ANGLE_BOOST.

Please refer to the ATC_ANGLE_BOOST documentation for detailed tuning instructions.

Q_A_ANG_LIM_TC (Angle Limit (to maintain altitude) Time Constant)

DEFAULT: AC_ATTITUDE_CONTROL_ANGLE_LIMIT_TC_DEFAULT	RANGE: 0.5 10.0	UNITS: null

Angle Limit (to maintain altitude) Time Constant

QuadPlane Note: This parameter configures the VTOL motors/behavior. It functions identically to the standard Copter parameter ATC_ANG_LIM_TC.

Please refer to the ATC_ANG_LIM_TC documentation for detailed tuning instructions.

Q_A_ANG_PIT_P (PID Proportional Gain)

DEFAULT: 0	RANGE: null	UNITS: null

PID Proportional Gain

QuadPlane Note: This parameter configures the VTOL motors/behavior. It functions identically to the standard Copter parameter ATC_ANG_PIT_P.

Please refer to the ATC_ANG_PIT_P documentation for detailed tuning instructions.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Q_A_ANG_RLL_P (PID Proportional Gain)

DEFAULT: 0	RANGE: null	UNITS: null

PID Proportional Gain

QuadPlane Note: This parameter configures the VTOL motors/behavior. It functions identically to the standard Copter parameter ATC_ANG_RLL_P.

Please refer to the ATC_ANG_RLL_P documentation for detailed tuning instructions.

Q_A_ANG_YAW_P (PID Proportional Gain)

DEFAULT: 0	RANGE: null	UNITS: null

PID Proportional Gain

QuadPlane Note: This parameter configures the VTOL motors/behavior. It functions identically to the standard Copter parameter ATC_ANG_YAW_P.

Please refer to the ATC_ANG_YAW_P documentation for detailed tuning instructions.

Q_A_INPUT_TC (Attitude control input time constant)

DEFAULT: AC_ATTITUDE_CONTROL_INPUT_TC_DEFAULT	RANGE: 0 1	UNITS: s

Attitude control input time constant

QuadPlane Note: This parameter configures the VTOL motors/behavior. It functions identically to the standard Copter parameter ATC_INPUT_TC.

Please refer to the ATC_INPUT_TC documentation for detailed tuning instructions.

Q_A_LAND_P_MULT (Landed pitch gain multiplier)

DEFAULT: 1.0	RANGE: 0.25 1.0	UNITS: null

Landed pitch gain multiplier

QuadPlane Note: This parameter configures the VTOL motors/behavior. It functions identically to the standard Copter parameter ATC_LAND_P_MULT.

Please refer to the ATC_LAND_P_MULT documentation for detailed tuning instructions.

Q_A_LAND_R_MULT (Landed roll gain multiplier)

DEFAULT: 1.0	RANGE: 0.25 1.0	UNITS: null

Landed roll gain multiplier

QuadPlane Note: This parameter configures the VTOL motors/behavior. It functions identically to the standard Copter parameter ATC_LAND_R_MULT.

Please refer to the ATC_LAND_R_MULT documentation for detailed tuning instructions.

Q_A_LAND_Y_MULT (Landed yaw gain multiplier)

DEFAULT: 1.0	RANGE: 0.25 1.0	UNITS: null

Landed yaw gain multiplier

QuadPlane Note: This parameter configures the VTOL motors/behavior. It functions identically to the standard Copter parameter ATC_LAND_Y_MULT.

Please refer to the ATC_LAND_Y_MULT documentation for detailed tuning instructions.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Q_A_RATE_FF_ENAB (Rate Feedforward Enable)

DEFAULT: AC_ATTITUDE_CONTROL_RATE_BF_FF_DEFAULT	RANGE: null	UNITS: null

Rate Feedforward Enable

QuadPlane Note: This parameter configures the VTOL motors/behavior. It functions identically to the standard Copter parameter ATC_RATE_FF_ENAB.

Please refer to the ATC_RATE_FF_ENAB documentation for detailed tuning instructions.

Q_A_RATE_P_MAX (Tiltrotor maximum VTOL angle)

DEFAULT: 45	RANGE: 20 80	UNITS: deg

Tiltrotor maximum VTOL angle

QuadPlane Note: This parameter configures the VTOL motors/behavior. It functions identically to the standard Copter parameter ATC_RATE_P_MAX.

Please refer to the ATC_RATE_P_MAX documentation for detailed tuning instructions.

Q_A_RATE_R_MAX (Tiltrotor maximum VTOL angle)

DEFAULT: 45	RANGE: 20 80	UNITS: deg

Tiltrotor maximum VTOL angle

QuadPlane Note: This parameter configures the VTOL motors/behavior. It functions identically to the standard Copter parameter ATC_RATE_R_MAX.

Please refer to the ATC_RATE_R_MAX documentation for detailed tuning instructions.

Q_A_RATE_Y_MAX (Tiltrotor maximum VTOL angle)

DEFAULT: 45	RANGE: 20 80	UNITS: deg

Tiltrotor maximum VTOL angle

QuadPlane Note: This parameter configures the VTOL motors/behavior. It functions identically to the standard Copter parameter ATC_RATE_Y_MAX.

Please refer to the ATC_RATE_Y_MAX documentation for detailed tuning instructions.

Q_A_RAT_PIT_D (PID Derivative Gain)

DEFAULT: 0	RANGE: null	UNITS: null

PID Derivative Gain

QuadPlane Note: This parameter configures the VTOL motors/behavior. It functions identically to the standard Copter parameter ATC_RAT_PIT_D.

Please refer to the ATC_RAT_PIT_D documentation for detailed tuning instructions.

Q_A_RAT_PIT_D_FF (FF FeedForward Gain)

DEFAULT: 0	RANGE: null	UNITS: null

FF FeedForward Gain

QuadPlane Note: This parameter configures the VTOL motors/behavior. It functions identically to the standard Copter parameter ATC_RAT_PIT_D_FF.

Please refer to the ATC_RAT_PIT_D_FF documentation for detailed tuning instructions.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Q_A_RAT_PIT_FF (FF FeedForward Gain)

DEFAULT: 0	RANGE: null	UNITS: null

FF FeedForward Gain

QuadPlane Note: This parameter configures the VTOL motors/behavior. It functions identically to the standard Copter parameter ATC_RAT_PIT_FF.

Please refer to the ATC_RAT_PIT_FF documentation for detailed tuning instructions.

Q_A_RAT_PIT_FLTD (PID D term filter frequency in Hz)

DEFAULT: AC_PID_DFILT_HZ_DEFAULT	RANGE: null	UNITS: Hz

PID D term filter frequency in Hz

QuadPlane Note: This parameter configures the VTOL motors/behavior. It functions identically to the standard Copter parameter ATC_RAT_PIT_FLTD.

Please refer to the ATC_RAT_PIT_FLTD documentation for detailed tuning instructions.

Q_A_RAT_PIT_FLTE (PID Error filter frequency in Hz)

DEFAULT: AC_PID_EFILT_HZ_DEFAULT	RANGE: null	UNITS: Hz

PID Error filter frequency in Hz

QuadPlane Note: This parameter configures the VTOL motors/behavior. It functions identically to the standard Copter parameter ATC_RAT_PIT_FLTE.

Please refer to the ATC_RAT_PIT_FLTE documentation for detailed tuning instructions.

Q_A_RAT_PIT_FLTT (PID Target filter frequency in Hz)

DEFAULT: AC_PID_TFILT_HZ_DEFAULT	RANGE: null	UNITS: Hz

PID Target filter frequency in Hz

QuadPlane Note: This parameter configures the VTOL motors/behavior. It functions identically to the standard Copter parameter ATC_RAT_PIT_FLTT.

Please refer to the ATC_RAT_PIT_FLTT documentation for detailed tuning instructions.

Q_A_RAT_PIT_I (PID Integral Gain)

DEFAULT: 0	RANGE: null	UNITS: null

PID Integral Gain

QuadPlane Note: This parameter configures the VTOL motors/behavior. It functions identically to the standard Copter parameter ATC_RAT_PIT_I.

Please refer to the ATC_RAT_PIT_I documentation for detailed tuning instructions.

Q_A_RAT_PIT_IMAX (PID Integral Maximum)

DEFAULT: 0	RANGE: null	UNITS: null

PID Integral Maximum

QuadPlane Note: This parameter configures the VTOL motors/behavior. It functions identically to the standard Copter parameter ATC_RAT_PIT_IMAX.

Please refer to the ATC_RAT_PIT_IMAX documentation for detailed tuning instructions.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Q_A_RAT_PIT_NEF (PID Error notch filter index)

DEFAULT: 0	RANGE: 1 8	UNITS: null

PID Error notch filter index

QuadPlane Note: This parameter configures the VTOL motors/behavior. It functions identically to the standard Copter parameter ATC_RAT_PIT_NEF.

Please refer to the ATC_RAT_PIT_NEF documentation for detailed tuning instructions.

Q_A_RAT_PIT_NTF (PID Target notch filter index)

DEFAULT: 0	RANGE: 1 8	UNITS: null

PID Target notch filter index

QuadPlane Note: This parameter configures the VTOL motors/behavior. It functions identically to the standard Copter parameter ATC_RAT_PIT_NTF.

Please refer to the ATC_RAT_PIT_NTF documentation for detailed tuning instructions.

Q_A_RAT_PIT_P (PID Proportional Gain)

DEFAULT: 0	RANGE: null	UNITS: null

PID Proportional Gain

QuadPlane Note: This parameter configures the VTOL motors/behavior. It functions identically to the standard Copter parameter ATC_RAT_PIT_P.

Please refer to the ATC_RAT_PIT_P documentation for detailed tuning instructions.

Q_A_RAT_PIT_PDMX (PD sum maximum)

DEFAULT: 0	RANGE: null	UNITS: null

PD sum maximum

QuadPlane Note: This parameter configures the VTOL motors/behavior. It functions identically to the standard Copter parameter ATC_RAT_PIT_PDMX.

Please refer to the ATC_RAT_PIT_PDMX documentation for detailed tuning instructions.

Q_A_RAT_PIT_SMAX (QuadPlane Pitch Axis Rate Slew Rate Limit)

DEFAULT: 0.0	RANGE: 0 200

QuadPlane Pitch Axis Rate Slew Rate Limit (Q_A_RAT_PIT_SMAX)

QuadPlane Note: This parameter configures the VTOL motors/behavior. It functions identically to the standard Copter parameter ATC_RAT_PIT_SMAX.

Please refer to the ATC_RAT_PIT_SMAX documentation for detailed tuning instructions.

Q_A_RAT_RLL_D (PID Derivative Gain)

DEFAULT: 0	RANGE: null	UNITS: null

PID Derivative Gain

QuadPlane Note: This parameter configures the VTOL motors/behavior. It functions identically to the standard Copter parameter ATC_RAT_RLL_D.

Please refer to the ATC_RAT_RLL_D documentation for detailed tuning instructions.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Q_A_RAT_RLL_D_FF (FF FeedForward Gain)

DEFAULT: 0	RANGE: null	UNITS: null

FF FeedForward Gain

QuadPlane Note: This parameter configures the VTOL motors/behavior. It functions identically to the standard Copter parameter ATC_RAT_RLL_D_FF.

Please refer to the ATC_RAT_RLL_D_FF documentation for detailed tuning instructions.

Q_A_RAT_RLL_FF (FF FeedForward Gain)

DEFAULT: 0	RANGE: null	UNITS: null

FF FeedForward Gain

QuadPlane Note: This parameter configures the VTOL motors/behavior. It functions identically to the standard Copter parameter ATC_RAT_RLL_FF.

Please refer to the ATC_RAT_RLL_FF documentation for detailed tuning instructions.

Q_A_RAT_RLL_FLTD (PID D term filter frequency in Hz)

DEFAULT: AC_PID_DFILT_HZ_DEFAULT	RANGE: null	UNITS: Hz

PID D term filter frequency in Hz

QuadPlane Note: This parameter configures the VTOL motors/behavior. It functions identically to the standard Copter parameter ATC_RAT_RLL_FLTD.

Please refer to the ATC_RAT_RLL_FLTD documentation for detailed tuning instructions.

Q_A_RAT_RLL_FLTE (PID Error filter frequency in Hz)

DEFAULT: AC_PID_EFILT_HZ_DEFAULT	RANGE: null	UNITS: Hz

PID Error filter frequency in Hz

QuadPlane Note: This parameter configures the VTOL motors/behavior. It functions identically to the standard Copter parameter ATC_RAT_RLL_FLTE.

Please refer to the ATC_RAT_RLL_FLTE documentation for detailed tuning instructions.

Q_A_RAT_RLL_FLTT (PID Target filter frequency in Hz)

DEFAULT: AC_PID_TFILT_HZ_DEFAULT	RANGE: null	UNITS: Hz

PID Target filter frequency in Hz

QuadPlane Note: This parameter configures the VTOL motors/behavior. It functions identically to the standard Copter parameter ATC_RAT_RLL_FLTT.

Please refer to the ATC_RAT_RLL_FLTT documentation for detailed tuning instructions.

Q_A_RAT_RLL_I (PID Integral Gain)

DEFAULT: 0	RANGE: null	UNITS: null

PID Integral Gain

QuadPlane Note: This parameter configures the VTOL motors/behavior. It functions identically to the standard Copter parameter ATC_RAT_RLL_I.

Please refer to the ATC_RAT_RLL_I documentation for detailed tuning instructions.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY

Q_A_RAT_RLL_IMAX (PID Integral Maximum)

DEFAULT: 0	RANGE: null	UNITS: null

PID Integral Maximum

QuadPlane Note: This parameter configures the VTOL motors/behavior. It functions identically to the standard Copter parameter ATC_RAT_RLL_IMAX.

Please refer to the ATC_RAT_RLL_IMAX documentation for detailed tuning instructions.

Q_A_RAT_RLL_NEF (PID Error notch filter index)

DEFAULT: 0	RANGE: 1 8	UNITS: null

PID Error notch filter index

QuadPlane Note: This parameter configures the VTOL motors/behavior. It functions identically to the standard Copter parameter ATC_RAT_RLL_NEF.

Please refer to the ATC_RAT_RLL_NEF documentation for detailed tuning instructions.

Q_A_RAT_RLL_NTF (PID Target notch filter index)

DEFAULT: 0	RANGE: 1 8	UNITS: null

PID Target notch filter index

QuadPlane Note: This parameter configures the VTOL motors/behavior. It functions identically to the standard Copter parameter ATC_RAT_RLL_NTF.

Please refer to the ATC_RAT_RLL_NTF documentation for detailed tuning instructions.

Q_A_RAT_RLL_P (PID Proportional Gain)

DEFAULT: 0	RANGE: null	UNITS: null

PID Proportional Gain

QuadPlane Note: This parameter configures the VTOL motors/behavior. It functions identically to the standard Copter parameter ATC_RAT_RLL_P.

Please refer to the ATC_RAT_RLL_P documentation for detailed tuning instructions.

Q_A_RAT_RLL_PDMX (PD sum maximum)

DEFAULT: 0	RANGE: null	UNITS: null

PD sum maximum

QuadPlane Note: This parameter configures the VTOL motors/behavior. It functions identically to the standard Copter parameter ATC_RAT_RLL_PDMX.

Please refer to the ATC_RAT_RLL_PDMX documentation for detailed tuning instructions.

Q_A_RAT_RLL_SMAX (QuadPlane Roll Axis Rate Slew Rate Limit)

DEFAULT: 0.0	RANGE: 0 200

QuadPlane Roll Axis Rate Slew Rate Limit (Q_A_RAT_RLL_SMAX)

QuadPlane Note: This parameter configures the VTOL motors/behavior. It functions identically to the standard Copter parameter ATC_RAT_RLL_SMAX.

Please refer to the ATC_RAT_RLL_SMAX documentation for detailed tuning instructions.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Q_A_RAT_YAW_D (PID Derivative Gain)

DEFAULT: 0	RANGE: null	UNITS: null

PID Derivative Gain

QuadPlane Note: This parameter configures the VTOL motors/behavior. It functions identically to the standard Copter parameter ATC_RAT_YAW_D.

Please refer to the ATC_RAT_YAW_D documentation for detailed tuning instructions.

Q_A_RAT_YAW_D_FF (FF FeedForward Gain)

DEFAULT: 0	RANGE: null	UNITS: null

FF FeedForward Gain

QuadPlane Note: This parameter configures the VTOL motors/behavior. It functions identically to the standard Copter parameter ATC_RAT_YAW_D_FF.

Please refer to the ATC_RAT_YAW_D_FF documentation for detailed tuning instructions.

Q_A_RAT_YAW_FF (FF FeedForward Gain)

DEFAULT: 0	RANGE: null	UNITS: null

FF FeedForward Gain

QuadPlane Note: This parameter configures the VTOL motors/behavior. It functions identically to the standard Copter parameter ATC_RAT_YAW_FF.

Please refer to the ATC_RAT_YAW_FF documentation for detailed tuning instructions.

Q_A_RAT_YAW_FLTD (PID D term filter frequency in Hz)

DEFAULT: AC_PID_DFILT_HZ_DEFAULT	RANGE: null	UNITS: Hz

PID D term filter frequency in Hz

QuadPlane Note: This parameter configures the VTOL motors/behavior. It functions identically to the standard Copter parameter ATC_RAT_YAW_FLTD.

Please refer to the ATC_RAT_YAW_FLTD documentation for detailed tuning instructions.

Q_A_RAT_YAW_FLTE (PID Error filter frequency in Hz)

DEFAULT: AC_PID_EFILT_HZ_DEFAULT	RANGE: null	UNITS: Hz

PID Error filter frequency in Hz

QuadPlane Note: This parameter configures the VTOL motors/behavior. It functions identically to the standard Copter parameter ATC_RAT_YAW_FLTE.

Please refer to the ATC_RAT_YAW_FLTE documentation for detailed tuning instructions.

Q_A_RAT_YAW_FLTT (PID Target filter frequency in Hz)

DEFAULT: AC_PID_TFILT_HZ_DEFAULT	RANGE: null	UNITS: Hz

PID Target filter frequency in Hz

QuadPlane Note: This parameter configures the VTOL motors/behavior. It functions identically to the standard Copter parameter ATC_RAT_YAW_FLTT.

Please refer to the ATC_RAT_YAW_FLTT documentation for detailed tuning instructions.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Q_A_RAT_YAW_I (PID Integral Gain)

DEFAULT: 0	RANGE: null	UNITS: null

PID Integral Gain

QuadPlane Note: This parameter configures the VTOL motors/behavior. It functions identically to the standard Copter parameter ATC_RAT_YAW_I.

Please refer to the ATC_RAT_YAW_I documentation for detailed tuning instructions.

Q_A_RAT_YAW_IMAX (PID Integral Maximum)

DEFAULT: 0	RANGE: null	UNITS: null

PID Integral Maximum

QuadPlane Note: This parameter configures the VTOL motors/behavior. It functions identically to the standard Copter parameter ATC_RAT_YAW_IMAX.

Please refer to the ATC_RAT_YAW_IMAX documentation for detailed tuning instructions.

Q_A_RAT_YAW_NEF (PID Error notch filter index)

DEFAULT: 0	RANGE: 1 8	UNITS: null

PID Error notch filter index

QuadPlane Note: This parameter configures the VTOL motors/behavior. It functions identically to the standard Copter parameter ATC_RAT_YAW_NEF.

Please refer to the ATC_RAT_YAW_NEF documentation for detailed tuning instructions.

Q_A_RAT_YAW_NTF (PID Target notch filter index)

DEFAULT: 0	RANGE: 1 8	UNITS: null

PID Target notch filter index

QuadPlane Note: This parameter configures the VTOL motors/behavior. It functions identically to the standard Copter parameter ATC_RAT_YAW_NTF.

Please refer to the ATC_RAT_YAW_NTF documentation for detailed tuning instructions.

Q_A_RAT_YAW_P (PID Proportional Gain)

DEFAULT: 0	RANGE: null	UNITS: null

PID Proportional Gain

QuadPlane Note: This parameter configures the VTOL motors/behavior. It functions identically to the standard Copter parameter ATC_RAT_YAW_P.

Please refer to the ATC_RAT_YAW_P documentation for detailed tuning instructions.

Q_A_RAT_YAW_PDMX (PD sum maximum)

DEFAULT: 0	RANGE: null	UNITS: null

PD sum maximum

QuadPlane Note: This parameter configures the VTOL motors/behavior. It functions identically to the standard Copter parameter ATC_RAT_YAW_PDMX.

Please refer to the ATC_RAT_YAW_PDMX documentation for detailed tuning instructions.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Q_A_RAT_YAW_SMAX (QuadPlane Yaw Axis Rate Slew Rate Limit)

DEFAULT: 0.0	**RANGE:** 0 200

QuadPlane Yaw Axis Rate Slew Rate Limit (Q_A_RAT_YAW_SMAX)

QuadPlane Note: This parameter configures the VTOL motors/behavior. It functions identically to the standard Copter parameter ATC_RAT_YAW_SMAX.

Please refer to the ATC_RAT_YAW_SMAX documentation for detailed tuning instructions.

Q_A_SLEW_YAW (Yaw target slew rate)

DEFAULT: AC_ATTITUDE_CONTROL_SLEW_YAW_DEFAULT_CDS	**RANGE:** 500 18000	**UNITS:** cdeg/s

Yaw target slew rate

QuadPlane Note: This parameter configures the VTOL motors/behavior. It functions identically to the standard Copter parameter ATC_SLEW_YAW.

Please refer to the ATC_SLEW_YAW documentation for detailed tuning instructions.

Q_A_THR_G_BOOST (Throttle-gain boost)

DEFAULT: 0.0f	**RANGE:** 0 1	**UNITS:** null

Throttle-gain boost

QuadPlane Note: This parameter configures the VTOL motors/behavior. It functions identically to the standard Copter parameter ATC_THR_G_BOOST.

Please refer to the ATC_THR_G_BOOST documentation for detailed tuning instructions.

Q_A_THR_MIX_MAN (Throttle Mix Manual)

DEFAULT: AC_ATTITUDE_CONTROL_MAN_DEFAULT	**RANGE:** 0.1 0.9	**UNITS:** null

Throttle Mix Manual

QuadPlane Note: This parameter configures the VTOL motors/behavior. It functions identically to the standard Copter parameter ATC_THR_MIX_MAN.

Please refer to the ATC_THR_MIX_MAN documentation for detailed tuning instructions.

Q_A_THR_MIX_MAX (Tiltrotor maximum VTOL angle)

DEFAULT: 45	**RANGE:** 20 80	**UNITS:** deg

Tiltrotor maximum VTOL angle

QuadPlane Note: This parameter configures the VTOL motors/behavior. It functions identically to the standard Copter parameter ATC_THR_MIX_MAX.

Please refer to the ATC_THR_MIX_MAX documentation for detailed tuning instructions.

Q_A_THR_MIX_MIN (Throttle Mix Minimum)

DEFAULT: AC_ATTITUDE_CONTROL_MIN_DEFAULT	**RANGE:** 0.1 0.25	**UNITS:** null

Throttle Mix Minimum

QuadPlane Note: This parameter configures the VTOL motors/behavior. It functions identically to the standard Copter parameter ATC_THR_MIX_MIN.

Please refer to the ATC_THR_MIX_MIN documentation for detailed tuning instructions.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Q_BACKTRANS_MS (QuadPlane Back Transition Time)

DEFAULT: 3000	RANGE: 0 10000	UNITS: ms

QuadPlane Back Transition Time (Q_BACKTRANS_MS)

Description

`Q_BACKTRANS_MS` sets how long the drone spends decelerating and mixing controls when switching from Plane mode to Copter mode (e.g., when you switch to QHOVER while flying fast).

A smoother transition gives the VTOL motors time to spin up before the wing loses lift.

Tuning & Behavior

- **Default Value:** 3000 ms (3 seconds).
- **Recommendation:** Set to **3000-5000** for smooth stops.

Q_BCK_PIT_LIM (Back Transition Pitch Limit)

DEFAULT: 30	RANGE: 0 45	UNITS: deg

Back Transition Pitch Limit (Q_BCK_PIT_LIM)

Description

`Q_BCK_PIT_LIM` prevents the drone from pointing its nose too high into the air while trying to slow down.

When you transition from a "Plane" to a "Copter," the autopilot pitches up to use the wings as airbrakes. This parameter ensures the pitch isn't so extreme that the drone stalls violently or flips.

Tuning & Behavior

- **Default Value:** 30 degrees.
- **Recommendation:** Leave at **30**. If your plane decelerates too slowly, you can increase this to **45**.

Q_ENABLE (Enable QuadPlane)

DEFAULT: 0	RANGE: 0 2

Enable QuadPlane (Q_ENABLE)

Description

`Q_ENABLE` transforms a standard ArduPlane installation into a **QuadPlane**.

When enabled, the autopilot allocates memory for the multicopter flight controllers (Attitude, Position, Motors) and mixes the output of vertical lift motors with the forward flight control surfaces.

- **0: Disable.** The aircraft flies as a standard fixed-wing plane.
- **1: Enable.** QuadPlane features are active. You can switch between VTOL modes (QLoiter, QHover) and Fixed-Wing modes (FBWA, Auto).
- **2: Enable VTOL AUTO.** Same as 1, but forces `AUTO` missions to default to VTOL behavior unless specified otherwise.

The Engineer's View

Defined in `ArduPlane/quadplane.cpp`.
Changing this parameter triggers a massive reallocation of internal objects (Motors, PID controllers, WPNav) during the next boot.
Reboot Required: You MUST reboot the flight controller after changing this parameter for the VTOL motors to function.

Tuning & Behavior

- **Default Value:** 0 (Disabled)
- **Recommendation:** Set to **1** for any hybrid VTOL aircraft.
- **Safety:** Ensure your frame configuration (`Q_FRAME_CLASS` , `Q_FRAME_TYPE`) is set correctly *before* plugging in the battery after enabling this.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Q_ESC_CAL (QuadPlane ESC Calibration)

DEFAULT: 0	RANGE: 0 1

QuadPlane ESC Calibration (Q_ESC_CAL)

Description

`Q_ESC_CAL` allows you to calibrate the vertical lift motors without affecting the forward pusher motor.

Tuning & Behavior

- **0 (Disabled):** Normal operation.
- **1 (Enabled):** On next boot, the VTOL motors will pass-through the throttle stick for calibration (High/Low endpoint setting). The forward motor will remain safe.

Q_FRAME_CLASS (QuadPlane Frame Class)

DEFAULT: 1	RANGE: 0 17

QuadPlane Frame Class (Q_FRAME_CLASS)

Description

`Q_FRAME_CLASS` tells the autopilot how many vertical lift motors you have and how they are arranged. This corresponds exactly to the `FRAME_CLASS` used in ArduCopter.

- **1: Quad.** 4 Motors in a square/rectangle. (Standard QuadPlane).
- **2: Hexa.** 6 Motors.
- **3: Octa.** 8 Motors.
- **7: Tri.** 3 Motors (Tricopter layout).
- **10: Tailsitter.** (Duo/Quad). The entire plane sits vertically on its tail for takeoff.

The Engineer's View

Used to initialize the `AP_Motors` matrix.
For **Tilt-Rotors** (like the V-22 Osprey or Convergence), you typically select the class that matches the number of *rotors* (e.g., **Tri** for a Convergence), and then use `Q_TILT_MASK` to define which ones can tilt.

Tuning & Behavior

- **Default Value:** 1 (Quad)
- **Reboot Required:** Yes.
- **Safety:** If set incorrectly (e.g., Hexa on a Quad), the motor mixing will be wrong, likely causing an immediate flip on takeoff.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Q_FRAME_TYPE (QuadPlane Frame Type)

DEFAULT: 1	RANGE: 0 19

QuadPlane Frame Type (Q_FRAME_TYPE)

Description

`Q_FRAME_TYPE` specifies the geometry of the frame class selected in `Q_FRAME_CLASS`.

- **1: X (Default).** Motors are at 45 degrees to the nose. (Standard for almost all QuadPlanes).
- **0: Plus (+).** Motors are at 0/90/180/270 degrees.
- **2: V.**
- **3: H.** H-frame quad.

The Engineer's View

Used by `AP_MotorsMatrix` to populate the roll/pitch/yaw factors for each motor output.
Most QuadPlanes use an **X** configuration because it keeps the propellers out of the camera view and allows the fuselage to pass between the rotors.

Tuning & Behavior

- **Default Value:** 1 (X)
- **Recommendation:** Leave at **1** for standard builds.
- **Reboot Required:** Yes.

Q_FWD_MANTHR_MAX (Pilot Manual Forward Throttle)

DEFAULT: 0	RANGE: 0 100	UNITS: %

Pilot Manual Forward Throttle (Q_FWD_MANTHR_MAX)

Description

`Q_FWD_MANTHR_MAX` allows the pilot to "Gas it" while hovering.

If you have a dedicated slider or knob on your transmitter mapped to the forward motor, this parameter sets the maximum power that slider can command while the drone is in a VTOL mode.

Tuning & Behavior

- **Default Value:** 0 (Disabled).
- **Recommendation:** Set to **100%** if you want full manual authority over the pusher motor for repositioning the drone quickly in QHOVER.

Q_FWD_PIT_LIM (QuadPlane Forward Pitch Limit)

DEFAULT: 5	RANGE: 0 15	UNITS: deg

QuadPlane Forward Pitch Limit (Q_FWD_PIT_LIM)

Description

`Q_FWD_PIT_LIM` restricts how much the nose can dip when `Q_FWD_THR_USE` is active.

Since the forward motor is doing the work of fighting the wind, the drone doesn't need to tilt much. Keeping the fuselage level (near 0 pitch) reduces drag and makes the drone more stable.

Tuning & Behavior

- **Default Value:** 5 degrees.
- **Recommendation:** Keep small (3-8 degrees).

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Q_FWD_THR_GAIN (QuadPlane Forward Throttle Gain)

DEFAULT: 0.5	RANGE: 0 5

QuadPlane Forward Throttle Gain (Q_FWD_THR_GAIN)

Description

`Q_FWD_THR_GAIN` determines how aggressively the forward motor spins up when the drone tilts forward in QLOITER.

If `Q_FWD_THR_USE` is enabled, this parameter translates the "tilt demand" (e.g., 5 degrees nose down) into a throttle percentage for the pusher prop.

Tuning & Behavior

- **Default Value:** 0.5
- **Recommendation:** Increase until the drone holds position in wind without needing to pitch down excessively.

Q_FWD_THR_USE (QuadPlane Forward Throttle Use)

DEFAULT: 0	RANGE: 0 3

QuadPlane Forward Throttle Use (Q_FWD_THR_USE)

Description

`Q_FWD_THR_USE` allows the forward pusher motor to help the drone hold position against strong winds while in a hover mode.

- **0 (Disabled):** The forward motor is off in VTOL modes.
- **1 (Position Hold):** The forward motor is used to fight headwind in QLOITER/QLAND.
- **2 (Distance):** Used for position hold and to speed up travel to waypoints in VTOL modes.

The Engineer's View

If enabled, the position controller calculates a pitch demand. If the pitch demand is "nose down" (fight wind or move forward), the mixer allocates some of that demand to the forward motor instead of just pitching the quadcopter rotors. This keeps the fuselage level and efficient.

Tuning & Behavior

- **Default Value:** 0 (Disabled).
- **Recommendation:** Set to **1** if you fly in windy conditions. It significantly improves position holding capability.

Q_FW_LND_APR_RAD (QuadPlane Fixed-Wing Landing Approach Radius)

DEFAULT: 0	RANGE: 0 1000	UNITS: m

QuadPlane Fixed-Wing Landing Approach Radius (Q_FW_LND_APR_RAD)

Description

`Q_FW_LND_APR_RAD` defines the tightness of the turn when lining up for a runway landing.

If set to 0, it uses `WP_LOITER_RAD`.

Tuning & Behavior

- **Default Value:** 0.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Q_GUIDED_MODE (Enable VTOL in GUIDED mode)

DEFAULT: 0	RANGE: 0 1

Enable VTOL in GUIDED mode (Q_GUIDED_MODE)

Description

`Q_GUIDED_MODE` determines how the QuadPlane behaves when receiving "Fly To" commands from a Ground Control Station (GCS) or companion computer.

- **0: Disabled (Fixed Wing).** The plane flies to the guided waypoint as a fixed-wing aircraft and circles (loiters) around it. It does not use the vertical lift motors.
- **1: Enabled (VTOL).** The plane flies to the guided waypoint. Once it arrives, it transitions to VTOL mode and hovers stationary at the target coordinates.

The Engineer's View

Used in `ModeGuided::update()`.
If enabled, the vehicle will execute a `DO_VTOL_TRANSITION` automatically upon reaching the target radius. This is essential for "Drone-in-a-box" applications where the vehicle must precisely hover over a landing pad or delivery point.

Tuning & Behavior

- **Default Value:** 0
- **Recommendation:**
  - **Mapping/Surveillance:** Keep at **0** to save battery (circling is much more efficient than hovering).
  - **Delivery/Precision Inspection:** Set to **1** to allow precise hovering at the target.

Q_LAND_ALTCHG (QuadPlane Land Altitude Change)

DEFAULT: 0	RANGE: 0 1

QuadPlane Land Altitude Change (Q_LAND_ALTCHG)

Description

`Q_LAND_ALTCHG` allows the drone to descend *while* it is traveling horizontally toward the landing spot.

- **0 (Default):** The drone flies at `Q_RTL_ALT` to above the home point, stops, then descends vertically. Safe but slow.
- **1 (Enabled):** The drone descends on a diagonal path. Faster, but requires clear airspace.

Tuning & Behavior

- **Default Value:** 0.

Q_LAND_FINAL_ALT (QuadPlane Land Final Altitude)

DEFAULT: 6	RANGE: 0 50	UNITS: m

Q_LAND_FINAL_ALT: QuadPlane Land Final Altitude

Description

The altitude (in meters) where the QuadPlane switches to its final descent speed.

Description

`Q_LAND_FINAL_ALT` is the VTOL equivalent of `LAND_ALT_LOW`.

When the QuadPlane descends to this height, it slows down to `Q_LAND_FINAL_SPD` to ensure a soft touchdown.

Tuning & Behavior

- **Default Value:** 6 meters.
- **Recommendation:** Increase to **10-15m** for heavy aircraft to give the vertical velocity controller more time to stabilize before contact.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Q_LAND_FINAL_SPD (QuadPlane Land Final Speed)

DEFAULT: 0.5	RANGE: 0.1 2.0	UNITS: m/s

QuadPlane Land Final Speed (Q_LAND_FINAL_SPD)

Description

`Q_LAND_FINAL_SPD` is the "Touchdown Velocity."

- **0.5 (Default):** 0.5 m/s. Safe for most skids and wheels.
- **0.3:** Very gentle.
- **1.0:** Firm landing.

Tuning & Behavior

- **Default Value:** 0.5 m/s.
- **Recommendation:** If you land on uneven ground, keep this low. If you land on a moving boat, you might need it higher to "stick" the landing.

Q_LAND_ICE_CUT (QuadPlane Land ICE Cut)

DEFAULT: 0	RANGE: 0 10	UNITS: m

QuadPlane Land ICE Cut (Q_LAND_ICE_CUT)

Description

`Q_LAND_ICE_CUT` automatically kills the gas engine during the final stage of a vertical landing.

This prevents the propeller from striking the ground if the plane tips over, and stops the engine from idling on the ground.

Tuning & Behavior

- **Default Value:** 0 (Disabled).
- **Recommendation:** Set to **2 meters** to cut the engine just before touchdown.

Q_LOIT_ACCEL (QuadPlane Loiter Horizontal Acceleration)

DEFAULT: 250	RANGE: 0 980	UNITS: cm/s/s

QuadPlane Loiter Horizontal Acceleration (Q_LOIT_ACCEL)

Description

`Q_LOIT_ACCEL` controls how quickly the QuadPlane speeds up and slows down in QLOITER mode.

- **Higher values** make the drone feel "snappy" and responsive to stick inputs.
- **Lower values** make the movement feel smooth and cinematic.

Tuning & Behavior

- **Default Value:** 250 cm/s/s.
- **Recommendation:** Set between **200 and 500**.

Q_LOIT_ACC_MAX (Tiltrotor maximum VTOL angle)

DEFAULT: 45	RANGE: 20 80	UNITS: deg

Tiltrotor maximum VTOL angle

QuadPlane Note: This parameter configures the VTOL motors/behavior. It functions identically to the standard Copter parameter LOIT_ACC_MAX.

Please refer to the LOIT_ACC_MAX documentation for detailed tuning instructions.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Q_LOIT_ANG_MAX (Tiltrotor maximum VTOL angle)

DEFAULT: 45	**RANGE:** 20 80	**UNITS:** deg

Tiltrotor maximum VTOL angle

QuadPlane Note: This parameter configures the VTOL motors/behavior. It functions identically to the standard Copter parameter LOIT_ANG_MAX.

Please refer to the LOIT_ANG_MAX documentation for detailed tuning instructions.

Q_LOIT_BRK_ACCEL (Loiter braking acceleration)

DEFAULT: LOITER_BRAKE_ACCEL_DEFAULT	**RANGE:** 25 250	**UNITS:** cm/s/s

Loiter braking acceleration

QuadPlane Note: This parameter configures the VTOL motors/behavior. It functions identically to the standard Copter parameter LOIT_BRK_ACCEL.

Please refer to the LOIT_BRK_ACCEL documentation for detailed tuning instructions.

Q_LOIT_BRK_DELAY (Loiter brake start delay (in seconds))

DEFAULT: LOITER_BRAKE_START_DELAY_DEFAULT	**RANGE:** 0 2	**UNITS:** s

Loiter brake start delay (in seconds)

QuadPlane Note: This parameter configures the VTOL motors/behavior. It functions identically to the standard Copter parameter LOIT_BRK_DELAY.

Please refer to the LOIT_BRK_DELAY documentation for detailed tuning instructions.

Q_LOIT_BRK_JERK (Loiter braking jerk)

DEFAULT: LOITER_BRAKE_JERK_DEFAULT	**RANGE:** 500 5000	**UNITS:** cm/s/s/s

Loiter braking jerk

QuadPlane Note: This parameter configures the VTOL motors/behavior. It functions identically to the standard Copter parameter LOIT_BRK_JERK.

Please refer to the LOIT_BRK_JERK documentation for detailed tuning instructions.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Q_LOIT_OPTIONS (QLoiter mode options)

DEFAULT: 0	RANGE: 0 1

QLoiter mode options (Q_LOIT_OPTIONS)

Description

`Q_LOIT_OPTIONS` modifies the behavior of the QuadPlane **QLOITER** (VTOL Loiter) flight mode.

- **Bit 0 (1):** Allow QLoiter to hold position without GPS (using Optical Flow or other non-GPS sources).

The Mathematics

$$\text{Behavior} = \sum 2^i \cdot b_i$$

The Engineer's View

Used in `QuadPlane::mode_qloiter` logic.
If Bit 0 is set, the strict check for a 3D GPS lock is relaxed, allowing the vehicle to enter and maintain QLOITER using alternative position sources like Optical Flow or Visual Odometry.

Tuning & Behavior

- **Default Value:** 0 (Require GPS)
- **Recommendation:**
  - **GPS Flight:** Leave at 0.
  - **Indoor/Non-GPS Flight:** Set to **1** if you have a valid Optical Flow or VIO setup and want to loiter.

Q_LOIT_SPEED (QuadPlane Loiter Horizontal Speed)

DEFAULT: 1250	RANGE: 0 2000	UNITS: cm/s

QuadPlane Loiter Horizontal Speed (Q_LOIT_SPEED)

Description

`Q_LOIT_SPEED` determines how fast the QuadPlane can move sideways or forwards while in QLOITER mode.

Tuning & Behavior

- **Default Value:** 1250 cm/s (12.5 m/s).
- **Recommendation:** Set based on your desired cruise speed in hover.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Q_MAV_TYPE (MAVLink type identifier)

DEFAULT: 0	RANGE: 0 21

MAVLink type identifier (Q_MAV_TYPE)

Description

`Q_MAV_TYPE` tells your Ground Control Station (GCS) what icon to display for your vehicle.

- **0: AUTO (Default).** ArduPilot decides based on the frame class (usually reports as "FIXED_WING").
- **20: VTOL_QUADROTOR.** Reports as a VTOL QuadPlane.
- **21: VTOL_TILTROTOR.** Reports as a Tiltrotor.

The Engineer's View

Some GCS software (like QGroundControl) changes its available flight modes or UI layout based on the reported MAV_TYPE. For example, if reported as "FIXED_WING", QGC might hide the "Takeoff" button slider that is normally present for Copters.

Tuning & Behavior

- **Default Value:** 0
- **Recommendation:** Leave at **0** unless your GCS is not displaying the correct vehicle icon or options.

Q_M_BAT_CURR_MAX (Tiltrotor maximum VTOL angle)

DEFAULT: 45	RANGE: 20 80	UNITS: deg

Tiltrotor maximum VTOL angle

QuadPlane Note: This parameter configures the VTOL motors/behavior. It functions identically to the standard Copter parameter MOT_BAT_CURR_MAX.

Please refer to the MOT_BAT_CURR_MAX documentation for detailed tuning instructions.

Q_M_BAT_CURR_TC (Motor Current Max Time Constant)

DEFAULT: AP_MOTORS_BAT_CURR_TC_DEFAULT	RANGE: 0 10	UNITS: s

Motor Current Max Time Constant

QuadPlane Note: This parameter configures the VTOL motors/behavior. It functions identically to the standard Copter parameter MOT_BAT_CURR_TC.

Please refer to the MOT_BAT_CURR_TC documentation for detailed tuning instructions.

Q_M_BAT_IDX (Battery compensation index)

DEFAULT: 0	RANGE: null	UNITS: null

Battery compensation index

QuadPlane Note: This parameter configures the VTOL motors/behavior. It functions identically to the standard Copter parameter MOT_BAT_IDX.

Please refer to the MOT_BAT_IDX documentation for detailed tuning instructions.

Q_M_BAT_VOLT_MAX (Tiltrotor maximum VTOL angle)

DEFAULT: 45	RANGE: 20 80	UNITS: deg

Tiltrotor maximum VTOL angle

QuadPlane Note: This parameter configures the VTOL motors/behavior. It functions identically to the standard Copter parameter MOT_BAT_VOLT_MAX.

Please refer to the MOT_BAT_VOLT_MAX documentation for detailed tuning instructions.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Q_M_BAT_VOLT_MIN (Battery voltage compensation minimum voltage)

DEFAULT: AP_MOTORS_BAT_VOLT_MIN_DEFAULT	RANGE: 6 42	UNITS: V

Battery voltage compensation minimum voltage

QuadPlane Note: This parameter configures the VTOL motors/behavior. It functions identically to the standard Copter parameter MOT_BAT_VOLT_MIN.

Please refer to the MOT_BAT_VOLT_MIN documentation for detailed tuning instructions.

Q_M_BOOST_SCALE (Compass1 scale factor)

DEFAULT: 0	RANGE: 0 1.3	UNITS: null

Compass1 scale factor

QuadPlane Note: This parameter configures the VTOL motors/behavior. It functions identically to the standard Copter parameter MOT_BOOST_SCALE.

Please refer to the MOT_BOOST_SCALE documentation for detailed tuning instructions.

Q_M_HOVER_LEARN (Learn compass offsets automatically)

DEFAULT: COMPASS_LEARN_DEFAULT	RANGE: null	UNITS: null

Learn compass offsets automatically

QuadPlane Note: This parameter configures the VTOL motors/behavior. It functions identically to the standard Copter parameter MOT_HOVER_LEARN.

Please refer to the MOT_HOVER_LEARN documentation for detailed tuning instructions.

Q_M_OPTIONS (quadplane options)

DEFAULT: 0	RANGE: null	UNITS: null

quadplane options

QuadPlane Note: This parameter configures the VTOL motors/behavior. It functions identically to the standard Copter parameter MOT_OPTIONS.

Please refer to the MOT_OPTIONS documentation for detailed tuning instructions.

Q_M_PWM_MAX (Tiltrotor maximum VTOL angle)

DEFAULT: 45	RANGE: 20 80	UNITS: deg

Tiltrotor maximum VTOL angle

QuadPlane Note: This parameter configures the VTOL motors/behavior. It functions identically to the standard Copter parameter MOT_PWM_MAX.

Please refer to the MOT_PWM_MAX documentation for detailed tuning instructions.

Q_M_PWM_MIN (PWM output minimum)

DEFAULT: 1000	RANGE: 0 2000	UNITS: PWM

PWM output minimum

QuadPlane Note: This parameter configures the VTOL motors/behavior. It functions identically to the standard Copter parameter MOT_PWM_MIN.

Please refer to the MOT_PWM_MIN documentation for detailed tuning instructions.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Q_M_PWM_TYPE (Tiltrotor type)

DEFAULT: TILT_TYPE_CONTINUOUS	RANGE: null	UNITS: null

Tiltrotor type

QuadPlane Note: This parameter configures the VTOL motors/behavior. It functions identically to the standard Copter parameter MOT_PWM_TYPE.

Please refer to the MOT_PWM_TYPE documentation for detailed tuning instructions.

Q_M_SAFE_DISARM (Motor PWM output disabled when disarmed)

DEFAULT: 0	RANGE: null	UNITS: null

Motor PWM output disabled when disarmed

QuadPlane Note: This parameter configures the VTOL motors/behavior. It functions identically to the standard Copter parameter MOT_SAFE_DISARM.

Please refer to the MOT_SAFE_DISARM documentation for detailed tuning instructions.

Q_M_SAFE_TIME (Time taken to disable and enable the motor PWM output when disarmed)

DEFAULT: AP_MOTORS_SAFE_TIME_DEFAULT	RANGE: 0 5	UNITS: s

Time taken to disable and enable the motor PWM output when disarmed and armed.

QuadPlane Note: This parameter configures the VTOL motors/behavior. It functions identically to the standard Copter parameter MOT_SAFE_TIME.

Please refer to the MOT_SAFE_TIME documentation for detailed tuning instructions.

Q_M_SLEW_DN_TIME (Output slew time for decreasing throttle)

DEFAULT: AP_MOTORS_SLEW_TIME_DEFAULT	RANGE: 0 .5	UNITS: s

Output slew time for decreasing throttle

QuadPlane Note: This parameter configures the VTOL motors/behavior. It functions identically to the standard Copter parameter MOT_SLEW_DN_TIME.

Please refer to the MOT_SLEW_DN_TIME documentation for detailed tuning instructions.

Q_M_SLEW_UP_TIME (Output slew time for increasing throttle)

DEFAULT: AP_MOTORS_SLEW_TIME_DEFAULT	RANGE: 0 .5	UNITS: s

Output slew time for increasing throttle

QuadPlane Note: This parameter configures the VTOL motors/behavior. It functions identically to the standard Copter parameter MOT_SLEW_UP_TIME.

Please refer to the MOT_SLEW_UP_TIME documentation for detailed tuning instructions.

Q_M_SPIN_ARM (Motor Spin armed)

DEFAULT: AP_MOTORS_SPIN_ARM_DEFAULT	RANGE: null	UNITS: null

Motor Spin armed

QuadPlane Note: This parameter configures the VTOL motors/behavior. It functions identically to the standard Copter parameter MOT_SPIN_ARM.

Please refer to the MOT_SPIN_ARM documentation for detailed tuning instructions.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Q_M_SPIN_MAX (Tiltrotor maximum VTOL angle)

DEFAULT: 45	RANGE: 20 80	UNITS: deg

Tiltrotor maximum VTOL angle

QuadPlane Note: This parameter configures the VTOL motors/behavior. It functions identically to the standard Copter parameter MOT_SPIN_MAX.

Please refer to the MOT_SPIN_MAX documentation for detailed tuning instructions.

Q_M_SPIN_MIN (Motor Spin minimum)

DEFAULT: AP_MOTORS_SPIN_MIN_DEFAULT	RANGE: null	UNITS: null

Motor Spin minimum

QuadPlane Note: This parameter configures the VTOL motors/behavior. It functions identically to the standard Copter parameter MOT_SPIN_MIN.

Please refer to the MOT_SPIN_MIN documentation for detailed tuning instructions.

Q_M_SPOOL_TIME (Spool up time)

DEFAULT: AP_MOTORS_SPOOL_UP_TIME_DEFAULT	RANGE: 0.05 2	UNITS: s

Spool up time

QuadPlane Note: This parameter configures the VTOL motors/behavior. It functions identically to the standard Copter parameter MOT_SPOOL_TIME.

Please refer to the MOT_SPOOL_TIME documentation for detailed tuning instructions.

Q_M_SPOOL_TIM_DN (Pressure error coefficient in negative Z direction (down))

DEFAULT: 0.0	RANGE: -1.0 1.0	UNITS: null

Pressure error coefficient in negative Z direction (down)

QuadPlane Note: This parameter configures the VTOL motors/behavior. It functions identically to the standard Copter parameter MOT_SPOOL_TIM_DN.

Please refer to the MOT_SPOOL_TIM_DN documentation for detailed tuning instructions.

Q_M_THST_EXPO (Controlled Expo)

DEFAULT: default_expo	RANGE: -0.5 1.0	UNITS: null

Controlled Expo

QuadPlane Note: This parameter configures the VTOL motors/behavior. It functions identically to the standard Copter parameter MOT_THST_EXPO.

Please refer to the MOT_THST_EXPO documentation for detailed tuning instructions.

Q_M_THST_HOVER (Thrust Hover Value)

DEFAULT: AP_MOTORS_THST_HOVER_DEFAULT	RANGE: 0.125 0.6875	UNITS: null

Thrust Hover Value

QuadPlane Note: This parameter configures the VTOL motors/behavior. It functions identically to the standard Copter parameter MOT_THST_HOVER.

Please refer to the MOT_THST_HOVER documentation for detailed tuning instructions.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Q_M_YAW_HEADROOM (Matrix Yaw Min)

DEFAULT: AP_MOTORS_YAW_HEADROOM_DEFAULT	RANGE: 0 500	UNITS: PWM

Matrix Yaw Min

QuadPlane Note: This parameter configures the VTOL motors/behavior. It functions identically to the standard Copter parameter MOT_YAW_HEADROOM.

Please refer to the MOT_YAW_HEADROOM documentation for detailed tuning instructions.

Q_M_YAW_SV_ANGLE (Tailsitter fixed wing transition angle)

DEFAULT: 45	RANGE: 5 80	UNITS: deg

Tailsitter fixed wing transition angle

QuadPlane Note: This parameter configures the VTOL motors/behavior. It functions identically to the standard Copter parameter MOT_YAW_SV_ANGLE.

Please refer to the MOT_YAW_SV_ANGLE documentation for detailed tuning instructions.

Q_NAVALT_MIN (QuadPlane Minimum Navigation Altitude)

DEFAULT: 0	RANGE: 0 50	UNITS: m

QuadPlane Minimum Navigation Altitude (Q_NAVALT_MIN)

Description

`Q_NAVALT_MIN` prevents the drone from leaning over to fly to a waypoint while it is still on the ground or taking off.

The drone must climb vertically to this height *before* it starts moving horizontally toward the first waypoint.

Tuning & Behavior

- **Default Value:** 0 (Disabled).
- **Recommendation:** Set to **3 or 5 meters** to ensure the drone clears obstacles/grass before tilting.

Q_OPTIONS (QuadPlane Options)

DEFAULT: 0	RANGE: 0 4294967295

QuadPlane Options (Q_OPTIONS)

Description

`Q_OPTIONS` enables specialized behaviors for VTOL flight.

- **Bit 0 (1):** Level Transitions. Keep wings level during transition (default is to bank).
- **Bit 1 (2):** Allow taking off in fixed-wing modes (auto-transition).
- **Bit 3 (8):** Disarm motors in QuadPlane modes on crash detection.
- **Bit 5 (32):** Thrusters always active in Q modes (for tailsitters).

Tuning & Behavior

- **Default Value:** 0.
- **Recommendation:** Set **Bit 16 (65536)** to suppress "QuadPlane warnings" if you know what you are doing.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Q_PILOT_ACCEL_Z (Pilot Vertical Acceleration)

DEFAULT: 250	RANGE: 50 500	UNITS: cm/s/s

Pilot Vertical Acceleration (Q_PILOT_ACCEL_Z)

Description

`Q_PILOT_ACCEL_Z` determines how "Snappy" the altitude control feels when you move the throttle stick in VTOL modes (like QHOVER or QLOITER).

It sets the maximum rate at which the vertical velocity can change. A higher value makes the drone react instantly to stick movements, while a lower value makes the climb/descent feel more gradual and "heavy."

Tuning & Behavior

- **Default Value:** 250 cm/s/s (0.25g).
- **High Value (500):** Very aggressive, jumpy altitude control.
- **Low Value (100):** Very smooth, slow transitions between climbing and descending.
- **Recommendation:** Keep at **250** for a balanced feel.

Q_PILOT_SPD_DN (Pilot Maximum Descent Rate)

DEFAULT: 0	RANGE: 0 500	UNITS: cm/s

Pilot Maximum Descent Rate (Q_PILOT_SPD_DN)

Description

`Q_PILOT_SPD_DN` sets the "Speed Limit" for going down in VTOL modes.

Tuning & Behavior

- **Default Value:** 0 (Uses `PILOT_SPEED_DN` or `Q_PILOT_SPD_UP` if 0).
- **Recommendation:** Set to **150-200 cm/s** for safe landings. Descending too fast can cause the drone to wobble in its own rotor wash (Vortex Ring State).

Q_PILOT_SPD_UP (Pilot Maximum Climb Rate)

DEFAULT: 250	RANGE: 50 1000	UNITS: cm/s

Pilot Maximum Climb Rate (Q_PILOT_SPD_UP)

Description

`Q_PILOT_SPD_UP` sets the "Speed Limit" for going up in VTOL modes.

When you push the throttle stick to the maximum, the drone will climb at exactly this rate. This ensures predictable behavior and prevents the pilot from over-working the lift motors.

Tuning & Behavior

- **Default Value:** 250 cm/s (2.5 m/s).
- **Recommendation:** Match this to your airframe's power-to-weight ratio. High-power quads can handle **500 or 1000 cm/s**.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Q_PLT_Y_EXPO (Pilot Yaw Exponential)

DEFAULT: 0	RANGE: 0 1

Pilot Yaw Exponential (Q_PLT_Y_EXPO)

Description

`Q_PLT_Y_EXPO` makes the yaw stick less sensitive around the center point.

This allows for very fine heading corrections with small stick movements, while still providing full rotation speed at the stick edges.

Tuning & Behavior

- **0:** Linear control (same sensitivity everywhere).
- **0.5:** Moderate expo.
- **Recommendation:** Use **0.15 - 0.25** for a natural control feel.

Q_PLT_Y_RATE (Pilot Commanded Yaw Rate)

DEFAULT: 0	RANGE: 0 500	UNITS: deg/s

Pilot Commanded Yaw Rate (Q_PLT_Y_RATE)

Description

`Q_PLT_Y_RATE` sets how fast the drone spins when the pilot pushes the rudder stick.

- **0 (Default):** Use `Q_YAW_RATE_MAX`.
- **Value:** Override with a specific rate for pilot input.

Q_PLT_Y_RATE_TC (Pilot Yaw Rate Time Constant)

DEFAULT: 0.25	RANGE: 0 1	UNITS: s

Pilot Yaw Rate Time Constant (Q_PLT_Y_RATE_TC)

Description

`Q_PLT_Y_RATE_TC` determines how quickly the drone reaches the commanded yaw speed.

It smooths out the "Start" and "Stop" of a rotation. A higher value makes the drone feel heavier and more graceful, while a lower value makes it feel more responsive and robotic.

Tuning & Behavior

- **Default Value:** 0.25 seconds.

Q_P_ANGLE_MAX (Angle Max)

DEFAULT: 3000	RANGE: 1000 8000	UNITS: cdeg

Angle Max

QuadPlane Note: This parameter configures the VTOL motors/behavior. It functions identically to the standard Copter parameter PSC_ANGLE_MAX.

Please refer to the PSC_ANGLE_MAX documentation for detailed tuning instructions.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY

Q_P_D_ACC_D (PID Derivative Gain)

DEFAULT: 0	RANGE: null	UNITS: null

PID Derivative Gain

QuadPlane Note: This parameter configures the VTOL motors/behavior. It functions identically to the standard Copter parameter PSC_ACCZ_D.

Please refer to the PSC_ACCZ_D documentation for detailed tuning instructions.

Q_P_D_ACC_D_FF (FF FeedForward Gain)

DEFAULT: 0	RANGE: null	UNITS: null

FF FeedForward Gain

QuadPlane Note: This parameter configures the VTOL motors/behavior. It functions identically to the standard Copter parameter PSC_ACCZ_D_FF.

Please refer to the PSC_ACCZ_D_FF documentation for detailed tuning instructions.

Q_P_D_ACC_FF (FF FeedForward Gain)

DEFAULT: 0	RANGE: null	UNITS: null

FF FeedForward Gain

QuadPlane Note: This parameter configures the VTOL motors/behavior. It functions identically to the standard Copter parameter PSC_ACCZ_FF.

Please refer to the PSC_ACCZ_FF documentation for detailed tuning instructions.

Q_P_D_ACC_FLTD (PID D term filter frequency in Hz)

DEFAULT: AC_PID_DFILT_HZ_DEFAULT	RANGE: null	UNITS: Hz

PID D term filter frequency in Hz

QuadPlane Note: This parameter configures the VTOL motors/behavior. It functions identically to the standard Copter parameter PSC_ACCZ_FLTD.

Please refer to the PSC_ACCZ_FLTD documentation for detailed tuning instructions.

Q_P_D_ACC_FLTE (PID Error filter frequency in Hz)

DEFAULT: AC_PID_EFILT_HZ_DEFAULT	RANGE: null	UNITS: Hz

PID Error filter frequency in Hz

QuadPlane Note: This parameter configures the VTOL motors/behavior. It functions identically to the standard Copter parameter PSC_ACCZ_FLTE.

Please refer to the PSC_ACCZ_FLTE documentation for detailed tuning instructions.

Q_P_D_ACC_FLTT (PID Target filter frequency in Hz)

DEFAULT: AC_PID_TFILT_HZ_DEFAULT	RANGE: null	UNITS: Hz

PID Target filter frequency in Hz

QuadPlane Note: This parameter configures the VTOL motors/behavior. It functions identically to the standard Copter parameter PSC_ACCZ_FLTT.

Please refer to the PSC_ACCZ_FLTT documentation for detailed tuning instructions.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Q_P_D_ACC_I (PID Integral Gain)

DEFAULT: 0	RANGE: null	UNITS: null

PID Integral Gain

QuadPlane Note: This parameter configures the VTOL motors/behavior. It functions identically to the standard Copter parameter PSC_ACCZ_I.

Please refer to the PSC_ACCZ_I documentation for detailed tuning instructions.

Q_P_D_ACC_IMAX (PID Integral Maximum)

DEFAULT: 0	RANGE: null	UNITS: null

PID Integral Maximum

QuadPlane Note: This parameter configures the VTOL motors/behavior. It functions identically to the standard Copter parameter PSC_ACCZ_IMAX.

Please refer to the PSC_ACCZ_IMAX documentation for detailed tuning instructions.

Q_P_D_ACC_NEF (PID Error notch filter index)

DEFAULT: 0	RANGE: 1 8	UNITS: null

PID Error notch filter index

QuadPlane Note: This parameter configures the VTOL motors/behavior. It functions identically to the standard Copter parameter PSC_ACCZ_NEF.

Please refer to the PSC_ACCZ_NEF documentation for detailed tuning instructions.

Q_P_D_ACC_NTF (PID Target notch filter index)

DEFAULT: 0	RANGE: 1 8	UNITS: null

PID Target notch filter index

QuadPlane Note: This parameter configures the VTOL motors/behavior. It functions identically to the standard Copter parameter PSC_ACCZ_NTF.

Please refer to the PSC_ACCZ_NTF documentation for detailed tuning instructions.

Q_P_D_ACC_P (PID Proportional Gain)

DEFAULT: 0	RANGE: null	UNITS: null

PID Proportional Gain

QuadPlane Note: This parameter configures the VTOL motors/behavior. It functions identically to the standard Copter parameter PSC_ACCZ_P.

Please refer to the PSC_ACCZ_P documentation for detailed tuning instructions.

Q_P_D_ACC_PDMX (PD sum maximum)

DEFAULT: 0	RANGE: null	UNITS: null

PD sum maximum

QuadPlane Note: This parameter configures the VTOL motors/behavior. It functions identically to the standard Copter parameter PSC_ACCZ_PDMX.

Please refer to the PSC_ACCZ_PDMX documentation for detailed tuning instructions.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Q_P_D_ACC_SMAX (Slew rate limit)

DEFAULT: 0	RANGE: 0 200	UNITS: null

Slew rate limit

QuadPlane Note: This parameter configures the VTOL motors/behavior. It functions identically to the standard Copter parameter PSC_ACCZ_SMAX.

Please refer to the PSC_ACCZ_SMAX documentation for detailed tuning instructions.

Q_P_D_POS_P (PID Proportional Gain)

DEFAULT: 0	RANGE: null	UNITS: null

PID Proportional Gain

QuadPlane Note: This parameter configures the VTOL motors/behavior. It functions identically to the standard Copter parameter PSC_POSZ_P.

Please refer to the PSC_POSZ_P documentation for detailed tuning instructions.

Q_P_D_VEL_D (PID Derivative Gain)

DEFAULT: 0	RANGE: null	UNITS: null

PID Derivative Gain

QuadPlane Note: This parameter configures the VTOL motors/behavior. It functions identically to the standard Copter parameter PSC_VELZ_D.

Please refer to the PSC_VELZ_D documentation for detailed tuning instructions.

Q_P_D_VEL_FF (FF FeedForward Gain)

DEFAULT: 0	RANGE: null	UNITS: null

FF FeedForward Gain

QuadPlane Note: This parameter configures the VTOL motors/behavior. It functions identically to the standard Copter parameter PSC_VELZ_FF.

Please refer to the PSC_VELZ_FF documentation for detailed tuning instructions.

Q_P_D_VEL_FLTD (PID D term filter frequency in Hz)

DEFAULT: AC_PID_DFILT_HZ_DEFAULT	RANGE: null	UNITS: Hz

PID D term filter frequency in Hz

QuadPlane Note: This parameter configures the VTOL motors/behavior. It functions identically to the standard Copter parameter PSC_VELZ_FLTD.

Please refer to the PSC_VELZ_FLTD documentation for detailed tuning instructions.

Q_P_D_VEL_FLTE (PID Error filter frequency in Hz)

DEFAULT: AC_PID_EFILT_HZ_DEFAULT	RANGE: null	UNITS: Hz

PID Error filter frequency in Hz

QuadPlane Note: This parameter configures the VTOL motors/behavior. It functions identically to the standard Copter parameter PSC_VELZ_FLTE.

Please refer to the PSC_VELZ_FLTE documentation for detailed tuning instructions.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Q_P_D_VEL_I (PID Integral Gain)

DEFAULT: 0	RANGE: null	UNITS: null

PID Integral Gain

QuadPlane Note: This parameter configures the VTOL motors/behavior. It functions identically to the standard Copter parameter PSC_VELZ_I.

Please refer to the PSC_VELZ_I documentation for detailed tuning instructions.

Q_P_D_VEL_IMAX (PID Integral Maximum)

DEFAULT: 0	RANGE: null	UNITS: null

PID Integral Maximum

QuadPlane Note: This parameter configures the VTOL motors/behavior. It functions identically to the standard Copter parameter PSC_VELZ_IMAX.

Please refer to the PSC_VELZ_IMAX documentation for detailed tuning instructions.

Q_P_D_VEL_P (PID Proportional Gain)

DEFAULT: 0	RANGE: null	UNITS: null

PID Proportional Gain

QuadPlane Note: This parameter configures the VTOL motors/behavior. It functions identically to the standard Copter parameter PSC_VELZ_P.

Please refer to the PSC_VELZ_P documentation for detailed tuning instructions.

Q_P_JERK_D (PID Derivative Gain)

DEFAULT: 0	RANGE: null	UNITS: null

PID Derivative Gain

QuadPlane Note: This parameter configures the VTOL motors/behavior. It functions identically to the standard Copter parameter PSC_JERK_D.

Please refer to the PSC_JERK_D documentation for detailed tuning instructions.

Q_P_NE_POS_P (PID Proportional Gain)

DEFAULT: 0	RANGE: null	UNITS: null

PID Proportional Gain

QuadPlane Note: This parameter configures the VTOL motors/behavior. It functions identically to the standard Copter parameter PSC_POSXY_P.

Please refer to the PSC_POSXY_P documentation for detailed tuning instructions.

Q_P_NE_VEL_D (PID Derivative Gain)

DEFAULT: 0	RANGE: null	UNITS: null

PID Derivative Gain

QuadPlane Note: This parameter configures the VTOL motors/behavior. It functions identically to the standard Copter parameter PSC_VELXY_D.

Please refer to the PSC_VELXY_D documentation for detailed tuning instructions.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Q_P_NE_VEL_FF (FF FeedForward Gain)

DEFAULT: 0	RANGE: null	UNITS: null

FF FeedForward Gain

QuadPlane Note: This parameter configures the VTOL motors/behavior. It functions identically to the standard Copter parameter PSC_VELXY_FF.

Please refer to the PSC_VELXY_FF documentation for detailed tuning instructions.

Q_P_NE_VEL_FLTD (PID D term filter frequency in Hz)

DEFAULT: AC_PID_DFILT_HZ_DEFAULT	RANGE: null	UNITS: Hz

PID D term filter frequency in Hz

QuadPlane Note: This parameter configures the VTOL motors/behavior. It functions identically to the standard Copter parameter PSC_VELXY_FLTD.

Please refer to the PSC_VELXY_FLTD documentation for detailed tuning instructions.

Q_P_NE_VEL_FLTE (PID Error filter frequency in Hz)

DEFAULT: AC_PID_EFILT_HZ_DEFAULT	RANGE: null	UNITS: Hz

PID Error filter frequency in Hz

QuadPlane Note: This parameter configures the VTOL motors/behavior. It functions identically to the standard Copter parameter PSC_VELXY_FLTE.

Please refer to the PSC_VELXY_FLTE documentation for detailed tuning instructions.

Q_P_NE_VEL_I (PID Integral Gain)

DEFAULT: 0	RANGE: null	UNITS: null

PID Integral Gain

QuadPlane Note: This parameter configures the VTOL motors/behavior. It functions identically to the standard Copter parameter PSC_VELXY_I.

Please refer to the PSC_VELXY_I documentation for detailed tuning instructions.

Q_P_NE_VEL_IMAX (PID Integral Maximum)

DEFAULT: 0	RANGE: null	UNITS: null

PID Integral Maximum

QuadPlane Note: This parameter configures the VTOL motors/behavior. It functions identically to the standard Copter parameter PSC_VELXY_IMAX.

Please refer to the PSC_VELXY_IMAX documentation for detailed tuning instructions.

Q_P_NE_VEL_P (PID Proportional Gain)

DEFAULT: 0	RANGE: null	UNITS: null

PID Proportional Gain

QuadPlane Note: This parameter configures the VTOL motors/behavior. It functions identically to the standard Copter parameter PSC_VELXY_P.

Please refer to the PSC_VELXY_P documentation for detailed tuning instructions.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Q_RC_SPEED (VTOL Motor Update Rate)

DEFAULT: 490	RANGE: 50 490	UNITS: Hz

VTOL Motor Update Rate (Q_RC_SPEED)

Description

`Q_RC_SPEED` determines how fast the flight controller talks to your vertical lift motors.

A higher update rate allows the autopilot to make faster corrections, leading to a much more stable hover, especially in wind.

- **490 Hz:** Standard for high-speed PWM ESCs.
- **50 Hz:** Legacy analog servos/ESCs.

Tuning & Behavior

- **Default Value:** 490 Hz.
- **Recommendation:** Leave at **490 Hz** for all modern brushless ESCs.
- **DShot/CAN:** If using DShot or CAN ESCs, this parameter is ignored, as those protocols have their own high-speed timing.

Q_RTL_ALT (QuadPlane RTL Altitude)

DEFAULT: 15	RANGE: 0 100	UNITS: m

QuadPlane RTL Altitude (Q_RTL_ALT)

Description

`Q_RTL_ALT` sets the "Safe Height" for the final VTOL descent.

When the plane returns home, it will transition to VTOL mode and climb or descend to this altitude before starting the final vertical landing.

Tuning & Behavior

- **Default Value:** 15 meters.
- **Recommendation:** Set high enough to clear any trees or obstacles near your landing zone.

Q_RTL_ALT_MIN (QuadPlane RTL Minimum Altitude)

DEFAULT: 0	RANGE: 0 50	UNITS: m

QuadPlane RTL Minimum Altitude (Q_RTL_ALT_MIN)

Description

`Q_RTL_ALT_MIN` ensures the plane doesn't descend too early.

If `RTL_ALT` (the cruise altitude) is very high, the plane might try to descend steeply to reach `Q_RTL_ALT` (the landing altitude). This parameter puts a floor on that descent so the plane stays safely above the ground until it is close to home.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Q_RTL_MODE (VTOL RTL mode)

DEFAULT: 0	RANGE: 0 3

Q_RTL_MODE: VTOL RTL mode

Description

Controls whether Return to Launch (RTL) behaves like a plane or a multicopter.

Description

`Q_RTL_MODE` decides how your QuadPlane comes home when you trigger an RTL (or when failsafe triggers it).

- **0: Disabled (Default).** Standard Plane RTL. The aircraft circles the home point at `RTL_ALTITUDE`. It will **NOT land**. You must take manual control to land it.
- **1: Enabled (Hybrid).** Plane RTL to home. Once within `RTL_RADIUS`, it switches to VTOL (QRTL) and lands vertically.
- **3: QRTL Always.** Immediately switches to VTOL mode and flies home like a multicopter. (Very inefficient for long range!).

The Engineer's View

Option **1 (Hybrid)** is the standard "Fire and Forget" safety mode. It uses the efficiency of the wing to get home, then the precision of the quad motors to land.

Tuning & Behavior

- **Default Value:** 0
- **Recommendation:** Set to **1** for autonomous safety.
- **Danger:** If you set this to **3** and you are 5km away, the drone will try to fly 5km in hover mode. It will likely run out of battery and crash before it gets there.

Q_TAILSIT_ANGLE (Tailsitter Transition Angle)

DEFAULT: 45	RANGE: 10 80	UNITS: deg

Tailsitter Transition Angle (Q_TAILSIT_ANGLE)

Description

`Q_TAILSIT_ANGLE` sets the "Switching Point" during a transition.

As the tailsitter leans forward to gain speed, there is a moment where it stops being a "Drone" and starts being a "Plane." This parameter defines that halfway point.

Tuning & Behavior

- **Default Value:** 45 degrees.
- **Recommendation:** Set to **45** for most designs. If the plane has a very high stall speed, you might need a higher angle (e.g. **60**) to keep the VTOL motors helping for longer.

Q_TAILSIT_ANG_VT (Tailsitter VT Angle)

DEFAULT: 0	RANGE: -30 30	UNITS: deg

Tailsitter VT Angle (Q_TAILSIT_ANG_VT)

Description

`Q_TAILSIT_ANG_VT` sets the "Nose Point" during the initial hover.

Standard tailsitters hover with the nose at 90 degrees (straight up). This parameter allows you to add an offset if your fuselage is naturally tilted or if you want the drone to hover with a slight forward lean.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Q_TAILSIT_DSKLD (Tailsitter Disk Loading)

DEFAULT: 0	RANGE: 0 100	UNITS: kg/m^2

Tailsitter Disk Loading (Q_TAILSIT_DSKLD)

Description

Q_TAILSIT_DSKLD provides a physical reference for the aerodynamic forces on the rotors.

Disk loading is the weight of the drone divided by the total area of the spinning props. High disk loading (small props on heavy drone) means high-speed propwash, which requires different control surface gains than low disk loading.

Q_TAILSIT_ENABLE (Tailsitter Enable)

DEFAULT: 0	RANGE: 0 1

Tailsitter Enable (Q_TAILSIT_ENABLE)

Description

Q_TAILSIT_ENABLE activates the specialized physics and control models for **Tailsitter** aircraft.

A Tailsitter is a VTOL plane that sits on its tail for takeoff and landing. Unlike a standard QuadPlane (which stays horizontal), a Tailsitter must rotate its entire body by 90 degrees to transition between hover and forward flight.

Tuning & Behavior

- **0:** Disabled. Standard QuadPlane/Fixed-Wing logic.
- **1:** Enabled. Activates 90-degree frame rotation and tailsitter-specific mixing.
- **Reboot Required:** Yes.

Q_TAILSIT_GSCMAX (Tailsitter Max Gain Scaling)

DEFAULT: 1.0	RANGE: 0 5

Tailsitter Max Gain Scaling (Q_TAILSIT_GSCMAX)

Description

Q_TAILSIT_GSCMAX defines the upper limit of the gain multiplier. When the plane is hovering (0 airspeed), the surfaces move at their maximum possible deflection to maintain control.

Q_TAILSIT_GSCMIN (Tailsitter Min Gain Scaling)

DEFAULT: 0.4	RANGE: 0 1

Tailsitter Min Gain Scaling (Q_TAILSIT_GSCMIN)

Description

Q_TAILSIT_GSCMIN is the "Floor" for the gain multiplier. When the plane is flying fast, we reduce the surface movement (gain) to prevent vibration. This parameter ensures we don't reduce it so much that the drone becomes unresponsive.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Q_TAILSIT_GSCMSK (Tailsitter Gain Scaling Mask)

DEFAULT: 0	RANGE: 0 4294967295

Tailsitter Gain Scaling Mask (Q_TAILSIT_GSCMSK)

Description

`Q_TAILSIT_GSCMSK` protects against "Control Surface Flutter" during transitions.

In a hover, there is no airspeed, so the elevons/ailerons must move a lot to tilt the plane. As the plane speeds up, those same large movements would cause the plane to flip or vibrate violently. This parameter tells ArduPilot which surfaces need to be automatically "Dampened" as the plane gets faster.

- **Bit 0 (1):** Roll
- **Bit 1 (2):** Pitch
- **Bit 2 (4):** Yaw

Tuning & Behavior

- **Default Value:** 0.
- **Recommendation:** Usually set to **7** (Roll+Pitch+Yaw) for all tailsitters with control surfaces.

Q_TAILSIT_INPUT (Tailsitter Input Type)

DEFAULT: 0	RANGE: 0 1

Tailsitter Input Type (Q_TAILSIT_INPUT)

Description

`Q_TAILSIT_INPUT` defines how the pilot's sticks behave when the plane is standing on its tail (vertical).

- **0 (Body Frame):** The sticks follow the drone's perspective. If you push the "Roll" stick, the drone rotates around its own spine, which looks like a "Yaw" spin to someone standing on the ground.
- **1 (Earth Frame):** The sticks follow the pilot's perspective. "Roll" always tilts the drone left/right relative to the horizon, regardless of how the drone is rotated.

Tuning & Behavior

- **Default Value:** 0.
- **Recommendation:** Most pilots find **1 (Earth Frame)** much more intuitive for hovering, as it makes the tailsitter feel like a standard multirotor.

Q_TAILSIT_MIN_VO (Tailsitter Min VO Speed)

DEFAULT: 0	RANGE: 0 10	UNITS: m/s

Tailsitter Min VO Speed (Q_TAILSIT_MIN_VO)

Description

`Q_TAILSIT_MIN_VO` defines the "Blind Speed." Below this airspeed, the autopilot relies entirely on VTOL motors for stabilization because the control surfaces (ailerons/elevons) are not yet effective.

Q_TAILSIT_MOTMX (Tailsitter Motor Max Output)

DEFAULT: 0	RANGE: 0 1

Tailsitter Motor Max Output (Q_TAILSIT_MOTMX)

Description

`Q_TAILSIT_MOTMX` caps how much work the motors do to keep the plane level, ensuring there is enough power left to keep the plane in the air.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Q_TAILSIT_RAT_FW (Tailsitter FW Rate)

DEFAULT: 0	RANGE: 0 100	UNITS: deg/s

Tailsitter FW Rate (Q_TAILSIT_RAT_FW)

Description

`Q_TAILSIT_RAT_FW` defines the "Snap" into forward flight.

Once the plane has leaned over enough to gain lift from the wings, this parameter sets how fast it performs the final rotation into the horizontal level position.

Q_TAILSIT_RAT_VT (Tailsitter Transition Rate)

DEFAULT: 0	RANGE: 0 100	UNITS: deg/s

Tailsitter Transition Rate (Q_TAILSIT_RAT_VT)

Description

`Q_TAILSIT_RAT_VT` controls how "Fast" the plane tips over during a transition.

- **Low Value (e.g., 20):** Very slow, graceful tip-over. Safe but uses more battery.
- **High Value (e.g., 60):** Aggressive, fast transition. More efficient.

Tuning & Behavior

- **Default Value:** 0 (Uses `Q_TRANSITION_MS` calculation).

Q_TAILSIT_RLL_MX (Tailsitter Max Roll Rate)

DEFAULT: 0	RANGE: 0 180	UNITS: deg/s

Tailsitter Max Roll Rate (Q_TAILSIT_RLL_MX)

Description

`Q_TAILSIT_RLL_MX` sets the "Sensitivity" of the yaw axis (which becomes body-roll in hover) for tailsitters.

Q_TAILSIT_THR_VT (Tailsitter Transition Throttle)

DEFAULT: 0	RANGE: 0 100	UNITS: %

Tailsitter Transition Throttle (Q_TAILSIT_THR_VT)

Description

`Q_TAILSIT_THR_VT` prevents the plane from trying to fly forward if the motors aren't spinning fast enough to provide lift.

Tuning & Behavior

- **Default Value:** 0.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Q_TAILSIT_VFGAIN (Tailsitter Vector Forward Gain)

DEFAULT: 0	RANGE: 0 5

Tailsitter Vector Forward Gain (Q_TAILSIT_VFGAIN)

Description

`Q_TAILSIT_VFGAIN` defines how much "Forward Push" the VTOL motors provide while the plane is still mostly vertical.

Q_TAILSIT_VHGAIN (Tailsitter Vector Handling Gain)

DEFAULT: 0.5	RANGE: 0 5

Tailsitter Vector Handling Gain (Q_TAILSIT_VHGAIN)

Description

`Q_TAILSIT_VHGAIN` is the "Stiffness" of the hover control for tailsitters.

Tuning & Behavior

- **Default Value:** 0.5.
- **Recommendation:** Increase if the tailsitter feels "loose" or "mushy" when trying to stay vertical.

Q_TAILSIT_VHPOW (Tailsitter Vector Handling Power)

DEFAULT: 2.5	RANGE: 0 10

Tailsitter Vector Handling Power (Q_TAILSIT_VHPOW)

Description

`Q_TAILSIT_VHPOW` sets the "Authority" of the VTOL motors over the fuselage orientation.

Q_TAILSIT_VT_P_P (Tailsitter VT Pitch P)

DEFAULT: 0	RANGE: 0 10

Tailsitter VT Pitch P (Q_TAILSIT_VT_P_P)

Description

`Q_TAILSIT_VT_P_P` is the stiffness of the pitch axis during hover.

Q_TAILSIT_VT_R_P (Tailsitter VT Roll P)

DEFAULT: 0	RANGE: 0 10

Tailsitter VT Roll P (Q_TAILSIT_VT_R_P)

Description

`Q_TAILSIT_VT_R_P` is the "Stiffness" of the roll axis during hover.

- **0 (Default):** Use standard multicopter gains (`Q_A_RAT_RLL_P`).
- **Non-Zero:** Overrides the standard gains with this specific value for hover only.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Q_TAILSIT_VT_Y_P (Tailsitter VT Yaw P)

DEFAULT: 0	RANGE: 0 10

Tailsitter VT Yaw P (Q_TAILSIT_VT_Y_P)

Description

`Q_TAILSIT_VT_Y_P` is the stiffness of the yaw axis during hover.

Q_THROTTLE_EXPO (Pilot Throttle Exponential)

DEFAULT: 0	RANGE: 0 1

Pilot Throttle Exponential (Q_THROTTLE_EXPO)

Description

`Q_THROTTLE_EXPO` makes the throttle stick less sensitive around the middle (hover) position.

This helps the pilot maintain a steady hover by preventing small accidental stick movements from causing the drone to climb or descend.

Tuning & Behavior

- **0:** Linear.
- **Recommendation:** Match your `MOT_THST_EXPO` if you want a consistent feel.

Q_TILT_ENABLE (Tilt Rotor Enable)

DEFAULT: 0	RANGE: 0 1

Tilt Rotor Enable (Q_TILT_ENABLE)

Description

`Q_TILT_ENABLE` activates the logic for VTOLs that rotate their motors.

Unlike a standard QuadPlane (which has separate lift and push motors), a Tilt Rotor uses servos to point the motors up for hover and forward for flight.

Tuning & Behavior

- **0:** Disabled.
- **1:** Enabled.
- **Reboot Required:** Yes.

Q_TILT_FIX_ANGLE (Tilt Rotor Fixed Angle)

DEFAULT: 0	RANGE: 0 90	UNITS: deg

Tilt Rotor Fixed Angle (Q_TILT_FIX_ANGLE)

Description

`Q_TILT_FIX_ANGLE` sets the "Park Position" for the motors.

When you disarm, do you want the motors pointing up (90) to keep props out of the grass, or forward (0) to look streamlined? This parameter controls that behavior.

Tuning & Behavior

- **Default Value:** 0.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Q_TILT_FIX_GAIN (Tilt Rotor Fixed Gain)

DEFAULT: 0	RANGE: 0 1

Tilt Rotor Fixed Gain (Q_TILT_FIX_GAIN)

Description

`Q_TILT_FIX_GAIN` is rarely used but allows for softer holding force when parked.

Q_TILT_MASK (Tilt Rotor Mask)

DEFAULT: 0	RANGE: 0 65535

Q_TILT_MASK: Tilt Rotor Mask

Description

A bitmask selecting which motors can tilt to provide forward thrust.

Description

`Q_TILT_MASK` defines which of your vertical lift motors are capable of rotating forward to fly like a plane.

- **0: None (Standard QuadPlane).** Separate lift motors and a separate forward pusher motor.
- **Bitmask:** Selects specific motors. For example, if Motors 1 and 2 (Front Left/Right) tilt, but the rear motor is fixed, you would select bits 0 and 1 (Value = 3).

The Mathematics

$$\text{Tilt Motors} = \sum 2^{i-1}$$

Where i is the motor number (1-based).

- **Convergence (Tri):** Front two motors tilt. Motor 1 and 2. Value = + 2 = \mathbf{3}$.
- **V-22 Osprey (Bicopter):** Both motors tilt. Motor 1 and 2. Value = **3**.
- **Quad Tilt:** All 4 motors tilt. Value = **15**.

The Engineer's View

Used in `Tiltrotor::setup()`.
If a motor is in this mask, ArduPilot will actuate the corresponding tilt servo (assigned via `SERVOx_FUNCTION = 41`) to transition the motor from vertical to horizontal as airspeed increases.

Tuning & Behavior

- **Default Value:** 0
- **Recommendation:** Set strictly according to your physical airframe.
- **Vectored Yaw:** If all lifting motors tilt, you can often use `Q_TILT_TYPE` to enable "Vectored Yaw," using the tilt servos to steer the plane in hover instead of differential torque.

Q_TILT_MAX (Tilt Rotor Maximum Angle)

DEFAULT: 45	RANGE: 0 90	UNITS: deg

Tilt Rotor Maximum Angle (Q_TILT_MAX)

Description

`Q_TILT_MAX` defines the "Up" position for the tilt mechanism.

Usually, this is 90 degrees (vertical) for hover. However, some airframes (like the E-Flite Convergence) might have a different geometry.

Tuning & Behavior

- **Default Value:** 45 degrees.
- **Recommendation:** Set to **90** for standard tilt-rotors.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY

Q_TILT_RATE_DN (Tilt Rotor Rate Down)

DEFAULT: 10	RANGE: 10 300	UNITS: deg/s

Tilt Rotor Rate Down (Q_TILT_RATE_DN)

Description

`Q_TILT_RATE_DN` controls how fast the motors rotate forward.

This directly affects acceleration. A lower value gives a smoother, more gradual transition.

Tuning & Behavior

- **Default Value:** 10 deg/s.
- **Recommendation:** Keep lower than `Q_TILT_RATE_UP` to prevent altitude loss during transition.

Q_TILT_RATE_UP (Tilt Rotor Rate Up)

DEFAULT: 40	RANGE: 10 300	UNITS: deg/s

Tilt Rotor Rate Up (Q_TILT_RATE_UP)

Description

`Q_TILT_RATE_UP` controls how fast the motors snap to vertical when you switch to QHOVER.

If this is too fast, the plane might balloon up (gain altitude rapidly). If too slow, the plane might stall before the motors can generate lift.

Tuning & Behavior

- **Default Value:** 40 deg/s.

Q_TILT_TYPE (Tilt Rotor Type)

DEFAULT: 0	RANGE: 0 3

Tilt Rotor Type (Q_TILT_TYPE)

Description

`Q_TILT_TYPE` tells ArduPilot how your tilting mechanism works.

- **0: Continuous.** A standard servo that can move smoothly to any angle between 0 and 90.
- **1: Binary.** A retract mechanism that only has two positions (Up/Down).
- **2: Vectored Yaw.** The tilt servos are also used for yaw control in forward flight (like a V-22 Osprey).

Tuning & Behavior

- **Default Value:** 0.

Q_TILT_WING_FLAP (Tilt Wing Flap)

DEFAULT: 0	RANGE: 0 100	UNITS: %

Tilt Wing Flap (Q_TILT_WING_FLAP)

Description

`Q_TILT_WING_FLAP` is for **Tilt-Wing** aircraft (where the entire wing rotates, not just the motors).

As the wing tilts up, the ailerons become less effective for roll control. This parameter mixes in some "Flap" to increase lift and control authority during the transition.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Q_TILT_YAW_ANGLE (Tilt Rotor Yaw Angle)

DEFAULT: 0	RANGE: 0 30	UNITS: deg

Tilt Rotor Yaw Angle (Q_TILT_YAW_ANGLE)

Description

`Q_TILT_YAW_ANGLE` enables "Vectored Yaw" for tilt-rotors.

If your tilt servos are strong enough, you can differential-tilt them (left motor up, right motor down) to yaw the aircraft without using a rudder. This parameter limits how much they can move for this purpose.

Tuning & Behavior

- **Default Value:** 0 (Disabled).
- **Recommendation:** Set to **10-15 degrees** if your airframe supports it (Requires `Q_TILT_TYPE` = 2).

Q_TKOFF_ARSP_LIM (QuadPlane Takeoff Airspeed Limit)

DEFAULT: 0	RANGE: 0 20	UNITS: m/s

QuadPlane Takeoff Airspeed Limit (Q_TKOFF_ARSP_LIM)

Description

`Q_TKOFF_ARSP_LIM` is a safety check for high-wind operations.

If you try to take off vertically in a hurricane, the wing will generate lift even though you are in Copter mode. This can confuse the controller. This parameter sets a limit: if the airspeed is higher than this value while taking off, the autopilot will restrict how far it leans back (to fight the wind) or abort the takeoff entirely to prevent a loss of control.

Tuning & Behavior

- **Default Value:** 0 (Disabled).
- **Recommendation:** Set to **15 m/s** if you operate in gusty environments.

Q_TKOFF_FAIL_SCL (QuadPlane Takeoff Failure Scaler)

DEFAULT: 0	RANGE: 0 5

QuadPlane Takeoff Failure Scaler (Q_TKOFF_FAIL_SCL)

Description

`Q_TKOFF_FAIL_SCL` adjusts how long the drone tries to climb before giving up.

If the drone is at full throttle but not gaining altitude (e.g., battery is dead or payload is too heavy), the autopilot will disarm to save the motors. This parameter extends that timeout.

Tuning & Behavior

- **Default Value:** 0 (Standard timeout).
- **High Value:** Allows longer struggle before disarming.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Q_TKOFF_RPM_MAX (QuadPlane Takeoff RPM Max)

DEFAULT: 0	RANGE: 0 20000	UNITS: RPM

QuadPlane Takeoff RPM Max (Q_TKOFF_RPM_MAX)

Description

`Q_TKOFF_RPM_MAX` caps the motor speed during launch.

This is useful for noise abatement or to protect fragile props from over-spinning during the high-power takeoff surge.

Tuning & Behavior

- **Default Value:** 0 (Disabled).

Q_TKOFF_RPM_MIN (QuadPlane Takeoff RPM Min)

DEFAULT: 0	RANGE: 0 10000	UNITS: RPM

QuadPlane Takeoff RPM Min (Q_TKOFF_RPM_MIN)

Description

`Q_TKOFF_RPM_MIN` is a safety check for RPM-controlled motors.

The autopilot will not release the landing gear or start the mission until the VTOL motors report at least this speed.

Tuning & Behavior

- **Default Value:** 0 (Disabled).

Q_TRANSITION_MS (Transition time)

DEFAULT: 5000	RANGE: 0 30000	UNITS: ms

Transition time (Q_TRANSITION_MS)

Description

`Q_TRANSITION_MS` is a safety buffer.

When your QuadPlane accelerates forward, it waits until it reaches `ARSPD_FBW_MIN` (Minimum Airspeed). Once it hits that speed, the wing *should* be flying. However, to be safe, the autopilot keeps the VTOL motors running for this extra amount of time to ensure the wing is fully generating lift before shutting them down.

- **5000ms (Default):** The quad motors stay on for 5 seconds after reaching minimum airspeed.

The Engineer's View

Used in `SLT_Transition::update()`.
During this phase (`TRANSITION_TIMER`), the quad motors' throttle is linearly ramped down from hover power to zero. This provides a smooth handoff from rotor-borne lift to wing-borne lift.

Tuning & Behavior

- **Default Value:** 5000 ms
- **Recommendation:**
  - **New Airframes:** Keep at **5000** or higher to be safe.
  - **Tuned Aircraft:** You can reduce this to **2000** or **3000** to save battery and reduce drag during transition.
  - **Warning:** Setting this too low (e.g., 0) can cause the plane to stall immediately after the transition if the airspeed sensor reading was momentarily noisy or if the wing wasn't truly ready.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Q_TRANS_DECEL (QuadPlane Transition Deceleration)

DEFAULT: 2.0	RANGE: 0.5 5.0	UNITS: m/s^2

QuadPlane Transition Deceleration (Q_TRANS_DECEL)

Description

`Q_TRANS_DECEL` determines how hard the drone "Brakes" when stopping.

When you switch from FBWA to QHOVER, the plane pitches up to slow down. This parameter sets how aggressively it pitches up.

Tuning & Behavior

- **Default Value:** 2.0 m/s^2.
- **Higher Value:** Faster stop, but the nose will pitch up sharply (Cobra maneuver).
- **Lower Value:** Smoother, gradual stop, but takes more distance.

Q_TRANS_FAIL (QuadPlane Transition Failure Time)

DEFAULT: 0	RANGE: 0 20	UNITS: s

QuadPlane Transition Failure Time (Q_TRANS_FAIL)

Description

`Q_TRANS_FAIL` is a safety timer.

If the plane tries to transition to forward flight but fails to reach the minimum airspeed (e.g. headwind too strong, motor failure), this timer triggers a failsafe action.

Tuning & Behavior

- **Default Value:** 0 (Disabled).
- **Recommendation:** Set to **10 seconds** for safety.

Q_TRANS_FAIL_ACT (QuadPlane Transition Failure Action)

DEFAULT: 0	RANGE: 0 1

QuadPlane Transition Failure Action (Q_TRANS_FAIL_ACT)

Description

`Q_TRANS_FAIL_ACT` decides what to do if `Q_TRANS_FAIL` triggers.

- **0:** Warn only (Text message on GCS).
- **1:** QLAND (Immediately land vertically).

Tuning & Behavior

- **Default Value:** 0.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Q_TRAN_PIT_MAX (QuadPlane Transition Pitch Max)

DEFAULT: 3	RANGE: 0 30	UNITS: deg

QuadPlane Transition Pitch Max (Q_TRAN_PIT_MAX)

Description

`Q_TRAN_PIT_MAX` keeps the nose down during acceleration.

When the pusher motor starts, the plane naturally wants to pitch up. This parameter clamps the pitch angle to ensure the plane stays level and accelerates efficiently.

Tuning & Behavior

- **Default Value:** 3 degrees.
- **Recommendation:** Increase to **10-15** for faster acceleration if your airframe has plenty of power.

Q_TRIM_PITCH (VTOL Pitch Trim)

DEFAULT: 0	RANGE: -20 20	UNITS: deg

VTOL Pitch Trim (Q_TRIM_PITCH)

Description

`Q_TRIM_PITCH` defines the "Neutral" attitude in hover.

If your QuadPlane naturally drifts forward while hovering, you can set a small positive trim to tilt the nose up slightly and stay stationary.

Tuning & Behavior

- **Default Value:** 0 degrees.
- **Recommendation:** Use this to ensure the drone stays stationary in calm air with zero stick input.

Q_VFWD_ALT (QuadPlane Forward Motor Altitude)

DEFAULT: 0	RANGE: 0 100	UNITS: m

QuadPlane Forward Motor Altitude (Q_VFWD_ALT)

Description

`Q_VFWD_ALT` prevents the forward pusher motor from spinning up while the drone is still taking off or low to the ground.

Tuning & Behavior

- **Default Value:** 0 (Active at all altitudes).
- **Recommendation:** Set to **3 or 5 meters** to ensure the drone is clear of grass and ground personnel before the main propeller starts.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY

Q_VFWD_GAIN (QuadPlane Forward Motor Gain)

DEFAULT: 0	RANGE: 0 0.5

QuadPlane Forward Motor Gain (Q_VFWD_GAIN)

Description

`Q_VFWD_GAIN` allows the drone to use its "Plane Propeller" to help it stay still in the wind.

Normally, a drone stays still by tilting into the wind. If the wind is very strong, the drone must tilt significantly. This parameter allows the autopilot to spin up the forward-facing motor to push against the wind, allowing the drone to stay more level while hovering.

Tuning & Behavior

- **Default Value:** 0 (Disabled).
- **Recommendation:** Set to **0.05 - 0.1** to improve hover stability in high winds.
- **Effect:** Reduces the pitch angle required to hold position.

Q_WP_ACCEL (Waypoint Acceleration)

DEFAULT: WPNAV_ACCELERATION	RANGE: 50 500	UNITS: cm/s/s

Waypoint Acceleration

QuadPlane Note: This parameter configures the VTOL motors/behavior. It functions identically to the standard Copter parameter WPNAV_ACCEL.

Please refer to the WPNAV_ACCEL documentation for detailed tuning instructions.

Q_WP_ACCEL_C (Waypoint Cornering Acceleration)

DEFAULT: 0.0	RANGE: 0 500	UNITS: cm/s/s

Waypoint Cornering Acceleration

QuadPlane Note: This parameter configures the VTOL motors/behavior. It functions identically to the standard Copter parameter WPNAV_ACCEL_C.

Please refer to the WPNAV_ACCEL_C documentation for detailed tuning instructions.

Q_WP_ACCEL_Z (Waypoint Vertical Acceleration)

DEFAULT: WPNAV_WP_ACCEL_Z_DEFAULT	RANGE: 50 500	UNITS: cm/s/s

Waypoint Vertical Acceleration

QuadPlane Note: This parameter configures the VTOL motors/behavior. It functions identically to the standard Copter parameter WPNAV_ACCEL_Z.

Please refer to the WPNAV_ACCEL_Z documentation for detailed tuning instructions.

Q_WP_JERK (Waypoint Jerk)

DEFAULT: 1.0f	RANGE: 1 20	UNITS: m/s^3

Waypoint Jerk

QuadPlane Note: This parameter configures the VTOL motors/behavior. It functions identically to the standard Copter parameter WPNAV_JERK.

Please refer to the WPNAV_JERK documentation for detailed tuning instructions.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Q_WP_RADIUS (Circular Fence Radius)

DEFAULT: AC_FENCE_CIRCLE_RADIUS_DEFAULT	RANGE: 30 10000	UNITS: m

Circular Fence Radius

QuadPlane Note: This parameter configures the VTOL motors/behavior. It functions identically to the standard Copter parameter WPNAV_RADIUS.

Please refer to the WPNAV_RADIUS documentation for detailed tuning instructions.

Q_WP_RFND_USE (Waypoint missions use rangefinder for terrain following)

DEFAULT: 1	RANGE: null	UNITS: null

Waypoint missions use rangefinder for terrain following

QuadPlane Note: This parameter configures the VTOL motors/behavior. It functions identically to the standard Copter parameter WPNAV_RFND_USE.

Please refer to the WPNAV_RFND_USE documentation for detailed tuning instructions.

Q_WP_SPEED (Loiter Horizontal Maximum Speed)

DEFAULT: LOITER_SPEED_DEFAULT	RANGE: 20 3500	UNITS: cm/s

Loiter Horizontal Maximum Speed

QuadPlane Note: This parameter configures the VTOL motors/behavior. It functions identically to the standard Copter parameter WPNAV_SPEED.

Please refer to the WPNAV_SPEED documentation for detailed tuning instructions.

Q_WP_SPEED_DN (Waypoint Descent Speed Target)

DEFAULT: WPNAV_WP_SPEED_DOWN	RANGE: 10 500	UNITS: cm/s

Waypoint Descent Speed Target

QuadPlane Note: This parameter configures the VTOL motors/behavior. It functions identically to the standard Copter parameter WPNAV_SPEED_DN.

Please refer to the WPNAV_SPEED_DN documentation for detailed tuning instructions.

Q_WP_SPEED_UP (Waypoint Climb Speed Target)

DEFAULT: WPNAV_WP_SPEED_UP	RANGE: 10 1000	UNITS: cm/s

Waypoint Climb Speed Target

QuadPlane Note: This parameter configures the VTOL motors/behavior. It functions identically to the standard Copter parameter WPNAV_SPEED_UP.

Please refer to the WPNAV_SPEED_UP documentation for detailed tuning instructions.

Q_WP_TER_MARGIN (Avoidance distance margin in GPS modes)

DEFAULT: 2.0f	RANGE: 1 10	UNITS: m

Avoidance distance margin in GPS modes

QuadPlane Note: This parameter configures the VTOL motors/behavior. It functions identically to the standard Copter parameter WPNAV_TER_MARGIN.

Please refer to the WPNAV_TER_MARGIN documentation for detailed tuning instructions.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Q_WVANE_ANG_MIN (QuadPlane Weathervane Min Angle)

DEFAULT: 1	RANGE: 0 10	UNITS: deg

QuadPlane Weathervane Min Angle (Q_WVANE_ANG_MIN)

Description

`Q_WVANE_ANG_MIN` creates a "Deadband" for the weathervane logic.

If the wind is only 2 degrees off the nose, you probably don't want the motors to spin up to correct it. This parameter prevents "Hunting" in variable winds.

Tuning & Behavior

- **Default Value:** 1 degree.

Q_WVANE_ENABLE (QuadPlane Weathervaning Enable)

DEFAULT: 0	RANGE: 0 4

QuadPlane Weathervaning Enable (Q_WVANE_ENABLE)

Description

`Q_WVANE_ENABLE` allows the drone to passively turn its nose into the wind while hovering.

Weathervaning reduces the workload on the roll/pitch motors. By pointing into the wind, the drone is more aerodynamic and uses less battery.

Tuning & Behavior

- **0:** Disabled.
- **1:** Nose into wind (Yaw only).
- **2:** Nose or Tail into wind (Yaw only).
- **3:** Nose into wind (Yaw + Roll).
- **4:** Nose or Tail into wind (Yaw + Roll).

Recommendation

Use **1** for standard QuadPlanes. Use **0** if you need precise heading control for photography.

Q_WVANE_GAIN (QuadPlane Weathervaning Gain)

DEFAULT: 1	RANGE: 0.1 5

QuadPlane Weathervaning Gain (Q_WVANE_GAIN)

Description

`Q_WVANE_GAIN` determines how aggressively the drone turns to face the wind.

Tuning & Behavior

- **Default Value:** 1.
- **High Value:** Snappy response, but might oscillate if the wind is gusty.
- **Low Value:** Slow, gentle turn into the wind.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Q_WVANE_HGT_MIN (QuadPlane Weathervane Min Height)

DEFAULT: 0	RANGE: 0 50	UNITS: m

QuadPlane Weathervane Min Height (Q_WVANE_HGT_MIN)

Description

`Q_WVANE_HGT_MIN` keeps the heading locked during takeoff.

You generally want the drone to launch straight up without twisting. Once it clears the ground obstacles (e.g. 5 meters up), it is safe to turn into the wind.

Tuning & Behavior

- **Default Value:** 0.
- **Recommendation:** Set to **2 or 3 meters**.

Q_WVANE_LAND (QuadPlane Weathervane Land)

DEFAULT: 0	RANGE: 0 1

QuadPlane Weathervane Land (Q_WVANE_LAND)

Description

`Q_WVANE_LAND` allows the drone to pivot into the wind while descending for a landing.

- **0:** Disabled (Maintains approach heading).
- **1:** Enabled (Turns into wind).

Tuning & Behavior

- **Default Value:** 0.
- **Recommendation:** Enable this if you fly in strong winds, as landing sideways can tip the drone over.

Q_WVANE_OPTIONS (QuadPlane Weathervane Options)

DEFAULT: 0	RANGE: 0 1

QuadPlane Weathervane Options (Q_WVANE_OPTIONS)

Description

`Q_WVANE_OPTIONS` enables specialized behaviors.

- **Bit 0:** Pitch into wind. Instead of yawing, the drone will pitch its nose down into the wind. This is useful for tailsitters or tilt-rotors that generate lift from their wings even in hover.

Tuning & Behavior

- **Default Value:** 0.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Q_WVANE_SPD_MAX (QuadPlane Weathervane Max Speed)

DEFAULT: 0	RANGE: 0 10	UNITS: m/s

QuadPlane Weathervane Max Speed (Q_WVANE_SPD_MAX)

Description

`Q_WVANE_SPD_MAX` disables weathervaning if the drone is moving fast.

If you are repositioning the drone (e.g. loitering at 5 m/s), you want it to point in the direction of travel, not the direction of the wind. This parameter ensures the navigation controller has priority over the wind controller when moving.

Tuning & Behavior

- **Default Value:** 0 (Disabled/Always Active).
- **Recommendation:** Set to **1 or 2 m/s**.

Q_WVANE_TAKEOFF (QuadPlane Weathervane Takeoff)

DEFAULT: 0	RANGE: 0 1

QuadPlane Weathervane Takeoff (Q_WVANE_TAKEOFF)

Description

`Q_WVANE_TAKEOFF` allows the drone to yaw into the wind *while* it is climbing out.

- **0:** Disabled (Heading locked until takeoff complete).
- **1:** Enabled.

Tuning & Behavior

- **Default Value:** 0.

Q_WVANE_VELZ_MAX (QuadPlane Weathervane Max Vertical Speed)

DEFAULT: 0	RANGE: 0 5	UNITS: m/s

QuadPlane Weathervane Max Vertical Speed (Q_WVANE_VELZ_MAX)

Description

`Q_WVANE_VELZ_MAX` prevents yawing during fast climbs or descents.

If you are rocketing up to altitude, you usually want to keep the heading locked. This parameter disables the wind-turning logic if the vertical speed exceeds the threshold.

Tuning & Behavior

- **Default Value:** 0 (Disabled).

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Q_YAW_RATE_MAX (Maximum Yaw Rate)

DEFAULT: 90	RANGE: 10 360	UNITS: deg/s

Maximum Yaw Rate (Q_YAW_RATE_MAX)

Description

`Q_YAW_RATE_MAX` sets the "Spin Speed" limit.

It determines how fast the drone will rotate when you push the yaw stick to the full deflection.

Tuning & Behavior

- **Default Value:** 90 deg/s.
- **High Value (180+):** Very fast rotation. Good for agile maneuvers but can cause EKF issues if the compass cannot keep up.
- **Low Value (45):** Slow, scale-like rotation.
- **Recommendation:** 90 deg/s is a safe and standard limit.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **QUAL** parameter group (specifically `QUAL_MIN` or `FLOW_QUAL_MIN`) defines the confidence requirements for visual sensors.

Key Concepts

1. Optical Flow Quality

Optical Flow sensors (like the PX4Flow or HereFlow) output a "Quality" score (0-255) for every sample. This score indicates how many distinct features the camera can see and track on the ground.

- `QUAL_MIN`: The minimum acceptable quality score.
- **Action:** If the live quality drops below this threshold (due to low light or flying over water/sand), the autopilot will reject the flow data and trigger a failsafe (usually falling back to AltHold mode).

Developer Notes

- **Library:** `libraries/AP_OpticalFlow`.
- **Tuning:** If you find your drone constantly dropping out of Loiter indoors, check the logs for `QUAL`. You may need to lower the threshold or improve lighting.

QUAL_MIN (FlowHold Flow quality minimum)

DEFAULT: 10	RANGE: 0 255

QUAL_MIN: FlowHold Flow quality minimum

Description

Minimum flow quality to use flow position hold.

Values

- **Range:** 0 to 255
- **Default:** 10

Description

This parameter sets the reliability threshold for the Optical Flow sensor in **FlowHold** mode.

- **Function:** The optical flow sensor reports a "quality" metric (0-255) with each reading, indicating how confident it is in the surface texture it sees.
- **Below Threshold:** If quality drops below `QUAL_MIN` (e.g., flying over water, low light, or featureless concrete), FlowHold stops trying to hold position and behaves like **AltHold** (drift with wind) to prevent runaway accumulation of errors.
- **Above Threshold:** FlowHold actively fights drift using the flow sensor data.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **QWIK** parameter group configures the **QuickTune** Lua script (and integrated library in newer versions). QuickTune is a faster, more automated alternative to the standard AutoTune mode for multicopters.

While AutoTune requires the pilot to toggle a switch and wait for individual axis twitches, QuickTune can automatically find gains while the pilot is flying naturally or during a mission.

Key Concepts

1. Oscillation Detection

QuickTune monitors the gyro data for high-frequency oscillations that indicate a PID gain is too high.

- `QWIK_ENABLE`: Master switch.
- `QWIK_AUTO_FILTER`: If enabled, the system also tries to set the optimal low-pass filter frequencies based on the noise profile.

2. Auto-Save (`QWIK_AUTO_SAVE`)

Determines if the new gains are applied permanently.

- **0:** Trial only.
- **1:** Save to EEPROM automatically once the tune converges.

Parameter Breakdown

- `QWIK_AXES`: Bitmask of axes to tune (Roll, Pitch, Yaw).
- `QWIK_GAIN_MARGIN`: The safety margin applied once an oscillation is detected (e.g., set gain to 60% of the oscillation point).

Integration Guide

1. **Script:** Ensure the `quiktune.lua` script is in the `scripts` folder on the SD card.
2. **Enable:** Set `QWIK_ENABLE = 1`.
3. **Tuning:** Fly the vehicle in Loiter or AltHold. Perform some rapid stick movements.
4. **Verify:** Check the GCS messages for "QuickTune: Roll P set to X."

Developer Notes

- **Library:** `libraries/AP_Scripting` (usually as a Lua implementation).

QWIK_ANGLE_MAX (Quicktune Max Attitude Error)

DEFAULT: 10	RANGE: 0 45	UNITS: deg

Quicktune Max Attitude Error (QWIK_ANGLE_MAX)

Description

`QWIK_ANGLE_MAX` is the "Emergency Kill Switch" for the tuning process.

If the drone's actual tilt angle differs from your stick command by more than this many degrees (indicating that the drone is losing control or oscillating wildly), Quicktune will instantly abort the tune and revert to your original safe gains.

Tuning & Behavior

- **Default Value:** 10 degrees.
- **Recommendation:** If you get "Attitude Error - Aborting" messages during tuning even when the drone seems stable, try increasing this to **15 or 20**.
- **Safety:** Do not set this too high (e.g. >30), as it protects the drone from a bad tune that could lead to a crash.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





QWIK_AUTO_FILTER (Quicktune Auto Filter Enable)

DEFAULT: 1	RANGE: 0 1

Quicktune Auto Filter Enable (QWIK_AUTO_FILTER)

Description

`QWIK_AUTO_FILTER` automates one of the most confusing parts of tuning: the PID filters.

For a sharp, high-performance tune, the PID filters (`FLTT`, `FLTD`, `FLTE`) should be related to the `INS_GYRO_FILTER`. When this parameter is enabled, Quicktune will automatically update your filter settings at the start of the tuning process to ensure they are optimal for your current hardware configuration.

Tuning & Behavior

- **Default Value:** 1 (Enabled).
- **Recommendation:** Keep enabled. It ensures that your filter settings evolve alongside your PID gains for a balanced tune.

QWIK_AUTO_SAVE (Quicktune Auto Save Time)

DEFAULT: 0	RANGE: 0 60	UNITS: s

Quicktune Auto Save Time (QWIK_AUTO_SAVE)

Description

`QWIK_AUTO_SAVE` allows the autopilot to permanently store the new PID gains without requiring a separate "Save" command from the pilot.

Normally, you have to toggle a switch on your remote to save the tune. If you set this parameter to **10**, the drone will wait for 10 seconds after it finishes tuning all axes and then automatically save the values to memory.

- **0 (Default):** Auto-save disabled. You must save manually using an RC switch or the Ground Station.

QWIK_AXES (Quicktune Axes)

DEFAULT: 7	RANGE: 0 7

Quicktune Axes (QWIK_AXES)

Description

`QWIK_AXES` defines which flight axes the Quicktune assistant will attempt to optimize.

- **Bit 0 (1):** Roll
- **Bit 1 (2):** Pitch
- **Bit 2 (4):** Yaw

Tuning & Behavior

- **Default Value:** 7 (All axes).
- **Recommendation:** Usually, it is best to tune all three axes in one flight. However, if you are experiencing issues with a specific axis (like a vibrating tail), you can uncheck Yaw to focus only on Roll and Pitch.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY



QWIK_DOUBLE_TIME (Quicktune Doubling Time)

DEFAULT: 10	RANGE: 5 20	UNITS: s

Quicktune Doubling Time (QWIK_DOUBLE_TIME)

Description

`QWIK_DOUBLE_TIME` controls the "Aggression" of the tuning process.

It determines how fast the autopilot increases the PID gains while searching for the oscillation point.

- **Low Value (5s):** Fast tune. The gains increase rapidly. Good for experienced pilots and stable frames.
- **High Value (20s):** Slow tune. Safer for large, heavy, or expensive drones where a sudden oscillation could be dangerous.

Tuning & Behavior

- **Default Value:** 10 seconds.
- **Recommendation:** If you get "Attitude Error - Aborting" messages during tuning, increase this value to **15 or 20** to give the drone more time to stabilize between gain steps.

QWIK_ENABLE (Quicktune Enable)

DEFAULT: 0	RANGE: 0 1

Quicktune Enable (QWIK_ENABLE)

Description

`QWIK_ENABLE` activates an automated, in-flight PID tuning assistant.

Quicktune is a C++ implementation of the popular "QuickTune" Lua script. It works by slowly increasing your drone's P and D gains while you fly until it detects a small oscillation. It then backs off the gain to a safe margin. This allows you to get a high-performance tune in minutes without needing to use the full AutoTune mode.

Tuning & Behavior

- **Default Value:** 0 (Disabled).
- **Usage:** Once enabled, you must assign an RC channel to `RCx_OPTION = 129` (QuickTune) to start/stop the tuning process in flight.
- **Reboot Required:** Yes.

QWIK_GAIN_MARGIN (Quicktune Gain Margin)

DEFAULT: 60	RANGE: 20 80	UNITS: %

Quicktune Gain Margin (QWIK_GAIN_MARGIN)

Description

`QWIK_GAIN_MARGIN` determines how "Conservative" or "Aggressive" the final tune will be.

During the tuning process, Quicktune increases your gains until the drone starts to shake (oscillate). It then takes that "Limit" value and reduces it by this percentage to find a safe, stable flying gain.

- **Higher Value (80%):** More conservative. The final gains will be lower and the drone will feel smoother but less "Locked in."
- **Lower Value (40%):** More aggressive. The final gains will be closer to the oscillation point, resulting in a very sharp, high-performance feel.

Tuning & Behavior

- **Default Value:** 60%.
- **Recommendation:** Start with **60%**. If the drone still feels a bit loose after tuning, try **50%**.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





QWIK_OPTIONS (Quicktune options)

DEFAULT: 0	RANGE: null	UNITS: Bitmask

Quicktune options (QWIK_OPTIONS)

Description

This parameter provides additional configuration for the Quicktune automated tuning system. Its primary use is to simplify the control logic for pilots who prefer using a standard 2-position toggle switch rather than the default 3-position behavior.

The Mathematics

The parameter is a bitmask (B):

- **Bit 0 (Value 1): Use Two-Position Switch.**
  - **Disabled (0):** Default 3-position logic.
    - Low: Abort/Revert.
    - Mid: Start Tuning.
    - High: Save Tune.
  - **Enabled (1):** Simplified 2-position logic.
    - Low: Disable/Abort.
    - High: Start Tuning.
    - *(Note: Requires `QWIK_AUTO_SAVE` to be configured to save the results automatically).*

The Engineer's View

In `AP_Quicktune::update()` (`libraries/AP_Quicktune/AP_Quicktune.cpp`), the logic branches based on the `OPTIONS_TWO_POSITION` flag.
If Bit 0 is set:

1. The `sw_pos_tune` target is shifted to `SwitchPos::HIGH`.
2. The `sw_pos_save` target is set to `SwitchPos::NONE`, meaning the switch itself cannot trigger a save.
3. The system relies on the `tune_done_time_ms` timer and the `auto_save` parameter to commit the learned gains to EEPROM after the tuning session successfully completes.

Tuning & Behavior

- **Default Value:** 0 (Standard 3-position switch logic).
- **When to Use:** If you are short on auxiliary channels or prefer a simple "Tune ON / Tune OFF" switch.
- **Safety Hint:** If using the 2-position mode, ensure `QWIK_AUTO_SAVE` is set to a non-zero value (e.g., 10 seconds), otherwise your new tune will be lost the moment you flip the switch back to Low.

QWIK_OSC_SMAX (Quicktune Oscillation Threshold)

DEFAULT: 4	RANGE: 1 10

Quicktune Oscillation Threshold (QWIK_OSC_SMAX)

Description

`QWIK_OSC_SMAX` defines how much "Shaking" the autopilot allows before it decides the gain is too high.

The system monitors the "Slew Rate" (how fast the motors are changing speed) to detect the onset of high-frequency oscillations.

- **Lower Value (1-2):** Very sensitive. Quicktune will stop increasing gains at the first hint of a vibration.
- **Higher Value (6-10):** Less sensitive. Allows the drone to shake more violently before backing off.

Tuning & Behavior

- **Default Value:** 4.
- **Recommendation:** Leave at **4**. Only increase this if you find that Quicktune is stopping prematurely due to noise from imbalanced propellers or a loose frame.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





QWIK_REDUCE_MAX (Quicktune Max Gain Reduction)

DEFAULT: 20	RANGE: 0 100	UNITS: %

Quicktune Max Gain Reduction (QWIK_REDUCE_MAX)

Description

`QWIK_REDUCE_MAX` is a "Safety Floor" for the tuning process.

If your drone already flies reasonably well, you don't want a "False Positive" oscillation detection (caused by wind or a bump) to trick the system into dropping your gains to dangerously low levels. This parameter prevents Quicktune from lowering your original gains by more than the specified percentage.

- **Default:** 20%.
- **0:** No reduction allowed. Quicktune can only increase your gains or leave them as they are.

QWIK_RP_PI_RATIO (Quicktune roll/pitch PI ratio)

DEFAULT: 1.0	RANGE: 1.0 2.0	UNITS: Ratio

Quicktune roll/pitch PI ratio (QWIK_RP_PI_RATIO)

Description

This parameter defines the fixed relationship (ratio) between the Proportional (P) and Integral (I) gains that Quicktune maintains while it searches for the optimal tune for Roll and Pitch.

In many flight control systems, the P and I gains are "coupled"—as you increase the responsiveness (P), you must also increase the "correction force over time" (I) to maintain a balanced feel and prevent drift. This parameter automates that coupling, ensuring that as Quicktune pushes for higher performance, it keeps the integrator balanced.

The Mathematics

During the tuning process, whenever Quicktune adjusts the P gain (K_p), it automatically calculates the new I gain (K_i) using this ratio (γ):

$$K_i = \frac{K_p}{\gamma}$$

Where γ is `QWIK_RP_PI_RATIO`.

For example, if the ratio is **1.0**, the I gain will always equal the P gain. If the ratio is **2.0**, the I gain will be exactly half of the P gain.

The Engineer's View

In `AP_Quicktune::adjust_gain()` (`libraries/AP_Quicktune/AP_Quicktune.cpp`):

1. The code identifies when a P gain is being modified.
2. It checks if FeedForward (`FF`) is present. If `FF > 0`, the system assumes a more advanced control model where `I` is matched to `FF` for a 1-second time constant, and this ratio is ignored.
3. If no FF is used, it reads `rp_pi_ratio` and performs the division: `set_param_value(iname, value/pi_ratio)`.
4. This ensures that the "Integrator Time Constant" remains consistent throughout the entire automated tuning session, preventing the aircraft from becoming sluggish or "drifty" while the P gain is being swept.

Tuning & Behavior

- **Default Value:** 1.0 (Balanced).
- **Range:** 1.0 to 2.0.
- **Effect of Increasing:** Results in a lower relative I gain. This can make the vehicle feel less "locked-in" but reduces the risk of low-frequency oscillations.
- **Effect of Decreasing:** Results in a higher relative I gain, providing stronger resistance to external disturbances (like wind) but potentially causing slow "wandering" if set too low.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY

QWIK_YAW_D_MAX (Quicktune Yaw D max)

DEFAULT: 0.01	RANGE: 0.001 1.0	UNITS: Gain

Quicktune Yaw D max (QWIK_YAW_D_MAX)

Description

This parameter defines the maximum limit for the Yaw Derivative (D) gain that the Quicktune system is allowed to set.

The D-gain is primarily used to "dampen" the tail of the aircraft, preventing it from overshooting when you stop a turn. However, yaw dynamics involve a lot of inertia, and setting the D-gain too high can cause high-frequency vibrations that result in hot motors or ESC failures. This parameter ensures that the automated tuning process never enters that dangerous high-gain territory.

The Mathematics

The system monitors for oscillations while increasing the D-gain, but will stop immediately if the gain reaches this ceiling:

$$K_d_yaw = \min(K_target, QWIK_YAW_D_MAX)$$

A value of **0.01** is a common starting point for medium-sized multirotors.

The Engineer's View

In `AP_Quicktune::gain_limit()` (`libraries/AP_Quicktune/AP_Quicktune.cpp`):

1. The code checks the tuning axis.
2. For the `YAW` axis, it specifically retrieves the `_yaw_d_max` parameter value.
3. This ceiling is used in the `limited` check within the `update()` function to terminate the "Search" phase for the D-gain stage of the tune.
4. By providing a hard limit, ArduPilot protects against "false negatives" where the algorithm might not detect a subtle high-frequency vibration that is still damaging to the hardware.

Tuning & Behavior

- **Default Value:** 0.01.
- **Range:** 0.001 to 1.0.
- **When to Increase:** If your vehicle has very little damping and "bounces" at the end of a yaw rotation after a Quicktune session.
- **When to Decrease:** If your motors are warm after a flight where you used Quicktune, or if you hear high-frequency "chirping" from the motors during yaw maneuvers.
- **Note:** Most vehicles require very little Yaw D-gain compared to Roll or Pitch.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





QWIK_YAW_P_MAX (Quicktune Yaw P max)

DEFAULT: 0.5	RANGE: 0.1 3.0	UNITS: Gain

Quicktune Yaw P max (QWIK_YAW_P_MAX)

Description

This parameter sets a "Safety Ceiling" for the Yaw Proportional (P) gain during the Quicktune process. It prevents the automated tuning algorithm from increasing the yaw responsiveness beyond a point that might be physically dangerous or cause mechanical fatigue, even if the software does not detect any oscillations.

Yaw control on many aircraft (especially multirotors) is achieved via motor torque differential, which is less efficient than roll or pitch. Setting a maximum limit ensures that the tune remains conservative and doesn't demand more from the motors than is safe for long-term operation.

The Mathematics

The tuning algorithm calculates an incremental target gain (K_target), but the final value applied to the vehicle's PID controller is always capped:

$$K_p_yaw = \min(K_target, QWIK_YAW_P_MAX)$$

This prevents the algorithm from "running away" if the oscillation detection threshold is set too high or if the airframe is exceptionally stiff.

The Engineer's View

In `AP_Quicktune::gain_limit()` (`libraries/AP_Quicktune/AP_Quicktune.cpp`):

1. The code checks if the axis being tuned is `AxisName::YAW`.
2. If true, it returns the value of `yaw_p_max`.
3. In the `update()` loop, this limit is used to check if the current test gain has reached the user-defined boundary (`limited = (limit > 0.0 && pval ≥ limit)`).
4. If the limit is hit, the tuning stage for that gain is marked as "Done" and the system moves to the next parameter.

Tuning & Behavior

- **Default Value:** 0.5.
- **Range:** 0.1 to 3.0.
- **When to Increase:** If the vehicle still feels "lazy" or slow to hold heading after a Quicktune session, and you are certain the mechanics can handle more aggressive torque changes.
- **When to Decrease:** If you have a very powerful, high-torque vehicle where even small gains result in violent yaw kicks.
- **Dependencies:** Works in conjunction with `QWIK_Y_PI_RATIO` to determine the final K_i for the yaw axis.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





QWIK_Y_PI_RATIO (Quicktune Yaw PI ratio)

DEFAULT: 10.0	RANGE: 1.0 20	UNITS: Ratio

Quicktune Yaw PI ratio (QWIK_Y_PI_RATIO)

Description

This parameter defines the fixed relationship (ratio) between the Proportional (P) and Integral (I) gains for the **Yaw** axis during the Quicktune process.

Because yaw dynamics are typically much slower and less sensitive than roll or pitch, the integrator gain (K_i) is usually set much lower than the proportional gain. Maintaining this ratio ensures that Quicktune doesn't over-power the yaw integrator, which could lead to slow, oscillating "wagging" of the tail.

The Mathematics

During the tuning process, whenever Quicktune adjusts the Yaw P gain (K_p_yaw), it automatically calculates the new Yaw I gain (K_i_yaw) using this ratio (γ_yaw):

$$K_i_yaw = \frac{K_p_yaw}{\gamma_yaw}$$

Where γ_yaw is `QWIK_Y_PI_RATIO`.

For example, with the default ratio of **10.0**, if Quicktune finds an optimal Yaw P of 0.5, it will automatically set the Yaw I to 0.05.

The Engineer's View

In `AP_Quicktune::adjust_gain()` (`libraries/AP_Quicktune/AP_Quicktune.cpp`):

1. The code identifies when the Yaw axis P gain is being modified.
2. It checks for FeedForward (`FF`). If `FF > 0`, the integrator is typically handled independently of P, and this ratio is bypassed.
3. Otherwise, it applies the `y_pi_ratio` specifically.
4. This high default value (10.0 vs 1.0 for roll/pitch) reflects the engineering reality that Yaw requires a much wider gap between instantaneous response (P) and long-term error correction (I) to remain stable on most airframes.

Tuning & Behavior

- **Default Value:** 10.0.
- **Range:** 1.0 to 20.0.
- **Effect of Increasing:** Results in a much lower relative I gain for yaw. Recommended for vehicles that "tail-wag" or feel nervous in the yaw axis.
- **Effect of Decreasing:** Results in a higher relative I gain. Only recommended if the vehicle fails to hold its heading against constant torque or wind.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY

Overview

The **RALLY** parameter group configures the **Rally Points** system. Rally points are alternate "Safe Havens" for the drone to return to during a Failsafe or RTL event, instead of returning all the way to the original takeoff (Home) point.

Rally points are essential for long-range missions (e.g., 20km pipelines) where the vehicle may not have enough battery to return home if a failsafe occurs at the far end of the mission.

Key Concepts

1. Nearest Point Logic

By default, if Rally Points are loaded, the autopilot will calculate the distance to all points and the Home point. It will then choose the **closest** valid point to return to.

- `RALLY_INCL_HOME`: Determines if the Home point is included in this "closest point" calculation.

2. Sanity Checks (`RALLY_LIMIT_KM`)

If the nearest Rally Point is extremely far away (e.g., left over from a previous flight at a different location), the autopilot will ignore it to prevent a "fly-away."

- `RALLY_LIMIT_KM`: The maximum allowable distance to a rally point (default 5km).

Parameter Breakdown

- `RALLY_TOTAL`: Read-only count of uploaded rally points.
- `RALLY_LIMIT_KM`: Distance safety limit.

Integration Guide

1. **Plan:** Use the "Plan" tab in Mission Planner to drop Rally Points along your mission path.
2. **Upload:** Write them to the flight controller.
3. **Verify:** Arm the vehicle and ensure the GCS shows the rally points on the map.

Developer Notes

- **Library:** `libraries/AP_Rally`.
- **Storage:** Stored in a dedicated region of EEPROM.

`RALLY_INCL_HOME` (Rally Include Home)

DEFAULT: 1	RANGE: 0 1

Rally Include Home (RALLY_INCL_HOME)

Description

`RALLY_INCL_HOME` defines the drone's decision logic for Return-to-Launch when multiple "Safe Zones" exist.

Rally Points are pre-programmed safe landing locations. When you trigger RTL, the drone normally finds the *closest* Rally point and flies there. This parameter determines if the drone's original takeoff point (**Home**) should also be considered in that "closest" calculation.

- **1: Enabled (Default).** The drone will fly to either Home or the nearest Rally point, whichever is physically closer to its current location.
- **0: Disabled.** The drone will ignore Home and **only** fly to a Rally point. This is useful if the takeoff point is no longer safe (e.g., you are launching from a moving boat or a crowded area).

Tuning & Behavior

- **Default:** 1.
- **Recommendation:** Leave at **1** for most users. Only disable if your landing site is strictly restricted to designated Rally zones.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





RALLY_LIMIT_KM (Rally Limit)

DEFAULT: 5	RANGE: 0 100	UNITS: km

Rally Limit (RALLY_LIMIT_KM)

Description

This is a safety parameter ("Sanity Check") for Rally Points. If the nearest Rally Point is further away than this limit, the autopilot will ignore it and default to returning to Home (RTL) instead. This prevents the dangerous situation where a drone tries to fly to a Rally Point saved from a previous mission at a distant location (e.g., a different flying field).

The Mathematics

Logic Check:

$$extTarget = \begin{cases} extRallyPoint & \text{if } extDist(extVehicle, extRally) < extRALLY_LIMIT_KM \\ extHome & \text{otherwise} \end{cases}$$

The Engineer's View

Defined in `libraries/AP_Rally/AP_Rally.cpp`.

- **0:** Disabled. The closest rally point is ALWAYS used, regardless of distance.
- **>0:** Enforces the distance limit.

Tuning & Behavior

- **Default Value:** 5 km
- **Recommendation:** Keep this set to a reasonable radius for your operation (e.g., 2-5km) to prevent fly-aways.

RALLY_TOTAL (Rally Total)

DEFAULT: 0	RANGE: 0 50

RALLY_TOTAL: Rally Total

Description

Number of rally points currently loaded.

Values

- **Range:** 0 to 50
- **Default:** 0

Description

This parameter indicates how many Rally Points (safe landing/loiter locations) are currently stored in the autopilot's persistent memory.

- **Usage:** This is typically updated automatically by the Ground Control Station (GCS) when you upload a Rally Point list. You should not normally edit this manually.
- **Function:** The autopilot uses this count to know how many points to check when looking for the nearest safe location during an RTL (Return to Launch).

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **RATE** parameter group (specifically `RATE_ENABLE` or `FSTRATE_ENABLE`) configures the execution architecture of the high-speed **Attitude Rate Controllers**.

Key Concepts

1. High-Priority Threading

Modern ArduPilot firmware can run the critical rate PID loops in a dedicated high-priority CPU thread.

- `RATE_ENABLE`: Enables the fast-rate threading logic.
  - **Benefit:** Ensures that motor updates are never delayed by lower-priority tasks like file I/O or MAVLink processing, resulting in a more stable and "locked-in" flight feel.

Developer Notes

- **Library:** `AP_Scheduler`.
- **See Also:** FSTRATE for more details.

RATE_ENABLE (Yaw rate enable)

```
DEFAULT: 0
```

RATE_ENABLE: Yaw rate enable

Description

Enable yaw rate controller for aerobatic flight.

Values

Value	Meaning
0	Disabled
1	Enabled

- **Default:** 0

Description

This parameter enables a dedicated rate-based controller for the Yaw axis on fixed-wing aircraft.

- **Function:** When enabled, the autopilot will try to maintain a specific yaw rate (degrees per second) proportional to the rudder stick input, rather than just passing the stick input directly to the servo.
- **Use Case:** Primarily for aerobatic planes or 3D flight where precise yaw rate control is desired during maneuvers like knife-edge or spins. For standard coordinated turns, the standard lateral acceleration controller is usually sufficient.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **RAW** parameter group (specifically `RAW_DATA` and `RAW_LOG_OPT`) configures the recording of unfiltered, low-level sensor information.

Key Concepts

1. GPS Raw Logging (`RAW_DATA`)

Normally, ArduPilot only logs the calculated Lat/Lon position. Enabling `RAW_DATA` tells the GPS driver to log the raw satellite measurements (pseudoranges).

- **Purpose:** Required for **PPK** (Post-Processed Kinematics) mapping applications.

2. IMU Raw Logging (`RAW_LOG_OPT`)

Allows for logging raw IMU samples at very high frequencies (e.g., the full 1kHz or 8kHz rate).

- **Purpose:** Used for high-fidelity vibration analysis and tuning the dynamic harmonic notch filters.

Developer Notes

- **Library:** `libraries/AP_GPS`, `libraries/AP_InertialSensor`.
- **Performance:** Enabling raw logging drastically increases the log file size and CPU load. Use only when troubleshooting or performing mapping missions.

RAW_DATA (Raw data logging)

DEFAULT: 0	RANGE: 0 5

RAW_DATA: Raw data logging

Description

This parameter enables the logging of raw GNSS satellite observations from the GPS module to the autopilot's DataFlash logs.

Values

Value	Meaning
0	Ignore (Disabled)
1	Always log
2	Stop logging when disarmed (SBF/Septentrio only)
5	Only log every five samples (u-blox only)

- **Default:** 0

Description

This parameter is essential for **PPK (Post-Processing Kinematics)**.

- **How it works:** Instead of just logging the calculated position, it logs the raw "carrier phase" and "pseudorange" data from the satellites.
- **Usage:** After a flight, you can use software like RTKLIB or Emlid Studio to process this raw log against a base station log to achieve centimeter-level accuracy for photo geotagging or survey paths.
- **Warning:** Enabling this generates much larger log files and increases CPU load. Use a high-speed SD card.

Source Code

ardupilot/libraries/AP_GPS/AP_GPS.cpp

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





RAW_LOG_OPT: Raw logging options

Description

Raw logging options bitmask.

Values

- **Bitmask:**
  - **Bit 0:** Log primary gyro only
  - **Bit 1:** Log all gyros
  - **Bit 2:** Post-filter (log data after the low-pass and notch filters)
  - **Bit 3:** Pre and post-filter (log both raw and filtered data)
- **Default:** 0

Description

This parameter controls the high-speed "Raw IMU" logging, which is used for analyzing vibrations and tuning the **Harmonic Notch Filter**.

- **Usage:** Set to **1** to see the raw noise on all gyros. Set to **3** (1 + 2) to see both raw noise and how well your filters are removing it.
- **Warning:** High-rate logging creates extremely large log files. **Only enable this during tuning flights**, then set it back to 0 for regular operation to avoid wearing out your SD card and potentially causing CPU spikes.

Source Code

ardupilot/libraries/AP_InertialSensor/AP_InertialSensor.cpp

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **RC** parameter group configures how the autopilot interprets signals from the **Radio Control (RC)** receiver. It handles channel calibration (Min/Max/Trim), reversal, and the assignment of auxiliary functions to transmitter switches.

ArduPilot supports up to 16 RC input channels.

Key Concepts

1. Channel Calibration (`RCn_MIN`, `RCn_MAX`, `RCn_TRIM`)

Defines the pulse-width range (PWM in microseconds) for each stick and switch.

- `MIN`: Pulse width at lowest stick position (typically ~1000us).
- `MAX`: Pulse width at highest stick position (typically ~2000us).
- `TRIM`: Pulse width at center/neutral position (typically ~1500us).
- `DZ`: Deadzone. Small range around the center where stick movement is ignored to prevent "jitter" or "drift."

2. Auxiliary Functions (`RCn_OPTION`)

This is the most powerful feature of the RC system. You can map any of ArduPilot's hundreds of functions to a switch on your transmitter.

- **Common Options:**
  - 7: Do Nothing.
  - 22: Parachute Release.
  - 28: Gripper.
  - 300: Camera Trigger.

3. Failsafe (`RC_FS_...`)

Configures how the autopilot detects that the radio link has been lost.

- `RC_FS_TIMEOUT`: How many seconds of no signal before triggering a failsafe action.

Parameter Breakdown

- `RC_PROTOCOLS`: Bitmask to select which digital protocols to search for (SBUS, CRSF, FPort, etc.).
- `RC_OPTIONS`: Bitmask for specialized input behaviors (e.g., ignore receiver failsafe bit).
- `RC_SPEED`: Update rate for standard PWM inputs (Hz).

Integration Guide

1. **Calibration:** Always perform "RC Calibration" in your GCS whenever you change your transmitter settings or receiver.
2. **Assignment:** Use the "Radio Calibration" or "Extended Tuning" screens to assign switch functions (`RCn_OPTION`).
3. **Reverse:** If the HUD shows the drone tilting the wrong way, use `RCn_REVERSED`.

Developer Notes

- **Library:** `libraries/AP_RCProtocol`.
- **Filtering:** Input signals are low-pass filtered to reduce noise.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





MNT1_RC_RATE (Mount RC Rate)

DEFAULT: 0	RANGE: 0 90	UNITS: deg/s

Mount RC Rate (MNT1_RC_RATE)

Description

`MNT1_RC_RATE` (often appearing in older configs as `RC_RATE`) controls how fast your camera gimbal rotates when you move the knobs or sticks on your transmitter.

- **0 (Disabled):** The gimbal is in **Position Mode**. If you move the knob to the middle, the camera points to the middle. If you let go, it stays where the knob is.
- **> 0:** The gimbal is in **Rate Mode**. The knob acts like a joystick. Moving it slightly makes the gimbal turn slowly; moving it all the way makes it turn at the maximum rate defined by this parameter. Letting go of the stick stops the rotation, and the gimbal stays at its current angle.

The Engineer's View

Defined in `AP_Mount_Params.cpp`.
This parameter selects between two fundamental control philosophies for cinematography. Most professional aerial photographers prefer **Rate Mode** because it allows for perfectly smooth, consistent panning speeds.

Tuning & Behavior

- **Default Value:** 0 (Position Mode)
- **Recommendation:**
  - **Cinematography:** Set to **10 or 20** deg/s for cinematic pans.
  - **FPV / Tracking:** Use **Position Mode (0)** so the camera direction matches your stick position exactly.

RC10_DZ (RC dead-zone)

DEFAULT: 0	RANGE: 0 200	UNITS: PWM

RC dead-zone

Note: This parameter functions identically to RC1_DZ.

RC10_MAX (RC max PWM)

DEFAULT: 1900	RANGE: 800 2200	UNITS: PWM

RC max PWM

Note: This parameter functions identically to RC1_MAX.

RC10_MIN (RC min PWM)

DEFAULT: 1100	RANGE: 800 2200	UNITS: PWM

RC min PWM

Note: This parameter functions identically to RC1_MIN.

RC10_OPTION (RC input option)

DEFAULT: 0	RANGE: null

RC input option

Note: This parameter functions identically to RC1_OPTION.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





RC10_REVERSED (RC reversed)

DEFAULT: 0	RANGE: null

RC reversed

Note: This parameter functions identically to RC1_REVERSED.

RC10_TRIM (RC trim PWM)

DEFAULT: 1500	RANGE: 800 2200	UNITS: PWM

RC trim PWM

Note: This parameter functions identically to RC1_TRIM.

RC11_DZ (RC dead-zone)

DEFAULT: 0	RANGE: 0 200	UNITS: PWM

RC dead-zone

Note: This parameter functions identically to RC1_DZ.

RC11_MAX (RC max PWM)

DEFAULT: 1900	RANGE: 800 2200	UNITS: PWM

RC max PWM

Note: This parameter functions identically to RC1_MAX.

RC11_MIN (RC min PWM)

DEFAULT: 1100	RANGE: 800 2200	UNITS: PWM

RC min PWM

Note: This parameter functions identically to RC1_MIN.

RC11_OPTION (RC input option)

DEFAULT: 0	RANGE: null

RC input option

Note: This parameter functions identically to RC1_OPTION.

RC11_REVERSED (RC reversed)

DEFAULT: 0	RANGE: null

RC reversed

Note: This parameter functions identically to RC1_REVERSED.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





RC11_TRIM (RC trim PWM)

DEFAULT: 1500	RANGE: 800 2200	UNITS: PWM

RC trim PWM

Note: This parameter functions identically to RC1_TRIM.

RC12_DZ (RC dead-zone)

DEFAULT: 0	RANGE: 0 200	UNITS: PWM

RC dead-zone

Note: This parameter functions identically to RC1_DZ.

RC12_MAX (RC max PWM)

DEFAULT: 1900	RANGE: 800 2200	UNITS: PWM

RC max PWM

Note: This parameter functions identically to RC1_MAX.

RC12_MIN (RC min PWM)

DEFAULT: 1100	RANGE: 800 2200	UNITS: PWM

RC min PWM

Note: This parameter functions identically to RC1_MIN.

RC12_OPTION (RC input option)

DEFAULT: 0	RANGE: null

RC input option

Note: This parameter functions identically to RC1_OPTION.

RC12_REVERSED (RC reversed)

DEFAULT: 0	RANGE: null

RC reversed

Note: This parameter functions identically to RC1_REVERSED.

RC12_TRIM (RC trim PWM)

DEFAULT: 1500	RANGE: 800 2200	UNITS: PWM

RC trim PWM

Note: This parameter functions identically to RC1_TRIM.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





RC13_DZ (RC dead-zone)

DEFAULT: 0	RANGE: 0 200	UNITS: PWM

RC dead-zone

Note: This parameter functions identically to RC1_DZ.

RC13_MAX (RC max PWM)

DEFAULT: 1900	RANGE: 800 2200	UNITS: PWM

RC max PWM

Note: This parameter functions identically to RC1_MAX.

RC13_MIN (RC min PWM)

DEFAULT: 1100	RANGE: 800 2200	UNITS: PWM

RC min PWM

Note: This parameter functions identically to RC1_MIN.

RC13_OPTION (RC input option)

DEFAULT: 0	RANGE: null

RC input option

Note: This parameter functions identically to RC1_OPTION.

RC13_REVERSED (RC reversed)

DEFAULT: 0	RANGE: null

RC reversed

Note: This parameter functions identically to RC1_REVERSED.

RC13_TRIM (RC trim PWM)

DEFAULT: 1500	RANGE: 800 2200	UNITS: PWM

RC trim PWM

Note: This parameter functions identically to RC1_TRIM.

RC14_DZ (RC dead-zone)

DEFAULT: 0	RANGE: 0 200	UNITS: PWM

RC dead-zone

Note: This parameter functions identically to RC1_DZ.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





RC14_MAX (RC max PWM)

DEFAULT: 1900	RANGE: 800 2200	UNITS: PWM

RC max PWM

Note: This parameter functions identically to RC1_MAX.

RC14_MIN (RC min PWM)

DEFAULT: 1100	RANGE: 800 2200	UNITS: PWM

RC min PWM

Note: This parameter functions identically to RC1_MIN.

RC14_OPTION (RC input option)

DEFAULT: 0	RANGE: null

RC input option

Note: This parameter functions identically to RC1_OPTION.

RC14_REVERSED (RC reversed)

DEFAULT: 0	RANGE: null

RC reversed

Note: This parameter functions identically to RC1_REVERSED.

RC14_TRIM (RC trim PWM)

DEFAULT: 1500	RANGE: 800 2200	UNITS: PWM

RC trim PWM

Note: This parameter functions identically to RC1_TRIM.

RC15_DZ (RC dead-zone)

DEFAULT: 0	RANGE: 0 200	UNITS: PWM

RC dead-zone

Note: This parameter functions identically to RC1_DZ.

RC15_MAX (RC max PWM)

DEFAULT: 1900	RANGE: 800 2200	UNITS: PWM

RC max PWM

Note: This parameter functions identically to RC1_MAX.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





RC15_MIN (RC min PWM)

DEFAULT: 1100	RANGE: 800 2200	UNITS: PWM

RC min PWM

Note: This parameter functions identically to RC1_MIN.

RC15_OPTION (RC input option)

DEFAULT: 0	RANGE: null

RC input option

Note: This parameter functions identically to RC1_OPTION.

RC15_REVERSED (RC reversed)

DEFAULT: 0	RANGE: null

RC reversed

Note: This parameter functions identically to RC1_REVERSED.

RC15_TRIM (RC trim PWM)

DEFAULT: 1500	RANGE: 800 2200	UNITS: PWM

RC trim PWM

Note: This parameter functions identically to RC1_TRIM.

RC16_DZ (RC dead-zone)

DEFAULT: 0	RANGE: 0 200	UNITS: PWM

RC dead-zone

Note: This parameter functions identically to RC1_DZ.

RC16_MAX (RC max PWM)

DEFAULT: 1900	RANGE: 800 2200	UNITS: PWM

RC max PWM

Note: This parameter functions identically to RC1_MAX.

RC16_MIN (RC min PWM)

DEFAULT: 1100	RANGE: 800 2200	UNITS: PWM

RC min PWM

Note: This parameter functions identically to RC1_MIN.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





RC16_OPTION (RC input option)

DEFAULT: 0	RANGE: null

RC input option

Note: This parameter functions identically to RC1_OPTION.

RC16_REVERSED (RC reversed)

DEFAULT: 0	RANGE: null

RC reversed

Note: This parameter functions identically to RC1_REVERSED.

RC16_TRIM (RC trim PWM)

DEFAULT: 1500	RANGE: 800 2200	UNITS: PWM

RC trim PWM

Note: This parameter functions identically to RC1_TRIM.

RC1_DZ (RC Input Channel 1 Deadzone)

DEFAULT: 30	RANGE: 0 200	UNITS: PWM

RC Input Channel 1 Deadzone (RC1_DZ)

Description

`RC1_DZ` creates a small "Null Zone" in the center of your stick. If the PWM signal changes by less than this amount, the autopilot ignores it. This prevents "Stick Jitter" or poor stick centering from causing the drone to twitch or drift.

RC1_MAX (RC Input Channel 1 Maximum)

DEFAULT: 1900	RANGE: 800 2200	UNITS: PWM

RC Input Channel 1 Maximum (RC1_MAX)

Description

`RC1_MAX` defines the highest PWM value your transmitter sends on the first channel. This is used to scale your stick inputs so that "Full Stick" equals "Maximum Demand" in the flight code.

RC1_MIN (RC Input Channel 1 Minimum)

DEFAULT: 1100	RANGE: 800 2200	UNITS: PWM

RC Input Channel 1 Minimum (RC1_MIN)

Description

`RC1_MIN` defines the lowest PWM value your transmitter sends on the first channel. This is calibrated automatically and ensures the autopilot knows the full physical range of your stick movement.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





RC1_OPTION (RC Input Channel 1 Auxiliary Option)

DEFAULT: 0	RANGE: 0 300

RC Input Channel 1 Auxiliary Option (RC1_OPTION)

Description

RC1_OPTION allows you to turn a radio channel into a "Function Switch."

While channels 1-4 are usually reserved for flight axes (Roll, Pitch, Throttle, Yaw), channels 5-16 can be used to trigger features like "Return to Launch," "Arm/Disarm," "AutoTune," or "Camera Trigger."

- **0: Do Nothing.**
- **7: RTL.**
- **300+: Lua Script Trigger.**

RC1_REVERSED (RC Input Channel 1 Reversal)

DEFAULT: 0	RANGE: 0 1

RC Input Channel 1 Reversal (RC1_REVERSED)

Description

RC1_REVERSED flips the logic of your stick. If you push the stick right but the GCS shows the bar moving left, you set this to 1.

Note: It is always preferred to fix this during Radio Calibration in the GCS.

RC1_TRIM (RC Input Channel 1 Trim)

DEFAULT: 1500	RANGE: 800 2200	UNITS: PWM

RC Input Channel 1 Trim (RC1_TRIM)

Description

RC1_TRIM defines the "Center" position for the first channel of your radio receiver (typically Roll).

When your transmitter sticks are centered, the receiver sends a specific PWM value. ArduPilot needs to know exactly what that value is so it doesn't think you are trying to turn when you are actually hands-off.

- **Standard Setting:** 1500 (µs).

Tuning & Behavior

- **Calibration:** This value is set automatically when you perform a **Radio Calibration** in your Ground Control Station. You should never need to adjust it manually.
- **Safety:** If your drone drifts when the sticks are centered, re-run the Radio Calibration rather than manually editing this parameter.

RC2_DZ (RC dead-zone)

DEFAULT: 0	RANGE: 0 200	UNITS: PWM

RC dead-zone

Note: This parameter functions identically to RC1_DZ.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





RC2_MAX (RC max PWM)

DEFAULT: 1900	RANGE: 800 2200	UNITS: PWM

RC max PWM

Note: This parameter functions identically to RC1_MAX.

RC2_MIN (RC min PWM)

DEFAULT: 1100	RANGE: 800 2200	UNITS: PWM

RC min PWM

Note: This parameter functions identically to RC1_MIN.

RC2_OPTION (RC input option)

DEFAULT: 0	RANGE: null

RC input option

Note: This parameter functions identically to RC1_OPTION.

RC2_REVERSED (RC reversed)

DEFAULT: 0	RANGE: null

RC reversed

Note: This parameter functions identically to RC1_REVERSED.

RC2_TRIM (RC trim PWM)

DEFAULT: 1500	RANGE: 800 2200	UNITS: PWM

RC trim PWM

Note: This parameter functions identically to RC1_TRIM.

RC3_DZ (RC dead-zone)

DEFAULT: 0	RANGE: 0 200	UNITS: PWM

RC dead-zone

Note: This parameter functions identically to RC1_DZ.

RC3_MAX (RC max PWM)

DEFAULT: 1900	RANGE: 800 2200	UNITS: PWM

RC max PWM

Note: This parameter functions identically to RC1_MAX.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





RC3_MIN (RC min PWM)

DEFAULT: 1100	RANGE: 800 2200	UNITS: PWM

RC min PWM

Note: This parameter functions identically to RC1_MIN.

RC3_OPTION (RC input option)

DEFAULT: 0	RANGE: null

RC input option

Note: This parameter functions identically to RC1_OPTION.

RC3_REVERSED (RC reversed)

DEFAULT: 0	RANGE: null

RC reversed

Note: This parameter functions identically to RC1_REVERSED.

RC3_TRIM (RC trim PWM)

DEFAULT: 1500	RANGE: 800 2200	UNITS: PWM

RC trim PWM

Note: This parameter functions identically to RC1_TRIM.

RC4_DZ (RC dead-zone)

DEFAULT: 0	RANGE: 0 200	UNITS: PWM

RC dead-zone

Note: This parameter functions identically to RC1_DZ.

RC4_MAX (RC max PWM)

DEFAULT: 1900	RANGE: 800 2200	UNITS: PWM

RC max PWM

Note: This parameter functions identically to RC1_MAX.

RC4_MIN (RC min PWM)

DEFAULT: 1100	RANGE: 800 2200	UNITS: PWM

RC min PWM

Note: This parameter functions identically to RC1_MIN.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





RC4_OPTION (RC input option)

DEFAULT: 0	RANGE: null

RC input option

Note: This parameter functions identically to RC1_OPTION.

RC4_REVERSED (RC reversed)

DEFAULT: 0	RANGE: null

RC reversed

Note: This parameter functions identically to RC1_REVERSED.

RC4_TRIM (RC trim PWM)

DEFAULT: 1500	RANGE: 800 2200	UNITS: PWM

RC trim PWM

Note: This parameter functions identically to RC1_TRIM.

RC5_DZ (RC dead-zone)

DEFAULT: 0	RANGE: 0 200	UNITS: PWM

RC dead-zone

Note: This parameter functions identically to RC1_DZ.

RC5_MAX (RC max PWM)

DEFAULT: 1900	RANGE: 800 2200	UNITS: PWM

RC max PWM

Note: This parameter functions identically to RC1_MAX.

RC5_MIN (RC min PWM)

DEFAULT: 1100	RANGE: 800 2200	UNITS: PWM

RC min PWM

Note: This parameter functions identically to RC1_MIN.

RC5_OPTION (RC input option)

DEFAULT: 0	RANGE: null

RC input option

Note: This parameter functions identically to RC1_OPTION.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





RC5_REVERSED (RC reversed)

DEFAULT: 0	RANGE: null

RC reversed

Note: This parameter functions identically to RC1_REVERSED.

RC5_TRIM (RC trim PWM)

DEFAULT: 1500	RANGE: 800 2200	UNITS: PWM

RC trim PWM

Note: This parameter functions identically to RC1_TRIM.

RC6_DZ (RC dead-zone)

DEFAULT: 0	RANGE: 0 200	UNITS: PWM

RC dead-zone

Note: This parameter functions identically to RC1_DZ.

RC6_MAX (RC max PWM)

DEFAULT: 1900	RANGE: 800 2200	UNITS: PWM

RC max PWM

Note: This parameter functions identically to RC1_MAX.

RC6_MIN (RC min PWM)

DEFAULT: 1100	RANGE: 800 2200	UNITS: PWM

RC min PWM

Note: This parameter functions identically to RC1_MIN.

RC6_OPTION (RC input option)

DEFAULT: 0	RANGE: null

RC input option

Note: This parameter functions identically to RC1_OPTION.

RC6_REVERSED (RC reversed)

DEFAULT: 0	RANGE: null

RC reversed

Note: This parameter functions identically to RC1_REVERSED.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





RC6_TRIM (RC trim PWM)

DEFAULT: 1500	RANGE: 800 2200	UNITS: PWM

RC trim PWM

Note: This parameter functions identically to RC1_TRIM.

RC7_DZ (RC dead-zone)

DEFAULT: 0	RANGE: 0 200	UNITS: PWM

RC dead-zone

Note: This parameter functions identically to RC1_DZ.

RC7_MAX (RC max PWM)

DEFAULT: 1900	RANGE: 800 2200	UNITS: PWM

RC max PWM

Note: This parameter functions identically to RC1_MAX.

RC7_MIN (RC min PWM)

DEFAULT: 1100	RANGE: 800 2200	UNITS: PWM

RC min PWM

Note: This parameter functions identically to RC1_MIN.

RC7_OPTION (RC input option)

DEFAULT: 0	RANGE: null

RC input option

Note: This parameter functions identically to RC1_OPTION.

RC7_REVERSED (RC reversed)

DEFAULT: 0	RANGE: null

RC reversed

Note: This parameter functions identically to RC1_REVERSED.

RC7_TRIM (RC trim PWM)

DEFAULT: 1500	RANGE: 800 2200	UNITS: PWM

RC trim PWM

Note: This parameter functions identically to RC1_TRIM.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





RC8_DZ (RC dead-zone)

DEFAULT: 0	RANGE: 0 200	UNITS: PWM

RC dead-zone

Note: This parameter functions identically to RC1_DZ.

RC8_MAX (RC max PWM)

DEFAULT: 1900	RANGE: 800 2200	UNITS: PWM

RC max PWM

Note: This parameter functions identically to RC1_MAX.

RC8_MIN (RC min PWM)

DEFAULT: 1100	RANGE: 800 2200	UNITS: PWM

RC min PWM

Note: This parameter functions identically to RC1_MIN.

RC8_OPTION (RC input option)

DEFAULT: 0	RANGE: null

RC input option

Note: This parameter functions identically to RC1_OPTION.

RC8_REVERSED (RC reversed)

DEFAULT: 0	RANGE: null

RC reversed

Note: This parameter functions identically to RC1_REVERSED.

RC8_TRIM (RC trim PWM)

DEFAULT: 1500	RANGE: 800 2200	UNITS: PWM

RC trim PWM

Note: This parameter functions identically to RC1_TRIM.

RC9_DZ (RC dead-zone)

DEFAULT: 0	RANGE: 0 200	UNITS: PWM

RC dead-zone

Note: This parameter functions identically to RC1_DZ.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





RC9_MAX (RC max PWM)

DEFAULT: 1900	RANGE: 800 2200	UNITS: PWM

RC max PWM

Note: This parameter functions identically to RC1_MAX.

RC9_MIN (RC min PWM)

DEFAULT: 1100	RANGE: 800 2200	UNITS: PWM

RC min PWM

Note: This parameter functions identically to RC1_MIN.

RC9_OPTION (RC input option)

DEFAULT: 0	RANGE: null

RC input option

Note: This parameter functions identically to RC1_OPTION.

RC9_REVERSED (RC reversed)

DEFAULT: 0	RANGE: null

RC reversed

Note: This parameter functions identically to RC1_REVERSED.

RC9_TRIM (RC trim PWM)

DEFAULT: 1500	RANGE: 800 2200	UNITS: PWM

RC trim PWM

Note: This parameter functions identically to RC1_TRIM.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





RC_FS_MSK (Servo RC Failsafe Mask)

DEFAULT: 0	RANGE: 0 4294967295

RC_FS_MSK: Servo RC Failsafe Mask

Description

Bitmask of scaled passthru output channels which will be set to their trim value during rc failsafe instead of holding their last position before failsafe.

Values

- **Bitmask:**
  - **Bit 0:** Servo 1
  - **Bit 1:** Servo 2
  - **...**
  - **Bit 31:** Servo 32
- **Default:** 0

Description

This parameter defines the behavior of "Pass-Through" channels (servos directly controlled by the pilot's RC sticks) when the radio link is lost (RC Failsafe).

- **Default (0):** All pass-through channels will "Hold Last Position." For example, if you were turning right when the signal was lost, the rudder stays right.
- **Enabled Bit:** If a channel's bit is set, that servo will jump to its **TRIM** value (usually center) immediately upon failsafe.
- **Use Case:** Primarily for auxiliary surfaces like flaps or landing gear where you want a predictable, neutral state during an emergency return home, rather than leaving them in a high-drag configuration.

RC_FS_TIMEOUT (RC Failsafe timeout)

DEFAULT: 1.0	RANGE: 0.5 10.0	UNITS: s

RC Failsafe timeout (RC_FS_TIMEOUT)

Description

`RC_FS_TIMEOUT` defines how long ArduPilot waits after losing contact with your radio receiver before it decides "The pilot is gone" and triggers a failsafe action (like RTL or Land).

A shorter timeout provides faster safety response but may cause "nuisance" failsafes if your radio link is momentarily blocked by an obstacle. A longer timeout allows for more robust connections in noisy environments but delays the automatic return home.

The Mathematics

The system monitors the "Last Pulse Time" for the RC channels. Failsafe triggers if:

$$\Delta t_\text{no_signal} > \text{RC_FS_TIMEOUT}$$

The Engineer's View

Defined in `RC_Channels_VarInfo.h` as `_fs_timeout`.
It is primarily used in `ArduCopter/radio.cpp` (and equivalent vehicle files) within the `read_radio()` loop. The timeout applies to both pulse-based failsafes (like SBUS signal loss) and throttle-low failsafes.

Tuning & Behavior

- **Default Value:** 1.0 second
- **Range:** 0.5 to 10.0 seconds
- **Recommendation:**
  - **Standard Flight:** Keep at **1.0s**. This is the best balance of safety and robustness.
  - **Long Range:** You might increase this to **2.0s or 3.0s** if you are flying near the limit of your radio range where "telemetry lost" warnings are frequent.
  - **Racing:** Set to **0.5s** for the absolute fastest response in a crash or signal loss.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





RC_OPTIONS (RC options)

DEFAULT: 32	RANGE: 0 16383

RC options (RC_OPTIONS)

Description

`RC_OPTIONS` is a bitmask used to enable specialized behaviors for your radio control link. It handles things like MAVLink overrides, Link Quality (LQ) sensing, and protocol-specific tweaks.

Key Bits:

- **Bit 0 (1): Ignore RC Receiver.** Effectively disables the radio input.
- **Bit 1 (2): Ignore MAVLink Overrides.** Prevents GCS or companion computers from injecting "fake" RC signals.
- **Bit 5 (32): Arming check throttle for 0 input.** (Enabled by Default). Requires the throttle to be zeroed before the drone will arm.
- **Bit 13 (8192): Use 420kbaud for ELRS.** Required for standard ExpressLRS setups on modern flight controllers to avoid synchronization errors.

The Mathematics

$$\text{Active Options} = \sum 2^i \cdot b_i$$

The Engineer's View

Defined in `RC_Channels_VarInfo.h`.
Commonly set during initial configuration. For example, many ExpressLRS users will have `RC_OPTIONS` set to **8192** (Bit 13) or **8736** (Bits 5, 8, 9, 13).

Tuning & Behavior

- **Default Value:** 32 (Arming check for 0 throttle).
- **Recommendation:**
  - **ELRS Users:** Ensure Bit 13 (8192) is set.
  - **Joystick Users (GCS):** Ensure Bit 1 (2) is NOT set, otherwise MAVLink joysticks won't work.
  - **Companion Computers:** If your drone is controlled via Python/ROS, you might want to set Bit 1 (2) to ensure the radio doesn't override your script during autonomous flight (or vice-versa).

RC_OVERRIDE_TIME (RC Override Timeout)

DEFAULT: 3.0	RANGE: 0 120	UNITS: s

RC Override Timeout (RC_OVERRIDE_TIME)

Description

`RC_OVERRIDE_TIME` is a safety timer for GCS/Companion Computer control.

If you control your drone using a Joystick plugged into Mission Planner, or via a Python script sending `RC_CHANNELS_OVERRIDE` messages, this timer determines what happens if that computer crashes or the telemetry link dies.

- **3.0s (Default):** If no override message is received for 3 seconds, the autopilot ignores the stale overrides and gives control back to the physical RC Receiver.
- **0.0:** Overrides are ignored immediately (effectively disables overrides).
- **-1.0:** Infinite timeout. The drone will hold the last received command forever (Danger!).

The Mathematics

$$\text{Active} = (t_now - t_last_msg) < \text{RC_OVERRIDE_TIME}$$

The Engineer's View

Defined in `RC_Channels_VarInfo.h` as `_override_timeout`.
This timeout applies to **all** channels simultaneously.

Tuning & Behavior

- **Default Value:** 3.0s
- **Recommendation:**
  - **Joystick Flying:** Set to **1.0s** for faster safety handover if the link drops.
  - **Companion Computer:** Set to **0.5s** to ensure the drone doesn't "hang" on an old command if the script crashes.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





RC_PROTOCOLS (RC Protocols Mask)

DEFAULT: 1	RANGE: 0 4294967295

RC Protocols Mask (RC_PROTOCOLS)

Description

`RC_PROTOCOLS` tells the autopilot which languages to listen for on the RC Input pin.

By default, ArduPilot scans for **All (1)** known protocols (SBUS, DSM, PPM, CRSF, etc.). This makes setup easy but can cause CPU load or false detections. If you know exactly what receiver you are using, you can disable the others to speed up the detection process and prevent glitches.

- **Bit 0 (1):** All Protocols (Default).
- **Bit 1 (2):** PPM.
- **Bit 2 (4):** IBUS.
- **Bit 3 (8):** SBUS.
- **Bit 4 (16):** SBUS Non-Inverted.
- **Bit 5 (32):** DSM / Spektrum.
- **Bit 6 (64):** SUMD.
- **Bit 7 (128):** SRXL.
- **Bit 8 (256):** SRXL2.
- **Bit 9 (512):** CRSF (Crossfire / ELRS).
- **Bit 10 (1024):** FPort.

The Engineer's View

Defined in `AP_RCProtocol`.
The autopilot runs a state machine that samples the input pin. If `RC_PROTOCOLS` has Bit 0 set, it iterates through all available backend decoders. If you specify a single bit (e.g., 512 for CRSF), the loop skips the other decoders, saving CPU cycles and reducing the chance of a "ghost" signal being detected on a noisy line.

Tuning & Behavior

- **Default Value:** 1 (All).
- **Recommendation:**
  - **ExpressLRS / Crossfire:** Set to **513** (All + CRSF) or just **512** (CRSF only) if you want to be strict.
  - **SBUS:** Set to **8**.
  - **Ghosting:** If your servos twitch randomly, try disabling "All" and selecting only your specific protocol bit.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





RC_SPEED (RC output speed in Hz)

DEFAULT: 490	RANGE: 50 490	UNITS: Hz

RC output speed in Hz (RC_SPEED)

Description

`RC_SPEED` determines how often the flight controller sends a new position update to your ESCs and servos when using standard **PWM** signaling.

- **Standard Servos:** Usually require **50Hz**. Higher values can burn them out.
- **Digital Servos:** Can often handle **250Hz - 330Hz**.
- **Standard ESCs:** Usually run at **490Hz** for fast response.

Note: This parameter has no effect if you are using digital protocols like **DShot**. DShot speed is determined by `MOT_PWM_TYPE` and the main loop rate.

The Mathematics

The frequency f in Hz determines the period T between pulses:

$$T = \frac{1}{f}$$

At 490Hz, a new command is sent every **2.04 milliseconds**. At 50Hz, a command is sent every **20 milliseconds**. Lower period (higher frequency) means lower latency and better stabilization.

The Engineer's View

Used during the hardware initialization of the PWM timer groups.
On most boards, outputs are grouped (e.g., Pins 1-4 share a timer). If you set `RC_SPEED` to 490Hz for your motors on Pin 1, any servo connected to Pin 4 must also be able to handle 490Hz. If it can't, you must move the servo to a different timer group.

Tuning & Behavior

- **Default Value:** 490 Hz (Optimized for multicopter ESCs)
- **Recommendation:**
  - **Multicopters (Standard ESCs):** Keep at **490Hz**.
  - **Planes (Analog Servos):** Set to **50Hz**.
  - **Helicopters (Digital Cyclic Servos):** Set to **125Hz or 250Hz** (check servo specs).

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

Key Concepts

RCIN1_PORT_OPTIONS (RC Input Port Options (Instance 1))

DEFAULT: 0	RANGE: 0 15

RC Input Port Options (RCIN1_PORT_OPTIONS)

Description

`RCIN1_PORT_OPTIONS` provides low-level electrical control over the primary radio receiver connector.

This is often required for modern RC protocols like **FPort** (which needs inversion and half-duplex) or **SBUS** (which needs inversion).

- **Bit 0 (1): Invert RX.** Necessary for SBUS on some boards.
- **Bit 1 (2): Invert TX.**
- **Bit 2 (4): Half-Duplex.** Necessary for FPort or single-wire protocols.
- **Bit 3 (8): Swap Pins.**

Tuning & Behavior

- **Default:** 0.
- **Reboot Required:** Yes.
- **Note:** This is usually handled automatically by the receiver protocol detection, but manual override is available for custom hardware setups.

RCIN_MSGRATE (RC Input Message Rate)

DEFAULT: 10	RANGE: 1 100	UNITS: Hz

RC Input Message Rate (RCIN_MSGRATE)

Description

`RCIN_MSGRATE` defines the "Update Speed" for radio control signals within the system.

RCIN_PORT (RC Input Port (AP_Periph))

DEFAULT: 0	RANGE: 0 10

RC Input Port (AP_Periph) (RCIN_PORT)

Description

`RCIN_PORT` specifies which UART connector on your peripheral hardware is plugged into the RC receiver.

This is primarily used on **AP_Periph** CAN nodes that are acting as "Remote RC Ports." It allows the node to know which serial hardware to initialize for protocol detection.

Tuning & Behavior

- **Default:** 0.
- **Setup:** Refer to the hardware manual for your CAN node (e.g. Matek L431) to find the correct port number.
- **Reboot Required:** Yes.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





RCIN_PROTOCOLS (Enabled RC Protocols (AP_Periph))

DEFAULT: 1	RANGE: 0 65535

Enabled RC Protocols (RCIN_PROTOCOLS)

Description

`RCIN_PROTOCOLS` is used on **AP_Periph** CAN nodes that are being used to bridge an RC receiver to the main flight controller.

By default, the system attempts to automatically detect the protocol of any receiver plugged into the serial port. However, if auto-detection is slow or failing, you can use this bitmask to force the system to only look for specific protocols.

- **1:** All protocols enabled (Auto-detect).
- **Bit 0:** PPM
- **Bit 1:** SBUS
- **Bit 2:** DSM
- **Bit 3:** SUMD
- **Bit 4:** SRXL
- **Bit 10:** CRSF (Crossfire/ELRS)

Tuning & Behavior

- **Default:** 1 (Auto).
- **Recommendation:** Leave at **1** unless your specific receiver is not being detected correctly.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **RCMAP** parameter group defines the default mapping of the four primary control axes (Roll, Pitch, Yaw, Throttle) to RC input channels.

By default, ArduPilot follows the **AETR** (Aileron, Elevator, Throttle, Rudder) or **TAER** standards. However, if your transmitter uses a different channel order, you can use these parameters to remap the logic without changing physical wires.

Key Concepts

1. Axis Mapping

- `RCMAP_ROLL`: Maps the Roll (Aileron) function to a channel (1-16).
- `RCMAP_PITCH`: Maps the Pitch (Elevator) function to a channel.
- `RCMAP_THROTTLE`: Maps the Throttle function to a channel.
- `RCMAP_YAW`: Maps the Yaw (Rudder) function to a channel.

Developer Notes

- **Library:** `libraries/AP_RCProtocol`.
- **Standard Defaults:**
  - Channel 1: Roll
  - Channel 2: Pitch
  - Channel 3: Throttle
  - Channel 4: Yaw

Pitch channel ()

RCMAP_PITCH: Pitch channel

Description

Pitch channel number. This is useful when you have a RC transmitter that can't change the channel order easily. Pitch is normally on channel 2, but you can move it to any channel with this parameter.

Values

- **Range:** 1 to 16
- **Increment:** 1
- **Default:** 2

Description

This parameter maps the physical radio channel for Pitch (Elevator) to the autopilot's internal Pitch control.

- **Usage:** If your radio outputs Elevator on Channel 2 (Default), leave this at 2. If it uses a different order (e.g., AETR vs TAER), adjust this to match.
- **Reboot Required:** You MUST reboot the flight controller for changes to this parameter to take effect.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





RCMAP_FORWARD (Forward Channel Mapping)

DEFAULT: 0	RANGE: 1 16

Forward Channel Mapping (RCMAP_FORWARD)

Description

`RCMAP_FORWARD` allows you to customize which stick or switch on your transmitter controls the vehicle's forward/reverse movement.

This is primarily used for **6-DOF multicopters** (omnicopters) and **Sub/ROV** applications where the standard 4-axis control (Roll, Pitch, Throttle, Yaw) is expanded to include direct translation.

Tuning & Behavior

- **Default:** 0 (Disabled/Unmapped).
- **Usage:** Set to the channel number (e.g. 6) that your transmitter is sending the forward-translation signal on.
- **Reboot Required:** Yes.

RCMAP_LATERAL (Lateral Channel Mapping)

DEFAULT: 0	RANGE: 1 16

Lateral Channel Mapping (RCMAP_LATERAL)

Description

`RCMAP_LATERAL` allows you to customize which stick or switch controls the vehicle's left/right strafing movement.

This is essential for vehicles capable of lateral translation without rolling, such as 6-DOF drones and ROVs.

Roll channel ()

RCMAP_ROLL: Roll channel

Description

Roll channel number. This is useful when you have a RC transmitter that can't change the channel order easily. Roll is normally on channel 1, but you can move it to any channel with this parameter.

Values

- **Range:** 1 to 16
- **Increment:** 1
- **Default:** 1

Description

This parameter maps the physical radio channel for Roll (Aileron) to the autopilot's internal Roll control.

- **Usage:** If your radio outputs Aileron on Channel 3 instead of Channel 1, set this parameter to 3.
- **Why it matters:** Allows ArduPilot to work with any radio brand (Futaba, Spektrum, FrSky) regardless of their native channel order (AETR, TAER, etc.).
- **Reboot Required:** You MUST reboot the flight controller for changes to this parameter to take effect.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





`Throttle channel` ()

RCMAP_THROTTLE: Throttle channel

Description

Throttle channel number. This is useful when you have a RC transmitter that can't change the channel order easily. Throttle is normally on channel 3, but you can move it to any channel with this parameter.

Values

- **Range:** 1 to 16
- **Increment:** 1
- **Default:** 3

Description

This parameter maps the physical radio channel for Throttle to the autopilot's internal Throttle control.

- **Usage:** If your radio outputs Throttle on Channel 3 (Default), leave this at 3. Adjust if using a non-standard radio configuration.
- **Reboot Required:** You MUST reboot the flight controller for changes to this parameter to take effect.

`Yaw channel` ()

RCMAP_YAW: Yaw channel

Description

Yaw channel number. This is useful when you have a RC transmitter that can't change the channel order easily. Yaw (also known as rudder) is normally on channel 4, but you can move it to any channel with this parameter.

Values

- **Range:** 1 to 16
- **Increment:** 1
- **Default:** 4

Description

This parameter maps the physical radio channel for Yaw (Rudder) to the autopilot's internal Yaw control.

- **Usage:** If your radio outputs Rudder on Channel 4 (Default), leave this at 4. If you have a 4-channel radio and want to swap Rudder and Aileron, you would change this and `RCMAP_ROLL`.
- **Reboot Required:** You MUST reboot the flight controller for changes to this parameter to take effect.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **RELAY** parameter group configures the **Digital Relay** subsystem. Relays are simple On/Off (3.3V/5V digital) switches used to control external hardware that does not require a PWM signal.

ArduPilot supports up to 6 standard relays (and more in some versions) that can be mapped to any GPIO-capable pin on the flight controller.

Key Concepts

1. Relay Assignment (`RELAYx_PIN`)

Maps a physical pin on the flight controller to a logical relay ID.

- `RELAY1_PIN`: The GPIO ID of the first relay (e.g., 50 for AUX 1).
- **Disabled (-1):** The relay is not active.

2. Logic Polarity (`RELAYx_INVERTED`)

Defines whether "On" is High (3.3V) or Low (0V).

- **0 (Normal):** On = High, Off = Low.
- **1 (Inverted):** On = Low, Off = High.

3. Default State (`RELAYx_DEFAULT`)

Defines the state of the relay immediately after the autopilot boots up.

- **0:** Off.
- **1:** On.
- **2:** No change (Maintain state across reboots if possible).

Parameter Breakdown

- `RELAY_ON`: (Legacy/Global) Global command to toggle.
- `RELAY1_FUNCTION`: Sets the purpose of the relay (e.g., Relay, Ignition, Brush).

Integration Guide

1. **Select Pin:** Identify an AUX pin you wish to use.
2. **GPIO Mask:** Ensure that pin is enabled in `BRD_PWM_COUNT` or `GPIO_MASK` (depending on firmware).
3. **Config:** Set `RELAY1_PIN` to the pin ID.
4. **Control:** Map an RC channel to `Relay 1` (RC Option 33) to toggle it with a switch.

Developer Notes

- **Library:** `libraries/AP_Relay`.
- **Capacity:** Useful for controlling lights, smoke systems, camera shutters, or high-power battery isolation contactors.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Camera relay ON value ()

RELAY_ON: Camera relay ON value

Description

This sets whether the relay goes high or low when it triggers. Note that you should also set `RELAY_DEFAULT` appropriately for your camera.

Values

Value	Meaning
0	Low (Ground)
1	High (3.3V or 5V)

- **Default:** 1

Description

This parameter configures the electrical signal logic used to trigger a camera via a relay.

- **Usage:** Set this to match your camera's shutter release requirement. Most cameras trigger on a transition.
- **Function:** When the "Take Photo" command is issued, the assigned relay pin will switch to the level defined here for the duration set in `CAM_DURATION`.

Source Code

ardupilot/libraries/AP_Camera/AP_Camera_Params.cpp

RELAY10_DEFAULT (Relay default state)

DEFAULT: (float	RANGE: null

Relay default state

Note: This parameter functions identically to RELAY1_DEFAULT.

RELAY10_FUNCTION (Relay function)

DEFAULT: (float	RANGE: null

Relay function

Note: This parameter functions identically to RELAY1_FUNCTION.

RELAY10_INVERTED (Relay invert output signal)

DEFAULT: false	RANGE: null

Relay invert output signal

Note: This parameter functions identically to RELAY1_INVERTED.

RELAY10_PIN (Relay pin)

DEFAULT: -1	RANGE: null

Relay pin

Note: This parameter functions identically to RELAY1_PIN.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





RELAY11_DEFAULT (Relay default state)

DEFAULT: (float	RANGE: null

Relay default state

Note: This parameter functions identically to RELAY1_DEFAULT.

RELAY11_FUNCTION (Relay function)

DEFAULT: (float	RANGE: null

Relay function

Note: This parameter functions identically to RELAY1_FUNCTION.

RELAY11_INVERTED (Relay invert output signal)

DEFAULT: false	RANGE: null

Relay invert output signal

Note: This parameter functions identically to RELAY1_INVERTED.

RELAY11_PIN (Relay pin)

DEFAULT: -1	RANGE: null

Relay pin

Note: This parameter functions identically to RELAY1_PIN.

RELAY12_DEFAULT (Relay default state)

DEFAULT: (float	RANGE: null

Relay default state

Note: This parameter functions identically to RELAY1_DEFAULT.

RELAY12_FUNCTION (Relay function)

DEFAULT: (float	RANGE: null

Relay function

Note: This parameter functions identically to RELAY1_FUNCTION.

RELAY12_INVERTED (Relay invert output signal)

DEFAULT: false	RANGE: null

Relay invert output signal

Note: This parameter functions identically to RELAY1_INVERTED.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





RELAY12_PIN (Relay pin)

DEFAULT: -1	RANGE: null

Relay pin

Note: This parameter functions identically to RELAY1_PIN.

RELAY13_DEFAULT (Relay default state)

DEFAULT: (float	RANGE: null

Relay default state

Note: This parameter functions identically to RELAY1_DEFAULT.

RELAY13_FUNCTION (Relay function)

DEFAULT: (float	RANGE: null

Relay function

Note: This parameter functions identically to RELAY1_FUNCTION.

RELAY13_INVERTED (Relay invert output signal)

DEFAULT: false	RANGE: null

Relay invert output signal

Note: This parameter functions identically to RELAY1_INVERTED.

RELAY13_PIN (Relay pin)

DEFAULT: -1	RANGE: null

Relay pin

Note: This parameter functions identically to RELAY1_PIN.

RELAY14_DEFAULT (Relay default state)

DEFAULT: (float	RANGE: null

Relay default state

Note: This parameter functions identically to RELAY1_DEFAULT.

RELAY14_FUNCTION (Relay function)

DEFAULT: (float	RANGE: null

Relay function

Note: This parameter functions identically to RELAY1_FUNCTION.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





RELAY14_INVERTED (Relay invert output signal)

DEFAULT: false	RANGE: null

Relay invert output signal

Note: This parameter functions identically to RELAY1_INVERTED.

RELAY14_PIN (Relay pin)

DEFAULT: -1	RANGE: null

Relay pin

Note: This parameter functions identically to RELAY1_PIN.

RELAY15_DEFAULT (Relay default state)

DEFAULT: (float	RANGE: null

Relay default state

Note: This parameter functions identically to RELAY1_DEFAULT.

RELAY15_FUNCTION (Relay function)

DEFAULT: (float	RANGE: null

Relay function

Note: This parameter functions identically to RELAY1_FUNCTION.

RELAY15_INVERTED (Relay invert output signal)

DEFAULT: false	RANGE: null

Relay invert output signal

Note: This parameter functions identically to RELAY1_INVERTED.

RELAY15_PIN (Relay pin)

DEFAULT: -1	RANGE: null

Relay pin

Note: This parameter functions identically to RELAY1_PIN.

RELAY16_DEFAULT (Relay default state)

DEFAULT: (float	RANGE: null

Relay default state

Note: This parameter functions identically to RELAY1_DEFAULT.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





RELAY16_FUNCTION (Relay function)

DEFAULT: (float	RANGE: null

Relay function

Note: This parameter functions identically to RELAY1_FUNCTION.

RELAY16_INVERTED (Relay invert output signal)

DEFAULT: false	RANGE: null

Relay invert output signal

Note: This parameter functions identically to RELAY1_INVERTED.

RELAY16_PIN (Relay pin)

DEFAULT: -1	RANGE: null

Relay pin

Note: This parameter functions identically to RELAY1_PIN.

RELAY1_DEFAULT (Relay Default State (Instance 1))

DEFAULT: 0	RANGE: 0 1

Relay Default State (RELAY1_DEFAULT)

Description

`RELAY1_DEFAULT` defines the "Power-On" state of the first relay output.

This is important for safety and automation. For example, if Relay 1 is connected to a camera power supply, you might want it to default to **ON (1)** so the camera starts recording immediately upon power-up. If it is connected to a loud siren or a high-voltage igniter, you would set it to **OFF (0)** to prevent accidental activation.

- **0: OFF.** The relay starts in the deactivated state.
- **1: ON.** The relay starts in the activated state.

Tuning & Behavior

- **Default:** 0.
- **Logic Link:** This value is applied immediately when the autopilot initializes.
- **Note:** If RELAY1_INVERTED is enabled, the physical output state will be flipped relative to this setting.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





RELAY1_FUNCTION (Relay Function (Instance 1))

DEFAULT: 0	RANGE: 0 100

Relay Function (RELAY1_FUNCTION)

Description

`RELAY1_FUNCTION` defines the "Job" of the first relay output.

Unlike physical PWM outputs which are usually used for motors and servos, Relays are simple digital ON/OFF switches. This parameter tells the autopilot what internal logic should control this switch.

Common Values

- **0: None.** Relay is disabled.
- **1: Relay.** Manual control via MAVLink or auxiliary switch.
- **2: Ignition.** Controlled by the Internal Combustion Engine (ICE) library.
- **3: Parachute.** Triggered when a crash is detected.
- **4: Camera.** Triggered by the camera shutter logic.

Tuning & Behavior

- **Default:** 0.
- **Note:** If you want to use the relay for a custom purpose (like turning on LED lights via a transmitter switch), set this to **1 (Relay)** and assign an RC channel to that relay instance.

RELAY1_INVERTED (Relay Inversion (Instance 1))

DEFAULT: 0	RANGE: 0 1

Relay Inversion (RELAY1_INVERTED)

Description

`RELAY1_INVERTED` calibrates the autopilot for your specific relay hardware.

Different relay modules have different "Trigger" requirements. Some turn ON when given a 3.3V signal (Active-High), while others turn ON when the signal is pulled to Ground (Active-Low).

- **0: Normal.** Active-High (Standard).
- **1: Inverted.** Active-Low.

Tuning & Behavior

- **Calibration:** If your relay stays ON when you command it OFF (and vice versa), toggle this parameter.

RELAY1_PIN (Relay Pin (Instance 1))

DEFAULT: -1	RANGE: -1 103

Relay Pin (RELAY1_PIN)

Description

`RELAY1_PIN` tells the autopilot which digital connector on the flight controller is wired to the relay hardware.

- **-1:** Disabled (No physical pin assigned).
- **0-103:** Physical GPIO pin number.

Tuning & Behavior

- **Setup:** Refer to your flight controller's "GPIOs" documentation to find the correct number for the pin you are using.
- **Hardware Constraint:** Standard servo outputs on many Pixhawk-style boards cannot be used as relays unless they are first "freed" by setting their `SERVOx_FUNCTION` to -1.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





RELAY2_DEFAULT (Relay default state)

DEFAULT: (float	RANGE: null

Relay default state

Note: This parameter functions identically to RELAY1_DEFAULT.

RELAY2_FUNCTION (Relay function)

DEFAULT: (float	RANGE: null

Relay function

Note: This parameter functions identically to RELAY1_FUNCTION.

RELAY2_INVERTED (Relay invert output signal)

DEFAULT: false	RANGE: null

Relay invert output signal

Note: This parameter functions identically to RELAY1_INVERTED.

RELAY2_PIN (Relay pin)

DEFAULT: -1	RANGE: null

Relay pin

Note: This parameter functions identically to RELAY1_PIN.

RELAY3_DEFAULT (Relay default state)

DEFAULT: (float	RANGE: null

Relay default state

Note: This parameter functions identically to RELAY1_DEFAULT.

RELAY3_FUNCTION (Relay function)

DEFAULT: (float	RANGE: null

Relay function

Note: This parameter functions identically to RELAY1_FUNCTION.

RELAY3_INVERTED (Relay invert output signal)

DEFAULT: false	RANGE: null

Relay invert output signal

Note: This parameter functions identically to RELAY1_INVERTED.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





RELAY3_PIN (Relay pin)

DEFAULT: -1	RANGE: null

Relay pin

Note: This parameter functions identically to RELAY1_PIN.

RELAY4_DEFAULT (Relay default state)

DEFAULT: (float	RANGE: null

Relay default state

Note: This parameter functions identically to RELAY1_DEFAULT.

RELAY4_FUNCTION (Relay function)

DEFAULT: (float	RANGE: null

Relay function

Note: This parameter functions identically to RELAY1_FUNCTION.

RELAY4_INVERTED (Relay invert output signal)

DEFAULT: false	RANGE: null

Relay invert output signal

Note: This parameter functions identically to RELAY1_INVERTED.

RELAY4_PIN (Relay pin)

DEFAULT: -1	RANGE: null

Relay pin

Note: This parameter functions identically to RELAY1_PIN.

RELAY5_DEFAULT (Relay default state)

DEFAULT: (float	RANGE: null

Relay default state

Note: This parameter functions identically to RELAY1_DEFAULT.

RELAY5_FUNCTION (Relay function)

DEFAULT: (float	RANGE: null

Relay function

Note: This parameter functions identically to RELAY1_FUNCTION.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





RELAY5_INVERTED (Relay invert output signal)

DEFAULT: false	RANGE: null

Relay invert output signal

Note: This parameter functions identically to RELAY1_INVERTED.

RELAY5_PIN (Relay pin)

DEFAULT: -1	RANGE: null

Relay pin

Note: This parameter functions identically to RELAY1_PIN.

RELAY6_DEFAULT (Relay default state)

DEFAULT: (float	RANGE: null

Relay default state

Note: This parameter functions identically to RELAY1_DEFAULT.

RELAY6_FUNCTION (Relay function)

DEFAULT: (float	RANGE: null

Relay function

Note: This parameter functions identically to RELAY1_FUNCTION.

RELAY6_INVERTED (Relay invert output signal)

DEFAULT: false	RANGE: null

Relay invert output signal

Note: This parameter functions identically to RELAY1_INVERTED.

RELAY6_PIN (Relay pin)

DEFAULT: -1	RANGE: null

Relay pin

Note: This parameter functions identically to RELAY1_PIN.

RELAY7_DEFAULT (Relay default state)

DEFAULT: (float	RANGE: null

Relay default state

Note: This parameter functions identically to RELAY1_DEFAULT.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





RELAY7_FUNCTION (Relay function)

DEFAULT: (float	RANGE: null

Relay function

Note: This parameter functions identically to RELAY1_FUNCTION.

RELAY7_INVERTED (Relay invert output signal)

DEFAULT: false	RANGE: null

Relay invert output signal

Note: This parameter functions identically to RELAY1_INVERTED.

RELAY7_PIN (Relay pin)

DEFAULT: -1	RANGE: null

Relay pin

Note: This parameter functions identically to RELAY1_PIN.

RELAY8_DEFAULT (Relay default state)

DEFAULT: (float	RANGE: null

Relay default state

Note: This parameter functions identically to RELAY1_DEFAULT.

RELAY8_FUNCTION (Relay function)

DEFAULT: (float	RANGE: null

Relay function

Note: This parameter functions identically to RELAY1_FUNCTION.

RELAY8_INVERTED (Relay invert output signal)

DEFAULT: false	RANGE: null

Relay invert output signal

Note: This parameter functions identically to RELAY1_INVERTED.

RELAY8_PIN (Relay pin)

DEFAULT: -1	RANGE: null

Relay pin

Note: This parameter functions identically to RELAY1_PIN.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





RELAY9_DEFAULT (Relay default state)

DEFAULT: (float	RANGE: null

Relay default state

Note: This parameter functions identically to RELAY1_DEFAULT.

RELAY9_FUNCTION (Relay function)

DEFAULT: (float	RANGE: null

Relay function

Note: This parameter functions identically to RELAY1_FUNCTION.

RELAY9_INVERTED (Relay invert output signal)

DEFAULT: false	RANGE: null

Relay invert output signal

Note: This parameter functions identically to RELAY1_INVERTED.

RELAY9_PIN (Relay pin)

DEFAULT: -1	RANGE: null

Relay pin

Note: This parameter functions identically to RELAY1_PIN.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **RLL** parameter group (often appearing as `RLL_` or `RLL2SRV_`) configures the **Roll Attitude Controller** for **ArduPlane**.

This library is responsible for maintaining the requested bank angle (e.g., "Bank 30 degrees left") and managing the rotation rate of the aircraft by driving the aileron servos.

Key Concepts

1. Roll-to-Servo (`RLL2SRV_...`)

The core PID controller for fixed-wing roll.

- `RLL2SRV_P/I/D`: Standard PID gains.
- `RLL2SRV_TCONST`: The time constant (seconds) for the roll response. Defines how long it takes to reach 63% of the target angle.

2. Rate Limits (`RLL2SRV_RMAX`)

Defines the maximum physical roll rate (deg/s) the autopilot is allowed to request. This protects the airframe from over-stress and ensures the pilot stays within safe maneuvering limits.

Parameter Breakdown

- `RLL2SRV_P`: Proportional gain.
- `RLL2SRV_TCONST`: Speed of response (typically 0.4s to 0.7s).
- `RLL_RATE_P`: (If present) Direct gyro-rate feedback gain.

Integration Guide

- **Tuning:** Use **AUTOTUNE** in ArduPlane. It will automatically find the best `RLL2SRV_P`, `I`, and `D` values by performing a series of controlled rolls.
- **Oscillation:** If the plane "waggles" its wings rapidly in FBWA mode, reduce `RLL2SRV_P` or increase `RLL2SRV_TCONST`.

Developer Notes

- **Library:** `libraries/APM_Control/AP_RollController.cpp`.

RLL2SRV_RMAX (Maximum Roll Rate)

DEFAULT: 0	RANGE: 0 180	UNITS: deg/s

Maximum Roll Rate (RLL2SRV_RMAX)

Description

`RLL2SRV_RMAX` acts as a "Speed Limit" for roll.

Even if the `TCONST` gain demands a roll rate of 300 deg/s to fix an error, this parameter caps the request.

- **0:** Disabled (No limit).
- **60:** Cap at 60 deg/s.

Tuning & Behavior

- **Default Value:** 0.
- **Recommendation:** Set to match the physical capabilities of your airframe. If you ask for more than the plane can do, the integrator will wind up.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





RLL2SRV_TCONST (Roll Time Constant)

DEFAULT: 0.5	**RANGE:** 0.4 1.0	**UNITS:** s

Roll Time Constant (RLL2SRV_TCONST)

Description

`RLL2SRV_TCONST` defines the "Lag" between demanding a bank angle and achieving it.

It converts the Angle Error into a Desired Rate.
$$ \text{Rate}_{cmd} = \frac{\text{Angle}_{error}}{\text{TCONST}} $$

- **Small Value (0.4):** High Gain. The plane tries to snap to the angle instantly. Can cause oscillation.
- **Large Value (0.8):** Low Gain. The plane rolls lazily to the angle.

Tuning & Behavior

- **Default Value:** 0.5 seconds.
- **Recommendation:**
  - **Aerobatic Plane:** 0.3 - 0.4.
  - **Large UAV / Glider:** 0.6 - 0.8.

RLL_RATE_D (Roll Rate D Gain)

DEFAULT: 0.004	**RANGE:** 0.001 0.03

Roll Rate D Gain (RLL_RATE_D)

Description

`RLL_RATE_D` acts as a "Shock Absorber" for the roll axis.

When the plane rolls quickly to the target angle, it has momentum. `D` predicts that the error is closing fast and applies "Counter-Steer" to stop the roll smoothly without overshooting.

Tuning & Behavior

- **Default Value:** 0.004.
- **Too High:** You will see rapid, jittery oscillations or hear the servos chattering.
- **Too Low:** The plane will bounce or "Wag" when you stop a roll.

RLL_RATE_D_FF (Roll Derivative Feed-Forward)

DEFAULT: 0	**RANGE:** 0 0.03

Roll Derivative Feed-Forward (RLL_RATE_D_FF)

Description

`RLL_RATE_D_FF` acts like an "Inertia Override."

When you flick the stick quickly, the autopilot sees a rapid change in your command. This parameter injects extra power immediately to overcome the drone's inertia and start the roll faster than a standard PID loop could.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





RLL_RATE_FF (Roll Rate Feed-Forward)

DEFAULT: 0.345	RANGE: 0 3.0

Roll Rate Feed-Forward (RLL_RATE_FF)

Description

`RLL_RATE_FF` is the most important gain for Fixed Wing aircraft.

It tells the autopilot: "If I want 30 degrees/sec roll rate, I know I need to move the ailerons to 15 degrees." It provides the bulk of the control surface movement directly from the stick input, bypassing the error-correction loop.

The Mathematics

$$\text{Output} = k_FF \cdot \text{Rate}_target$$

Tuning & Behavior

- **Default Value:** 0.345.
- **Recommendation:** Tune this *first* (or use Autotune). If `FF` is correct, the P and I terms have very little work to do.

RLL_RATE_FLTD (Roll Rate Derivative Filter)

DEFAULT: 0	RANGE: 0 50	UNITS: Hz

Roll Rate Derivative Filter (RLL_RATE_FLTD)

Description

`RLL_RATE_FLTD` cleans up the gyro signal before the D-term calculates the rate of change.

The Derivative term amplifies noise. Without this filter, engine vibrations would cause the D-term to command erratic servo movements, leading to "Jitter" and hot servos.

Tuning & Behavior

- **Default Value:** 0 (Disabled).
- **Recommendation:** Always set this if using D-gain.
  - **Large Plane:** 5 - 10 Hz.
  - **Small Plane:** 10 - 20 Hz.

RLL_RATE_FLTE (Roll Rate Error Filter)

DEFAULT: 0	RANGE: 0 50	UNITS: Hz

Roll Rate Error Filter (RLL_RATE_FLTE)

Description

`RLL_RATE_FLTE` smooths out the "Error" signal (the difference between desired and actual rate). This helps prevent high-frequency noise from triggering jerky control surface movements.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





RLL_RATE_FLTT (Roll Rate Target Filter)

DEFAULT: 0	RANGE: 0 50	UNITS: Hz

Roll Rate Target Filter (RLL_RATE_FLTT)

Description

`RLL_RATE_FLTT` smooths the pilot's stick inputs before they reach the PID controller.

- **0:** Disabled.
- **5 Hz:** Very smooth, "cinematic" feel.
- **20 Hz:** Sharp, responsive feel.

Tuning & Behavior

- **Default Value:** 0 (Disabled).
- **Recommendation:** Use **3 - 5 Hz** if you want to fly smooth lines.

RLL_RATE_I (Roll Rate I Gain)

DEFAULT: 0.15	RANGE: 0.01 0.6

Roll Rate I Gain (RLL_RATE_I)

Description

`RLL_RATE_I` handles "Drift" and "Trim" on the roll axis.

If one wing is heavier than the other, or if your ailerons are slightly warped, the plane will constantly want to roll to one side. The I-term builds up aileron pressure over time to cancel out this bias and keep the roll rate at zero (level flight).

Tuning & Behavior

- **Default Value:** 0.15.
- **Recommendation:** Usually set equal to `RLL_RATE_FF` or found via Autotune.

RLL_RATE_IMAX (Roll Rate I Max)

DEFAULT: 0.666	RANGE: 0 1

Roll Rate I Max (RLL_RATE_IMAX)

Description

`RLL_RATE_IMAX` limits how far the autopilot can move the ailerons to correct for a persistent error (like a bent wing or extreme crosswind).

- **Default (0.666):** Allows the I-term to use up to 66% of the servo travel.

Tuning & Behavior

- **Recommendation:** Leave at default. If set too low, the plane might not be able to hold level flight if it is badly out of trim.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





RLL_RATE_P (Roll Rate P Gain)

DEFAULT: 0.15	RANGE: 0.08 0.35

Roll Rate P Gain (RLL_RATE_P)

Description

`RLL_P` is the "Instant Reaction" gain for your plane's roll axis.

When you move the aileron stick, you are commanding a roll rate (degrees per second). If the plane isn't rolling fast enough, `P` adds more aileron deflection immediately.

- **Too Low:** The plane feels sluggish and disconnected. It takes a moment to start rolling.
- **Too High:** The plane oscillates (wobbles) quickly in roll.

The Mathematics

$$ \text{Output} = k_P \cdot (\text{Rate}*{target} - |*text{Rate}*{actual}) $$

Tuning & Behavior

- **Default Value:** 0.15.
- **Recommendation:** Use **Autotune** to find the perfect value for your airframe.
- **Manual Tuning:** Increase P until you see oscillations, then reduce by 30%.

RLL_RATE_PDMX (Roll Rate PD Max)

DEFAULT: 0	RANGE: 0 1.0

Roll Rate PD Max (RLL_RATE_PDMX)

Description

`RLL_RATE_PDMX` limits the total effort of the P and D terms.

It ensures that the high-speed portion of the PID loop (which reacts to quick movement) doesn't over-saturate the servos, leaving room for the slower, long-term I-term to do its job of keeping the plane level.

- **0 (Default):** Disabled (No limit).
- **0.5:** Limits the PD sum to 50% of the total servo travel.

RLL_RATE_SMAX (Roll Slew Rate Limit)

DEFAULT: 0	RANGE: 0 200

Roll Slew Rate Limit (RLL_RATE_SMAX)

Description

`RLL_RATE_SMAX` prevents the servos from moving too fast.

If the PID loop commands a jump from 0% to 100% aileron in 1 millisecond, this limiter slows it down to a more reasonable speed.

- **0:** Disabled (Unlimited).
- **Value:** Maximum percentage change per second.

Tuning & Behavior

- **Recommendation:** Usually left at **0** for planes. Can be used if you have very slow servos or fragile linkages.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **RLL2SRV** parameter group contains the core gains for the ArduPlane roll controller.

For full documentation on these parameters, please see the RLL group.

RLL2SRV_RMAX (Steering Max Rate)

DEFAULT: 0	RANGE: 0 180	UNITS: deg/s

Steering Max Rate (RLL2SRV_RMAX)

Description

`RLL2SRV_RMAX` limits how fast the steering servo or motor is allowed to move.

- **0:** Disabled (No limit).
- **Value:** Degrees per second.

RLL2SRV_TCONST (Steering Time Constant)

DEFAULT: 0.75	RANGE: 0.4 1.0	UNITS: s

Steering Time Constant (RLL2SRV_TCONST)

Description

`RLL2SRV_TCONST` defines the "Responsiveness" of the steering system for Rovers and the ground-handling of Planes.

A lower value makes the steering more aggressive and "snappy," forcing the vehicle to reach its target angle as quickly as possible. A higher value makes the steering feel "lazy" or more gradual.

Tuning & Behavior

- **Default:** 0.75 seconds.
- **Recommendation (Rover):** Leave at **0.75**.
- **Recommendation (Plane):** Use **0.5** for better ground handling during high-speed takeoff runs.
- **Note:** If set too low, the steering may oscillate (weaving) even at low speeds.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY

RANGEFINDER CONFIGURATION (RNGFND)

Overview

The **RNGFND** parameter group configures the **Rangefinder** (Lidar, Ultrasonic, or Radar) subsystem. Rangefinders provide precise altitude data (distance to ground) or obstacle distance.

ArduPilot supports up to 10 simultaneous rangefinder instances, which can be oriented in any direction (Down, Forward, Up, etc.).

Key Concepts

1. Distance Sensing Types (`RNGFNDn_TYPE`)

Defines the hardware protocol.

- **1 (Analog):** Old ultrasonic sensors.
- **2 (MaxbotixI2C):** I2C Ultrasonic.
- **4 (LightWareI2C):** High-end Lidar.
- **15 (MAVLink):** Distance data from a companion computer or separate MAVLink device.
- **24 (DroneCAN):** CAN-based sensors (e.g., HereFlow, Benewake CAN).

2. Orientation (`RNGFNDn_ORIENT`)

Determines which direction the sensor is pointing.

- **25 (Down):** Used for precision landing, terrain following, and automated takeoff/landing.
- **0 (Forward):** Used for simple obstacle avoidance.
- **6 (Up):** Used for ceiling avoidance or indoor altitude hold.

3. Scaling and Offsets

- `RNGFNDn_SCALING`: Multiplier for analog sensors.
- `RNGFNDn_OFFSET`: Physical distance (meters) between the sensor and the zero-point of the vehicle.
- `RNGFND_GNDCLEAR`: (Down only) The distance from the sensor to the ground when the vehicle is physically resting on its landing gear. **Critical for proper landing detection.**

Parameter Breakdown

- `RNGFNDn_MIN / MAX`: The valid operating range of the sensor (centimeters). Values outside this range are ignored by the EKF.
- `RNGFND_FILT`: Cutoff frequency for the rangefinder data filter (Hz).

Integration Guide

Setting up a Downward Lidar

1. **Mount:** Face the sensor straight down with a clear view of the ground.
2. **Enable:** Set `RNGFND1_TYPE` to your sensor model.
3. **Orient:** Set `RNGFND1_ORIENT = 25`.
4. **Measure:** Measure the distance to the ground while landed and set `RNGFND_GNDCLEAR`.
5. **Verify:** Check the `sonar_range` or `rngfnd1` value in the GCS. It should read correctly as you lift the drone.

Developer Notes

- **Library:** `libraries/AP_RangeFinder`.
- **Usage:** Fused by `AP_NavEKF3` to improve vertical position accuracy and for "Terrain Following" mission logic.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY

GNDCLEAR (Rangefinder Ground Clearance)

DEFAULT: 10	RANGE: 1 100	UNITS: cm

Rangefinder Ground Clearance (GNDCLEAR)

Description

`GNDCLEAR` tells the autopilot how high the Lidar sensor is sitting when the drone is on the ground.

When the drone is stationary on its landing gear, the Lidar reading isn't zero (because the sensor is usually a few centimeters off the dirt). This parameter allows the flight controller to subtract that height so that the reported "Altitude" is exactly zero on the ground.

Tuning & Behavior

- **Default Value:** 10 cm.
- **Recommendation:** Measure the distance from the sensor face to the floor and enter it here.

RNGFND1_ADDR (Rangefinder Address (Sensor 1))

DEFAULT: 0	RANGE: 0 127

Rangefinder Address (RNGFND1_ADDR)

Description

`RNGFND1_ADDR` sets the unique identifier used to communicate with the sensor over a digital bus.

- **For I2C sensors:** This is the 7-bit I2C address (e.g., 0x62 for LIDAR-Lite).
- **For DroneCAN sensors:** This is the Node ID assigned to the sensor.
- **For Serial/Analog sensors:** This parameter is generally ignored.

Tuning & Behavior

- **Default:** 0 (Use the driver's default address).
- **Setup:** Only change this if you have multiple identical sensors on the same bus and have reconfigured one to a non-default address.

RNGFND1_FUNCTION (Rangefinder Voltage Function)

DEFAULT: 0	RANGE: 0 2

Rangefinder Voltage Function (RNGFND1_FUNCTION)

Description

`RNGFND1_FUNCTION` defines the relationship between the sensor's output voltage and the actual physical distance. This is only used for **Analog** sensors.

- **0: Linear (Standard).** $\text{Dist} = (\text{Volt} - \text{Offset}) \times \text{Scaling}$. Used for most ultrasonic and infrared sensors.
- **1: Inverted.** $\text{Dist} = (\text{Offset} - \text{Volt}) \times \text{Scaling}$. Used for sensors where voltage decreases as distance increases.
- **2: Hyperbolic.** $\text{Dist} = \frac{\text{Scaling}}{\text{Volt} - \text{Offset}}$.

Tuning & Behavior

- **Default:** 0 (Linear).
- **Recommendation:** Check your sensor's data sheet. If the graph of "Distance vs Voltage" is a straight line, use **Linear**. If it's a curve, you may need **Hyperbolic**.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





RNGFND1_OFFSET (Rangefinder Offset (Sensor 1))

DEFAULT: 0	RANGE: -1.0 1.0	UNITS: V

Rangefinder Offset (RNGFND1_OFFSET)

Description

`RNGFND1_OFFSET` provides a linear correction to the distance measurement.

For **Analog** sensors, this is an electrical offset in Volts (e.g. the sensor outputs 0.2V even at 0cm). For **PWM** sensors, this is an additive offset in centimeters (e.g. to account for the sensor being mounted 10cm inside the fuselage).

$$\text{Distance} = (\text{Voltage} \times \text{Scaling}) + \text{Offset}$$

Tuning & Behavior

- **Default:** 0.
- **Calibration:** Place the vehicle at a known distance from a wall. If the reported distance is consistently too short, increase the offset.

RNGFND1_ORIENT (Rangefinder orientation (Sensor 1))

DEFAULT: 25	RANGE: 0 25

Rangefinder orientation (RNGFND1_ORIENT)

Description

`RNGFND1_ORIENT` defines the pointing vector of the distance sensor.

Correct orientation is vital for the EKF and flight controllers to know whether they are measuring altitude (Down), distance to an obstacle (Forward/Side), or height above the vehicle (Up).

- **25: Down.** (Most common). Used for altitude hold and precision landing.
- **0: Forward.** Used for simple object avoidance.
- **24: Up.** Used for ceiling-following or indoor flight.

Tuning & Behavior

- **Default:** 25 (Down).
- **Warning:** If the orientation is set incorrectly (e.g., set to Forward when pointing Down), the drone may attempt to "climb" to avoid what it thinks is an obstacle in front of it.

RNGFND1_PIN (Rangefinder Pin)

DEFAULT: -1	RANGE: -1 100

Rangefinder Pin (RNGFND1_PIN)

Description

`RNGFND1_PIN` sets the Analog Input pin used for **Analog** rangefinders (Type 1).

- **-1: Not used.**
- **0-9: Analog Pin Number.** (e.g. 0 for A0).
- **Note:** This parameter is only relevant if `RNGFND1_TYPE` = 1.

Tuning & Behavior

- **Default:** -1

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





RNGFND1_PWRRNG (Rangefinder Power Save Range (Sensor 1))

DEFAULT: 0	RANGE: 0 32767	UNITS: m

Rangefinder Power Save Range (RNGFND1_PWRRNG)

Description

`RNGFND1_PWRRNG` allows the autopilot to turn off the distance sensor when the drone is flying high.

Some Lidar sensors consume significant power. If you are flying a long-range mapping mission at 100m altitude, but your Lidar only works up to 20m, keeping it active is wasteful. This parameter puts the sensor into a low-power "Sleep" state when the estimated terrain clearance exceeds this value.

- **0:** Disabled (Sensor is always ON).
- **Value (m):** Altitude above terrain at which the sensor sleeps.

Tuning & Behavior

- **Default:** 0.
- **Usage:** Set to slightly less than your sensor's maximum effective range.
- **Note:** This requires the sensor driver to support a low-power mode (e.g. Lightware).

RNGFND1_RMETRIC (Rangefinder Ratiometric Enable)

DEFAULT: 1	RANGE: 0 1

Rangefinder Ratiometric Enable (RNGFND1_RMETRIC)

Description

`RNGFND1_RMETRIC` calibrates the autopilot's ADC (Analog-to-Digital Converter) behavior for **Analog** distance sensors.

- **1: Yes (Standard).** Most simple analog sensors change their output voltage relative to the 5V supply. If the 5V rail drops slightly, the sensor's output drops too. The autopilot compensates for this by comparing the sensor voltage to the rail voltage.
- **0: No.** Use this if the sensor has its own internal voltage regulator and outputs a constant voltage regardless of the input power (e.g. some high-end Lightware sensors).

Tuning & Behavior

- **Default:** 1.
- **Recommendation:** Leave at **1** unless your sensor's manual explicitly states it has an internal voltage regulator.

RNGFND1_SCALING (Rangefinder Scaling)

DEFAULT: 3.0	RANGE: 0 100	UNITS: m/V

Rangefinder Scaling (RNGFND1_SCALING)

Description

`RNGFND1_SCALING` calibrates **Analog** rangefinders (Type 1). It defines how many meters of distance correspond to 1.0 Volt of output.

$$\text{Distance} = (\text{Voltage} \times \text{Scaling}) + \text{Offset}$$

- **Example:** If 1V = 1m, set Scaling to 1.0.
- **Example:** Maxbotix EZ4 outputs ~3.3V at 7m. Scaling ≈ 2.12.

Tuning & Behavior

- **Default:** 3.0
- **Calculation:** Measure the voltage at a known distance. $Scaling = \frac{\text{Distance}}{\text{Voltage}}$.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY

RNGFND1_SNR_MIN (Rangefinder Minimum SNR (Sensor 1))

DEFAULT: 0	RANGE: 0 255

Rangefinder Minimum SNR (RNGFND1_SNR_MIN)

Description

`RNGFND1_SNR_MIN` defines the "Quality Threshold" for your distance sensor.

Digital rangefinders (like those using DroneCAN) often report the strength of the reflection (SNR). In environments with poor reflectivity (e.g. over tall grass or dark surfaces) or at the very edge of the sensor's range, the data can become noisy and unreliable. This parameter tells the autopilot to ignore any distance reading if its quality score falls below this limit.

Tuning & Behavior

- **Default:** 0 (Trust all data).
- **Recommendation:** If you see your altitude "flickering" or jumping when the drone is high up or over difficult terrain, increase this value to **10** or **20**.
- **Note:** Setting this too high will cause the rangefinder to "drop out" (status 0) even when it might still be reporting valid data.
- **Context:** This is primarily used by DroneCAN/UAVCAN rangefinders.

RNGFND1_STOP_PIN (Rangefinder Stop Pin)

DEFAULT: -1	RANGE: -1 103

Rangefinder Stop Pin (RNGFND1_STOP_PIN)

Description

`RNGFND1_STOP_PIN` allows ArduPilot to turn the rangefinder sensor ON and OFF dynamically.

This is primarily used to save battery power or to prevent the sensor from producing noisy signals when the vehicle is flying too high for the sensor to work (Out of Range).

- **-1:** Disabled (Sensor is always ON).
- **0-103:** Physical GPIO pin number used to control the sensor's enable/stop line.

Tuning & Behavior

- **Default:** -1.
- **Usage:** If a pin is configured, the autopilot will set the pin HIGH to enable the sensor and LOW to disable it when the estimated altitude is well beyond the sensor's maximum range.

RNGFND1_TYPE (Rangefinder Type (Sensor 1))

DEFAULT: 0	RANGE: 0 42

Rangefinder Type (RNGFND1_TYPE)

Description

`RNGFND1_TYPE` enables the primary altitude/distance sensor.

- **0: None.** Disabled.
- **1: Analog.** Voltage-based sensors.
- **2: Maxbotix I2C.**
- **4: PulsedLight I2C.** (LIDAR-Lite).
- **10: MavLink.**
- **15: TFmini.** (Serial).
- **24: DroneCAN.**

Tuning & Behavior

- **Default:** 0
- **Reboot Required:** Yes.
- **Setup:** After setting the type, you must reboot, then configure `RNGFND1_MIN_CM` and `RNGFND1_MAX_CM`.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY

RNGFND1_WSP_AVG (Wasp-LRF Multi-Pulse Average)

DEFAULT: 2	RANGE: 0 255

Wasp-LRF Multi-Pulse Average (RNGFND1_WSP_AVG)

Description

`RNGFND1_WSP_AVG` improves the accuracy of the **Wasp-LRF** laser rangefinder.

In this mode, the sensor fires a quick sequence of laser pulses and averages the results before sending a single distance value to the autopilot. This helps filter out "Outliers" or noise caused by dust, rain, or difficult surfaces.

- **Higher Value:** More accurate, but reduces the effective update frequency.
- **Lower Value:** Faster updates, but more sensitive to noise.

Tuning & Behavior

- **Default:** 2.
- **Recommendation:** Leave at **2** for general flight. Increase to **5** or **10** if you are performing slow-speed precision tasks (like automatic docking) where accuracy is more important than speed.
- **Dependency:** Only active if RNGFND1_TYPE is set to 18 (WASP-LRF).

RNGFND1_WSP_BAUD (Wasp-LRF Baud Rate)

DEFAULT: 0	RANGE: 0 1

Wasp-LRF Baud Rate (RNGFND1_WSP_BAUD)

Description

`RNGFND1_WSP_BAUD` sets the UART speed for the **Wasp-LRF** laser sensor.

- **0: Low Speed (115200).** Standard for most telemetry ports.
- **1: High Speed (921600).** Required if you are using extremely high update frequencies (>1000 Hz).

RNGFND1_WSP_FRQ (Wasp-LRF Measurement Frequency)

DEFAULT: 20	RANGE: 20 10000	UNITS: Hz

Wasp-LRF Measurement Frequency (RNGFND1_WSP_FRQ)

Description

`RNGFND1_WSP_FRQ` controls how many times per second the Wasp-LRF laser rangefinder takes a distance measurement.

This is a hardware-specific parameter for the WASP-LRF sensor. A higher frequency provides more data for the autopilot to filter, which is useful for high-speed terrain following.

Tuning & Behavior

- **Default:** 20 Hz.
- **Maximum:** Up to 10,000 Hz (if supported by the sensor and communication baud rate).
- **Dependency:** Only active if RNGFND1_TYPE is set to 18 (WASP-LRF).

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





RNGFND1_WSP_MAVG (Wasp-LRF Moving Average)

DEFAULT: 4	RANGE: 0 255	UNITS: Samples

Wasp-LRF Moving Average (RNGFND1_WSP_MAVG)

Description

`RNGFND1_WSP_MAVG` defines the window size for a smoothing filter inside the **Wasp-LRF** sensor driver.

A higher number of samples in the average leads to more stable altitude readings but introduces a small amount of latency (lag) as the drone moves vertically.

RNGFND1_WSP_MEDF (Wasp-LRF Median Filter)

DEFAULT: 4	RANGE: 0 255	UNITS: Samples

Wasp-LRF Median Filter (RNGFND1_WSP_MEDF)

Description

`RNGFND1_WSP_MEDF` enables a **Median Filter**, which is excellent at removing "Spikes" from the data.

Unlike a standard average, a median filter picks the middle value in a set. This ensures that a single bad reading (e.g., a laser reflection from a leaf or a piece of dust) doesn't cause the altitude estimate to jump.

RNGFND1_WSP_THR (Wasp-LRF Sensitivity Threshold)

DEFAULT: -1	RANGE: -1 255

Wasp-LRF Sensitivity Threshold (RNGFND1_WSP_THR)

Description

`RNGFND1_WSP_THR` controls the "Gain" of the laser receiver.

- **Higher Value:** Higher sensitivity. Can see farther or over darker surfaces, but more prone to false readings from fog or dust.
- **-1:** Automatic threshold adjustment.

RNGFND2_ADDR (Bus address of sensor)

DEFAULT: 0	RANGE: 0 127

Bus address of sensor

Note: This parameter functions identically to RNGFND1_ADDR.

RNGFND2_FUNCTION (Rangefinder function)

DEFAULT: 0	RANGE: null

Rangefinder function

Note: This parameter functions identically to RNGFND1_FUNCTION.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





RNGFND2_OFFSET (rangefinder offset)

DEFAULT: 0.0f	RANGE: null	UNITS: V

rangefinder offset

Note: This parameter functions identically to RNGFND1_OFFSET.

RNGFND2_ORIENT (Rangefinder orientation)

DEFAULT: AP_RANGEFINDER_DEFAULT_ORIENTATION	RANGE: null

Rangefinder orientation

Note: This parameter functions identically to RNGFND1_ORIENT.

RNGFND2_PIN (Rangefinder pin)

DEFAULT: -1	RANGE: null

Rangefinder pin

Note: This parameter functions identically to RNGFND1_PIN.

RNGFND2_PWRRNG (Powersave range)

DEFAULT: 0	RANGE: 0 32767	UNITS: m

Powersave range

Note: This parameter functions identically to RNGFND1_PWRRNG.

RNGFND2_RMETRIC (Ratiometric)

DEFAULT: 1	RANGE: null

Ratiometric

Note: This parameter functions identically to RNGFND1_RMETRIC.

RNGFND2_SCALING (Rangefinder scaling)

DEFAULT: 3.0f	RANGE: null	UNITS: m/V

Rangefinder scaling

Note: This parameter functions identically to RNGFND1_SCALING.

RNGFND2_STOP_PIN (Rangefinder stop pin)

DEFAULT: -1	RANGE: null

Rangefinder stop pin

Note: This parameter functions identically to RNGFND1_STOP_PIN.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





RNGFND2_TYPE (Rangefinder type)

DEFAULT: 0	RANGE: null

Rangefinder type

Note: This parameter functions identically to RNGFND1_TYPE.

RNGFND3_ADDR (Bus address of sensor)

DEFAULT: 0	RANGE: 0 127

Bus address of sensor

Note: This parameter functions identically to RNGFND1_ADDR.

RNGFND3_FUNCTION (Rangefinder function)

DEFAULT: 0	RANGE: null

Rangefinder function

Note: This parameter functions identically to RNGFND1_FUNCTION.

RNGFND3_OFFSET (rangefinder offset)

DEFAULT: 0.0f	RANGE: null	UNITS: V

rangefinder offset

Note: This parameter functions identically to RNGFND1_OFFSET.

RNGFND3_ORIENT (Rangefinder orientation)

DEFAULT: AP_RANGEFINDER_DEFAULT_ORIENTATION	RANGE: null

Rangefinder orientation

Note: This parameter functions identically to RNGFND1_ORIENT.

RNGFND3_PIN (Rangefinder pin)

DEFAULT: -1	RANGE: null

Rangefinder pin

Note: This parameter functions identically to RNGFND1_PIN.

RNGFND3_PWRRNG (Powersave range)

DEFAULT: 0	RANGE: 0 32767	UNITS: m

Powersave range

Note: This parameter functions identically to RNGFND1_PWRRNG.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





RNGFND3_RMETRIC (Ratiometric)

DEFAULT: 1	RANGE: null

Ratiometric

Note: This parameter functions identically to RNGFND1_RMETRIC.

RNGFND3_SCALING (Rangefinder scaling)

DEFAULT: 3.0f	RANGE: null	UNITS: m/V

Rangefinder scaling

Note: This parameter functions identically to RNGFND1_SCALING.

RNGFND3_STOP_PIN (Rangefinder stop pin)

DEFAULT: -1	RANGE: null

Rangefinder stop pin

Note: This parameter functions identically to RNGFND1_STOP_PIN.

RNGFND3_TYPE (Rangefinder type)

DEFAULT: 0	RANGE: null

Rangefinder type

Note: This parameter functions identically to RNGFND1_TYPE.

RNGFND4_ADDR (Bus address of sensor)

DEFAULT: 0	RANGE: 0 127

Bus address of sensor

Note: This parameter functions identically to RNGFND1_ADDR.

RNGFND4_FUNCTION (Rangefinder function)

DEFAULT: 0	RANGE: null

Rangefinder function

Note: This parameter functions identically to RNGFND1_FUNCTION.

RNGFND4_OFFSET (rangefinder offset)

DEFAULT: 0.0f	RANGE: null	UNITS: V

rangefinder offset

Note: This parameter functions identically to RNGFND1_OFFSET.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





RNGFND4_ORIENT (Rangefinder orientation)

DEFAULT: AP_RANGEFINDER_DEFAULT_ORIENTATION	RANGE: null

Rangefinder orientation

Note: This parameter functions identically to RNGFND1_ORIENT.

RNGFND4_PIN (Rangefinder pin)

DEFAULT: -1	RANGE: null

Rangefinder pin

Note: This parameter functions identically to RNGFND1_PIN.

RNGFND4_PWRRNG (Powersave range)

DEFAULT: 0	RANGE: 0 32767	UNITS: m

Powersave range

Note: This parameter functions identically to RNGFND1_PWRRNG.

RNGFND4_RMETRIC (Ratiometric)

DEFAULT: 1	RANGE: null

Ratiometric

Note: This parameter functions identically to RNGFND1_RMETRIC.

RNGFND4_SCALING (Rangefinder scaling)

DEFAULT: 3.0f	RANGE: null	UNITS: m/V

Rangefinder scaling

Note: This parameter functions identically to RNGFND1_SCALING.

RNGFND4_STOP_PIN (Rangefinder stop pin)

DEFAULT: -1	RANGE: null

Rangefinder stop pin

Note: This parameter functions identically to RNGFND1_STOP_PIN.

RNGFND4_TYPE (Rangefinder type)

DEFAULT: 0	RANGE: null

Rangefinder type

Note: This parameter functions identically to RNGFND1_TYPE.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





RNGFND5_ADDR (Bus address of sensor)

DEFAULT: 0	RANGE: 0 127

Bus address of sensor

Note: This parameter functions identically to RNGFND1_ADDR.

RNGFND5_FUNCTION (Rangefinder function)

DEFAULT: 0	RANGE: null

Rangefinder function

Note: This parameter functions identically to RNGFND1_FUNCTION.

RNGFND5_OFFSET (rangefinder offset)

DEFAULT: 0.0f	RANGE: null	UNITS: V

rangefinder offset

Note: This parameter functions identically to RNGFND1_OFFSET.

RNGFND5_ORIENT (Rangefinder orientation)

DEFAULT: AP_RANGEFINDER_DEFAULT_ORIENTATION	RANGE: null

Rangefinder orientation

Note: This parameter functions identically to RNGFND1_ORIENT.

RNGFND5_PIN (Rangefinder pin)

DEFAULT: -1	RANGE: null

Rangefinder pin

Note: This parameter functions identically to RNGFND1_PIN.

RNGFND5_PWRRNG (Powersave range)

DEFAULT: 0	RANGE: 0 32767	UNITS: m

Powersave range

Note: This parameter functions identically to RNGFND1_PWRRNG.

RNGFND5_RMETRIC (Ratiometric)

DEFAULT: 1	RANGE: null

Ratiometric

Note: This parameter functions identically to RNGFND1_RMETRIC.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





RNGFND5_SCALING (Rangefinder scaling)

DEFAULT: 3.0f	RANGE: null	UNITS: m/V

Rangefinder scaling

Note: This parameter functions identically to RNGFND1_SCALING.

RNGFND5_STOP_PIN (Rangefinder stop pin)

DEFAULT: -1	RANGE: null

Rangefinder stop pin

Note: This parameter functions identically to RNGFND1_STOP_PIN.

RNGFND5_TYPE (Rangefinder type)

DEFAULT: 0	RANGE: null

Rangefinder type

Note: This parameter functions identically to RNGFND1_TYPE.

RNGFND6_ADDR (Bus address of sensor)

DEFAULT: 0	RANGE: 0 127

Bus address of sensor

Note: This parameter functions identically to RNGFND1_ADDR.

RNGFND6_FUNCTION (Rangefinder function)

DEFAULT: 0	RANGE: null

Rangefinder function

Note: This parameter functions identically to RNGFND1_FUNCTION.

RNGFND6_OFFSET (rangefinder offset)

DEFAULT: 0.0f	RANGE: null	UNITS: V

rangefinder offset

Note: This parameter functions identically to RNGFND1_OFFSET.

RNGFND6_ORIENT (Rangefinder orientation)

DEFAULT: AP_RANGEFINDER_DEFAULT_ORIENTATION	RANGE: null

Rangefinder orientation

Note: This parameter functions identically to RNGFND1_ORIENT.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





RNGFND6_PIN (Rangefinder pin)

DEFAULT: -1	RANGE: null

Rangefinder pin

Note: This parameter functions identically to RNGFND1_PIN.

RNGFND6_PWRRNG (Powersave range)

DEFAULT: 0	RANGE: 0 32767	UNITS: m

Powersave range

Note: This parameter functions identically to RNGFND1_PWRRNG.

RNGFND6_RMETRIC (Ratiometric)

DEFAULT: 1	RANGE: null

Ratiometric

Note: This parameter functions identically to RNGFND1_RMETRIC.

RNGFND6_SCALING (Rangefinder scaling)

DEFAULT: 3.0f	RANGE: null	UNITS: m/V

Rangefinder scaling

Note: This parameter functions identically to RNGFND1_SCALING.

RNGFND6_STOP_PIN (Rangefinder stop pin)

DEFAULT: -1	RANGE: null

Rangefinder stop pin

Note: This parameter functions identically to RNGFND1_STOP_PIN.

RNGFND6_TYPE (Rangefinder type)

DEFAULT: 0	RANGE: null

Rangefinder type

Note: This parameter functions identically to RNGFND1_TYPE.

RNGFND7_ADDR (Bus address of sensor)

DEFAULT: 0	RANGE: 0 127

Bus address of sensor

Note: This parameter functions identically to RNGFND1_ADDR.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





RNGFND7_FUNCTION (Rangefinder function)

DEFAULT: 0	RANGE: null

Rangefinder function

Note: This parameter functions identically to RNGFND1_FUNCTION.

RNGFND7_OFFSET (rangefinder offset)

DEFAULT: 0.0f	RANGE: null	UNITS: V

rangefinder offset

Note: This parameter functions identically to RNGFND1_OFFSET.

RNGFND7_ORIENT (Rangefinder orientation)

DEFAULT: AP_RANGEFINDER_DEFAULT_ORIENTATION	RANGE: null

Rangefinder orientation

Note: This parameter functions identically to RNGFND1_ORIENT.

RNGFND7_PIN (Rangefinder pin)

DEFAULT: -1	RANGE: null

Rangefinder pin

Note: This parameter functions identically to RNGFND1_PIN.

RNGFND7_PWRRNG (Powersave range)

DEFAULT: 0	RANGE: 0 32767	UNITS: m

Powersave range

Note: This parameter functions identically to RNGFND1_PWRRNG.

RNGFND7_RMETRIC (Ratiometric)

DEFAULT: 1	RANGE: null

Ratiometric

Note: This parameter functions identically to RNGFND1_RMETRIC.

RNGFND7_SCALING (Rangefinder scaling)

DEFAULT: 3.0f	RANGE: null	UNITS: m/V

Rangefinder scaling

Note: This parameter functions identically to RNGFND1_SCALING.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





RNGFND7_STOP_PIN (Rangefinder stop pin)

DEFAULT: -1	RANGE: null

Rangefinder stop pin

Note: This parameter functions identically to RNGFND1_STOP_PIN.

RNGFND7_TYPE (Rangefinder type)

DEFAULT: 0	RANGE: null

Rangefinder type

Note: This parameter functions identically to RNGFND1_TYPE.

RNGFND8_ADDR (Bus address of sensor)

DEFAULT: 0	RANGE: 0 127

Bus address of sensor

Note: This parameter functions identically to RNGFND1_ADDR.

RNGFND8_FUNCTION (Rangefinder function)

DEFAULT: 0	RANGE: null

Rangefinder function

Note: This parameter functions identically to RNGFND1_FUNCTION.

RNGFND8_OFFSET (rangefinder offset)

DEFAULT: 0.0f	RANGE: null	UNITS: V

rangefinder offset

Note: This parameter functions identically to RNGFND1_OFFSET.

RNGFND8_ORIENT (Rangefinder orientation)

DEFAULT: AP_RANGEFINDER_DEFAULT_ORIENTATION	RANGE: null

Rangefinder orientation

Note: This parameter functions identically to RNGFND1_ORIENT.

RNGFND8_PIN (Rangefinder pin)

DEFAULT: -1	RANGE: null

Rangefinder pin

Note: This parameter functions identically to RNGFND1_PIN.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





RNGFND8_PWRRNG (Powersave range)

DEFAULT: 0	RANGE: 0 32767	UNITS: m

Powersave range

Note: This parameter functions identically to RNGFND1_PWRRNG.

RNGFND8_RMETRIC (Ratiometric)

DEFAULT: 1	RANGE: null

Ratiometric

Note: This parameter functions identically to RNGFND1_RMETRIC.

RNGFND8_SCALING (Rangefinder scaling)

DEFAULT: 3.0f	RANGE: null	UNITS: m/V

Rangefinder scaling

Note: This parameter functions identically to RNGFND1_SCALING.

RNGFND8_STOP_PIN (Rangefinder stop pin)

DEFAULT: -1	RANGE: null

Rangefinder stop pin

Note: This parameter functions identically to RNGFND1_STOP_PIN.

RNGFND8_TYPE (Rangefinder type)

DEFAULT: 0	RANGE: null

Rangefinder type

Note: This parameter functions identically to RNGFND1_TYPE.

RNGFND9_ADDR (Bus address of sensor)

DEFAULT: 0	RANGE: 0 127

Bus address of sensor

Note: This parameter functions identically to RNGFND1_ADDR.

RNGFND9_FUNCTION (Rangefinder function)

DEFAULT: 0	RANGE: null

Rangefinder function

Note: This parameter functions identically to RNGFND1_FUNCTION.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





RNGFND9_OFFSET (rangefinder offset)

DEFAULT: 0.0f	RANGE: null	UNITS: V

rangefinder offset

Note: This parameter functions identically to RNGFND1_OFFSET.

RNGFND9_ORIENT (Rangefinder orientation)

DEFAULT: AP_RANGEFINDER_DEFAULT_ORIENTATION	RANGE: null

Rangefinder orientation

Note: This parameter functions identically to RNGFND1_ORIENT.

RNGFND9_PIN (Rangefinder pin)

DEFAULT: -1	RANGE: null

Rangefinder pin

Note: This parameter functions identically to RNGFND1_PIN.

RNGFND9_PWRRNG (Powersave range)

DEFAULT: 0	RANGE: 0 32767	UNITS: m

Powersave range

Note: This parameter functions identically to RNGFND1_PWRRNG.

RNGFND9_RMETRIC (Ratiometric)

DEFAULT: 1	RANGE: null

Ratiometric

Note: This parameter functions identically to RNGFND1_RMETRIC.

RNGFND9_SCALING (Rangefinder scaling)

DEFAULT: 3.0f	RANGE: null	UNITS: m/V

Rangefinder scaling

Note: This parameter functions identically to RNGFND1_SCALING.

RNGFND9_STOP_PIN (Rangefinder stop pin)

DEFAULT: -1	RANGE: null

Rangefinder stop pin

Note: This parameter functions identically to RNGFND1_STOP_PIN.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





RNGFND9_TYPE (Rangefinder type)

DEFAULT: 0	RANGE: null

Rangefinder type

Note: This parameter functions identically to RNGFND1_TYPE.

RNGFNDA_ADDR (Bus address of sensor)

DEFAULT: 0	RANGE: 0 127

Bus address of sensor

Note: This parameter functions identically to RNGFND1_ADDR.

RNGFNDA_FUNCTION (Rangefinder function)

DEFAULT: 0	RANGE: null

Rangefinder function

Note: This parameter functions identically to RNGFND1_FUNCTION.

RNGFNDA_OFFSET (rangefinder offset)

DEFAULT: 0.0f	RANGE: null	UNITS: V

rangefinder offset

Note: This parameter functions identically to RNGFND1_OFFSET.

RNGFNDA_ORIENT (Rangefinder orientation)

DEFAULT: AP_RANGEFINDER_DEFAULT_ORIENTATION	RANGE: null

Rangefinder orientation

Note: This parameter functions identically to RNGFND1_ORIENT.

RNGFNDA_PIN (Rangefinder pin)

DEFAULT: -1	RANGE: null

Rangefinder pin

Note: This parameter functions identically to RNGFND1_PIN.

RNGFNDA_PWRRNG (Powersave range)

DEFAULT: 0	RANGE: 0 32767	UNITS: m

Powersave range

Note: This parameter functions identically to RNGFND1_PWRRNG.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





RNGFNDA_RMETRIC (Ratiometric)

DEFAULT: 1	RANGE: null

Ratiometric

Note: This parameter functions identically to RNGFND1_RMETRIC.

RNGFNDA_SCALING (Rangefinder scaling)

DEFAULT: 3.0f	RANGE: null	UNITS: m/V

Rangefinder scaling

Note: This parameter functions identically to RNGFND1_SCALING.

RNGFNDA_STOP_PIN (Rangefinder stop pin)

DEFAULT: -1	RANGE: null

Rangefinder stop pin

Note: This parameter functions identically to RNGFND1_STOP_PIN.

RNGFNDA_TYPE (Rangefinder type)

DEFAULT: 0	RANGE: null

Rangefinder type

Note: This parameter functions identically to RNGFND1_TYPE.

RNGFND_FILT (Rangefinder Low-Pass Filter)

DEFAULT: 0.5	RANGE: 0 5	UNITS: Hz

Rangefinder Low-Pass Filter (RNGFND_FILT)

Description

`RNGFND_FILT` smooths the "Jitter" from your distance sensor.

Many rangefinders (especially ultrasonic or low-cost Lidars) produce noisy data that can bounce around by several centimeters even when the drone is stationary. This filter removes that high-frequency noise to provide a stable altitude for the flight controller.

- **Higher Frequency:** More responsive to rapid terrain changes, but more "jittery" altitude hold.
- **Lower Frequency:** Very smooth altitude, but the drone might react slowly to a sudden obstacle or drop in terrain.

Tuning & Behavior

- **Default:** 0.5 Hz.
- **Recommendation:** Leave at **0.5 Hz** for most sensors. If your drone is "bobbing" up and down rapidly while using the rangefinder, try lowering this to **0.25 Hz**.
- **Disabled:** Setting this to **0** disables the filter (raw data is used directly).

RNGFND_GNDCLEAR (Distance (in cm) from the range finder to the ground)

DEFAULT: RANGEFINDER_GROUND_CLEARANCE_CM_DEFAULT	RANGE: 5 127	UNITS: cm

Distance (in cm) from the range finder to the ground

Note: This parameter functions identically to RNGFND1_GNDCLEAR.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





RNGFND_LANDING (Enable Rangefinder (Phase Specific))

DEFAULT: 0	RANGE: 0 1

Enable Rangefinder (Phase Specific) (RNGFND_LANDING)

Description

`RNGFND_LANDING` is the master switch for integrating distance sensor data into the **Plane** and **QuadPlane** flight logic.

Even if a rangefinder is configured and reporting data, ArduPilot will not use it for altitude control during landing or takeoff unless this parameter is set. This provides an safety layer, allowing pilots to verify sensor health before trusting it for critical landing maneuvers.

- **0: Disabled.** Rangefinder data is logged but ignored for flight control.
- **1: Enabled.** Used for landing approach, final flare, and VTOL assist.

Tuning & Behavior

- **Default:** 0.
- **Safety:** Only enable this after verifying that your rangefinder reports accurate distances and doesn't "glitch" or drop out while the vehicle is in motion.

RNGFND_LND_ORNT (Rangefinder Landing Orientation)

DEFAULT: 25	RANGE: 0 38

Rangefinder Landing Orientation (RNGFND_LND_ORNT)

Description

`RNGFND_LND_ORNT` tells the landing controller which way the "Ground" sensor is facing.

- **Standard Plane/Copter:** Set to **25 (Pitch 270 / Down)**.
- **Tailsitter (Landing on tail):** Set to **4 (Back)**.

Tuning & Behavior

- **Default:** 25 (Down).
- **Significance:** If you have multiple rangefinders (e.g. one pointing forward for obstacle avoidance and one pointing down for landing), this parameter ensures the landing logic only listens to the downward-facing one.

RNGFND_MAX_CM (Rangefinder maximum distance)

DEFAULT: 700	UNITS: cm

RNGFND_MAX_CM: Rangefinder maximum distance

Description

Maximum distance in centimeters that rangefinder can reliably read.

Values

- **Units:** cm
- **Increment:** 1
- **Default:** 700

Description

This parameter defines the "Range Limit" for the rangefinder.

- **Function:** Any reading above this value is ignored. If the sensor reports a value higher than this, ArduPilot treats it as "Out of Range" (e.g., looking into clear sky).
- **Usage:** Set this to slightly *less* than the maximum distance listed in your sensor's datasheet to ensure high reliability. If your sensor is rated for 10 meters, setting this to **900** (9m) can prevent the autopilot from acting on weak, noisy signals at the edge of the sensor's capability.
- **Altitude Limit:** If the drone flies higher than this altitude, the EKF will stop using the rangefinder for terrain following or precision landing and switch back to Barometer/GPS altitude.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





RNGFND_MAX_RATE (Rangefinder Max Update Rate)

DEFAULT: 50	RANGE: 0 200	UNITS: Hz

Rangefinder Max Update Rate (RNGFND_MAX_RATE)

Description

`RNGFND_MAX_RATE` acts as a "Data Limiter" for your distance sensor.

While high-end Lidars can report data hundreds of times per second, sending all that data over a CAN or MAVLink network can cause congestion (too much traffic). This parameter ensures that the sensor node only sends a fresh measurement at this maximum frequency, keeping the network efficient.

- **0:** No limit (Send as fast as possible).
- **50 (Default):** Sufficient for most navigation and landing tasks.

Tuning & Behavior

- **Default:** 50 Hz.
- **Recommendation:** Leave at **50** for most setups. If you have multiple distance sensors on a slow CAN bus, you might reduce this to **20** to free up bandwidth.
- **Note:** This is primarily used on **AP_Periph** CAN nodes.

RNGFND_MIN_CM (Rangefinder minimum distance)

DEFAULT: 20	UNITS: cm

RNGFND_MIN_CM: Rangefinder minimum distance

Description

Minimum distance in centimeters that rangefinder can reliably read.

Values

- **Units:** cm
- **Increment:** 1
- **Default:** 20

Description

This parameter defines the "Blind Spot" of your rangefinder.

- **Function:** Any distance reading reported by the sensor that is *less* than this value will be ignored by the autopilot and treated as "Out of Range."
- **Why it matters:** Most Sonar and Lidar sensors have a physical limit where they cannot distinguish between zero distance and a very close object. If this value is set too low, the drone might think it is at ground level while still several inches in the air, leading to a premature motor cut or a hard landing.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





RNGFND_POS (Rangefinder Position Offset (Sensor 1))

DEFAULT: 0	RANGE: -5 5	UNITS: m

Rangefinder Position Offset (RNGFND_POS)

Description

`RNGFND_POS` (often appearing as `RNGFND1_POS_X/Y/Z`) tells the autopilot exactly where the sensor is mounted on the frame.

Correctly setting the Z-offset is particularly important for landing. If the sensor is mounted 10cm below the center of gravity but this is set to 0, the EKF will think the drone is 10cm higher than it actually is, potentially causing a hard landing.

- **X:** Positive = Forward.
- **Y:** Positive = Right.
- **Z:** Positive = Down.

Tuning & Behavior

- **Default:** 0.
- **Measurement:** Measure from the flight controller (center) to the face of the sensor.

RNGFND_SQ_MIN (Rangefinder Minimum Signal Quality)

DEFAULT: 0	RANGE: 0 100	UNITS: %

Rangefinder Minimum Signal Quality (RNGFND_SQ_MIN)

Description

`RNGFND_SQ_MIN` defines the "Certainty Threshold" for your distance measurements.

Many sensors (especially acoustics/sonars used in **ArduSub**) report a confidence level or signal quality. If the return signal is weak (due to bubbles, silt, or extreme angles), the sensor may report a "Confidence" percentage. This parameter tells the autopilot to reject any measurement if the sensor's own confidence is below this value.

- **0:** Trust all data (Default).
- **50:** Reject data if the sensor is less than 50% certain.

Tuning & Behavior

- **Recommendation:** Set to **50%** if you find your depth or altitude readings "glitching" in difficult environments.
- **Context:** This is functionally similar to RNGFND1_SNR_MIN but uses a normalized percentage (0-100) instead of a raw SNR value.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY

Overview

Key Concepts

RNGFNDA_ADDR (Bus address of sensor)

DEFAULT: 0	RANGE: 0 127	UNITS: null

Bus address of sensor

Note: This parameter configures instance A. It functions identically to RNGFND1_ADDR.

RNGFNDA_FUNCTION (Function for button)

DEFAULT: 0	RANGE: null	UNITS: null

Function for button

Note: This parameter configures instance A. It functions identically to RNGFND1_FUNCTION.

RNGFNDA_MAX (Tiltrotor maximum VTOL angle)

DEFAULT: 45	RANGE: 20 80	UNITS: deg

Tiltrotor maximum VTOL angle

Note: This parameter is functionally identical to RNGFND1_MAX.

Please refer to the documentation for the primary instance for detailed configuration instructions.

RNGFNDA_MIN (Minimum RPM)

DEFAULT: 10	RANGE: null	UNITS: null

Minimum RPM

Note: This parameter is functionally identical to RNGFND1_MIN.

Please refer to the documentation for the primary instance for detailed configuration instructions.

RNGFNDA_OFFSET (Airspeed offset)

DEFAULT: 0	RANGE: null	UNITS: null

Airspeed offset

Note: This parameter configures instance A. It functions identically to RNGFND1_OFFSET.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





RNGFNDA_ORIENT (Camera Orientation)

DEFAULT: AC_PRECLAND_ORIENT_DEFAULT	RANGE: null	UNITS: null

Camera Orientation

Note: This parameter is functionally identical to RNGFND1_ORIENT.

Please refer to the documentation for the primary instance for detailed configuration instructions.

RNGFNDA_PIN (Airspeed pin)

DEFAULT: 0	RANGE: null	UNITS: null

Airspeed pin

Note: This parameter configures instance A. It functions identically to RNGFND1_PIN.

RNGFNDA_PWRRNG (Powersave range)

DEFAULT: 0	RANGE: 0 32767	UNITS: m

Powersave range

Note: This parameter configures instance A. It functions identically to RNGFND1_PWRRNG.

RNGFNDA_RECV_ID (CAN receive ID)

DEFAULT: 0	RANGE: 0 65535	UNITS: null

CAN receive ID

Note: This parameter configures instance A. It functions identically to RNGFND1_RECV_ID.

RNGFNDA_RMETRIC (Ratiometric)

DEFAULT: 1	RANGE: null	UNITS: null

Ratiometric

Note: This parameter configures instance A. It functions identically to RNGFND1_RMETRIC.

RNGFNDA_SCALING (RPM scaling)

DEFAULT: 1.0f	RANGE: null	UNITS: null

RPM scaling

Note: This parameter configures instance A. It functions identically to RNGFND1_SCALING.

RNGFNDA_SNR_MIN (Minimum RPM)

DEFAULT: 10	RANGE: null	UNITS: null

Minimum RPM

Note: This parameter configures instance A. It functions identically to RNGFND1_SNR_MIN.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





RNGFNDA_STOP_PIN (Airspeed pin)

DEFAULT: 0	RANGE: null	UNITS: null

Airspeed pin

Note: This parameter configures instance A. It functions identically to RNGFND1_STOP_PIN.

RNGFNDA_TYPE (Tiltrotor type)

DEFAULT: TILT_TYPE_CONTINUOUS	RANGE: null	UNITS: null

Tiltrotor type

Note: This parameter is functionally identical to RNGFND1_TYPE.

Please refer to the documentation for the primary instance for detailed configuration instructions.

RNGFNDA_WSP_MEDF (Moving Median Filter)

DEFAULT: 4	RANGE: 0 255	UNITS: null

Moving Median Filter

Note: This parameter configures instance A. It functions identically to RNGFND1_WSP_MEDF.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

Key Concepts

`Mount Roll angle maximum ()`

ROLL_MAX: Mount Roll angle maximum

Description

Mount Roll angle maximum.

Values

- **Range:** -180 to 180
- **Units:** deg
- **Default:** 30

Description

This parameter defines the physical "Right Tilt" limit for your camera gimbal's roll axis.

- **Usage:** Set this to the maximum angle your gimbal can tilt to the right before hitting its mechanical stops or twisting the cables too far.
- **Function:** The autopilot will prevent any automatic or manual command from exceeding this angle.

`Mount Roll angle minimum ()`

ROLL_MIN: Mount Roll angle minimum

Description

Mount Roll angle minimum.

Values

- **Range:** -180 to 180
- **Units:** deg
- **Default:** -30

Description

This parameter defines the physical "Left Tilt" limit for your camera gimbal's roll axis.

- **Usage:** Set this to the maximum angle your gimbal can tilt to the left before hitting its mechanical stops or twisting the cables too far.
- **Function:** The autopilot will prevent any automatic or manual command from exceeding this angle.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **RPM** parameter group configures the **Rotations Per Minute** (RPM) sensors. This system monitors the speed of motors or engines.

RPM data is critical for:

1. **Heli Governor:** Maintaining main rotor speed.
2. **Harmonic Notch Filter:** Tracking motor vibration frequency.
3. **Engine Health:** Monitoring ICE (Gas) engine state.

Key Concepts

1. Sensor Types (`RPMn_TYPE`)

- **1 (PWM Pin):** Connects to a Hall Effect sensor or Optical tachometer that sends a pulse for every rotation.
- **2 (ESC Telemetry):** Pulls RPM data from a BLHeli_32 or DShot ESC.
- **4 (DroneCAN):** For CAN-based RPM sensors.

2. Scaling (`RPMn_SCALING`)

Converts pulses per second into actual RPM.

- **For Hall Effect:** Set to the number of magnets on the shaft.
- **For BLHeli ESC:** Set to the number of motor poles divided by 2.

Parameter Breakdown

- `RPM1_PIN` : The GPIO pin assigned to the sensor.
- `RPM1_MAX` : Maximum expected RPM (sanity check).

Integration Guide

1. **ESC Setup:** If using DShot telemetry, set `RPM1_TYPE = 2` and ensure `SERVO_BLH_BDMASK` includes the motor you want to track.
2. **Verification:** Check the `RPM1` value in the GCS Status tab while the motor is spinning.

Developer Notes

- **Library:** `libraries/AP_RPM` .
- **Usage:** Often fed into `AP_ICEngine` or `AP_InertialSensor_HarmonicNotch` .

RPM sensor message rate ()

RPM_MSG_RATE: RPM sensor message rate

Description

This is the rate RPM sensor data is sent in Hz. Zero means no send. Each sensor with a set ID is sent in turn.

Values

- **Range:** 0 to 200
- **Units:** Hz
- **Default:** 0

Description

This parameter controls the frequency of RPM telemetry messages sent by the AP_Periph device (e.g., a CAN-based RPM sensor or ESC).

- **Usage:** Set this to a non-zero value (e.g., **10Hz or 20Hz**) to see motor or rotor RPM on your GCS or OSD.
- **Function:** The peripheral will sample the RPM and broadcast it on the CAN bus at this interval.
- **0 (Default):** Disables RPM telemetry from this device.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





RPM1_DC_ID (DroneCAN RPM Message ID)

DEFAULT: 0	RANGE: 0 65535

DroneCAN RPM Message ID (RPM1_DC_ID)

Description

`RPM1_DC_ID` identifies which specific piece of hardware on the DroneCAN bus is providing the RPM data for this instance.

This allows you to map multiple DroneCAN RPM sensors (e.g., from different ESCs) to specific slots in the autopilot's telemetry system.

- **0:** Auto-detection.
- **Value:** Specific Node ID.

RPM1_ESC_INDEX (RPM ESC Telemetry Index)

DEFAULT: 0	RANGE: 0 32

RPM ESC Telemetry Index (RPM1_ESC_INDEX)

Description

`RPM1_ESC_INDEX` allows you to "Inject" external RPM data into the ESC telemetry system.

If you have a motor that doesn't provide internal RPM telemetry (like a gas engine or an older ESC), but you have an external RPM sensor (optical or hall effect), you can use this parameter to tell the autopilot: "Take the RPM from this sensor and record it as if it were coming from ESC X."

Tuning & Behavior

- **Default:** 0 (Disabled).
- **Usage:** Set to the motor number (e.g. 1) to have this sensor's data appear in the `ESC[0]` telemetry logs and GCS fields.

RPM1_ESC_MASK (RPM ESC Telemetry Mask)

DEFAULT: 0	RANGE: 0 4294967295

RPM ESC Telemetry Mask (RPM1_ESC_MASK)

Description

`RPM1_ESC_MASK` allows a single RPM sensor to represent multiple motors.

This is useful for vehicles where multiple motors are physically linked to a single shaft (e.g. a geared twin-motor setup) or for helicopters where one RPM sensor on the main gear represents the speed for the entire power system.

- **Bit 0 (1):** Motor 1
- **Bit 1 (2):** Motor 2
- **Example:** Set to **3** (1+2) if this sensor's data applies to both Motors 1 and 2.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





RPM1_MAX (Maximum Reported RPM (Sensor 1))

DEFAULT: 100000	RANGE: 0 200000	UNITS: RPM

Maximum Reported RPM (RPM1_MAX)

Description

`RPM1_MAX` is a safety filter to prevent "Spikes" or unrealistic data from affecting the autopilot.

Electrical interference or a loose sensor wire can sometimes cause a burst of pulses that look like the motor is spinning at an impossibly high speed (e.g. 500,000 RPM). If the autopilot used this data for a governor or dynamic notch filter, it could cause a crash. This parameter sets the "sanity limit" – any reading higher than this is ignored.

Tuning & Behavior

- **Default:** 100,000 RPM.
- **Recommendation:** Set this to roughly **20% higher** than your motor's maximum possible physical RPM. For most small multicopters, 50,000 is plenty. For large gas engines, 10,000 might be appropriate.
- **Context:** Only applies when using GPIO-based (pulse counting) RPM sensors.

RPM1_MIN (Minimum Reported RPM (Sensor 1))

DEFAULT: 10	RANGE: 0 1000	UNITS: RPM

Minimum Reported RPM (RPM1_MIN)

Description

`RPM1_MIN` acts as a "Noise Gate" for the RPM sensor.

At extremely low speeds or when the motor is stopped, electrical noise on the sensor pin can sometimes look like a valid pulse, causing the GCS to show "Ghost RPM" (e.g. 2 or 3 RPM). This parameter tells the autopilot to ignore any calculated RPM below this value and simply report **0**.

Tuning & Behavior

- **Default:** 10 RPM.
- **Usage:** For large, slow-moving rotors (like on a large helicopter), you might need to lower this to **5**. For high-speed racing motors, you can increase this to **100** or more to ensure a clean zero-reading when disarmed.
- **Context:** Only applies when using GPIO-based (pulse counting) RPM sensors.

RPM1_MIN_QUAL (Minimum RPM Quality (Sensor 1))

DEFAULT: 0.5	RANGE: 0 1.0

Minimum RPM Quality (RPM1_MIN_QUAL)

Description

`RPM1_MIN_QUAL` defines the "Certainty Threshold" for digital RPM sensors.

Some RPM sensors (especially those reporting via DroneCAN) calculate a confidence or quality score based on the stability of the signal. If the signal is weak or inconsistent (due to electrical noise or poor sensor alignment), the quality score drops. This parameter tells the autopilot to reject any data if the quality falls below this level.

- **0.5 (Default):** 50% confidence required.
- **1.0:** Perfectly clean signal required.

Tuning & Behavior

- **Recommendation:** Leave at **0.5**. If you see your RPM readings "flickering" to 0 during high-speed flight, you may need to decrease this value slightly or check your wiring for noise.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





RPM1_PIN (RPM Input Pin (Sensor 1))

DEFAULT: -1	RANGE: -1 103

RPM Input Pin (RPM1_PIN)

Description

`RPM1_PIN` tells the autopilot which digital port on the flight controller is connected to your RPM sensor (e.g., an optical sensor, a hall effect sensor, or the RPM output from an ESC).

- **-1:** Disabled.
- **0-103:** The pin number assigned to the physical input.

Tuning & Behavior

- **Default:** -1.
- **Setup:** Refer to your flight controller's documentation to find the correct GPIO pin number for the port you are using.
- **Requirement:** Only used if `RPM1_TYPE` is set to **1 (GPIO)**.

RPM1_SCALING (RPM Scaling (Sensor 1))

DEFAULT: 1.0	RANGE: 0 100

RPM Scaling (RPM1_SCALING)

Description

`RPM1_SCALING` is the "Gear Ratio" or "Pole Count" setting for your RPM sensor. It tells the autopilot how many pulses from the sensor equal one full revolution of the motor or shaft.

$$\text{RPM} = \text{Sensor Rate} \times \text{RPM1_SCALING}$$

- **Example (Hall Effect):** If you have 1 magnet on the shaft, set Scaling to **1.0**.
- **Example (Pole Count):** If using an ESC sensor on a 14-pole motor, the sensor sends 7 pulses per rev. Set Scaling to $1/7 approx$ **0.142**.

Tuning & Behavior

- **Default:** 1.0.
- **Accuracy:** Verify the reported RPM against a handheld tachometer. If the reported RPM is exactly double the actual RPM, divide your Scaling value by 2.

RPM1_TYPE (RPM Sensor Type (Sensor 1))

DEFAULT: 0	RANGE: 0 7

RPM Sensor Type (RPM1_TYPE)

Description

`RPM1_TYPE` enables real-time monitoring of motor or rotor revolutions per minute (RPM).

This is highly recommended for Helicopters (to monitor head speed) and for large Internal Combustion Engine (ICE) vehicles. It is also used by the Harmonic Notch Filter to dynamically track and eliminate motor-induced vibrations.

- **0: None.** Disabled.
- **2: GPIO.** For Hall effect or optical sensors connected to a digital pin.
- **4: Harmonic Notch.** Derived from internal motor controllers (SITL/Special use).
- **5: ESC Telemetry.** Averages RPM data from all ESCs reporting telemetry.
- **7: DroneCAN.** For CAN-based RPM sensors.

Tuning & Behavior

- **Reboot Required:** Yes.
- **Integration:** Once enabled, you must configure the input pin (RPM1_PIN) if using GPIO, and the scaling (RPM1_SCALING).

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY

RPM2_DC_ID (DroneCAN Sensor ID)

DEFAULT: -1	RANGE: -1 10

DroneCAN Sensor ID

Note: This parameter functions identically to RPM1_DC_ID.

RPM2_ESC_INDEX (ESC Telemetry Index to write RPM to)

DEFAULT: 0	RANGE: 0 10

ESC Telemetry Index to write RPM to

Note: This parameter functions identically to RPM1_ESC_INDEX.

RPM2_ESC_MASK (Bitmask of ESC telemetry channels to average)

DEFAULT: 0	RANGE: null

Bitmask of ESC telemetry channels to average

Note: This parameter functions identically to RPM1_ESC_MASK.

RPM2_MAX (Maximum RPM)

DEFAULT: 100000	RANGE: null

Maximum RPM

Note: This parameter functions identically to RPM1_MAX.

RPM2_MIN (Minimum RPM)

DEFAULT: 10	RANGE: null

Minimum RPM

Note: This parameter functions identically to RPM1_MIN.

RPM2_MIN_QUAL (Minimum Quality)

DEFAULT: 0.5	RANGE: null

Minimum Quality

Note: This parameter functions identically to RPM1_MIN_QUAL.

RPM2_PIN (Input pin number)

DEFAULT: -1	RANGE: null

Input pin number

Note: This parameter functions identically to RPM1_PIN.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





RPM2_SCALING (RPM scaling)

DEFAULT: 1.0f	RANGE: null

RPM scaling

Note: This parameter functions identically to RPM1_SCALING.

RPM2_TYPE (RPM type)

DEFAULT: 0	RANGE: null

RPM type

Note: This parameter functions identically to RPM1_TYPE.

RPM3_DC_ID (DroneCAN Sensor ID)

DEFAULT: -1	RANGE: -1 10

DroneCAN Sensor ID

Note: This parameter functions identically to RPM1_DC_ID.

RPM3_ESC_INDEX (ESC Telemetry Index to write RPM to)

DEFAULT: 0	RANGE: 0 10

ESC Telemetry Index to write RPM to

Note: This parameter functions identically to RPM1_ESC_INDEX.

RPM3_ESC_MASK (Bitmask of ESC telemetry channels to average)

DEFAULT: 0	RANGE: null

Bitmask of ESC telemetry channels to average

Note: This parameter functions identically to RPM1_ESC_MASK.

RPM3_MAX (Maximum RPM)

DEFAULT: 100000	RANGE: null

Maximum RPM

Note: This parameter functions identically to RPM1_MAX.

RPM3_MIN (Minimum RPM)

DEFAULT: 10	RANGE: null

Minimum RPM

Note: This parameter functions identically to RPM1_MIN.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





RPM3_MIN_QUAL (Minimum Quality)

DEFAULT: 0.5	RANGE: null

Minimum Quality

Note: This parameter functions identically to RPM1_MIN_QUAL.

RPM3_PIN (Input pin number)

DEFAULT: -1	RANGE: null

Input pin number

Note: This parameter functions identically to RPM1_PIN.

RPM3_SCALING (RPM scaling)

DEFAULT: 1.0f	RANGE: null

RPM scaling

Note: This parameter functions identically to RPM1_SCALING.

RPM3_TYPE (RPM type)

DEFAULT: 0	RANGE: null

RPM type

Note: This parameter functions identically to RPM1_TYPE.

RPM4_DC_ID (DroneCAN Sensor ID)

DEFAULT: -1	RANGE: -1 10

DroneCAN Sensor ID

Note: This parameter functions identically to RPM1_DC_ID.

RPM4_ESC_INDEX (ESC Telemetry Index to write RPM to)

DEFAULT: 0	RANGE: 0 10

ESC Telemetry Index to write RPM to

Note: This parameter functions identically to RPM1_ESC_INDEX.

RPM4_ESC_MASK (Bitmask of ESC telemetry channels to average)

DEFAULT: 0	RANGE: null

Bitmask of ESC telemetry channels to average

Note: This parameter functions identically to RPM1_ESC_MASK.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





RPM4_MAX (Maximum RPM)

DEFAULT: 100000	RANGE: null

Maximum RPM

Note: This parameter functions identically to RPM1_MAX.

RPM4_MIN (Minimum RPM)

DEFAULT: 10	RANGE: null

Minimum RPM

Note: This parameter functions identically to RPM1_MIN.

RPM4_MIN_QUAL (Minimum Quality)

DEFAULT: 0.5	RANGE: null

Minimum Quality

Note: This parameter functions identically to RPM1_MIN_QUAL.

RPM4_PIN (Input pin number)

DEFAULT: -1	RANGE: null

Input pin number

Note: This parameter functions identically to RPM1_PIN.

RPM4_SCALING (RPM scaling)

DEFAULT: 1.0f	RANGE: null

RPM scaling

Note: This parameter functions identically to RPM1_SCALING.

RPM4_TYPE (RPM type)

DEFAULT: 0	RANGE: null

RPM type

Note: This parameter functions identically to RPM1_TYPE.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **RSSI** parameter group configures the **Received Signal Strength Indication**. This system allows the autopilot to monitor the quality of the radio link between the RC transmitter and the vehicle.

Key Concepts

1. RSSI Types (`RSSI_TYPE`)

How does the receiver tell the autopilot the signal strength?

- **1 (Analog Pin):** Legacy. High voltage (3.3V) on a dedicated wire means full signal.
- **2 (PWM/RC Channel):** The receiver sends RSSI as a standard PWM value on a spare channel (e.g., Channel 8).
- **3 (Receiver Protocol):** Modern. RSSI is embedded directly in the digital data stream (SBUS, CRSF, FPort). **Highly Recommended.**

2. Scaling (`RSSI_CHAN_LOW` / `HIGH`)

Maps the raw input (PWM or Voltage) to 0-100%.

Parameter Breakdown

- `RSSI_TYPE`: Input method selection.
- `RSSI_ANA_PIN`: (Analog only) ADC pin number.

Integration Guide

- **Modern Setup:** If using CRSF (ELRS) or SBus, set `RSSI_TYPE = 3`. No further calibration is needed.
- **Display:** Ensure the "RSSI" widget is enabled in your OSD to monitor link health during flight.

Developer Notes

- **Library:** `libraries/AP_RSSI`.

`RSSI_ANA_PIN` (Receiver RSSI Sensing Pin)

DEFAULT: 0	RANGE: -1 103

Receiver RSSI Sensing Pin (RSSI_ANA_PIN)

Description

`RSSI_ANA_PIN` tells the autopilot where to look for the "Signal Strength" wire from your radio receiver.

This is only used if you are physically wiring an RSSI output from your receiver to the flight controller.

- **Analog Input:** Use an ADC-capable pin (like an analog airspeed port).
- **PWM Input:** Use a digital GPIO pin (typically a servo output pin configured as a GPIO).

Tuning & Behavior

- **Default:** -1 (Disabled).
- **Calibration:** You must also set the RSSI_PIN_LOW and RSSI_PIN_HIGH voltages to calibrate the signal correctly.
- **Note:** If you are using MAVLink-based RSSI (common with ELRS/Crossfire), this parameter is ignored.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





RSSI_CHAN_HIGH (RSSI Strong Signal PWM)

DEFAULT: 2000	RANGE: 0 2000	UNITS: PWM

RSSI Strong Signal PWM (RSSI_CHAN_HIGH)

Description

`RSSI_CHAN_HIGH` calibrates the "Maximum" point for **PWM-based** RSSI sensors or auxiliary channels.

If your receiver is sending RSSI as a PWM signal, this parameter defines the 100% threshold.

Tuning & Behavior

- **Default:** 2000.
- **Calibration:** Place your transmitter near the drone and observe the RSSI PWM value in the GCS. Enter that value here.

RSSI_CHAN_LOW (RSSI Weak Signal PWM)

DEFAULT: 1000	RANGE: 0 2000	UNITS: PWM

RSSI Weak Signal PWM (RSSI_CHAN_LOW)

Description

`RSSI_CHAN_LOW` calibrates the "Minimum" point for **PWM-based** RSSI sensors or auxiliary channels.

If your receiver is sending RSSI as a PWM signal (where a shorter pulse means a weaker signal), this parameter defines the 0% threshold.

Tuning & Behavior

- **Default:** 1000.
- **Calibration:** Turn off your transmitter and observe the RSSI PWM value in the GCS status tab. Enter that value here.

RSSI_PIN_HIGH (RSSI Strong Signal Voltage)

DEFAULT: 3.3	RANGE: 0 5.0	UNITS: V

RSSI Strong Signal Voltage (RSSI_PIN_HIGH)

Description

`RSSI_PIN_HIGH` calibrates the "100% Signal" point for analog RSSI sensors.

You measure this by placing the transmitter immediately next to the drone's receiver. The maximum voltage output by the sensor is entered here.

Tuning & Behavior

- **Default:** 3.3V.
- **Accuracy:** Correctly setting this ensures your GCS shows a full signal bar when you are flying nearby.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





RSSI_PIN_LOW (RSSI Weak Signal Voltage)

DEFAULT: 0	RANGE: 0 5.0	UNITS: V

RSSI Weak Signal Voltage (RSSI_PIN_LOW)

Description

`RSSI_PIN_LOW` calibrates the "0% Signal" point for analog RSSI sensors.

You measure this by powering on the drone but keeping the transmitter turned off (or at extreme distance). The voltage reported by the sensor at this state is entered here.

Tuning & Behavior

- **Inversion:** If your receiver outputs a *higher* voltage for a *weaker* signal (inverted), simply set `RSSI_PIN_LOW` to the high voltage and `RSSI_PIN_HIGH` to the low voltage.

RSSI_TYPE (RSSI Type)

DEFAULT: 0	RANGE: 0 5

RSSI_TYPE: RSSI Type

Description

Radio Receiver RSSI type. This selects the method used to read the signal strength (RSSI) from your radio link.

Values

Value	Meaning
0	Disabled
1	Analog Pin (Voltage on a specific flight controller pin)
2	RC Channel PWM (RSSI value embedded in a spare RC channel)
3	Receiver Protocol (Digital info from SBUS, CRSF, or FPort)
4	PWM Input Pin (RSSI sent as a high-frequency PWM signal)
5	Telemetry Radio RSSI (RSSI reported back from SiK/MAVLink radios)

- **Default:** 0

Description

`RSSI_TYPE` tells the autopilot how to measure the "health" of your radio control link.

- **Receiver Protocol (3):** The modern standard. If you are using ELRS, Crossfire, or FrSky FPort, the RSSI is sent digitally inside the data stream. Set this to 3 and no other wiring is needed.
- **RC Channel (2):** Common for older FrSky setups where you map RSSI to an auxiliary channel (e.g., Channel 8 or 16).
- **Analog (1):** Used for very old receivers that output a 0-3.3V signal.

Source Code

ardupilot/libraries/AP_RSSI/AP_RSSI.cpp

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **RST** parameter group (specifically `RST_SWITCH_CH`) configures the manual **Reset** behavior for ArduRover.

Key Concepts

1. The Reset Switch

In some scenarios, a rover might be stuck in a failsafe state or a geofence breach. This parameter allows the pilot to assign a specific RC channel to a "Reset" button.

- **Action:** When the switch is toggled, the autopilot attempts to clear the current error state and return to the last active flight mode.

Developer Notes

- **Library:** `ArduRover/Parameters.cpp`.
- **Context:** Only relevant for **ArduRover**.

RST_SWITCH_CH (Reset Switch Channel)

DEFAULT: 0	RANGE: 0 16

Reset Switch Channel (RST_SWITCH_CH)

Description

This parameter assigns a specific RC switch channel to the "Reset" function. This is primarily used in Rover to clear a Geofence Breach state or to return control to the pilot after a failsafe event has triggered an autonomous mode change.

The Mathematics

N/A. Logic trigger.

The Engineer's View

Defined in `Rover/Parameters.cpp`. Checks for a PWM change (low to high) on the specified channel.

Tuning & Behavior

- **Default Value:** 0 (Disabled)
- **Set to:** An unused channel (e.g., 7 or 8) mapped to a momentary switch on your transmitter.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **RTC** parameter group configures the **System Clock** management. Maintaining accurate UTC time is essential for logging, timestamping mission waypoints, and ensuring GNSS synchronization.

Since most flight controllers do not have a dedicated battery-backed clock chip, the autopilot must synchronize its time from external sources.

Key Concepts

1. Time Sources (`RTC_TYPES`)

Defines which sources the autopilot is allowed to trust to set the time.

- **Bit 0 (GPS):** The most accurate source. Sets time once a 3D lock is achieved.
- **Bit 1 (MAVLink):** Receives time from the Ground Control Station.
- **Bit 2 (Hardware):** Internal clock chip (if present).

2. Timezone (`RTC_TZ_MIN`)

Allows setting a local time offset (in minutes) for the OSD or logging.

- **Note:** Internal logs are always kept in UTC; this only affects displayed time.

Parameter Breakdown

- `RTC_TYPES`: Bitmask of allowed sources.
- `RTC_TZ_MIN`: Offset from UTC (minutes).

Developer Notes

- **Library:** `libraries/AP_RTC`.

RTC_TYPES (Allowed sources of RTC time)

DEFAULT: 1	RANGE: 0 7	UNITS: Bitmask

Allowed sources of RTC time (RTC_TYPES)

Description

This parameter controls which external sources are trusted to set the autopilot's internal Real Time Clock (RTC). Maintaining accurate UTC time is essential for log timestamping and GNSS operations.

The Mathematics

Bitmask.

The Engineer's View

Defined in `libraries/AP_RTC/AP_RTC.cpp`.

- **Bit 0:** GPS (Preferred).
- **Bit 1:** MAVLink `SYSTEM_TIME` messages (from GCS).
- **Bit 2:** Hardware RTC (on-board backup battery).

Tuning & Behavior

- **Default Value:** 1 (GPS Only)
- **Recomendation:** Enable Bit 2 if your board has a backup battery for the RTC. Enable Bit 1 if flying indoors without GPS but connected to a Ground Station.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY

`Timezone offset from UTC` ()

RTC_TZ_MIN: Timezone offset from UTC

Description

Adds offset in +- minutes from UTC to calculate local time.

Values

- **Range:** -720 to +840 (minutes)
- **Increment:** 1
- **Default:** 0

Description

This parameter adjusts the autopilot's internal clock to match your local time zone.

- **Usage:** Set this if you want the logs or the On-Screen Display (OSD) to show local time instead of UTC.
- **Example:** For Eastern Standard Time (EST, UTC-5), set this to **-300**. For Central European Summer Time (CEST, UTC+2), set this to **120**.
- **Effect:** Primarily affects the `SYSTEM_TIME` MAVLink message and the time displayed on OSD widgets. It does NOT affect the EKF or navigation, which always use UTC.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





RETURN TO LAUNCH CONFIGURATION (RTL)

Overview

The **RTL** parameter group configures the **Return to Launch** (RTL) flight mode. This is the primary safety mode for ArduPilot vehicles, providing an autonomous way for the vehicle to return to its starting point and land.

RTL is often triggered automatically by Failsafes (Radio, Battery, GCS) or manually by the pilot.

Key Concepts

1. Return Altitude (`RTL_ALT`)

Defines the altitude the vehicle will climb (or descend) to before starting its trip back to Home.

- **Safety:** This must be high enough to clear any trees, buildings, or hills between the drone and Home.

2. Final Phase (`RTL_ALT_FINAL`)

Determines what the drone does once it arrives back over Home.

- **0:** Land immediately.
- **>0:** Loiter at this altitude (meters) until the pilot takes over or the battery fails.

3. Climb Behavior (`RTL_CLIMB_MIN`)

Ensures that if the vehicle is already above `RTL_ALT`, it doesn't dive until it gets home. It also ensures a minimum climb if already low.

Parameter Breakdown

- `RTL_ALT`: Return altitude (centimeters or meters, check units).
- `RTL_SPEED`: Speed (m/s) during the return trip.
- `RTL_AUTOLAND`: (Plane) Automatically starts a landing sequence upon arrival.

Integration Guide

- **Setup:** Always set `RTL_ALT` to at least 20 meters higher than the tallest obstacle in your flying area.
- **Verify:** Perform a manual RTL test early in your flight to ensure the vehicle behaves as expected.

Developer Notes

- **Library:** `ArduCopter/mode_rtl.cpp`, `ArduPlane/mode_rtl.cpp`.
- **Navigation:** Uses the standard position/velocity controllers to execute the path home.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





RTL_ALT (RTL Altitude)

DEFAULT: 1500	RANGE: 0 8000	UNITS: cm

RTL Altitude (RTL_ALT)

Description

`RTL_ALT` sets the "Safe Height" for Return-to-Launch.

When RTL is triggered (e.g. by a failsafe or switch), the drone will climb to this altitude before flying home. This ensures it clears trees, buildings, and hills.

- **0:** Stay at current altitude (Dangerous if low).
- **1500 (Default):** 15 meters.
- **Recommendation:** Set to at least **2000-3000 (20-30m)** to clear most suburban obstacles.

The Mathematics

$$\text{Target Alt} = \max(\text{Current Alt}, \text{RTL_ALT})$$

Tuning & Behavior

- **Default Value:** 1500 cm.
- **Note:** If the drone is already *higher* than this value, it will stay at its current altitude to fly home. It will never descend to `RTL_ALT`.

RTL_ALTITUDE (RTL Altitude)

DEFAULT: 15	RANGE: -1 500	UNITS: m

RTL Altitude (RTL_ALTITUDE)

Description

`RTL_ALTITUDE` defines your "Safe Return Height."

When the drone is commanded to Return-to-Launch (RTL), it will first check its current height. If it is below this value, it will climb to this altitude before starting the journey home. This ensures the drone flies over trees, buildings, and power lines instead of through them.

- **-1:** Maintains the drone's current altitude (No climb or descent).
- **0:** Drone will fly home at its current altitude.
- **Value (m):** Target altitude above the takeoff point.

Tuning & Behavior

- **Default:** 15 meters.
- **Recommendation:** Set this to at least **10 meters higher** than the tallest obstacle in your flight area.
- **Note:** If the drone is already *above* this altitude when RTL is triggered, it will stay at that higher altitude while returning home (unless `RTL_CONE` logic is used).

RTL_ALT_FINAL (RTL Final Altitude)

DEFAULT: 0	RANGE: 0 1000	UNITS: cm

RTL Final Altitude (RTL_ALT_FINAL)

Description

`RTL_ALT_FINAL` determines what the drone does after it arrives home.

- **0 (Default):** Land immediately.
- **> 0:** Hover at this altitude.

Tuning & Behavior

- **Default Value:** 0.
- **Recommendation:** Use **0** for automated safety. Use **200 (2m)** if you want the drone to come home and wait for you to catch it or manually land it.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





RTL_ALT_TYPE (RTL Altitude Type)

DEFAULT: 0	RANGE: 0 1

RTL Altitude Type (RTL_ALT_TYPE)

Description

`RTL_ALT_TYPE` enables Terrain Following during RTL.

- **0 (Default):** Relative to Home. The drone flies at `RTL_ALT` above the takeoff point.
- **1:** Terrain. The drone flies at `RTL_ALT` above the ground beneath it (requires a Rangefinder or Terrain Database).

Tuning & Behavior

- **Default Value:** 0.
- **Recommendation:** Use **1** if you are flying in hilly terrain to ensure the drone climbs over ridges on its way home.

RTL_AUTOLAND (RTL auto land)

DEFAULT: 0	RANGE: 0 3

RTL auto land (RTL_AUTOLAND)

Description

This parameter controls the behavior of the aircraft when it reaches the Return-To-Launch (RTL) destination. Instead of simply loitering at the home or rally point, the aircraft can be configured to automatically initiate a landing sequence. This feature allows for fully autonomous recovery missions but requires careful setup of `DO_LAND_START` mission items to define the approach path.

The Mathematics

The parameter acts as a state machine configuration for the RTL flight mode controller.

$$\text{Behavior} = \begin{cases} \text{Loiter at Home} & \text{if RTL_AUTOLAND} = 0 \\ \text{Land (Rally)} & \text{if RTL_AUTOLAND} = 1 \\ \text{Land (Home)} & \text{if RTL_AUTOLAND} \end{cases}$$

The Engineer's View

Defined in `ArduPlane/Parameters.cpp`. The values correspond to the `RtlAutoland` enum:

- `RTL_DISABLE` (0): Standard RTL behavior (loiter over Home/Rally).
- `RTL_RALLY` (1): Land at the nearest Rally point if defined and in range.
- `RTL_HOME` (2): Land at the Home position.
- `RTL_DISABLE_WO_CHECK` (3): Do not land, and suppress the pre-arm check that normally requires a `DO_LAND_START` item if `RTL_AUTOLAND` is active.

Tuning & Behavior

- **0 (Disabled):** The aircraft will return to home (or rally point) and loiter indefinitely until the battery fails or manual control is taken.
- **1 (Rally):** Initiates landing at the nearest valid Rally Point.
- **2 (Home):** Initiates landing at Home.
- **3 (Disable w/o Check):** Useful for using `DO_LAND_START` as a pure "Go Around" or "Abort" waypoint without forcing an automatic landing at the end of an RTL.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





RTL_CLIMB_MIN (RTL Minimum Climb)

DEFAULT: 0	RANGE: 0 1000	UNITS: cm

RTL Minimum Climb (RTL_CLIMB_MIN)

Description

`RTL_CLIMB_MIN` ensures the drone does *something* vertical when RTL is triggered.

If `RTL_ALT` is 15m and you are at 20m, the drone would normally just fly home. If you set `RTL_CLIMB_MIN` to 500 (5m), the drone will climb to 25m first. This is a visual cue to the pilot that RTL has engaged.

Tuning & Behavior

- **Default Value:** 0.
- **Recommendation:** Set to **0** unless you specifically want this "Jump" behavior.

RTL_CONE_SLOPE (RTL Cone Slope)

DEFAULT: 3.0	RANGE: 0.5 10.0

RTL Cone Slope (RTL_CONE_SLOPE)

Description

`RTL_CONE_SLOPE` creates a virtual funnel above the landing spot.

If the drone is very close to home but very high up, it doesn't need to climb to `RTL_ALT`. It can just descend. This parameter defines how steep that descent can be.

- **Low Value:** Shallow cone. Drone flies further out before descending.
- **High Value:** Steep cone. Drone can descend steeply.

Tuning & Behavior

- **Default Value:** 3.0 (3:1 slope).
- **0:** Disabled (Always climb to `RTL_ALT` first).

RTL_LOIT_TIME (RTL Loiter Time)

DEFAULT: 5000	RANGE: 0 60000	UNITS: ms

RTL Loiter Time (RTL_LOIT_TIME)

Description

`RTL_LOIT_TIME` gives you a chance to take control.

When the drone arrives overhead, it will pause for this duration. This is your opportunity to switch modes if you want to land manually.

Tuning & Behavior

- **Default Value:** 5000 ms (5 seconds).
- **Recommendation:** Keep at least **5000**.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





RTL_OPTIONS (RTL Mode Options)

DEFAULT: 0	RANGE: 0 4

RTL Mode Options (RTL_OPTIONS)

Description

`RTL_OPTIONS` allows you to customize how the drone behaves while flying home.

- **Bit 2 (4): Ignore Pilot Yaw.**
  - **Unset (0):** The pilot can yaw (spin) the drone while it is returning home. This is useful for looking around with a camera during the flight.
  - **Set (1):** The drone ignores yaw stick inputs. The nose will remain pointed towards the home point (or as defined by `WP_YAW_BEHAVIOR`).

Tuning & Behavior

- **Default Value:** 0.
- **Recommendation:** Leave at **0** to allow pilot intervention/camera pointing.

RTL_RADIUS (RTL Loiter Radius)

DEFAULT: 0	RANGE: 0 500	UNITS: m

RTL Loiter Radius (RTL_RADIUS)

Description

`RTL_RADIUS` defines how tight the plane circles when it gets home.

- **0 (Default):** Use `WP_LOITER_RAD`.
- **> 0:** Use this specific radius for RTL only.

Tuning & Behavior

- **Default Value:** 0.
- **Recommendation:** Keep at **0** to ensure consistent behavior with standard loiter modes.

RTL_SPEED (RTL Speed)

DEFAULT: 0	RANGE: 0 2000	UNITS: cm/s

RTL Speed (RTL_SPEED)

Description

`RTL_SPEED` sets the "Get Home Fast" velocity.

- **0 (Default):** Use `WPNAV_SPEED` (Waypoint Speed).
- **> 0:** Use this specific speed for RTL.

Tuning & Behavior

- **Default Value:** 0.
- **Recommendation:** Set slightly higher than your standard waypoint speed if you want the drone to return quickly in an emergency, but ensure it doesn't drain the battery so fast it crashes before arriving.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **RUDD** parameter group configures specific behaviors for the **Rudder** control axis on fixed-wing aircraft.

Key Concepts

1. Dual-Tail Gains (`RUDD_DT_GAIN`)

This parameter is specific to aircraft with **Dual Tail** (Twin Rudder) configurations. It allows for independent scaling of the rudder authority when the tails are acting together vs. opposing each other.

Developer Notes

- **Library:** `ArduPlane/servos.cpp`.
- **See Also:** RLL and PTCH for main flight control surfaces.

`rudder differential thrust gain` ()

RUDD_DT_GAIN: rudder differential thrust gain

Description

Gain control from rudder to differential thrust.

Values

- **Range:** 0 to 100
- **Units:** %
- **Increment:** 1
- **Default:** 10

Description

This parameter is used on twin-engine aircraft to assist with ground steering and yaw control using **Differential Thrust**.

- **How it works:** When you move the rudder stick, the autopilot increases the RPM of one motor and decreases the other.
- **Usage:** Set this to a non-zero value if you have motors assigned to `ThrottleLeft` and `ThrottleRight`.
- **Recommendation:** Start with **10%** and increase if the aircraft is sluggish to turn on the ground. Be careful with high values (>40%) as they can cause high-speed instability if one motor fails or responds differently.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

Key Concepts

RUDDER_ONLY (Rudder only aircraft)

DEFAULT: 0	RANGE: 0 1

Rudder only aircraft (RUDDER_ONLY)

Description

This parameter configures ArduPlane for "2-Channel" aircraft (Throttle + Rudder/Elevator) that lack ailerons, such as polyhedral gliders or vintage designs. When enabled, the autopilot's roll control logic drives the rudder servo instead of aileron servos to turn the aircraft.

The Mathematics

The roll controller output (PID) is re-routed:

$$\text{Rudder Output} = \text{Yaw Control} + \text{Roll Control} \times \text{KFF_RDDRMIX}$$

The Engineer's View

Defined in `ArduPlane/Parameters.cpp`.

- **Setup:** The transmitter's Aileron stick should be mapped to the `RCMAP_YAW` channel. The Rudder servo connects to the Yaw output.
- **Constraint:** Ground steering logic is disabled or altered in this mode.
- **Damping:** Requires `YAW2SRV_DAMP` tuning to prevent oscillation.

Tuning & Behavior

- **Default Value:** 0 (Disabled)
- **1:** Enable Rudder Only mode.
- **Note:** You must ensure `KFF_RDDRMIX` is set (typically 1.0) for the roll controller to have authority over the rudder.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **SAFE** parameter group (specifically `SAFE_DISARM_PWM`) configures the output safety state for the **Antenna Tracker**.

Key Concepts

1. Disarmed Safety

When the tracker is disarmed or in "Stop" mode, it is important to define what the servos do.

- `SAFE_DISARM_PWM`:
  - **0:** Zero PWM. Servos are unpowered/limp.
  - **1:** Trim PWM. Servos hold their center (neutral) position.

Developer Notes

- **Library:** `AntennaTracker/Parameters.cpp`.

SAFE_DISARM_PWM (PWM that will be output when disarmed or in stop mode)

DEFAULT: 0	RANGE: 0 1

PWM that will be output when disarmed or in stop mode (SAFE_DISARM_PWM)

Description

This parameter determines the signal sent to the servos when the Antenna Tracker is in a disarmed state or "Stop" mode. It is a safety feature to ensure the tracker does not move unexpectedly or holds a neutral position when not actively tracking.

The Mathematics

The parameter acts as a selector for the output logic:

- **0:** Zero PWM (Output = 0). Depending on the hardware/HAL, this may mean sending no signal pulses (servos go limp) or sending a 0us pulse width.
- **1:** Trim PWM. The servos are commanded to their trim (center) position.

The Engineer's View

Defined in `AntennaTracker/Parameters.cpp` and used in `tracking.cpp`. The values map to the `PWMDisarmed` enum:

- `ZERO` (0)
- `TRIM` (1)

Tuning & Behavior

- **Default Value:** 0
- **0 (Zero PWM):** Servos are typically unpowered (signal-wise) and can be moved by hand (if digital servos support this) or stay in their last position.
- **1 (Trim PWM):** Servos are actively held at their centered/neutral position.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **SAIL** parameter group configures the **Autonomous Sailing** logic for **ArduRover**.

This library allows rovers (Boats) to navigate using wind power. It manages complex sailing behaviors like tacking, jibing, and sail trim based on wind direction.

Key Concepts

1. Sail Control Types (`SAIL_TYPE`)

Defines the hardware used for sail adjustment.

- **0:** Manual (Servo direct).
- **1:** Winch (Standard servo/winch).
- **2:** Wing Sail (Rotating mast/wing).

2. Upwind Navigation (`SAIL_NO_GO_ANGLE`)

Boats cannot sail directly into the wind. If a waypoint is upwind, the autopilot will autonomously calculate a "tack" (zig-zag) path to reach it.

- `SAIL_NO_GO_ANGLE`: The angle relative to the wind (typically 45°) where the boat cannot maintain speed.

3. Safety (`SAIL_HEEL_MAX`)

To prevent capsizing, the autopilot monitors the roll angle. If the boat heels past this limit, the sails are automatically eased to dump wind pressure.

Parameter Breakdown

- `SAIL_ENABLE`: Master switch.
- `SAIL_ANGLE_IDEAL`: Optimal sail angle for maximum thrust.
- `SAIL_WNDSPD_MIN`: Minimum wind speed required to attempt sailing.

Integration Guide

1. **Hardware:** Requires a **Wind Vane** sensor (`WNDVN_TYPE`) to see the wind direction.
2. **Enable:** Set `SAIL_ENABLE = 1`.
3. **Calibrate:** Ensure the wind vane is aligned with the boat's nose.
4. **Mode:** Use `AUTO`, `GUIDED`, or `ACRO` to see the sailing logic in action.

Developer Notes

- **Library:** `Rover/sailboat.cpp`.
- **Navigation:** Integrated with the L1 controller to handle tacking and crosstrack limits.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





SAIL_ANGLE_IDEAL (Sail ideal angle)

DEFAULT: 25	RANGE: 0 90	UNITS: deg

Sail ideal angle (SAIL_ANGLE_IDEAL)

Description

This parameter defines the optimal angle of attack for the mainsail to generate maximum thrust. The autopilot uses this value, combined with the wind vane reading (apparent wind direction), to automatically trim the sail servo.

The Mathematics

$$\theta_\text{sail} = \theta_\text{wind} - \text{SAIL_ANGLE_IDEAL}$$

(Simplified logic for sail setting).

The Engineer's View

Defined in `Rover/sailboat.cpp`. Used in the `update_mainsail()` function.

- **Physics:** Represents the angle where the lift/drag ratio of the sail is maximized.

Tuning & Behavior

- **Default Value:** 25 deg
- **Adjustment:**
  - **Soft Sails:** Typically 20-30 degrees.
  - **Wing Sails:** May require different settings based on the airfoil profile.

SAIL_ANGLE_MAX (Sail Maximum Angle)

DEFAULT: 90	RANGE: 0 90	UNITS: deg

Sail Maximum Angle (SAIL_ANGLE_MAX)

Description

`SAIL_ANGLE_MAX` defines the mechanical limit of your sailboat's rig.

It represents the angle of the boom relative to the boat's longitudinal axis when the mainsheet servo is at its maximum travel (loose). This value is used by the sailing controller to calculate how much to "Sheet In" or "Sheet Out" based on the wind direction.

- **Standard Setting:** 90 degrees (Boom can swing fully perpendicular to the boat).

Tuning & Behavior

- **Default:** 90.
- **Measurement:** Let your mainsheet out completely and measure the angle of the boom. If your rigging prevents the boom from reaching 90 degrees (e.g. hitting the shrouds), enter the actual maximum angle here.
- **Note:** For rotating masts (wing sails), this parameter defines the rotation limit.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





SAIL_ANGLE_MIN (Sail Minimum Angle)

DEFAULT: 0	RANGE: 0 90	UNITS: deg

Sail Minimum Angle (SAIL_ANGLE_MIN)

Description

`SAIL_ANGLE_MIN` defines the "Fully In" position of the sailboat's rig.

It represents the angle of the boom relative to the boat's centerline when the mainsheet servo is at its minimum travel (tight). This is used when sailing "Close-Hauled" (upwind) to generate maximum lift from the sail.

- **Standard Setting:** 0 degrees (Boom is perfectly aligned with the boat).

Tuning & Behavior

- **Default:** 0.
- **Measurement:** Tighten your mainsheet completely and measure the angle. If your mechanical setup doesn't allow the boom to reach the center (0°), enter the actual minimum angle here.

SAIL_ENABLE (Enable Sailboat)

DEFAULT: 0	RANGE: 0 1

Enable Sailboat (SAIL_ENABLE)

Description

`SAIL_ENABLE` turns on the specialized sailboat flight controller.

When enabled, the vehicle will use wind direction data (from a wind vane) to automatically trim sails and perform "tacking" maneuvers (zigzagging) when navigating directly into the wind (the "No Go Zone"). This is only relevant for wind-powered aquatic vehicles.

- **0: Disable.** The vehicle behaves like a standard motor-powered Rover/Boat.
- **1: Enable.** Enables sail servos and sailboat navigation logic.

Tuning & Behavior

- **Reboot Required:** Yes.
- **Dependencies:** Requires a wind vane (WNDVN_TYPE) to be configured and functional.

SAIL_HEEL_MAX (Sail Maximum Heel Angle)

DEFAULT: 15	RANGE: 0 90	UNITS: deg

Sail Maximum Heel Angle (SAIL_HEEL_MAX)

Description

`SAIL_HEEL_MAX` is the "Capsize Prevention" setting for autonomous sailboats.

When the boat banks over (heels) due to wind pressure, there is a risk of losing control or taking on water. If the measured heel angle exceeds this value, the autopilot will automatically ease the sails (let the sheet out) to reduce the sideways force and level the boat, even if it means losing speed.

Tuning & Behavior

- **Default:** 15 degrees.
- **Recommendation:** Set this based on your boat's stability. For a wide-beamed boat or one with a heavy keel, you might increase this to **25** or **30** degrees. For a light catamaran, keep it lower (e.g., **10** or **15**).
- **Safety:** This is a critical safety parameter for long-distance autonomous voyages.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





SAIL_LOIT_RADIUS (Sail Loiter Radius)

DEFAULT: 5	RANGE: 0 50	UNITS: m

Sail Loiter Radius (SAIL_LOIT_RADIUS)

Description

`SAIL_LOIT_RADIUS` defines the size of the "Waiting Area" for a sailboat.

Since a sailboat cannot simply stop and hover (unless it drops an anchor), it "Loiters" by sailing in a continuous circle or figure-eight pattern. This parameter sets the radius of that pattern.

Tuning & Behavior

- **Default:** 5 meters.
- **Recommendation:** Set this to a value at least **3 to 5 times** the length of your boat. If the radius is too small, the boat will spend all its time turning (tacking) and may lose momentum and get stuck.
- **Safety:** Ensure the radius is large enough for the boat to successfully complete a turn (tack or jibe) without exiting the loiter boundary.

SAIL_NO_GO_ANGLE (Sail No-Go Zone Angle)

DEFAULT: 45	RANGE: 30 60	UNITS: deg

Sail No-Go Zone Angle (SAIL_NO_GO_ANGLE)

Description

`SAIL_NO_GO_ANGLE` defines the "Upwind Limit" of your boat.

A sailboat cannot sail directly into the wind; if it tries, the sails will simply luff (flap) and the boat will stop. This parameter defines the boundary of that unusable sector. When the autopilot needs to reach a waypoint that is directly upwind, it will use this angle to calculate the "Tacks" (zig-zagging) required to get there.

Tuning & Behavior

- **Default:** 45 degrees.
- **High Performance:** For a fast racing yacht with high-quality sails, you can decrease this to **35** or **40** degrees.
- **Casual/Draggy:** For a bulky boat or one with less efficient sails, you might need to increase this to **50** or **55** degrees.
- **Effect:** If this value is too low, the boat will constantly lose speed and "Stall" while trying to point too high into the wind. If too high, the boat will be inefficient and take much longer to reach upwind targets.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





SAIL_TYPE (Sailboat simulation sail type)

DEFAULT: 0	RANGE: 0 1

SAIL_TYPE: Sailboat simulation sail type

Description

This parameter selects the mechanical configuration of the sail for the SITL (Software-In-The-Loop) sailboat simulation.

Values

Value	Meaning
0	Mainsail with Sheet (Conventional cloth sail)
1	Directly Actuated Wing (Rigid wing sail)

- **Default:** 0 (Mainsail)

Description

`SAIL_TYPE` tells the simulator which physics model to use for the sailboat's propulsion.

- **Mainsail (0):** Simulates a flexible sail controlled by a winch (sheet). The sail billows and generates lift based on the tension of the sheet and wind angle.
- **Wing (1):** Simulates a rigid wing (similar to those on high-performance racing catamarans) that is directly rotated to a specific angle of attack by a servo.

SAIL_WNDSPD_MIN (Sailboat Minimum Wind Speed)

DEFAULT: 0	RANGE: 0 5.0	UNITS: m/s

Sailboat Minimum Wind Speed (SAIL_WNDSPD_MIN)

Description

`SAIL_WNDSPD_MIN` defines the "Calm Water" transition point for autonomous boats.

If the measured wind speed (via WNDVN_SPEED) falls below this value, the autopilot assumes there isn't enough wind to sail effectively. If the boat has an auxiliary motor, the autopilot will engage the motor to maintain its mission speed. If there is no motor, the boat will simply drift or hold position until the wind picks up.

Tuning & Behavior

- **Default:** 0 (Always try to sail).
- **Recommendation:** Set to **0.5 m/s** to **1.0 m/s** to prevent the boat from "Gilled" (stuck with no speed) in very light air.
- **Dependency:** Requires a functional wind speed sensor and an auxiliary motor if motoring is desired.

SAIL_XTRACK_MAX (Sailboat Max Cross-Track Error)

DEFAULT: 10	RANGE: 5 50	UNITS: m

Sailboat Max Cross-Track Error (SAIL_XTRACK_MAX)

Description

`SAIL_XTRACK_MAX` defines the "Corridor Width" for sailing missions.

When sailing upwind, a boat must zig-zag (tack) to reach its target. This parameter sets how far the boat can move to either side of the direct line (the "Cross-Track") before the autopilot triggers a tack to come back toward the center.

- **Default:** 10 meters.
- **Recommendation:** For large open water, you can increase this to **20 or 30** meters to reduce the number of tacks and improve efficiency. For narrow channels, keep it at **5 or 10**.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **SAVE** parameter group (specifically `SAVE_CFG` or `GPS_SAVE_CFG`) manages the non-volatile storage of external sensor settings.

Key Concepts

1. Flash Synchronization

When the autopilot configures a peripheral (like a u-Blox GPS), those settings are lost when the GPS powers down unless a "Save" command is sent to the GPS's internal flash.

- `SAVE_CFG`:
  - **0:** Don't save. Autopilot re-configures the sensor on every boot.
  - **1:** Save. Tells the sensor to store the current configuration permanently.

Developer Notes

- **Library:** `libraries/AP_GPS`.

SAVE_CFG (Save GPS configuration)

DEFAULT: 2	RANGE: 0 2

SAVE_CFG: Save GPS configuration

Description

Determines whether the configuration for this GPS should be written to non-volatile memory on the GPS.

Values

Value	Meaning
0	Do not save config
1	Save config
2	Save only when needed

- **Default:** 2

Description

This parameter controls whether the autopilot's GPS configuration commands (baud rate, update rate, protocol settings) are permanently stored in the GPS module's own memory (Flash/EEPROM).

- **Benefit:** If the config is saved on the GPS, it will boot up in the correct state even if the autopilot's serial port hasn't fully initialized.
- **Normal Operation (2):** The autopilot checks if the current GPS configuration matches what it wants. If not, it sends the new config and commands the GPS to save it. This is the most efficient and recommended setting.
- **Save Only (1):** Forces a save command after every boot.
- **Disable (0):** Useful if you have a custom GPS config that you don't want the autopilot to overwrite permanently.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **SB** parameter group (specifically `SB_H_OFS` and `SB_V_EXT`) provides layout adjustments for the **OSD Sidebars** (the moving tapes for Speed and Altitude).

Key Concepts

1. Tape Placement

On analog OSDs, the "Speed Tape" and "Altitude Tape" are rendered as sidebars. Depending on your goggles' field of view, these tapes might be too close to the center or clipped at the edges.

- `SB_H_OFS`: Adjusts the horizontal spacing of the tapes.
- `SB_V_EXT`: Adjusts the vertical length/extension of the tapes.

Developer Notes

- **Library:** `libraries/AP_OSD`.
- **Context:** Only relevant for vehicles with active OSD rendering.

SB_H_OFS (Sidebar horizontal offset)

DEFAULT: 0	RANGE: 0 20

SB_H_OFS: Sidebar horizontal offset

Description

Extends the spacing between the sidebar elements (like the artificial horizon and the tapes) by this amount of columns. Positive values increase the width to the right of the screen.

Values

- **Range:** 0 to 20
- **Default:** 0

Description

This parameter is used to fine-tune the layout of the **OSD (On-Screen Display)** sidebars.

- **Function:** If you find that the speed and altitude tapes are too close to the center of the screen (obscuring the horizon or the crosshair), increasing this value will push them further towards the edges.
- **Unit:** Each increment corresponds to one character column width on the OSD screen.
- **Alignment:** This offset is typically applied to both left and right sidebars to maintain symmetry.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





SB_V_EXT (Sidebar vertical extension)

DEFAULT: 0	RANGE: 0 10

SB_V_EXT: Sidebar vertical extension

Description

Increases the vertical length of the sidebar elements (like the airspeed and altitude tapes) by this many character lines. The extension is applied equally both above and below the default center point.

Values

- **Range:** 0 to 10
- **Default:** 0

Description

This parameter controls the "height" of the data tapes displayed in the **OSD (On-Screen Display)**.

- **Function:** If you want a taller speed or altitude tape to see a wider range of values at a glance, increasing this parameter will grow the tapes vertically.
- **Unit:** Each increment adds two character lines total (one above the center and one below).
- **Layout:** Note that making the sidebars too tall may cause them to overlap with other OSD elements at the top or bottom of the screen.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

Key Concepts

SBAS_MODE (SBAS Mode)

DEFAULT: 2	RANGE: 0 2

SBAS_MODE: SBAS Mode

Description

This parameter selects the behavior for SBAS (Satellite Based Augmentation System) like WAAS, EGNOS, or MSAS.

Values

Value	Meaning
0	Disabled (Turn off SBAS on the GPS)
1	Enabled (Turn on SBAS on the GPS)
2	No Change (Use the GPS module's current internal setting)

- **Default:** 2

Description

SBAS uses a network of ground stations and geostationary satellites to provide corrections to your GPS receiver, potentially improving accuracy from ~2.5m down to ~1m.

- **Usage:** In North America (WAAS), Europe (EGNOS), and Japan (MSAS), enabling this (1) usually improves position stability.
- **Why Disable?** In some regions, SBAS satellites are very low on the horizon, or the ground station baseline is so long that the "corrections" actually add more noise than accuracy.
- **Default (2):** This is the safest setting, as it leaves the decision to the GPS manufacturer's factory defaults.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

Key Concepts

SBP_LOGMASK (Swift Binary Protocol Logging Mask)

`DEFAULT: -256`

SBP_LOGMASK: Swift Binary Protocol Logging Mask

Description

This parameter acts as a filter for logging Swift Binary Protocol (SBP) messages from Swift Navigation GPS modules (like the Piksi Multi) to the autopilot's DataFlash logs.

Values

Value	Meaning
0	None (0x0000) - No SBP messages logged
-1	All (0xFFFF) - All SBP messages logged
-256	External only (0xFF00) - Only log messages originating from external sources

- **Default:** -256

Description

For users of Swift Navigation RTK GPS systems, `SBP_LOGMASK` controls the verbosity of the raw GPS data stream captured in the logs.

- **Function:** It performs a bitwise AND between this mask and the SBP message type ID. If the result is non-zero, the message is logged as `SBR1` or `SBR2` packets.
- **Diagnostic Use:** If you are having trouble with RTK fix stability or GPS timing, setting this to **-1** captures everything for detailed analysis by Swift Navigation support, but it will significantly increase the log file size.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY

CONTROL SCALING CONFIGURATION (SCALING)

Overview

The **SCALING** parameter group (specifically `SCALING_SPEED`) is a fundamental configuration for **ArduPlane** (Fixed-Wing).

Aerodynamics are highly speed-dependent. As a plane flies faster, the same amount of control surface movement (e.g., 5 degrees of aileron) produces significantly more roll force. To maintain stable flight across a wide speed range, the autopilot must automatically scale (reduce) its servo movements as airspeed increases.

Key Concepts

1. Speed Scaling

- `SCALING_SPEED`: The "Reference Speed" (m/s) at which your tuned PID gains are exactly 100% effective.
  - **Cruise:** Usually set to your typical cruise speed.
  - **Autotune:** Automatically set by the Autotune process to the speed at which the tune was performed.

2. The Inverse Relationship

The autopilot calculates a multiplier: `Scaler = SCALING_SPEED / Current_Airspeed`.

- **Faster:** If the plane flies twice as fast as `SCALING_SPEED`, the control movements are cut in half.
- **Slower:** If the plane flies slower, the control movements increase to maintain authority.

Developer Notes

- **Library:** `ArduPlane/Attitude.cpp`.
- **Fallback:** If no airspeed sensor is present, the system uses a throttle-based estimation of speed scaling.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





SCALING_SPEED (Speed scaling reference speed)

DEFAULT: 15.0	RANGE: 0 50	UNITS: m/s

Speed scaling reference speed (SCALING_SPEED)

Description

This parameter sets the "Baseline Airspeed" for the control surface scaling system.

As a fixed-wing aircraft flies faster, its control surfaces (ailerons, elevator, rudder) become more effective, generating more force for the same amount of movement. To prevent the aircraft from becoming twitchy or oscillating at high speeds, ArduPilot automatically reduces the control surface movement as speed increases.

`SCALING_SPEED` is the speed at which your tuned PID gains are applied exactly as written (Multiplier = 1.0).

- **Faster than `SCALING_SPEED`:** Surfaces move LESS than the PID requests (Multiplier < 1.0).
- **Slower than `SCALING_SPEED`:** Surfaces move MORE than the PID requests (Multiplier > 1.0).

Critical: This value is typically set to your **Cruise Speed** or the speed at which you performed **Autotune**.

The Mathematics

The system calculates a scaling factor (K_scale) that is applied to the P, I, and D terms of the attitude controllers.

$$K_scale = \frac{SCALING_SPEED}{V_true}$$

Where V_true is the current True Airspeed (TAS).

- If $V_true = SCALING_SPEED$, $K_scale = 1.0$
- If $V_true = 2 \times SCALING_SPEED$, $K_scale = 0.5$ (Half deflection for same error).

This inverse-linear relationship is derived from the physics of steady-state rate control, where the required deflection for a given turn rate scales inversely with velocity ($\delta \propto \frac{p}{V}$).

The Engineer's View

In `Plane::calc_speed_scaler()` (`ArduPlane/Attitude.cpp`), this scaler is computed and then constrained.

- **Upper Limit:** The scaler is capped at roughly `2.0` (or calculated based on `AIRSPEED_MIN`) to prevent excessive surface throw at stall speeds.
- **Lower Limit:** The scaler is floored at `0.5` (or calculated based on `AIRSPEED_MAX`) to prevent loss of control at high speeds.
- **No Airspeed Sensor:** If no airspeed sensor is present, the system estimates speed scaling using `THROTTLE_CRUISE` vs actual Throttle, applying a square root approximation: $Scale = \sqrt{\frac{THROTTLE_CRUISE}{Throttle_out}}$.

Tuning & Behavior

- **Default Value:** 15.0 m/s
- **Range:** 5 to 50 m/s
- **Effect of Changing:**
  - **Increasing `SCALING_SPEED`:** Makes the surfaces move **more** at all speeds (higher effective gain).
  - **Decreasing `SCALING_SPEED`:** Makes the surfaces move **less** at all speeds (lower effective gain).
- **Golden Rule:** If you change `SCALING_SPEED` after tuning, you invalidate your tune. You must re-tune (Autotune) or manually scale your P/I/D gains by the ratio of the change.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **SCAN** parameter group configures the behavior of the **Scan** mode for the **Antenna Tracker**.

Scan mode is used when the tracker does not have a valid vehicle position to follow. Instead of sitting idle, the tracker can be programmed to oscillate or sweep a specific area to help the ground station radios re-acquire a signal.

Key Concepts

1. Sweep Speeds

- `SCAN_SPEED_YAW`: The rotation speed (deg/s) for the horizontal sweep.
- `SCAN_SPEED_PIT`: The rotation speed (deg/s) for the vertical sweep.

Integration Guide

- **Usage:** If you fly out of range and the tracker stops moving, switching to `SCAN` mode will help sweep the sky with your directional antenna to catch a "glimmer" of telemetry data.

Developer Notes

- **Library:** `AntennaTracker/Parameters.cpp`.

SCAN_SPEED_PIT (Antenna Tracker Scan Pitch Speed)

DEFAULT: 2	RANGE: 0 100	UNITS: deg/s

Antenna Tracker Scan Pitch Speed (SCAN_SPEED_PIT)

Description

`SCAN_SPEED_PIT` defines the vertical sweep speed for the Antenna Tracker while searching for a signal.

SCAN_SPEED_YAW (Antenna Tracker Scan Yaw Speed)

DEFAULT: 2	RANGE: 0 100	UNITS: deg/s

Antenna Tracker Scan Yaw Speed (SCAN_SPEED_YAW)

Description

`SCAN_SPEED_YAW` defines how fast the Antenna Tracker "looks around" when it loses its connection to the drone.

In **SCAN** mode, the tracker performs a pre-defined sweeping pattern to re-acquire the telemetry signal. This parameter sets the horizontal sweep speed.

Tuning & Behavior

- **Default:** 2 deg/s.
- **Recommendation:** Keep this relatively low. If the tracker moves too fast, it might sweep past the signal before the radio can achieve a "Link."

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





REAL-TIME SCHEDULER (SCHED)

Overview

The **SCHED** parameter group configures the core **Task Scheduler** for ArduPilot. The scheduler is the heartbeat of the autopilot, responsible for prioritizing and running hundreds of distinct tasks (IMU sampling, PID loops, MAVLink, Logging) at precise intervals.

ArduPilot uses a cooperative multi-tasking model where each task is expected to run and return within a tiny time budget (e.g., 50 microseconds).

Key Concepts

1. Loop Rate (`SCHED_LOOP_RATE`)

Defines the main frequency (Hz) of the attitude control loop.

- **Multicopter:** Usually 400Hz (2.5ms loop) or 800Hz.
- **Fixed Wing:** Usually 50Hz (20ms loop) or 100Hz.
- **Importance:** Higher rates provide lower latency for control but require more CPU power.

2. Scheduler Options (`SCHED_OPTIONS`)

A bitmask for advanced scheduler features.

- **Fast Loop:** Toggles the dedicated high-priority rate thread on H7 processors.

3. Debugging (`SCHED_DEBUG`)

Allows developers to monitor "Slips" (when the scheduler runs late) or "Overruns" (when a task takes too long).

Parameter Breakdown

- `SCHED_LOOP_RATE`: The primary control frequency.
- `SCHED_DEBUG`: Verbosity of timing diagnostics.

Integration Guide

- **Warning: Do not change** `SCHED_LOOP_RATE` unless instructed by hardware documentation. Setting it too high can overload the CPU, causing "Main Loop Slow" errors and potentially a crash.

Developer Notes

- **Library:** `libraries/AP_Scheduler`.
- **Priority:** The scheduler uses a fixed table of tasks with assigned priorities. `INS` (sensors) always run first.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





SCHED_DEBUG (Scheduler debug level)

DEFAULT: 0	RANGE: 0 3

SCHED_DEBUG: Scheduler debug level

Description

Set to non-zero to enable scheduler debug messages.

Values

Value	Meaning
0	Disabled
2	Show Slips (Delayed tasks)
3	Show Overruns (Long-running tasks)

- **Default:** 0

Description

This parameter is a diagnostic tool for monitoring the real-time performance of the autopilot's operating system (the Scheduler).

- **Function:** If the CPU becomes overloaded, the scheduler may have to delay certain tasks ("Slips") or a specific task may take longer than its allocated time budget ("Overruns").
- **Usage:** Set to **2 or 3** during troubleshooting to see warning messages in the MAVLink console or Ground Control Station. These messages indicate that the flight controller is struggling to keep up with the workload.
- **Normal Operation:** Should be left at **0** to avoid flooding the communication link with debug data.

SCHED_LOOP_RATE (Scheduling main loop rate)

DEFAULT: 400	RANGE: 50 400	UNITS: Hz

Scheduling main loop rate (SCHED_LOOP_RATE)

Description

`SCHED_LOOP_RATE` defines the "Heartbeat" speed of the flight controller. It determines how many times per second the autopilot reads the sensors, runs the PID math, and updates the motor outputs.

- **400Hz (Default for Copter):** The drone updates its state every 2.5 milliseconds. This is required for stable, responsive multicopter flight.
- **50Hz (Default for Plane/Rover):** The vehicle updates its state every 20 milliseconds. Sufficient for slower-moving vehicles like cars or fixed-wing aircraft.

The Engineer's View

Defined in `AP_Scheduler.cpp`.
This parameter sets the base tick interval for the `AP_Scheduler` system. All other tasks (GPS reading, Battery monitoring, Logging) are scheduled as multiples of this rate.

Warning: Increasing this above 400Hz is considered experimental. It significantly increases CPU load and may cause "Scheduler Slips," where the drone becomes unstable because it cannot finish the math in time for the next loop.

Tuning & Behavior

- **Default Value:** 400 (Copter), 50 (Plane/Rover).
- **Recommendation:** Leave at **400** for all modern multicopters. Only reduce it if you are using extremely old hardware (like a Pixhawk 1) and experiencing high `LOAD` values (>80%).
- **Reboot Required:** Yes. This changes the fundamental timing of the processor.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





SCHED_OPTIONS (Scheduling options)

DEFAULT: 0	RANGE: 0 1

Scheduling options (SCHED_OPTIONS)

Description

`SCHED_OPTIONS` enables specialized performance tracking within the autopilot's operating system.

- **Bit 0 (1): Enable per-task perf info.** If enabled, ArduPilot will track exactly how many microseconds every individual task (like "Baro Update" or "Compass Read") takes to complete. This is vital for developers or users experiencing unexplained twitches or CPU spikes.

The Engineer's View

When Bit 0 is set, the `AP::PerfInfo` module allocates extra memory to store timing statistics for every item in the scheduler task table. These statistics can be viewed in real-time by downloading the `@SYS/tasks.txt` file via MAVLink FTP.

Tuning & Behavior

- **Default Value:** 0 (Disabled to save memory and CPU).
- **Recommendation:** Leave at **0** for normal flight. Enable only if you are diagnosing "Internal Error 0x1000" or high system load.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





LUA SCRIPTING CONFIGURATION (SCR)

Overview

The **SCR** parameter group configures the **On-Board Lua Scripting** engine. This is a powerful feature that allows users to add custom behaviors, flight modes, or hardware drivers to ArduPilot without needing to recompile the C++ source code.

Lua scripts run in a sandboxed environment on the autopilot's SD card, interacting with the system via a controlled API.

Key Concepts

1. The Sandbox (`SCR_HEAP_SIZE`)

Lua scripts require dedicated RAM.

- `SCR_HEAP_SIZE`: The total amount of memory (in bytes) allocated for the Lua engine.
  - **Pixhawk 1:** Not supported (Insufficient RAM).
  - **F7/H7 Boards:** Typically 40kB to 200kB.
  - **Requirement:** If you load many or complex scripts, you must increase this value.

2. User Parameters (`SCR_USERx`)

Allows scripts to have their own tunable parameters.

- `SCR_USER1` .. `6`: Integer values.
- `SCR_USER_FLOAT`: Floating point values.
- **Usage:** A script can read `SCR_USER1` and use it as a gain or a threshold.

3. Scripting Devices (`SCR_SDEV_...`)

Enables scripts to access specialized serial or I2C protocols.

Parameter Breakdown

- `SCR_ENABLE`: Master switch.
- `SCR_DEBUG_OPTS`: Toggles error reporting and memory usage logs.

Integration Guide

1. **Preparation:** Use an H7 flight controller (e.g., Orange Cube).
2. **Enable:** Set `SCR_ENABLE = 1`.
3. **Allocate:** Ensure `SCR_HEAP_SIZE` is sufficient (default 40,960 is usually okay for 1-2 scripts).
4. **Install:** Copy your `.lua` files to the `scripts/` folder on the SD card.
5. **Reboot.**

Developer Notes

- **Library:** `libraries/AP_Scripting`.
- **Safety:** Lua is run in a separate low-priority thread to ensure it can't interfere with the flight control PIDs.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





SCR_DEBUG_OPTS (Scripting Debug Options)

DEFAULT: 0	RANGE: null	UNITS: Bitmask

Scripting Debug Options (SCR_DEBUG_OPTS)

Description

This parameter is a bitmask used to control various debugging and maintenance features of the Lua scripting engine. It allows developers to monitor performance, manage security checksums, and suppress or enable specific status messages.

The Mathematics

The value is treated as a set of individual binary flags:

- **Bit 0 (1): Suppress "No Scripts" message.** Disables the GCS notification if the system finds zero scripts to run.
- **Bit 1 (2): Runtime telemetry messages.** Periodically sends GCS text messages showing current RAM usage and execution time for each running script.
- **Bit 2 (4): Suppress Log Writing.** Prevents scripts from being written to the `.BIN` log file (saves SD card space).
- **Bit 3 (8): Continuous Data Logging.** Enables high-frequency logging of script performance metrics to the DataFlash log.
- **Bit 4 (16): Disable Pre-Arm Check.** Allows the vehicle to arm even if a script has an error or is missing. **(Caution: Safety Risk)**.
- **Bit 5 (32): Auto-save Checksums.** A "one-shot" command. When set, the system calculates the current script CRCs, saves them to `SCR_LD_CHECKSUM` and `SCR_RUN_CHECKSUM`, and then automatically clears this bit.
- **Bit 6 (64): Disable Heap Expansion.** Prevents the Lua VM from trying to grow its memory pool if a script runs out of space.

The Engineer's View

In `AP_Scripting.cpp`:

- **Checkpoints:** The `arming_checks()` function respects Bit 4.
- **One-Shot Logic:** The `save_checksum()` function checks Bit 5 at 1Hz. If active, it performs the XOR-CRC calculations and immediately calls `option_clear(DebugOption::SAVE_CHECKSUM)` to prevent accidental overwrites.
- **Performance Tracking:** Bit 1 and 3 are handled inside the Lua executor loop (`lua_scripts.cpp`) to inject timing probes around the VM execution window.

Tuning & Behavior

- **Development Stage:** Set to **2 (Bit 1)** to get constant feedback on your script's memory footprint and efficiency.
- **Production Stage:** Set to **0** for a clean GCS experience.
- **Security Stage:** Use **32 (Bit 5)** to "lock in" your verified script set, ensuring that any unauthorized modification to the scripts on the SD card prevents the vehicle from arming.

SCR_DIR_DISABLE (Scripting Directory Disable)

DEFAULT: 0	RANGE: 0 1

Scripting Directory Disable (SCR_DIR_DISABLE)

Description

`SCR_DIR_DISABLE` is a security and performance setting for the Lua engine.

It prevents the flight controller from searching certain directories on the SD card for scripts.

- **0:** Scan all standard locations.
- **1:** Disable advanced directory scanning.

Tuning & Behavior

- **Recommendation:** Leave at **0** for standard users.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





SCR_ENABLE (Scripting Enable)

DEFAULT: 0	RANGE: 0 1

Scripting Enable (SCR_ENABLE)

Description

`SCR_ENABLE` activates the internal Lua interpreter.

Lua scripting allows you to add custom features to ArduPilot without needing to recompile the C++ source code. You can write simple scripts to control LEDs, automate complex flight patterns, or interface with unique hardware.

- **0: Disabled.**
- **1: Enabled.**

Tuning & Behavior

- **Default Value:** 0.
- **Reboot Required:** Yes.
- **Requirement:** Scripts must be placed in the `/scripts` folder on the SD card.

SCR_HEAP_SIZE (Scripting Heap Size)

DEFAULT: 40960	RANGE: 10240 1000000	UNITS: bytes

Scripting Heap Size (SCR_HEAP_SIZE)

Description

`SCR_HEAP_SIZE` defines the memory limit for your Lua scripts.

If your scripts are large or use many complex variables, they may run out of memory. This parameter allows you to allocate more of the flight controller's RAM to the scripting engine.

Tuning & Behavior

- **Default Value:** Usually 40,960 bytes (40 KB).
- **Constraint:** On memory-constrained boards (like those with only 1MB flash), you may not be able to increase this significantly. High-end boards (Cube Orange) can support much larger values.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY

SCR_LD_CHECKSUM (Loaded script checksum)

DEFAULT: -1	RANGE: null	UNITS: Hex

Loaded script checksum (SCR_LD_CHECKSUM)

Description

This parameter is a security and safety feature that ensures the set of Lua scripts on your SD card matches a "Known Good" configuration. It stores an aggregate CRC32 checksum of every script the autopilot has successfully found and compiled during boot.

If you enable this check, ArduPilot will prevent the vehicle from arming if even a single character in any script has been changed, or if a script has been added or removed from the `APM/scripts` folder. This is vital for commercial or industrial operations where script integrity is a requirement for flight certification.

The Mathematics

The system calculates a standard CRC32 for each individual script file. The total checksum (C_total) is the bitwise XOR accumulation of all individual script checksums (C_n):

$$C_total = C_1 \oplus C_2 \oplus \cdots \oplus C_n$$

The resulting 32-bit value is then masked by `0x7FFFFFFF` to fit within the signed integer parameter range.

The Engineer's View

In `AP_Scripting::arming_checks()` (`libraries/AP_Scripting/AP_Scripting.cpp`):

1. The code checks if `_required_loaded_checksum` is not -1.
2. It retrieves the current session's checksum from the Lua compiler backend.
3. If they do not match, it returns a `false` status with the message: `"Scripting: loaded CRC incorrect"`.
4. This check ensures that the scripts *on the disk* haven't been tampered with since the last time the "Master" checksum was saved.

Tuning & Behavior

- **Default Value:** -1 (Check disabled).
- **How to Set:** Use `SCR_DEBUG_OPTS` Bit 5 (Value 32) to automatically calculate and save the current checksum to this parameter.
- **Usage:** Once your scripts are fully tested and frozen, save the checksum. From that point on, any change to the scripts will block arming until you explicitly update the parameter again.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





SCR_RUN_CHECKSUM (Running script checksum)

DEFAULT: -1	RANGE: null	UNITS: Hex

Running script checksum (SCR_RUN_CHECKSUM)

Description

This parameter provides a dynamic layer of safety for Lua scripting by verifying that the correct set of scripts is not only present but is actually **running**.

While `SCR_LD_CHECKSUM` verifies that scripts exist and compiled correctly, `SCR_RUN_CHECKSUM` ensures that every intended script has successfully entered its execution loop and hasn't crashed or been manually stopped. If the "Running Checksum" calculated by the autopilot doesn't match this parameter, the vehicle will refuse to arm.

The Mathematics

The system calculates a bitwise XOR accumulation of the CRC32 values (C_n) for all scripts that are currently in the `running` state within the Lua VM.

$$C_running = C_1 \oplus C_2 \oplus \cdots \oplus C_running_n$$

The value is masked with `0x7FFFFFFF` to ensure compatibility with standard GCS parameter displays.

The Engineer's View

In `AP_Scripting::arming_checks()` (`libraries/AP_Scripting/AP_Scripting.cpp`):

1. The system identifies which scripts are active in the Lua scheduler.
2. It calculates the current aggregate CRC.
3. If this fails to match `_required_running_checksum` (and it's not -1), the pre-arm check fails with the message: `"Scripting: running CRC incorrect"`.
4. This is a critical distinction from the "Loaded" check: a script that has a syntax error will fail the "Loaded" check, but a script that compiles but then crashes (e.g., due to an infinite loop or memory limit) will fail the "Running" check.

Tuning & Behavior

- **Default Value:** -1 (Check disabled).
- **How to Set:** Set `SCR_DEBUG_OPTS` Bit 5 (Value 32) while all your desired scripts are running healthy. ArduPilot will save the current running CRC to this parameter automatically.
- **Reliability:** This parameter is essential for missions that depend on Lua scripts for safety (e.g., custom fail-safes or terrain-following logic). It guarantees that the vehicle won't arm if the safety-critical script is not active.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





SCR_SDEV1_PROTO (Scripting serial protocol 1)

DEFAULT: -1	RANGE: null	UNITS: Protocol

Scripting serial protocol 1 (SCR_SDEV1_PROTO)

Description

This parameter defines the logical protocol for the first virtual serial device (`SDEV1`). When using Lua scripting to interact with serial hardware, you must first tell the system what kind of data is being handled.

Setting this to a specific protocol (like MAVLink, GPS, or NMEA) allows the virtual port to benefit from ArduPilot's internal parsing and buffer management before the data is handed off to your Lua script.

The Mathematics

This is a standard protocol index (P):

- **-1:** None/Disabled.
- **1:** MAVLink 1.
- **2:** MAVLink 2.
- **5:** GPS.
- *(Refer to the SERIALx_PROTOCOL list for all available options).*

The Engineer's View

In `AP_Scripting.cpp` , this parameter is part of the `_serialdevice` subgroup.

- **Mapping:** It maps to `ports[0].state.protocol` .
- **Serial Manager Integration:** When `AP_SerialManager` scans for devices, it includes these virtual `SDEV` instances.
- **Lua Access:** A script then uses `serial:find_serial(0)` to obtain a handle to this port. The internal driver will then automatically handle the baud rate and framing required by the chosen protocol.

Tuning & Behavior

- **Default Value:** -1 (Disabled).
- **Reboot Required:** Yes. Virtual device registration is a boot-time operation.
- **Dependencies:** `SCR_SDEV_EN` must be set to 1.

SCR_SDEV2_PROTO (Scripting serial protocol 2)

DEFAULT: -1	RANGE: null	UNITS: Protocol

Scripting serial protocol 2 (SCR_SDEV2_PROTO)

Description

This parameter defines the logical protocol for the second virtual serial device (`SDEV2`). It functions exactly like `SCR_SDEV1_PROTO` , allowing for a second independent serial integration via Lua.

The Mathematics

This is a standard protocol index (P):

- **-1:** None/Disabled.
- **GPS (5):** If you are synthesizing a second GPS source in Lua, set this to 5.
- **MAVLink (2):** If your script needs to speak MAVLink to a companion device.

The Engineer's View

In `AP_Scripting.cpp` , this maps to `ports[1].state.protocol` .

- **Port Mapping:** Internally, this virtual port is assigned the index `AP_SERIALMANAGER_SCR_PORT_1 + 1` .
- **Lua Access:** Obtained via `serial:find_serial(1)` .

Tuning & Behavior

- **Default Value:** -1 (Disabled).
- **Reboot Required:** Yes.
- **Dependencies:** `SCR_SDEV_EN` must be 1.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





SCR_SDEV3_PROTO (Scripting serial protocol 3)

DEFAULT: -1	RANGE: null	UNITS: Protocol

Scripting serial protocol 3 (SCR_SDEV3_PROTO)

Description

This parameter defines the logical protocol for the third virtual serial device (`SDEV3`). It allows for a third independent serial integration via Lua, following the same logic as `SCR_SDEV1_PROTO`.

The Mathematics

This is a standard protocol index (P):

- **-1:** None/Disabled.
- **Range:** Any valid ArduPilot `SERIALx_PROTOCOL` value.

The Engineer's View

In `AP_Scripting.cpp`, this maps to `ports[2].state.protocol`.

- **Port Mapping:** Internally, this virtual port is assigned the index `AP_SERIALMANAGER_SCR_PORT_1 + 2`.
- **Lua Access:** Obtained via `serial:find_serial(2)`.

Tuning & Behavior

- **Default Value:** -1 (Disabled).
- **Reboot Required:** Yes.
- **Dependencies:** `SCR_SDEV_EN` must be 1.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





SCR_SDEV_EN (Scripting serial device enable)

DEFAULT: 0	RANGE: 0 1	UNITS: Option

Scripting serial device enable (SCR_SDEV_EN)

Description

This parameter enables the use of "Scripting Serial Devices." These are virtual serial ports that allow Lua scripts running on the autopilot to communicate directly with external hardware or sensors connected via a physical UART, without requiring a C++ driver.

When enabled, it allocates resources for up to three virtual serial instances (`SDEV1`, `SDEV2`, `SDEV3`) which can be bound to Lua `serial` objects. This is a powerful feature for rapid prototyping of custom hardware integrations.

The Mathematics

This parameter acts as a memory allocation gate (G_mem):

$$\text{IF } SCR_SDEV_EN == 1 \rightarrow \text{Allocate } [N \times \text{SerialPortState}]$$

Where N is defined by `AP_SCRIPTING_SERIALDEVICE_NUM_PORTS` (typically 3). Each port then maps to a virtual index in the `AP_SerialManager` device list.

The Engineer's View

In `AP_Scripting.cpp`, this maps to the `_serialdevice.enable` flag.

- **Initialization:** During boot, `AP_Scripting::init_serialdevice_ports()` checks this parameter. If active, it registers virtual ports with the Serial Manager.
- **Lua Binding:** In a Lua script, these ports are accessed using `serial:find_serial(instance)`, where `instance` corresponds to the `SDEV` number.
- **Data Integrity:** Enabling this also activates internal buffers for script-to-UART communication, which are cleared during script restarts to prevent stale data from being sent.

Tuning & Behavior

- **Default Value:** 0 (Disabled).
- **0:** No virtual ports are created; saves a small amount of RAM.
- **1:** Enables the `SCR_SDEVx_PROTO` parameters for configuration.
- **Reboot Required:** Yes. Virtual port registration happens only during the early boot phase.
- **Dependencies:** `SCR_ENABLE` must be 1. You must also set the physical UART protocol (e.g., `SERIAL2_PROTOCOL`) to **Scripting (28)** and then configure the corresponding `SCR_SDEVx_PROTO`.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY

SCR_THD_PRIORITY (Scripting thread priority)

DEFAULT: 0	RANGE: 0 8	UNITS: Priority Level

Scripting thread priority (SCR_THD_PRIORITY)

Description

This parameter determines how much "CPU Importance" is given to the Lua scripting engine relative to other autopilot tasks (like reading sensors, calculating motors, or handling radio telemetry).

By default, scripting runs at a low priority to ensure that even if a script is poorly written or very complex, it cannot "choke" the critical flight control logic. However, for specialized applications where a script needs to handle high-speed data or tight control loops, this priority can be raised.

WARNING: Raising this priority unnecessarily can lead to system instability, watchdog resets, or loss of control if your scripts consume too much CPU time.

The Mathematics

The priority levels are mapped to the underlying RTOS (Real-Time Operating System) thread scheduler:

- **0: Normal.** Standard background priority.
- **1: IO.** Matches Input/Output tasks.
- **3: UART.** Matches Serial communication priority.
- **7: Main.** Matches the main flight control loop. **(Extremely Dangerous)**.
- **8: Boost.** Highest possible priority.

The Engineer's View

In `AP_Scripting::init()` (`libraries/AP_Scripting/AP_Scripting.cpp`), the value of `_thd_priority` is translated into a `hal_priority` via an internal `priority_map`.
This priority is then passed to the `hal.scheduler->thread_create()` call.

Note that all Lua scripts run within the **same single thread**. If you raise the priority, you are raising it for every script on your SD card simultaneously. You must audit your scripts for "busy-waits" or long execution times before increasing this setting.

Tuning & Behavior

- **Default Value:** 0 (Normal).
- **When to Increase:** Only if your script is failing to meet timing requirements (e.g., missing high-speed serial data or failing to update a custom LED driver at the desired rate).
- **When to Decrease:** Never; 0 is already the safest background level.
- **Reboot Required:** Yes. Thread creation happens only during the boot sequence.

SCR_USER1 (Scripting User Parameter 1)

DEFAULT: 0	RANGE: -10000 10000

Scripting User Parameter (SCR_USER)

Description

`SCR_USER1` through `SCR_USER6` are "Input Boxes" for your custom Lua scripts.

Instead of hard-coding values like "Speed" or "Distance" inside your script, you can write the script to look at these parameters. This allows you to change how your script behaves using your Ground Station (Mission Planner/QGC) while in the field.

Tuning & Behavior

- **Usage:** These parameters do nothing by themselves. Their meaning is entirely defined by the specific Lua script you are running.

SCR_USER2 (Scripting User Parameter2)

DEFAULT: 0.0	RANGE: null

Scripting User Parameter2

Note: This parameter configures instance 2. It functions identically to SCR_USER1.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





SCR_USER3 (Scripting User Parameter3)

DEFAULT: 0.0	RANGE: null

Scripting User Parameter3

Note: This parameter configures instance 3. It functions identically to SCR_USER1.

SCR_USER4 (Scripting User Parameter4)

DEFAULT: 0.0	RANGE: null

Scripting User Parameter4

Note: This parameter configures instance 4. It functions identically to SCR_USER1.

SCR_USER5 (Scripting User Parameter5)

DEFAULT: 0.0	RANGE: null

Scripting User Parameter5

Note: This parameter configures instance 5. It functions identically to SCR_USER1.

SCR_USER6 (Scripting User Parameter6)

DEFAULT: 0.0	RANGE: null

Scripting User Parameter6

Note: This parameter configures instance 6. It functions identically to SCR_USER1.

SCR_USER_FLOAT (Scripting User Parameter (Float))

DEFAULT: 0

Scripting User Parameter (SCR_USER_FLOAT)

Description

`SCR_USER_FLOAT` (and its INT8/INT16 cousins) provides a way for **Lua Scripts** to have their own configurable settings in the standard parameter list.

Since ArduPilot supports custom user-written scripts, those scripts often need variables that the user can change (like a timer duration or a sensitivity factor). These "User Parameters" are placeholders that the scripts can read and write.

Tuning & Behavior

- **Usage:** Only relevant if you have a Lua script installed on your drone that is designed to use these parameters. Consult the documentation for your specific script to see what value to enter here.

SCR_USER_INT16 (Scripting User Parameter (Int16))

DEFAULT: 0	RANGE: -32768 32767

Scripting User Parameter (SCR_USER_INT16)

Description

Provides 16-bit integer storage for custom Lua scripts.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





SCR_USER_INT8 (Scripting User Parameter (Int8))

DEFAULT: 0	RANGE: -128 127

Scripting User Parameter (SCR_USER_INT8)

Description

Provides 8-bit integer storage for custom Lua scripts.

SCR_VM_I_COUNT (Scripting Instruction Count)

DEFAULT: 10000	RANGE: 1000 100000

Scripting Instruction Count (SCR_VM_I_COUNT)

Description

`SCR_VM_I_COUNT` acts as a "Speed Governor" for scripts to ensure they don't slow down the flight control loops.

If a script enters an infinite loop or tries to do too much math at once, the autopilot will pause the script after it reaches this many instructions, allowing the main flight stabilization code to run.

- **Default:** 10,000 instructions.
- **Recommendation:** Leave at default unless you have a very complex script that is triggering "Scripting: slow" warnings.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





SERIAL PORT CONFIGURATION (SERIAL)

Overview

The **SERIAL** parameter group configures the physical **UART (Universal Asynchronous Receiver-Transmitter)** ports on the flight controller.

Effective serial configuration is the backbone of drone integration, allowing the autopilot to talk to GPS units, Telemetry Radios, Companion Computers, ESCs, and OSDs.

Key Concepts

1. Serial Port Indexing

- `SERIAL0`: Usually the USB port.
- `SERIAL1`: Usually Telemetry 1.
- `SERIAL3`: Usually GPS 1.
  (Note: Mapping varies by board. Check your board's Wiki page).

2. Protocol Selection (`SERIALx_PROTOCOL`)

Tells the autopilot what "Language" to speak on each port.

- **1:** MAVLink 1.
- **2:** MAVLink 2 (Recommended for telemetry).
- **5:** GPS (NMEA/u-Blox).
- **10:** FrSky Passthrough.
- **16:** ESC Telemetry.
- **33:** MSP (Digital OSD).

3. Baud Rate (`SERIALx_BAUD`)

The communication speed. Both the autopilot and the connected device must match.

- **57:** 57,600 (Standard for SiK Radios).
- **115:** 115,200 (Standard for GPS/OSD).
- **921:** 921,600 (Recommended for Companion Computers/ROS).

4. Options (`SERIALx_OPTIONS`)

A bitmask for advanced UART features.

- **Bit 0:** Invert TX.
- **Bit 1:** Invert RX.
- **Bit 2:** Swap TX/RX pins (handy if you cross your wires!).
- **Bit 3:** Single-wire mode.

Parameter Breakdown

- `SERIALx_PROTOCOL`: Function of the port.
- `SERIALx_BAUD`: Speed of the port.
- `SERIAL_PASS1 / 2`: Used for "Serial Passthrough," allowing you to update GPS or ESC firmware through the autopilot.

Integration Guide

1. **Requirement:** Check your device manual for its default baud rate.
2. **Config:** Set the protocol and baud rate.
3. **Reboot: Critical.** Serial settings almost always require a full reboot to initialize the new drivers.

Developer Notes

- **Library:** `libraries/AP_SerialManager`.
- **DMA:** High-speed ports (921k+) often utilize DMA (Direct Memory Access) to reduce CPU load.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





SERIAL0_BAUD (Console/USB Baud Rate)

DEFAULT: 115	RANGE: 1 12500000

Console/USB Baud Rate (SERIAL0_BAUD)

Description

`SERIAL0_BAUD` defines the speed of the primary connection to your computer.

On most flight controllers, `SERIAL0` is the **USB** port. While USB itself doesn't have a "Baud Rate" in the traditional sense (it runs at high speed), the autopilot uses this parameter to determine the speed of the virtual UART bridged over USB.

- **115 (Default):** 115200 baud. standard for most connections.

Tuning & Behavior

- **Usage:** For most users, leave at **115**.
- **ESP32-S3 / Linux:** On boards where the primary console is a physical UART instead of a dedicated USB-Serial bridge, setting this correctly is essential for accessing the bootloader and terminal.

SERIAL0_PROTOCOL (Console/USB Protocol)

DEFAULT: 2	RANGE: -1 42

Console/USB Protocol (SERIAL0_PROTOCOL)

Description

`SERIAL0_PROTOCOL` tells the autopilot what "Language" to speak on the primary USB/Console port.

- **2 (Default):** MAVLink 2. Standard protocol for Ground Control Stations (Mission Planner, QGC).
- **-1:** None / Disabled.
- **1:** MAVLink 1.

Tuning & Behavior

- **Recommendation:** Leave at **2** for all normal operations.
- **Caution:** If you change this to a non-MAVLink protocol, you may lose the ability to connect to the flight controller via USB to change parameters. You would then need to connect via a different telemetry port to fix the setting.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





SERIAL1_BAUD (Telem1 Baud Rate)

DEFAULT: 57	RANGE: 1 12500000

Telem1 Baud Rate (SERIAL1_BAUD)

Description

`SERIAL1_BAUD` sets the speed of the serial connection.

The value is generally in **baud / 1000** (e.g., 57 = 57600 baud). However, very high speeds (like 2Mbaud) can be entered directly as integers in some GCS implementations, but the parameter storage is compressed.

- **57:** 57600 (Standard for SiK Telemetry Radios).
- **115:** 115200 (Standard for most GPS units and Wifi bridges).
- **921:** 921600 (High speed, good for fast logging or offboard control).
- **1500:** 1.5 Mbaud.

The Mathematics

$$\text{Baud Rate} = \text{SERIAL1_BAUD} \times 1000$$

(For values < 2000. Values > 2000 are interpreted literally).

The Engineer's View

Used in `AP_SerialManager::init()`.
The port configuration (start bits, stop bits, parity) is usually determined by the Protocol, but Baud is set here.
MAVLink usually requires 8N1 (8 data bits, No parity, 1 stop bit). S.Bus requires 100k baud, 8E2.

Tuning & Behavior

- **Default Value:** 57 (57600)
- **Recommendation:**
  - **SiK Radio:** 57
  - **GPS:** 115 or 230 (Ensure the GPS is configured to match, or use auto-baud).
  - **ESP8266/ESP32 Wifi:** 921 (for fast Mavlink).
  - **RC Receiver (ELRS/Crossfire):** Usually auto-negotiated or fixed by protocol (e.g., CRSF runs at ~400k).

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





SERIAL1_OPTIONS (Telem1 options)

DEFAULT: 0	RANGE: 0 8191

Telem1 options (SERIAL1_OPTIONS)

Description

`SERIAL1_OPTIONS` is a bitmask to configure low-level hardware properties of the UART port.

- **Bit 0 (1): Invert RX.** Inverts the logic level of the Receive pin. Used for S.Bus and some telemetry protocols.
- **Bit 1 (2): Invert TX.** Inverts the Transmit pin.
- **Bit 2 (4): Half Duplex.** Combines RX and TX onto a single wire (usually the TX pin). Used for FPort, S.Port, and some older protocols.
- **Bit 3 (8): Swap TX/RX.** Swaps the physical pins (if supported by the STM32 hardware).
- **Bit 4 (16): Pull Down RX.** Enables internal pull-down resistor.
- **Bit 5 (32): Pull Up RX.** Enables internal pull-up resistor.
- **Bit 6 (64): Pull Down TX.**
- **Bit 7 (128): Pull Up TX.**

The Mathematics

$$\text{Options} = \sum 2^i \cdot b_i$$

Example: For **FPort** (Inverted + Half Duplex):
Inverted TX (2) + Half Duplex (4) = **6**? Or sometimes Inverted RX (1) + Half Duplex (4) = **5**?
(Note: Half Duplex usually implies TX pin usage. The inversion requirement depends on whether the receiver sends inverted S.Port or un-inverted FPort).

The Engineer's View

These bits map directly to the HAL `UARTDriver::set_options()` method.
This provides software control over signal inversion, which historically required hardware inverters on older Flight Controllers (like Pixhawk 1). Modern F7/H7 chips have built-in inverters.

Tuning & Behavior

- **Default Value:** 0 (Standard 8N1, Full Duplex)
- **Common Use Cases:**
  - **S.Bus:** Usually requires **Invert RX (1)** if connected to a UART (not the RCIN pin).
  - **S.Port / FPort:** Requires **Half Duplex (4)** and often **Invert (2)** depending on the receiver firmware.
  - **GPS/Telemetry:** Usually **0**.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





SERIAL1_PROTOCOL (Telem1 protocol selection)

DEFAULT: 2	RANGE: -1 49

Telem1 protocol selection (SERIAL1_PROTOCOL)

Description

`SERIAL1_PROTOCOL` determines what language ArduPilot speaks on the first telemetry port.

- **2: MAVLink2 (Default).** Modern telemetry protocol. Use this for SiK Radios, Wifi Bridges, and Onboard Computers (Raspberry Pi/Jetson).
- **5: GPS.** Connect an external GPS unit here.
- **10: FrSky SPort Passthrough.** For OpenTX/EdgeTX telemetry on Taranis/Radiomaster radios.
- **23: RCIN.** If you want to connect your ELRS/Crossfire receiver to this UART instead of the dedicated RCIN pin.

The Mathematics

This parameter is an enumeration (integer ID) that selects the driver backend instantiated for this UART.

The Engineer's View

Used in `AP_SerialManager::init()`.
Changing this parameter usually requires a **Reboot** to initialize the correct driver.
ArduPilot supports concurrent protocols on different ports (e.g., MAVLink on Serial 1, GPS on Serial 3, FrSky on Serial 4).

Tuning & Behavior

- **Default Value:** 2 (MAVLink2)
- **Recommendation:**
  - **Telemetry Radio:** 2
  - **GPS:** 5
  - **Crossfire/ELRS:** 23 (RCIN) - Note: Crossfire VTX control uses protocol 29.
  - **DJI FPV (OSD):** 33
  - **Disabled:** -1 (Use to free up resources or if port is broken).

SERIAL2_BAUD (Telemetry 2 Baud Rate)

DEFAULT: DEFAULT_SERIAL2_BAUD	RANGE: null

Telemetry 2 Baud Rate

Note: This parameter functions identically to SERIAL1_BAUD.

SERIAL2_OPTIONS (Telem2 options)

DEFAULT: DEFAULT_SERIAL2_OPTIONS	RANGE: null

Telem2 options

Note: This parameter functions identically to SERIAL1_OPTIONS.

SERIAL2_PROTOCOL (Telemetry 2 protocol selection)

DEFAULT: DEFAULT_SERIAL2_PROTOCOL	RANGE: null

Telemetry 2 protocol selection

Note: This parameter functions identically to SERIAL1_PROTOCOL.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





SERIAL3_BAUD (Serial 3 (GPS) Baud Rate)

DEFAULT: DEFAULT_SERIAL3_BAUD	RANGE: null

Serial 3 (GPS) Baud Rate

Note: This parameter functions identically to SERIAL1_BAUD.

SERIAL3_OPTIONS (Serial3 options)

DEFAULT: DEFAULT_SERIAL3_OPTIONS	RANGE: null

Serial3 options

Note: This parameter functions identically to SERIAL1_OPTIONS.

SERIAL3_PROTOCOL (Serial 3 (GPS) protocol selection)

DEFAULT: DEFAULT_SERIAL3_PROTOCOL	RANGE: null

Serial 3 (GPS) protocol selection

Note: This parameter functions identically to SERIAL1_PROTOCOL.

SERIAL4_BAUD (Serial 4 Baud Rate)

DEFAULT: DEFAULT_SERIAL4_BAUD	RANGE: null

Serial 4 Baud Rate

Note: This parameter functions identically to SERIAL1_BAUD.

SERIAL4_OPTIONS (Serial4 options)

DEFAULT: DEFAULT_SERIAL4_OPTIONS	RANGE: null

Serial4 options

Note: This parameter functions identically to SERIAL1_OPTIONS.

SERIAL4_PROTOCOL (Serial4 protocol selection)

DEFAULT: DEFAULT_SERIAL4_PROTOCOL	RANGE: null

Serial4 protocol selection

Note: This parameter functions identically to SERIAL1_PROTOCOL.

SERIAL5_BAUD (Serial 5 Baud Rate)

DEFAULT: DEFAULT_SERIAL5_BAUD	RANGE: null

Serial 5 Baud Rate

Note: This parameter functions identically to SERIAL1_BAUD.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





SERIAL5_OPTIONS (Serial5 options)

DEFAULT: DEFAULT_SERIAL5_OPTIONS	RANGE: null

Serial5 options

Note: This parameter functions identically to SERIAL1_OPTIONS.

SERIAL5_PROTOCOL (Serial5 protocol selection)

DEFAULT: DEFAULT_SERIAL5_PROTOCOL	RANGE: null

Serial5 protocol selection

Note: This parameter functions identically to SERIAL1_PROTOCOL.

SERIAL6_BAUD (Serial 6 Baud Rate)

DEFAULT: DEFAULT_SERIAL6_BAUD	RANGE: null

Serial 6 Baud Rate

Note: This parameter functions identically to SERIAL1_BAUD.

SERIAL6_OPTIONS (Serial6 options)

DEFAULT: DEFAULT_SERIAL6_OPTIONS	RANGE: null

Serial6 options

Note: This parameter functions identically to SERIAL1_OPTIONS.

SERIAL6_PROTOCOL (Serial6 protocol selection)

DEFAULT: DEFAULT_SERIAL6_PROTOCOL	RANGE: null

Serial6 protocol selection

Note: This parameter functions identically to SERIAL1_PROTOCOL.

SERIAL7_BAUD (Serial 7 Baud Rate)

DEFAULT: DEFAULT_SERIAL7_BAUD	RANGE: null

Serial 7 Baud Rate

Note: This parameter functions identically to SERIAL1_BAUD.

SERIAL7_OPTIONS (Serial7 options)

DEFAULT: 0	RANGE: null

Serial7 options

Note: This parameter functions identically to SERIAL1_OPTIONS.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





SERIAL7_PROTOCOL (Serial7 protocol selection)

DEFAULT: DEFAULT_SERIAL7_PROTOCOL	RANGE: null

Serial7 protocol selection

Note: This parameter functions identically to SERIAL1_PROTOCOL.

SERIAL8_BAUD (Serial 8 Baud Rate)

DEFAULT: DEFAULT_SERIAL8_BAUD	RANGE: null

Serial 8 Baud Rate

Note: This parameter functions identically to SERIAL1_BAUD.

SERIAL8_OPTIONS (Serial8 options)

DEFAULT: 0	RANGE: null

Serial8 options

Note: This parameter functions identically to SERIAL1_OPTIONS.

SERIAL8_PROTOCOL (Serial8 protocol selection)

DEFAULT: DEFAULT_SERIAL8_PROTOCOL	RANGE: null

Serial8 protocol selection

Note: This parameter functions identically to SERIAL1_PROTOCOL.

SERIAL9_BAUD (Serial 9 Baud Rate)

DEFAULT: DEFAULT_SERIAL9_BAUD	RANGE: null

Serial 9 Baud Rate

Note: This parameter functions identically to SERIAL1_BAUD.

SERIAL9_OPTIONS (Serial9 options)

DEFAULT: DEFAULT_SERIAL9_OPTIONS	RANGE: null

Serial9 options

Note: This parameter functions identically to SERIAL1_OPTIONS.

SERIAL9_PROTOCOL (Serial9 protocol selection)

DEFAULT: DEFAULT_SERIAL9_PROTOCOL	RANGE: null

Serial9 protocol selection

Note: This parameter functions identically to SERIAL1_PROTOCOL.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





SERIAL_PASS1 (Serial Passthrough Port 1)

DEFAULT: -1	RANGE: -1 10

Serial Passthrough Port 1 (SERIAL_PASS1)

Description

`SERIAL_PASS1` and `SERIAL_PASS2` create a "Virtual Wire" inside the flight controller.

When you set both of these to valid port numbers (e.g., 0 for USB and 3 for GPS), the flight controller stops processing the data on those ports and simply forwards it back and forth. This is used to "talk through" the drone to reach external sensors for configuration (like using the u-blox u-center software to update GPS firmware while it is plugged into the drone).

Tuning & Behavior

- **Default:** -1 (Disabled).
- **Usage:**
  1. Set `SERIAL_PASS1` to **0** (USB).
  2. Set `SERIAL_PASS2` to the index of the port you want to reach (e.g. 3 for SERIAL3).
  3. Connect with your external software.
- **Safety:** normal flight functionality on these ports is **Disabled** while passthrough is active. Do not attempt to fly while this is enabled.

SERIAL_PASS2 (Serial Passthrough Port 2)

DEFAULT: -1	RANGE: -1 10

Serial Passthrough Port 2 (SERIAL_PASS2)

Description

Defines the other end of the serial bridge initiated by SERIAL_PASS1.

SERIAL_PASSTIMO (Serial Passthrough Timeout)

DEFAULT: 0	RANGE: 0 120	UNITS: s

Serial Passthrough Timeout (SERIAL_PASSTIMO)

Description

`SERIAL_PASSTIMO` is a safety timer for the serial bridge.

If no data is sent or received on the bridged ports for this many seconds, the autopilot will automatically close the bridge and revert the ports to their normal flight functions. This prevents the drone from being left in a "Non-Flight" state if you forget to manually disable the passthrough.

- **0:** Disabled (No automatic timeout).
- **Value:** Timeout in seconds.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **SERVO** parameter group (often appearing as `SERVOx_` or `MOT_`) configures the physical **PWM / DShot / SBus** outputs of the flight controller.

While `RCMAP` handles *Input*, the `SERVO` group handles *Output*. It defines what each pin on the servo rail actually does (e.g., Motor 1, Aileron, Camera Trigger, etc.) and how the signal is scaled.

Key Concepts

1. Servo Functions (`SERVOx_FUNCTION`)

The most important setting. Tells the autopilot what is plugged into the pin.

- **0:** Disabled.
- **33:** Motor 1 (Multicopter).
- **4:** Aileron (Plane).
- **10:** Camera Trigger.
- **51:** RC Passthrough (Directly copies an input channel to this output).

2. Output Calibration (`SERVOx_MIN`, `MAX`, `TRIM`)

Defines the pulse-width limits (microseconds).

- `MIN`: The "Low" endpoint (typically 1100us).
- `MAX`: The "High" endpoint (typically 1900us).
- `TRIM`: The "Neutral" position (typically 1500us).
- `REVERSED`: Flips the direction of travel (0=Normal, 1=Reversed).

3. BLHeli / DShot Pass-through (`SERVO_BLH_...`)

Configures the interface for BLHeli_32 ESCs, allowing the autopilot to perform firmware updates and read RPM telemetry (Bi-directional DShot).

Parameter Breakdown

- `SERVO_RATE`: The update frequency for standard PWM servos (default 50Hz).
- `SERVO_DSHOT_RATE`: The speed of digital DShot outputs.
- `SERVO_GPIO_MASK`: Converts PWM pins into simple digital High/Low pins.

Integration Guide

Setting up a New Motor

1. Connect ESC to Output 1.
2. Set `SERVO1_FUNCTION = 33` (Motor 1).
3. Set `MOT_PWM_TYPE` to your desired protocol (e.g., DShot600).
4. **Test:** Use the "Motor Test" tab in the GCS to verify rotation direction. If wrong, use `SERVO1_REVERSED` (for PWM) or BLHeliSuite (for DShot).

Developer Notes

- **Library:** `libraries/SRV_Channel`.
- **Safety:** PWM outputs are disabled until the vehicle is Armed or the Hardware Safety Switch is pressed.

SERVO10_FUNCTION (Servo output function)

DEFAULT: 0	RANGE: null

Servo output function

Note: This parameter functions identically to SERVO1_FUNCTION.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





SERVO10_MAX (Maximum PWM)

DEFAULT: 1900	RANGE: 800 2200	UNITS: PWM

Maximum PWM

Note: This parameter functions identically to SERVO1_MAX.

SERVO10_MIN (Minimum PWM)

DEFAULT: 1100	RANGE: 800 2200	UNITS: PWM

Minimum PWM

Note: This parameter functions identically to SERVO1_MIN.

SERVO10_REVERSED (Servo reverse)

DEFAULT: 0	RANGE: null

Servo reverse

Note: This parameter functions identically to SERVO1_REVERSED.

SERVO10_TRIM (Trim PWM)

DEFAULT: 1500	RANGE: 800 2200	UNITS: PWM

Trim PWM

Note: This parameter functions identically to SERVO1_TRIM.

SERVO11_FUNCTION (Servo output function)

DEFAULT: 0	RANGE: null

Servo output function

Note: This parameter functions identically to SERVO1_FUNCTION.

SERVO11_MAX (Maximum PWM)

DEFAULT: 1900	RANGE: 800 2200	UNITS: PWM

Maximum PWM

Note: This parameter functions identically to SERVO1_MAX.

SERVO11_MIN (Minimum PWM)

DEFAULT: 1100	RANGE: 800 2200	UNITS: PWM

Minimum PWM

Note: This parameter functions identically to SERVO1_MIN.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





SERVO11_REVERSED (Servo reverse)

DEFAULT: 0	RANGE: null

Servo reverse

Note: This parameter functions identically to SERVO1_REVERSED.

SERVO11_TRIM (Trim PWM)

DEFAULT: 1500	RANGE: 800 2200	UNITS: PWM

Trim PWM

Note: This parameter functions identically to SERVO1_TRIM.

SERVO12_FUNCTION (Servo output function)

DEFAULT: 0	RANGE: null

Servo output function

Note: This parameter functions identically to SERVO1_FUNCTION.

SERVO12_MAX (Maximum PWM)

DEFAULT: 1900	RANGE: 800 2200	UNITS: PWM

Maximum PWM

Note: This parameter functions identically to SERVO1_MAX.

SERVO12_MIN (Minimum PWM)

DEFAULT: 1100	RANGE: 800 2200	UNITS: PWM

Minimum PWM

Note: This parameter functions identically to SERVO1_MIN.

SERVO12_REVERSED (Servo reverse)

DEFAULT: 0	RANGE: null

Servo reverse

Note: This parameter functions identically to SERVO1_REVERSED.

SERVO12_TRIM (Trim PWM)

DEFAULT: 1500	RANGE: 800 2200	UNITS: PWM

Trim PWM

Note: This parameter functions identically to SERVO1_TRIM.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





SERVO13_FUNCTION (Servo output function)

DEFAULT: 0	RANGE: null

Servo output function

Note: This parameter functions identically to SERVO1_FUNCTION.

SERVO13_MAX (Maximum PWM)

DEFAULT: 1900	RANGE: 800 2200	UNITS: PWM

Maximum PWM

Note: This parameter functions identically to SERVO1_MAX.

SERVO13_MIN (Minimum PWM)

DEFAULT: 1100	RANGE: 800 2200	UNITS: PWM

Minimum PWM

Note: This parameter functions identically to SERVO1_MIN.

SERVO13_REVERSED (Servo reverse)

DEFAULT: 0	RANGE: null

Servo reverse

Note: This parameter functions identically to SERVO1_REVERSED.

SERVO13_TRIM (Trim PWM)

DEFAULT: 1500	RANGE: 800 2200	UNITS: PWM

Trim PWM

Note: This parameter functions identically to SERVO1_TRIM.

SERVO14_FUNCTION (Servo output function)

DEFAULT: 0	RANGE: null

Servo output function

Note: This parameter functions identically to SERVO1_FUNCTION.

SERVO14_MAX (Maximum PWM)

DEFAULT: 1900	RANGE: 800 2200	UNITS: PWM

Maximum PWM

Note: This parameter functions identically to SERVO1_MAX.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





SERVO14_MIN (Minimum PWM)

DEFAULT: 1100	RANGE: 800 2200	UNITS: PWM

Minimum PWM

Note: This parameter functions identically to SERVO1_MIN.

SERVO14_REVERSED (Servo reverse)

DEFAULT: 0	RANGE: null

Servo reverse

Note: This parameter functions identically to SERVO1_REVERSED.

SERVO14_TRIM (Trim PWM)

DEFAULT: 1500	RANGE: 800 2200	UNITS: PWM

Trim PWM

Note: This parameter functions identically to SERVO1_TRIM.

SERVO15_FUNCTION (Servo output function)

DEFAULT: 0	RANGE: null

Servo output function

Note: This parameter functions identically to SERVO1_FUNCTION.

SERVO15_MAX (Maximum PWM)

DEFAULT: 1900	RANGE: 800 2200	UNITS: PWM

Maximum PWM

Note: This parameter functions identically to SERVO1_MAX.

SERVO15_MIN (Minimum PWM)

DEFAULT: 1100	RANGE: 800 2200	UNITS: PWM

Minimum PWM

Note: This parameter functions identically to SERVO1_MIN.

SERVO15_REVERSED (Servo reverse)

DEFAULT: 0	RANGE: null

Servo reverse

Note: This parameter functions identically to SERVO1_REVERSED.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





SERVO15_TRIM (Trim PWM)

DEFAULT: 1500	RANGE: 800 2200	UNITS: PWM

Trim PWM

Note: This parameter functions identically to SERVO1_TRIM.

SERVO16_FUNCTION (Servo output function)

DEFAULT: 0	RANGE: null

Servo output function

Note: This parameter functions identically to SERVO1_FUNCTION.

SERVO16_MAX (Maximum PWM)

DEFAULT: 1900	RANGE: 800 2200	UNITS: PWM

Maximum PWM

Note: This parameter functions identically to SERVO1_MAX.

SERVO16_MIN (Minimum PWM)

DEFAULT: 1100	RANGE: 800 2200	UNITS: PWM

Minimum PWM

Note: This parameter functions identically to SERVO1_MIN.

SERVO16_REVERSED (Servo reverse)

DEFAULT: 0	RANGE: null

Servo reverse

Note: This parameter functions identically to SERVO1_REVERSED.

SERVO16_TRIM (Trim PWM)

DEFAULT: 1500	RANGE: 800 2200	UNITS: PWM

Trim PWM

Note: This parameter functions identically to SERVO1_TRIM.

SERVO17_FUNCTION (Servo output function)

DEFAULT: 0	RANGE: null

Servo output function

Note: This parameter functions identically to SERVO1_FUNCTION.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





SERVO17_MAX (Maximum PWM)

DEFAULT: 1900	RANGE: 800 2200	UNITS: PWM

Maximum PWM

Note: This parameter functions identically to SERVO1_MAX.

SERVO17_MIN (Minimum PWM)

DEFAULT: 1100	RANGE: 800 2200	UNITS: PWM

Minimum PWM

Note: This parameter functions identically to SERVO1_MIN.

SERVO17_REVERSED (Servo reverse)

DEFAULT: 0	RANGE: null

Servo reverse

Note: This parameter functions identically to SERVO1_REVERSED.

SERVO17_TRIM (Trim PWM)

DEFAULT: 1500	RANGE: 800 2200	UNITS: PWM

Trim PWM

Note: This parameter functions identically to SERVO1_TRIM.

SERVO18_FUNCTION (Servo output function)

DEFAULT: 0	RANGE: null

Servo output function

Note: This parameter functions identically to SERVO1_FUNCTION.

SERVO18_MAX (Maximum PWM)

DEFAULT: 1900	RANGE: 800 2200	UNITS: PWM

Maximum PWM

Note: This parameter functions identically to SERVO1_MAX.

SERVO18_MIN (Minimum PWM)

DEFAULT: 1100	RANGE: 800 2200	UNITS: PWM

Minimum PWM

Note: This parameter functions identically to SERVO1_MIN.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





SERVO18_REVERSED (Servo reverse)

DEFAULT: 0	RANGE: null

Servo reverse

Note: This parameter functions identically to SERVO1_REVERSED.

SERVO18_TRIM (Trim PWM)

DEFAULT: 1500	RANGE: 800 2200	UNITS: PWM

Trim PWM

Note: This parameter functions identically to SERVO1_TRIM.

SERVO19_FUNCTION (Servo output function)

DEFAULT: 0	RANGE: null

Servo output function

Note: This parameter functions identically to SERVO1_FUNCTION.

SERVO19_MAX (Maximum PWM)

DEFAULT: 1900	RANGE: 800 2200	UNITS: PWM

Maximum PWM

Note: This parameter functions identically to SERVO1_MAX.

SERVO19_MIN (Minimum PWM)

DEFAULT: 1100	RANGE: 800 2200	UNITS: PWM

Minimum PWM

Note: This parameter functions identically to SERVO1_MIN.

SERVO19_REVERSED (Servo reverse)

DEFAULT: 0	RANGE: null

Servo reverse

Note: This parameter functions identically to SERVO1_REVERSED.

SERVO19_TRIM (Trim PWM)

DEFAULT: 1500	RANGE: 800 2200	UNITS: PWM

Trim PWM

Note: This parameter functions identically to SERVO1_TRIM.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





SERVO1_FUNCTION (Servo 1 Output Function)

DEFAULT: 0	RANGE: 0 300

Servo 1 Output Function (SERVO1_FUNCTION)

Description

`SERVO1_FUNCTION` tells the autopilot what is physically plugged into the first output pin on your flight controller.

By default, pins do nothing. You must assign a function to each pin so the autopilot knows where to send the motor commands, steering signals, or auxiliary triggers.

Common Values

- **0: Disabled.** Output remains at neutral/off.
- **33-40: Motor 1 - Motor 8.** (Standard multicopter motor mapping).
- **4: Aileron.** (Fixed-wing).
- **19: Elevator.** (Fixed-wing).
- **21: Rudder.** (Fixed-wing).
- **51-66: RCIN Passthrough.** Directly mirrors the signal from an RC channel.
- **120+: Relay.** Simple digital ON/OFF control.

Tuning & Behavior

- **Default:** 0.
- **Safety:** Never change these values while a battery is connected and propellers are attached. A change in function can cause a motor to spin up unexpectedly.
- **Redundancy:** You can assign the same function to multiple pins (e.g., two pins set to "Aileron") if you have dual servos on a single control surface.

SERVO1_MAX (Servo 1 Maximum)

DEFAULT: 1900	RANGE: 800 2200	UNITS: PWM

Servo 1 Maximum (SERVO1_MAX)

Description

`SERVO1_MAX` sets the "Upper Limit" for a servo or ESC.

It defines the longest pulse width that the autopilot will ever send to this pin.

- **Standard Setting:** 1900 (µs).
- **Recommendation:** Match this to the physical limit of your aircraft's hardware to prevent mechanical binding.

SERVO1_MIN (Servo 1 Minimum)

DEFAULT: 1100	RANGE: 800 2200	UNITS: PWM

Servo 1 Minimum (SERVO1_MIN)

Description

`SERVO1_MIN` sets the "Lower Limit" for a servo or ESC.

It defines the shortest pulse width that the autopilot will ever send to this pin. This is used to prevent a servo from trying to push past a physical wall or to set the "Off" point for an ESC.

- **Standard Setting:** 1100 (µs).
- **Safety:** Ensure that at this setting, the servo is not "stalling" (making a buzzing sound) against a mechanical limit.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





SERVO1_REVERSED (Servo 1 Reversal)

DEFAULT: 0	RANGE: 0 1

Servo 1 Reversal (SERVO1_REVERSED)

Description

`SERVO1_REVERSED` allows you to flip the direction of a motor or servo output without swapping wires.

If you move your transmitter stick right, but the drone rolls left, you can use this parameter to correct the behavior. This is safer than reversing the channel on your transmitter, as it ensures that the autopilot's internal stabilization logic also moves in the correct direction.

- **0: Normal.**
- **1: Reversed.**

Tuning & Behavior

- **Safety:** Always verify your control surface directions on the ground before taking off.
- **Motors:** For DShot motors, it is usually better to reverse the direction in the ESC firmware (using BLHeliSuite), but this parameter can also be used for simple reversals.

SERVO1_TRIM (Servo 1 Trim)

DEFAULT: 1500	RANGE: 800 2200	UNITS: PWM

Servo 1 Trim (SERVO1_TRIM)

Description

`SERVO1_TRIM` defines the "Center" position of a servo or motor.

When the autopilot is not demanding any movement (e.g., sticks are centered in Stabilize mode), it sends this PWM value to the output.

- **Standard Setting:** 1500 (µs).
- **Usage:** Use this to mechanically center a servo arm if it's slightly off-center, or to set the zero-thrust point for ESCs that support reverse thrust.

Tuning & Behavior

- **Calibration:** Most Ground Control Stations have a "Radio Calibration" or "Servo Output" screen where you can adjust this value to level your control surfaces.
- **Safety:** Do not use large trim offsets to fix a crooked mechanical linkage. It is always better to adjust the linkage physically so that the trim remains near 1500.

SERVO20_FUNCTION (Servo output function)

DEFAULT: 0	RANGE: null

Servo output function

Note: This parameter functions identically to SERVO1_FUNCTION.

SERVO20_MAX (Maximum PWM)

DEFAULT: 1900	RANGE: 800 2200	UNITS: PWM

Maximum PWM

Note: This parameter functions identically to SERVO1_MAX.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





SERVO20_MIN (Minimum PWM)

DEFAULT: 1100	RANGE: 800 2200	UNITS: PWM

Minimum PWM

Note: This parameter functions identically to SERVO1_MIN.

SERVO20_REVERSED (Servo reverse)

DEFAULT: 0	RANGE: null

Servo reverse

Note: This parameter functions identically to SERVO1_REVERSED.

SERVO20_TRIM (Trim PWM)

DEFAULT: 1500	RANGE: 800 2200	UNITS: PWM

Trim PWM

Note: This parameter functions identically to SERVO1_TRIM.

SERVO21_FUNCTION (Servo output function)

DEFAULT: 0	RANGE: null

Servo output function

Note: This parameter functions identically to SERVO1_FUNCTION.

SERVO21_MAX (Maximum PWM)

DEFAULT: 1900	RANGE: 800 2200	UNITS: PWM

Maximum PWM

Note: This parameter functions identically to SERVO1_MAX.

SERVO21_MIN (Minimum PWM)

DEFAULT: 1100	RANGE: 800 2200	UNITS: PWM

Minimum PWM

Note: This parameter functions identically to SERVO1_MIN.

SERVO21_REVERSED (Servo reverse)

DEFAULT: 0	RANGE: null

Servo reverse

Note: This parameter functions identically to SERVO1_REVERSED.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





SERVO21_TRIM (Trim PWM)

DEFAULT: 1500	RANGE: 800 2200	UNITS: PWM

Trim PWM

Note: This parameter functions identically to SERVO1_TRIM.

SERVO22_FUNCTION (Servo output function)

DEFAULT: 0	RANGE: null

Servo output function

Note: This parameter functions identically to SERVO1_FUNCTION.

SERVO22_MAX (Maximum PWM)

DEFAULT: 1900	RANGE: 800 2200	UNITS: PWM

Maximum PWM

Note: This parameter functions identically to SERVO1_MAX.

SERVO22_MIN (Minimum PWM)

DEFAULT: 1100	RANGE: 800 2200	UNITS: PWM

Minimum PWM

Note: This parameter functions identically to SERVO1_MIN.

SERVO22_REVERSED (Servo reverse)

DEFAULT: 0	RANGE: null

Servo reverse

Note: This parameter functions identically to SERVO1_REVERSED.

SERVO22_TRIM (Trim PWM)

DEFAULT: 1500	RANGE: 800 2200	UNITS: PWM

Trim PWM

Note: This parameter functions identically to SERVO1_TRIM.

SERVO23_FUNCTION (Servo output function)

DEFAULT: 0	RANGE: null

Servo output function

Note: This parameter functions identically to SERVO1_FUNCTION.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





SERVO23_MAX (Maximum PWM)

DEFAULT: 1900	RANGE: 800 2200	UNITS: PWM

Maximum PWM

Note: This parameter functions identically to SERVO1_MAX.

SERVO23_MIN (Minimum PWM)

DEFAULT: 1100	RANGE: 800 2200	UNITS: PWM

Minimum PWM

Note: This parameter functions identically to SERVO1_MIN.

SERVO23_REVERSED (Servo reverse)

DEFAULT: 0	RANGE: null

Servo reverse

Note: This parameter functions identically to SERVO1_REVERSED.

SERVO23_TRIM (Trim PWM)

DEFAULT: 1500	RANGE: 800 2200	UNITS: PWM

Trim PWM

Note: This parameter functions identically to SERVO1_TRIM.

SERVO24_FUNCTION (Servo output function)

DEFAULT: 0	RANGE: null

Servo output function

Note: This parameter functions identically to SERVO1_FUNCTION.

SERVO24_MAX (Maximum PWM)

DEFAULT: 1900	RANGE: 800 2200	UNITS: PWM

Maximum PWM

Note: This parameter functions identically to SERVO1_MAX.

SERVO24_MIN (Minimum PWM)

DEFAULT: 1100	RANGE: 800 2200	UNITS: PWM

Minimum PWM

Note: This parameter functions identically to SERVO1_MIN.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





SERVO24_REVERSED (Servo reverse)

DEFAULT: 0	RANGE: null

Servo reverse

Note: This parameter functions identically to SERVO1_REVERSED.

SERVO24_TRIM (Trim PWM)

DEFAULT: 1500	RANGE: 800 2200	UNITS: PWM

Trim PWM

Note: This parameter functions identically to SERVO1_TRIM.

SERVO25_FUNCTION (Servo output function)

DEFAULT: 0	RANGE: null

Servo output function

Note: This parameter functions identically to SERVO1_FUNCTION.

SERVO25_MAX (Maximum PWM)

DEFAULT: 1900	RANGE: 800 2200	UNITS: PWM

Maximum PWM

Note: This parameter functions identically to SERVO1_MAX.

SERVO25_MIN (Minimum PWM)

DEFAULT: 1100	RANGE: 800 2200	UNITS: PWM

Minimum PWM

Note: This parameter functions identically to SERVO1_MIN.

SERVO25_REVERSED (Servo reverse)

DEFAULT: 0	RANGE: null

Servo reverse

Note: This parameter functions identically to SERVO1_REVERSED.

SERVO25_TRIM (Trim PWM)

DEFAULT: 1500	RANGE: 800 2200	UNITS: PWM

Trim PWM

Note: This parameter functions identically to SERVO1_TRIM.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





SERVO26_FUNCTION (Servo output function)

DEFAULT: 0	RANGE: null

Servo output function

Note: This parameter functions identically to SERVO1_FUNCTION.

SERVO26_MAX (Maximum PWM)

DEFAULT: 1900	RANGE: 800 2200	UNITS: PWM

Maximum PWM

Note: This parameter functions identically to SERVO1_MAX.

SERVO26_MIN (Minimum PWM)

DEFAULT: 1100	RANGE: 800 2200	UNITS: PWM

Minimum PWM

Note: This parameter functions identically to SERVO1_MIN.

SERVO26_REVERSED (Servo reverse)

DEFAULT: 0	RANGE: null

Servo reverse

Note: This parameter functions identically to SERVO1_REVERSED.

SERVO26_TRIM (Trim PWM)

DEFAULT: 1500	RANGE: 800 2200	UNITS: PWM

Trim PWM

Note: This parameter functions identically to SERVO1_TRIM.

SERVO27_FUNCTION (Servo output function)

DEFAULT: 0	RANGE: null

Servo output function

Note: This parameter functions identically to SERVO1_FUNCTION.

SERVO27_MAX (Maximum PWM)

DEFAULT: 1900	RANGE: 800 2200	UNITS: PWM

Maximum PWM

Note: This parameter functions identically to SERVO1_MAX.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





SERVO27_MIN (Minimum PWM)

DEFAULT: 1100	RANGE: 800 2200	UNITS: PWM

Minimum PWM

Note: This parameter functions identically to SERVO1_MIN.

SERVO27_REVERSED (Servo reverse)

DEFAULT: 0	RANGE: null

Servo reverse

Note: This parameter functions identically to SERVO1_REVERSED.

SERVO27_TRIM (Trim PWM)

DEFAULT: 1500	RANGE: 800 2200	UNITS: PWM

Trim PWM

Note: This parameter functions identically to SERVO1_TRIM.

SERVO28_FUNCTION (Servo output function)

DEFAULT: 0	RANGE: null

Servo output function

Note: This parameter functions identically to SERVO1_FUNCTION.

SERVO28_MAX (Maximum PWM)

DEFAULT: 1900	RANGE: 800 2200	UNITS: PWM

Maximum PWM

Note: This parameter functions identically to SERVO1_MAX.

SERVO28_MIN (Minimum PWM)

DEFAULT: 1100	RANGE: 800 2200	UNITS: PWM

Minimum PWM

Note: This parameter functions identically to SERVO1_MIN.

SERVO28_REVERSED (Servo reverse)

DEFAULT: 0	RANGE: null

Servo reverse

Note: This parameter functions identically to SERVO1_REVERSED.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





SERVO28_TRIM (Trim PWM)

DEFAULT: 1500	RANGE: 800 2200	UNITS: PWM

Trim PWM

Note: This parameter functions identically to SERVO1_TRIM.

SERVO29_FUNCTION (Servo output function)

DEFAULT: 0	RANGE: null

Servo output function

Note: This parameter functions identically to SERVO1_FUNCTION.

SERVO29_MAX (Maximum PWM)

DEFAULT: 1900	RANGE: 800 2200	UNITS: PWM

Maximum PWM

Note: This parameter functions identically to SERVO1_MAX.

SERVO29_MIN (Minimum PWM)

DEFAULT: 1100	RANGE: 800 2200	UNITS: PWM

Minimum PWM

Note: This parameter functions identically to SERVO1_MIN.

SERVO29_REVERSED (Servo reverse)

DEFAULT: 0	RANGE: null

Servo reverse

Note: This parameter functions identically to SERVO1_REVERSED.

SERVO29_TRIM (Trim PWM)

DEFAULT: 1500	RANGE: 800 2200	UNITS: PWM

Trim PWM

Note: This parameter functions identically to SERVO1_TRIM.

SERVO2_FUNCTION (Servo output function)

DEFAULT: 0	RANGE: null

Servo output function

Note: This parameter functions identically to SERVO1_FUNCTION.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





SERVO2_MAX (Maximum PWM)

DEFAULT: 1900	RANGE: 800 2200	UNITS: PWM

Maximum PWM

Note: This parameter functions identically to SERVO1_MAX.

SERVO2_MIN (Minimum PWM)

DEFAULT: 1100	RANGE: 800 2200	UNITS: PWM

Minimum PWM

Note: This parameter functions identically to SERVO1_MIN.

SERVO2_REVERSED (Servo reverse)

DEFAULT: 0	RANGE: null

Servo reverse

Note: This parameter functions identically to SERVO1_REVERSED.

SERVO2_TRIM (Trim PWM)

DEFAULT: 1500	RANGE: 800 2200	UNITS: PWM

Trim PWM

Note: This parameter functions identically to SERVO1_TRIM.

SERVO30_FUNCTION (Servo output function)

DEFAULT: 0	RANGE: null

Servo output function

Note: This parameter functions identically to SERVO1_FUNCTION.

SERVO30_MAX (Maximum PWM)

DEFAULT: 1900	RANGE: 800 2200	UNITS: PWM

Maximum PWM

Note: This parameter functions identically to SERVO1_MAX.

SERVO30_MIN (Minimum PWM)

DEFAULT: 1100	RANGE: 800 2200	UNITS: PWM

Minimum PWM

Note: This parameter functions identically to SERVO1_MIN.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





SERVO30_REVERSED (Servo reverse)

DEFAULT: 0	RANGE: null

Servo reverse

Note: This parameter functions identically to SERVO1_REVERSED.

SERVO30_TRIM (Trim PWM)

DEFAULT: 1500	RANGE: 800 2200	UNITS: PWM

Trim PWM

Note: This parameter functions identically to SERVO1_TRIM.

SERVO31_FUNCTION (Servo output function)

DEFAULT: 0	RANGE: null

Servo output function

Note: This parameter functions identically to SERVO1_FUNCTION.

SERVO31_MAX (Maximum PWM)

DEFAULT: 1900	RANGE: 800 2200	UNITS: PWM

Maximum PWM

Note: This parameter functions identically to SERVO1_MAX.

SERVO31_MIN (Minimum PWM)

DEFAULT: 1100	RANGE: 800 2200	UNITS: PWM

Minimum PWM

Note: This parameter functions identically to SERVO1_MIN.

SERVO31_REVERSED (Servo reverse)

DEFAULT: 0	RANGE: null

Servo reverse

Note: This parameter functions identically to SERVO1_REVERSED.

SERVO31_TRIM (Trim PWM)

DEFAULT: 1500	RANGE: 800 2200	UNITS: PWM

Trim PWM

Note: This parameter functions identically to SERVO1_TRIM.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





SERVO32_FUNCTION (Servo output function)

DEFAULT: 0	RANGE: null

Servo output function

Note: This parameter functions identically to SERVO1_FUNCTION.

SERVO32_MAX (Maximum PWM)

DEFAULT: 1900	RANGE: 800 2200	UNITS: PWM

Maximum PWM

Note: This parameter functions identically to SERVO1_MAX.

SERVO32_MIN (Minimum PWM)

DEFAULT: 1100	RANGE: 800 2200	UNITS: PWM

Minimum PWM

Note: This parameter functions identically to SERVO1_MIN.

SERVO32_REVERSED (Servo reverse)

DEFAULT: 0	RANGE: null

Servo reverse

Note: This parameter functions identically to SERVO1_REVERSED.

SERVO32_TRIM (Trim PWM)

DEFAULT: 1500	RANGE: 800 2200	UNITS: PWM

Trim PWM

Note: This parameter functions identically to SERVO1_TRIM.

SERVO3_FUNCTION (Servo output function)

DEFAULT: 0	RANGE: null

Servo output function

Note: This parameter functions identically to SERVO1_FUNCTION.

SERVO3_MAX (Maximum PWM)

DEFAULT: 1900	RANGE: 800 2200	UNITS: PWM

Maximum PWM

Note: This parameter functions identically to SERVO1_MAX.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





SERVO3_MIN (Minimum PWM)

DEFAULT: 1100	RANGE: 800 2200	UNITS: PWM

Minimum PWM

Note: This parameter functions identically to SERVO1_MIN.

SERVO3_REVERSED (Servo reverse)

DEFAULT: 0	RANGE: null

Servo reverse

Note: This parameter functions identically to SERVO1_REVERSED.

SERVO3_TRIM (Trim PWM)

DEFAULT: 1500	RANGE: 800 2200	UNITS: PWM

Trim PWM

Note: This parameter functions identically to SERVO1_TRIM.

SERVO4_FUNCTION (Servo output function)

DEFAULT: 0	RANGE: null

Servo output function

Note: This parameter functions identically to SERVO1_FUNCTION.

SERVO4_MAX (Maximum PWM)

DEFAULT: 1900	RANGE: 800 2200	UNITS: PWM

Maximum PWM

Note: This parameter functions identically to SERVO1_MAX.

SERVO4_MIN (Minimum PWM)

DEFAULT: 1100	RANGE: 800 2200	UNITS: PWM

Minimum PWM

Note: This parameter functions identically to SERVO1_MIN.

SERVO4_REVERSED (Servo reverse)

DEFAULT: 0	RANGE: null

Servo reverse

Note: This parameter functions identically to SERVO1_REVERSED.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





SERVO4_TRIM (Trim PWM)

DEFAULT: 1500	RANGE: 800 2200	UNITS: PWM

Trim PWM

Note: This parameter functions identically to SERVO1_TRIM.

SERVO5_FUNCTION (Servo output function)

DEFAULT: 0	RANGE: null

Servo output function

Note: This parameter functions identically to SERVO1_FUNCTION.

SERVO5_MAX (Maximum PWM)

DEFAULT: 1900	RANGE: 800 2200	UNITS: PWM

Maximum PWM

Note: This parameter functions identically to SERVO1_MAX.

SERVO5_MIN (Minimum PWM)

DEFAULT: 1100	RANGE: 800 2200	UNITS: PWM

Minimum PWM

Note: This parameter functions identically to SERVO1_MIN.

SERVO5_REVERSED (Servo reverse)

DEFAULT: 0	RANGE: null

Servo reverse

Note: This parameter functions identically to SERVO1_REVERSED.

SERVO5_TRIM (Trim PWM)

DEFAULT: 1500	RANGE: 800 2200	UNITS: PWM

Trim PWM

Note: This parameter functions identically to SERVO1_TRIM.

SERVO6_FUNCTION (Servo output function)

DEFAULT: 0	RANGE: null

Servo output function

Note: This parameter functions identically to SERVO1_FUNCTION.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





SERVO6_MAX (Maximum PWM)

DEFAULT: 1900	RANGE: 800 2200	UNITS: PWM

Maximum PWM

Note: This parameter functions identically to SERVO1_MAX.

SERVO6_MIN (Minimum PWM)

DEFAULT: 1100	RANGE: 800 2200	UNITS: PWM

Minimum PWM

Note: This parameter functions identically to SERVO1_MIN.

SERVO6_REVERSED (Servo reverse)

DEFAULT: 0	RANGE: null

Servo reverse

Note: This parameter functions identically to SERVO1_REVERSED.

SERVO6_TRIM (Trim PWM)

DEFAULT: 1500	RANGE: 800 2200	UNITS: PWM

Trim PWM

Note: This parameter functions identically to SERVO1_TRIM.

SERVO7_FUNCTION (Servo output function)

DEFAULT: 0	RANGE: null

Servo output function

Note: This parameter functions identically to SERVO1_FUNCTION.

SERVO7_MAX (Maximum PWM)

DEFAULT: 1900	RANGE: 800 2200	UNITS: PWM

Maximum PWM

Note: This parameter functions identically to SERVO1_MAX.

SERVO7_MIN (Minimum PWM)

DEFAULT: 1100	RANGE: 800 2200	UNITS: PWM

Minimum PWM

Note: This parameter functions identically to SERVO1_MIN.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





SERVO7_REVERSED (Servo reverse)

DEFAULT: 0	RANGE: null

Servo reverse

Note: This parameter functions identically to SERVO1_REVERSED.

SERVO7_TRIM (Trim PWM)

DEFAULT: 1500	RANGE: 800 2200	UNITS: PWM

Trim PWM

Note: This parameter functions identically to SERVO1_TRIM.

SERVO8_FUNCTION (Servo output function)

DEFAULT: 0	RANGE: null

Servo output function

Note: This parameter functions identically to SERVO1_FUNCTION.

SERVO8_MAX (Maximum PWM)

DEFAULT: 1900	RANGE: 800 2200	UNITS: PWM

Maximum PWM

Note: This parameter functions identically to SERVO1_MAX.

SERVO8_MIN (Minimum PWM)

DEFAULT: 1100	RANGE: 800 2200	UNITS: PWM

Minimum PWM

Note: This parameter functions identically to SERVO1_MIN.

SERVO8_REVERSED (Servo reverse)

DEFAULT: 0	RANGE: null

Servo reverse

Note: This parameter functions identically to SERVO1_REVERSED.

SERVO8_TRIM (Trim PWM)

DEFAULT: 1500	RANGE: 800 2200	UNITS: PWM

Trim PWM

Note: This parameter functions identically to SERVO1_TRIM.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





SERVO9_FUNCTION (Servo output function)

DEFAULT: 0	RANGE: null

Servo output function

Note: This parameter functions identically to SERVO1_FUNCTION.

SERVO9_MAX (Maximum PWM)

DEFAULT: 1900	RANGE: 800 2200	UNITS: PWM

Maximum PWM

Note: This parameter functions identically to SERVO1_MAX.

SERVO9_MIN (Minimum PWM)

DEFAULT: 1100	RANGE: 800 2200	UNITS: PWM

Minimum PWM

Note: This parameter functions identically to SERVO1_MIN.

SERVO9_REVERSED (Servo reverse)

DEFAULT: 0	RANGE: null

Servo reverse

Note: This parameter functions identically to SERVO1_REVERSED.

SERVO9_TRIM (Trim PWM)

DEFAULT: 1500	RANGE: 800 2200	UNITS: PWM

Trim PWM

Note: This parameter functions identically to SERVO1_TRIM.

SERVO_32_ENABLE (Enable 32 Servo Channels)

DEFAULT: 0	RANGE: 0 1

Enable 32 Servo Channels (SERVO_32_ENABLE)

Description

`SERVO_32_ENABLE` unlocks the full potential of high-end flight controllers (like the Cube Orange or Durandal) that can support more than the standard 16 outputs.

- **0:** 16 Channels (Standard).
- **1:** 32 Channels.

Tuning & Behavior

- **Default Value:** 0.
- **Recommendation:** Leave at **0** unless you have a massive rover or plane with more than 16 physical servos.
- **Reboot Required:** Yes.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





SERVO_AUTO_TRIM (Servo Auto Trim)

DEFAULT: 0	RANGE: 0 1

Servo Auto Trim (SERVO_AUTO_TRIM)

Description

`SERVO_AUTO_TRIM` simplifies the process of trimming a plane.

Instead of manually adjusting linkages or radio trims, you can fly the plane level, and the autopilot will slowly learn the new center positions for the servos.

Tuning & Behavior

- **0:** Disabled.
- **1:** Enabled.
- **Note:** Usually activated via an RC switch option (`RCx_OPTION` = 155) rather than setting this parameter permanently.

SERVO_BLH_3DMASK (BLHeli bitmask of 3D channels)

DEFAULT: 0	RANGE: null	UNITS: null

BLHeli bitmask of 3D channels

Legacy Note: `SERVO_*` parameters have been renamed to `OUT_*` in newer firmware versions. This parameter functions identically to OUT_BLH_3DMASK.

SERVO_BLH_AUTO (BLHeli pass-thru auto-enable)

DEFAULT: 0	RANGE: 0 1

BLHeli pass-thru auto-enable (SERVO_BLH_AUTO)

Description

`SERVO_BLH_AUTO` is a "smart" version of `SERVO_BLH_MASK`. Instead of you having to manually calculate the bitmask for your motors, ArduPilot looks at your `SERVOx_FUNCTION` settings.

If a pin is set to **Motor 1, Motor 2, etc.**, ArduPilot will automatically enable DShot and BLHeli pass-thru on those pins.

The Engineer's View

If set to **1**, the initialization logic in `AP_BLHeli` automatically adds all outputs with functions 33 through 40 (standard motors) and 82 through 85 (extra motors) to the digital output mask.

Tuning & Behavior

- **Default Value:** 0
- **Recommendation:** Set to **1** for most modern drone builds. it simplifies configuration and ensures that if you move a motor to a different pin, the digital protocol follows the function mapping automatically.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





SERVO_BLH_BDMASK (BLHeli Bidirectional Mask)

DEFAULT: 0	RANGE: 0 4294967295

BLHeli Bidirectional Mask (SERVO_BLH_BDMASK)

Description

`SERVO_BLH_BDMASK` enables high-speed RPM feedback over the existing motor signal wires.

This is required for the "Harmonic Notch Filter" to track motor speed perfectly. Unlike standard telemetry, this updates thousands of times per second.

Tuning & Behavior

- **Default Value:** 0.
- **Recommendation:** Enable for all motors if your ESCs support BLHeli_32 or Bluejay.

SERVO_BLH_DEBUG (BLHeli Debug Enable)

DEFAULT: 0	RANGE: 0 1

BLHeli Debug Enable (SERVO_BLH_DEBUG)

Description

`SERVO_BLH_DEBUG` floods the log with ESC traffic.

Use this only if your ESCs are not connecting or telemetry is failing. It records the raw DShot or serial bytes exchanged between the flight controller and the ESCs.

Tuning & Behavior

- **0:** Disabled.
- **1:** Enabled.
- **Warning:** Can consume significant CPU and SD card bandwidth. Disable after troubleshooting.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





SERVO_BLH_MASK (BLHeli Channel Bitmask)

DEFAULT: 0	RANGE: 0 4294967295

BLHeli Channel Bitmask (SERVO_BLH_MASK)

Description

`SERVO_BLH_MASK` tells ArduPilot which physical output pins are connected to digital **BLHeli_S**, **BLHeli_32**, or **AM32** ESCs.

- **PWM (Default):** If an output is NOT in this mask (and `SERVO_BLH_AUTO` is 0), it sends a standard analog PWM signal.
- **DShot:** If an output IS in this mask, it sends a high-speed digital DShot signal (as defined by `MOT_PWM_TYPE`).

This mask is also required for **BLHeli Pass-Thru**, which allows you to configure your ESCs (like changing motor direction or updating firmware) directly through the flight controller using the BLHeliSuite software.

The Mathematics

The parameter is a 32-bit integer bitmask.

$$\text{Active Channels} = \sum 2^i \cdot b_i$$

- **Bit 0 (1):** Output 1
- **Bit 1 (2):** Output 2
- **Bit 3 (8):** Output 4
- **Example:** To enable DShot on Outputs 1, 2, 3, and 4: $1 + 2 + 4 + 8 = \mathbf{15}$.

The Engineer's View

Used in `AP_BLHeli::init()`.
Setting this bitmask triggers the underlying HAL to reconfigure the timer pins for that channel into DShot mode.
Note: On many boards, outputs are grouped into "Timer Groups." Enabling DShot on one pin in a group may force all other pins in that group to also use DShot or be disabled.

Tuning & Behavior

- **Default Value:** 0
- **Recommendation:**
  - **Manual Config:** Use the bitmask to select exactly which motors are digital.
  - **Automatic Config:** For standard quads, it is often easier to set `SERVO_BLH_AUTO = 1`, which automatically enables this mask for all outputs assigned as "Motors" (`SERVOx_FUNCTION` 33-40).
- **Reboot Required:** You MUST reboot after changing this mask for the hardware pins to reconfigure.

SERVO_BLH_OTYPE (BLHeli output type override)

DEFAULT: 0	RANGE: null	UNITS: null

BLHeli output type override

Legacy Note: `SERVO_*` parameters have been renamed to `OUT_*` in newer firmware versions. This parameter functions identically to OUT_BLH_OTYPE.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





SERVO_BLH_POLES (BLHeli Motor Poles)

DEFAULT: 14	RANGE: 2 100

BLHeli Motor Poles (SERVO_BLH_POLES)

Description

`SERVO_BLH_POLES` ensures the RPM displayed on your OSD is correct.

ESCs report "Electrical RPM" (how fast the field spins). To get "Mechanical RPM" (how fast the bell spins), you must divide by half the number of magnetic poles.

Tuning & Behavior

- **Default Value:** 14 (Standard for 22xx/23xx motors).
- **Procedure:** Count the magnets on the bell of your motor. Enter that number here.

SERVO_BLH_PORT (BLHeli Serial Port)

DEFAULT: 0	RANGE: 0 10

BLHeli Serial Port (SERVO_BLH_PORT)

Description

`SERVO_BLH_PORT` is for legacy BLHeli setups.

Modern BLHeli_32/S ESCs use the DShot signal wire for passthrough. However, if you are using an older serial-based ESC programmer connected to a UART, this parameter selects which serial port it is plugged into.

Tuning & Behavior

- **Default Value:** 0 (Disabled/USB).

SERVO_BLH_RVMASK (BLHeli bitmask of reversed channels)

DEFAULT: 0	RANGE: null	UNITS: null

BLHeli bitmask of reversed channels

Legacy Note: `SERVO_*` parameters have been renamed to `OUT_*` in newer firmware versions. This parameter functions identically to OUT_BLH_RVMASK.

SERVO_BLH_TEST (BLHeli Motor Test)

DEFAULT: 0	RANGE: 0 1

BLHeli Motor Test (SERVO_BLH_TEST)

Description

`SERVO_BLH_TEST` runs the DShot beacon.

Setting this to **1** sends a command to all ESCs to beep. This confirms that the DShot communication link is working, even if the motors aren't spinning.

Tuning & Behavior

- **0:** Disabled.
- **1:** Test (Beep).

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





SERVO_BLH_TMOUT (BLHeli Passthrough Timeout)

DEFAULT: 0	RANGE: 0 600	UNITS: s

BLHeli Passthrough Timeout (SERVO_BLH_TMOUT)

Description

`SERVO_BLH_TMOUT` is a safety feature for the BLHeliSuite bridge.

When you connect to BLHeliSuite, the autopilot enters a special "Passthrough" mode where it stops sending motor commands and lets the PC talk to the ESCs. This parameter ensures the autopilot exits this mode automatically if you disconnect or crash the PC app, preventing a locked-up flight controller.

Tuning & Behavior

- **Default Value:** 0 (Disabled/No timeout).

SERVO_BLH_TRATE (BLHeli Telemetry Rate)

DEFAULT: 10	RANGE: 0 100	UNITS: Hz

BLHeli Telemetry Rate (SERVO_BLH_TRATE)

Description

`SERVO_BLH_TRATE` controls how often the ESC reports Voltage, Current, RPM, and Temperature.

Tuning & Behavior

- **Default Value:** 10 Hz.
- **Recommendation:** Set to **10 Hz** for general monitoring. Higher rates are not needed unless you are doing detailed analysis, and they can clog the serial port.

SERVO_DSHOT_ESC (DShot ESC Type)

DEFAULT: 0	RANGE: 0 4

DShot ESC Type (SERVO_DSHOT_ESC)

Description

`SERVO_DSHOT_ESC` tells the autopilot exactly what kind of DShot-capable ESCs you have connected.

While standard DShot works without this parameter, enabling the specific type allows for advanced features like **EDT (Extended DShot Telemetry)**, which sends temperature, voltage, and current data back to the flight controller over the single signal wire.

- **0: None.** Standard DShot.
- **1: BLHeli32 / AM32.** Supports standard telemetry commands.
- **2: BLHeli_S.**
- **3: BLHeli32 EDT.** Enables Extended DShot Telemetry (if supported by firmware).

Tuning & Behavior

- **Default Value:** 0.
- **Recommendation:** Set to **1** for most modern BLHeli_32/AM32 ESCs.
- **Reboot Required:** Yes.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





SERVO_DSHOT_RATE (DShot Update Rate)

DEFAULT: 0	RANGE: 0 4

DShot Update Rate (SERVO_DSHOT_RATE)

Description

`SERVO_DSHOT_RATE` selects the speed of the digital motor protocol.

- **0:** Disabled (Use PWM/OneShot).
- **1:** DShot150 (150kHz). Slowest, most robust against noise.
- **2:** DShot300 (300kHz). Standard.
- **3:** DShot600 (600kHz). High performance.
- **4:** DShot1200 (1200kHz). Experimental, requires very short signal wires.

Tuning & Behavior

- **Default Value:** 0.
- **Recommendation:** Use **DShot600** (Value 3) for most modern quads.

SERVO_FN (Torqeedo Servo Output Function)

DEFAULT: 70	RANGE: 70 74

SERVO_FN: Torqeedo Servo Output Function

Description

Torqeedo Servo Output Function.

Values

Value	Meaning
70	Throttle
73	Throttle Left
74	Throttle Right

- **Default:** 70 (Throttle)

Description

This parameter is part of the **Torqeedo** interface (for electric boat motors).

- **Function:** It tells the Torqeedo backend which virtual "servo channel" output from the flight controller should be used to command the motor's power.
- **Normal Usage:** In a single-motor boat, this is set to **70 (Throttle)**.
- **Skid Steering:** In a dual-motor boat, one motor's backend might be set to **73 (Throttle Left)** and the other to **74 (Throttle Right)**.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





SERVO_FTW_MASK (FETtec OneWire Motor Mask)

DEFAULT: 0	RANGE: 0 65535

FETtec OneWire Motor Mask (SERVO_FTW_MASK)

Description

`SERVO_FTW_MASK` enables the high-speed digital **FETtec OneWire** protocol on specific output channels.

OneWire is a specialized ESC protocol (similar to DShot but using a different electrical standard) that allows for bidirectional communication (RPM, voltage, current) over a single signal wire. By setting a bit in this mask, you tell ArduPilot to use the OneWire driver for that specific motor.

- **Bit 0 (1):** Motor 1
- **Bit 1 (2):** Motor 2
- ...

Tuning & Behavior

- **Default Value:** 0 (Disabled).
- **Reboot Required:** Yes.
- **Requirement:** You must also set the `SERIALx_PROTOCOL` to **43** (FETtec OneWire) on the UART connected to the ESCs.

SERVO_FTW_POLES (FETtec OneWire Motor Poles)

DEFAULT: 14	RANGE: 2 48

FETtec OneWire Motor Poles (SERVO_FTW_POLES)

Description

`SERVO_FTW_POLES` ensures your RPM readings are accurate.

ESCs measure "Electrical RPM." To convert this to real "Mechanical RPM," ArduPilot needs to know how many magnets (poles) are inside the motor.

Tuning & Behavior

- **Default Value:** 14.
- **Common Values:** 14 for standard 2207/2306 drone motors. 28+ for larger industrial motors.

SERVO_FTW_RVMASK (FETtec OneWire Reverse Mask)

DEFAULT: 0	RANGE: 0 65535

FETtec OneWire Reverse Mask (SERVO_FTW_RVMASK)

Description

`SERVO_FTW_RVMASK` allows you to reverse your motor directions in software.

Instead of swapping two wires on your motor, you can set a bit in this mask to tell the FETtec ESC to spin the motor in the opposite direction.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





SERVO_GPIO_MASK (Servo GPIO Mask)

DEFAULT: 0	RANGE: 0 4294967295

Servo GPIO Mask (SERVO_GPIO_MASK)

Description

`SERVO_GPIO_MASK` converts PWM servo headers into digital switches.

By setting a bit in this mask, the corresponding servo output pin stops sending PWM pulses and becomes a standard GPIO pin. You can then use it with `RELAY_PIN` or Lua scripts to toggle lights, cameras, or other digital devices.

Tuning & Behavior

- **Default Value:** 0.
- **Bit 0:** Servo 1.
- **Bit 1:** Servo 2.
- **Reboot Required:** Yes.

SERVO_PITCH_TYPE (Pitch Servo Actuator Type)

DEFAULT: 0	RANGE: 0 1

Pitch Servo Actuator Type (SERVO_PITCH_TYPE)

Description

`SERVO_PITCH_TYPE` defines the control hardware used for an **Antenna Tracker's** vertical axis.

- **0: Position Servo.** Standard proportional RC servo. The autopilot commands a specific angle.
- **1: On/Off Actuator.** A simple motor or relay. The autopilot "pulses" the motor based on ONOFF_PITCH_RATE to reach the target angle.

Tuning & Behavior

- **Default:** 0.
- **Usage:** Only change to **1** if you are using a heavy-duty antenna tracker with industrial motor drivers that do not support standard PWM servo signals.

SERVO_RATE (Servo Output Rate)

DEFAULT: 50	RANGE: 50 400	UNITS: Hz

Servo Output Rate (SERVO_RATE)

Description

`SERVO_RATE` sets the "Frame Rate" for your servos.

- **50Hz (Default):** Standard for analog servos and fixed-wing aircraft.
- **400Hz:** High speed for digital servos and multicopter ESCs using PWM.

Warning: Sending 400Hz to an analog servo will cause it to overheat and burn out.

Tuning & Behavior

- **Default Value:** 50 Hz.
- **Recommendation:**
  - **Plane:** 50 Hz.
  - **Copter:** 400 Hz (if using PWM).
  - **DShot:** Ignored (DShot runs at its own protocol speed).

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





SERVO_RC_FS_MSK (Servo RC Failsafe Mask)

DEFAULT: 0	RANGE: 0 65535

Servo RC Failsafe Mask (SERVO_RC_FS_MSK)

Description

`SERVO_RC_FS_MSK` controls what happens to your "Passthrough" servos when you lose radio signal.

If you have a camera gimbal or a pan/tilt mechanism controlled directly by a knob on your transmitter (`RCx_OPTION = PassThru`), what should it do if the radio link dies?

- **Bit Unset (0):** The servo holds its last known position (Freeze).
- **Bit Set (1):** The servo moves to its Trim/Center position.

The Mathematics

$$\text{Output} = \begin{cases} \text{Trim} & \text{if Bit Set} \land \text{Failsafe Last Known} \\ & \text{otherwise} \end{cases}$$

Tuning & Behavior

- **Default Value:** 0 (Hold Last).
- **Recommendation:**
  - **Gimbals:** Leave at **0** (Hold) to keep the camera pointing at the target.
  - **Landing Gear:** Leave at **0** (Hold) so the gear doesn't accidentally retract/deploy.
  - **Safety Mechanisms:** Set bit to **1** if you want a mechanism to return to a safe "Neutral" state on signal loss.

SERVO_ROB_POSMAX (Robotis Servo Position Max)

DEFAULT: 4095	RANGE: 0 4095

Robotis Servo Position Max (SERVO_ROB_POSMAX)

Description

`SERVO_ROB_POSMAX` sets the upper travel limit for **Robotis** servos.

Tuning & Behavior

- **Default Value:** 4095.
- **Recommendation:** Reduce this if the servo hits a hard stop before reaching the full 360-degree (or configured) range.

SERVO_ROB_POSMIN (Robotis Servo Position Min)

DEFAULT: 0	RANGE: 0 4095

Robotis Servo Position Min (SERVO_ROB_POSMIN)

Description

`SERVO_ROB_POSMIN` sets the lower travel limit for **Robotis Dynamixel** smart servos.

Unlike standard PWM servos (which use 1000-2000µs), Robotis servos use a digital position value, typically 0 to 4095 for a 12-bit resolution.

Tuning & Behavior

- **Default Value:** 0.
- **Recommendation:** Set to the mechanical limit of your actuator to prevent binding.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





SERVO_SBUS_RATE (SBUS Output Rate)

DEFAULT: 50	RANGE: 50 333	UNITS: Hz

SBUS Output Rate (SERVO_SBUS_RATE)

Description

`SERVO_SBUS_RATE` controls the speed of the SBUS output stream (if enabled via `BRD_SBUS_OUT`).

Standard SBUS is 50Hz (analog servo compatible). High-speed SBUS is often 200Hz or 333Hz.

Tuning & Behavior

- **Default Value:** 50 Hz.
- **Recommendation:** Ensure your servos can handle high frame rates before increasing this.

SERVO_VOLZ_MASK (Volz Servo Mask)

DEFAULT: 0	RANGE: 0 4294967295

Volz Servo Mask (SERVO_VOLZ_MASK)

Description

`SERVO_VOLZ_MASK` allows you to control high-end industrial actuators from **Volz**.

These servos use a proprietary RS-485 serial protocol instead of PWM. By setting a bit in this mask, you tell ArduPilot to send position commands for that channel over the configured serial port (`SERIALx_PROTOCOL = 14` or similar) addressed to the specific actuator ID.

The Engineer's View

Defined in `AP_Volz_Protocol.cpp`.
The Volz driver runs a separate thread to handle the 115200 baud serial traffic.

- **Bit 0 (1):** Channel 1
- **Bit 1 (2):** Channel 2
- ...

The driver calculates the required CRC and sends the "Extended Position" command.

Tuning & Behavior

- **Default Value:** 0 (Disabled).
- **Recommendation:** Set bits corresponding to the servo channels you have wired to the RS-485 bus.
- **Prerequisite:** You must configure a Serial Port for the Volz protocol.

SERVO_VOLZ_RANGE (Volz Servo Travel Range)

DEFAULT: 200	RANGE: 40 200	UNITS: deg

Volz Servo Travel Range (SERVO_VOLZ_RANGE)

Description

`SERVO_VOLZ_RANGE` is a scaling parameter for high-end **Volz digital servos**.

These servos use a digital protocol but can be commanded via a virtual PWM signal from the autopilot. This parameter defines how many degrees of physical rotation correspond to the standard 1000-2000µs PWM range.

- **200 (Default):** Maps ± 100 degrees of rotation.
- **40:** Maps ± 20 degrees. This provides much higher "Resolution" (finer movements) if your control surface doesn't need full travel.

Tuning & Behavior

- **Default:** 200.
- **Significance:** If you decrease this value, the servo will move more slowly and precisely for the same amount of stick movement.
- **Requirement:** Only relevant if you are using the Volz protocol (`SERIALx_PROTOCOL` = 20) and have Volz actuators installed.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY

SERVO_YAW_TYPE (Yaw Servo Actuator Type)

DEFAULT: 0	RANGE: 0 1

Yaw Servo Actuator Type (SERVO_YAW_TYPE)

Description

`SERVO_YAW_TYPE` defines the control hardware used for an **Antenna Tracker's** horizontal (pan) axis.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **SF** parameter group (specifically `SF_JS_AXISx`) is a **SITL-only** tool used to map physical USB Gamepads or Joysticks to the simulator using the **SFML** library.

Key Concepts

1. Direct Control

When developers want to test flight modes using a real controller rather than keyboard commands or GCS sliders, they use the SFML backend to pipe the USB HID data directly into the simulated RC input stack.

- `SF_JS_AXIS1` .. `8`: Maps physical joystick axes (0, 1, 2...) to virtual RC channels (Throttle, Roll, etc.).

Developer Notes

- **Library:** `libraries/SITL`.
- **Requirement:** Requires the autopilot to be compiled with `SFML` support.

SF_JS_AXIS1 (SFML Joystick Axis 1)

DEFAULT: 0	RANGE: 0 8

SFML Joystick Axis 1 (SF_JS_AXIS1)

Description

`SF_JS_AXIS1` allows you to use a real USB Joystick or Gamepad to fly the simulator.

If you are using the SFML build of SITL, this parameter maps the first axis of your physical controller to the internal RC input system.

Tuning & Behavior

- **Default Value:** 0.
- **Usage:** Usually handled by the `-j` flag in `sim_vehicle.py`, but can be manually configured here.

SF_JS_AXIS2 (null)

DEFAULT: sf::Joystick::Axis::Y	RANGE: null	UNITS: null

None

Note: This parameter functions identically to SF_JS_AXIS1.

SF_JS_AXIS3 (null)

DEFAULT: sf::Joystick::Axis::Z	RANGE: null	UNITS: null

None

Note: This parameter functions identically to SF_JS_AXIS1.

SF_JS_AXIS4 (null)

DEFAULT: sf::Joystick::Axis::U	RANGE: null	UNITS: null

None

Note: This parameter functions identically to SF_JS_AXIS1.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





SF_JS_AXIS5 (null)

DEFAULT: sf::Joystick::Axis::V	RANGE: null	UNITS: null

None

Note: This parameter functions identically to SF_JS_AXIS1.

SF_JS_AXIS6 (null)

DEFAULT: sf::Joystick::Axis::R	RANGE: null	UNITS: null

None

Note: This parameter functions identically to SF_JS_AXIS1.

SF_JS_AXIS7 (null)

DEFAULT: sf::Joystick::Axis::PovX	RANGE: null	UNITS: null

None

Note: This parameter functions identically to SF_JS_AXIS1.

SF_JS_AXIS8 (null)

DEFAULT: sf::Joystick::Axis::PovY	RANGE: null	UNITS: null

None

Note: This parameter functions identically to SF_JS_AXIS1.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





SYSTEM IDENTIFICATION CONFIGURATION (SID)

Overview

The **SID** parameter group configures the **System Identification** flight mode. This mode is used by engineers to mathematically model the vehicle's dynamics by injecting specific frequency sweeps (chirps) or step inputs into the control loops.

The data gathered during a SysID flight is used to calculate precise inertia, motor time constants, and aerodynamic coefficients, which can then be used to tune the PID controllers to a "Gold Standard" level of performance.

Key Concepts

1. Excitation Signals

- `SID_AXIS`: Selects which axis to test (Roll, Pitch, Yaw, Throttle).
- `SID_MAGNITUDE`: The strength of the injected signal (e.g., deg/s for rate sweeps).
- `SID_F_START_HZ` / `STOP_HZ`: The frequency range of the chirp signal.

2. Time Constants

- `SID_T_REC`: Total duration of the test recording.
- `SID_T_FADE_IN / OUT`: Smoothly ramps the signal to prevent sudden jerks.

Parameter Breakdown

- `SID_AXIS`: Axis selector.
- `SID_MAGNITUDE`: Signal amplitude.
- `SID_XY_CTRL_MUL`: Safety multiplier to detune the position controller during the test.

Integration Guide

- **Preparation:** System ID should only be performed after a basic manual tune is established.
- **Safety:** Ensure you have plenty of altitude. The vehicle will twitch and vibrate aggressively during the sweep.
- **Analysis:** Use tools like **UAVVID** or Matlab/Octave to process the DataFlash logs and generate a Bode plot.

Developer Notes

- **Library:** `libraries/AC_SystemID`.
- **Mode:** Triggered by switching to `SYSTEM_ID` (Mode 25).

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





SID_AXIS (System identification axis)

DEFAULT: 0	RANGE: 0 19

SID_AXIS: System identification axis

Description

This parameter selects the specific control loop axis to be "excited" during a System Identification flight test.

Values

- **0:** None (Disabled)
- **1:** Input Roll Angle
- **2:** Input Pitch Angle
- **3:** Input Yaw Angle
- **4:** Recovery Roll Angle
- **5:** Recovery Pitch Angle
- **6:** Recovery Yaw Angle
- **7:** Rate Roll
- **8:** Rate Pitch
- **9:** Rate Yaw
- **10:** Mixer Roll
- **11:** Mixer Pitch
- **12:** Mixer Yaw
- **13:** Mixer Thrust
- **14:** Measured Lateral Position
- **15:** Measured Longitudinal Position
- **16:** Measured Lateral Velocity
- **17:** Measured Longitudinal Velocity
- **18:** Input Lateral Velocity
- **19:** Input Longitudinal Velocity

Description

System ID Mode is an advanced diagnostic tool used to measure the actual physical response of the aircraft to control inputs.

- **How it works:** When you switch into System ID mode, the autopilot injects a "chirp" (a sine wave of increasing frequency) into the control loop selected by `SID_AXIS`.
- **Purpose:** By logging the input chirp and the resulting motion, engineers can calculate the precise "Transfer Function" of the aircraft, which is used to create a perfect PID tune or a digital twin model.
- **Safety:** Always start with small magnitudes (`SID_MAGNITUDE`) and rates (`SID_AXIS = 7, 8, or 9`) before trying angle or position loops.

SID_F_START_HZ (System identification Start Frequency)

DEFAULT: 0.5	RANGE: 0.01 100	UNITS: Hz

SID_F_START_HZ: System identification Start Frequency

Description

Frequency at the start of the sweep.

Values

- **Range:** 0.01 to 100
- **Units:** Hz
- **Default:** 0.5

Description

This parameter defines the lowest frequency of the "Chirp" signal used during System ID.

- **Function:** The chirp starts at this frequency and slowly increases until it hits `SID_F_STOP_HZ`.
- **Purpose:** Low frequencies are used to identify the aircraft's rigid-body dynamics and long-term stability.
- **Tuning: 0.5 Hz** is a good starting point for most vehicles. For very large, slow aircraft, you might reduce this to **0.1 Hz**.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





SID_F_STOP_HZ (System identification Stop Frequency)

DEFAULT: 40	RANGE: 0.01 100	UNITS: Hz

SID_F_STOP_HZ: System identification Stop Frequency

Description

Frequency at the end of the sweep.

Values

- **Range:** 0.01 to 100
- **Units:** Hz
- **Default:** 40

Description

This parameter defines the highest frequency of the "Chirp" signal used during System ID.

- **Purpose:** High frequencies are used to identify the aircraft's fast dynamics, such as motor response, propeller inertia, and frame flexibility.
- **Tuning: 40 Hz** (default) or **50 Hz** is sufficient for most multicopters. For very large vehicles, you might stop at **20 Hz** to avoid exciting structural resonances.
- **Warning:** Do not set this higher than half of your `SCHED_LOOP_RATE` (Nyquist limit). For a 400Hz loop, stay below 100Hz.

SID_MAGNITUDE (System identification Chirp Magnitude)

DEFAULT: 15	RANGE: 1 50

SID_MAGNITUDE: System identification Chirp Magnitude

Description

This parameter sets the "Strength" of the excitation signal (the chirp).

Values

- **Units:** Degrees (for angle/rate loops) or 0.0-1.0 (for mixer loops).
- **Default:** 15

Description

`SID_MAGNITUDE` determines how much the drone will oscillate during the System ID test.

- **Effect:** A larger magnitude provides a cleaner data signal (higher signal-to-noise ratio), which is better for analysis. However, a magnitude that is too large can cause the drone to hit its mechanical limits or become unstable.
- **Tuning Tip:** Start with a low value (e.g., **5**) to ensure the drone is safe, then gradually increase until the motion is clearly visible but the drone remains well within its flight envelope.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





SID_T_FADE_IN (System identification Fade in time)

DEFAULT: 15	RANGE: 0 20	UNITS: s

SID_T_FADE_IN: System identification Fade in time

Description

Time to reach maximum amplitude of sweep.

Values

- **Range:** 0 to 20
- **Units:** s
- **Default:** 15

Description

This parameter defines the "Ramp-Up" duration at the beginning of the System ID test.

- **Function:** Instead of instantly jumping to full magnitude, the autopilot slowly increases the magnitude of the chirp signal over this time period.
- **Safety:** This gives the pilot time to assess whether the drone is handling the excitation correctly and to abort if the oscillations become too violent.
- **Data Quality:** A longer fade-in ensures that any "startup transients" in the physics don't corrupt the analysis of the steady-state frequency response.

SID_T_FADE_OUT (System identification Fade out time)

DEFAULT: 2	RANGE: 0 5	UNITS: s

SID_T_FADE_OUT: System identification Fade out time

Description

Time to reach zero amplitude at the end of the sweep.

Values

- **Range:** 0 to 5
- **Units:** s
- **Default:** 2

Description

This parameter defines the "Cool-Down" duration at the end of the System ID test.

- **Function:** After the chirp finishes its frequency sweep, the autopilot slowly reduces the magnitude to zero over this time period.
- **Purpose:** Prevents a sudden, jerky stop of the aircraft's motion, which could destabilize it or cause stress on the frame. It ensures a smooth transition back to the pilot's manual control or the original flight mode's behavior.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





SID_T_REC (System identification Total Sweep length)

DEFAULT: 70	RANGE: 0 255	UNITS: s

SID_T_REC: System identification Total Sweep length

Description

Time taken to complete the sweep.

Values

- **Range:** 0 to 255
- **Units:** s
- **Default:** 70

Description

This parameter defines the total duration of the System ID experiment (excluding fade-in and fade-out).

- **Function:** A longer sweep time means the autopilot spends more time at each frequency, resulting in higher resolution and more accurate data for the System ID algorithm.
- **Requirement:** You must have enough battery life and airspace to fly in one direction (or hover) for this duration while the test is running.
- **Tuning: 70 to 100 seconds** is usually sufficient for a high-quality model.
- **Note:** If the drone hits a fence or the pilot takes manual control, the test is aborted.

SID_XY_CTRL_MUL (System identification XY control multiplier)

DEFAULT: 0.1	RANGE: 0.05 1.0

System identification XY control multiplier (SID_XY_CTRL_MUL)

Description

This safety parameter detunes (softens) the horizontal position controller during VTOL System Identification maneuvers. When injecting chirp or step signals to model the vehicle dynamics, the position controller normally tries to fight these disturbances. Reducing its gain allows the test signal to propagate effectively while still providing enough position holding to keep the vehicle from drifting away entirely.

The Mathematics

$$K_\text{effective} = K_\text{standard} \times \text{SID_XY_CTRL_MUL}$$

The Engineer's View

Defined in `ArduPlane/systemid.cpp`.

- **Context:** Only active during the `SYSTEM_ID` flight mode when performing axis sweeps.

Tuning & Behavior

- **Default Value:** 0.1 (10% of normal gain)
- **Increase:** If the vehicle drifts dangerously fast during SysID.
- **Decrease:** If the SysID data is contaminated by the position controller fighting the test signal.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY

Overview

The **SIM** parameter group configures the **Software-In-The-Loop (SITL)** simulator environment. These parameters allow developers to simulate hardware failures, environmental conditions (wind/tides), and sensor noise without risking a physical vehicle.

Note: These parameters only exist when running the SITL binary; they are not present in physical flight controller hardware.

Key Concepts

1. Sensor Failures (`SIM_..._FAIL`)

Developers can simulate a sensor "dying" in mid-air to test failsafe logic.

- `SIM_GPS_DISABLE`: Turns off GPS signal.
- `SIM_ACCEL_FAIL`: Simulates an accelerometer failure.

2. Environmental Conditions

- `SIM_WIND_SPD` / `DIR`: Injects constant wind.
- `SIM_WIND_TURB`: Adds turbulence to the wind model.
- `SIM_WAVE_AMP`: (Rover/Boat) Simulates water waves affecting the hull.

3. Sensor Noise and Bias

Allows for adding realistic imperfections to the simulated sensors to test EKF resilience.

- `SIM_GYR_RND`: Random noise on the gyros.
- `SIM_MAG_ALY`: Simulated magnetic anomaly.

Parameter Breakdown

- `SIM_SPEEDUP`: Multiplier for simulation speed (e.g., 5.0 runs the sim at 5x real-time).
- `SIM_RATE_HZ`: Update frequency of the physics engine.
- `SIM_VICON_...`: Configures simulated Motion Capture (Vicon/Optitrack) indoor positioning.

Integration Guide

- **Safety Testing:** Use `SIM_GPS_JAM = 1` while in an autonomous mission to verify the vehicle correctly switches to non-GPS modes or triggers RTL.

Developer Notes

- **Library:** `libraries/SITL`.
- **Frontends:** SITL can be viewed in 3D using Gazebo, AirSim, or RealFlight.

SIM_ACC1_BIAS (Simulation Accelerometer 1 Bias)

DEFAULT: 0	RANGE: -10 10	UNITS: m/s^2

Simulation Accelerometer 1 Bias (SIM_ACC1_BIAS)

Description

`SIM_ACC1_BIAS` allows you to test how ArduPilot handles a "Drifty" sensor.

Real accelerometers always have a slight offset (e.g. they might report 0.1 m/s^2 even when sitting perfectly level). ArduPilot calibrates this out during the "Accel Cal" process. By setting this parameter in SITL, you can intentionally introduce an error to verify that your calibration process or the EKF can handle it.

Tuning & Behavior

- **Default Value:** 0.
- **Testing:** Set to **0.5**. If you haven't performed an Accel Cal in the simulation, the drone will think it is tilted and try to roll away.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





SIM_ACC1_RND (Simulated Accelerometer 1 Noise)

DEFAULT: 0	RANGE: 0 10.0	UNITS: m/s^2

Simulated Accelerometer 1 Noise (SIM_ACC1_RND)

Description

`SIM_ACC1_RND` allows you to test the robustness of the autopilot's navigation filters (EKF) by injecting artificial noise into the first simulated accelerometer.

In a real drone, motors and propellers create significant vibration. By increasing this parameter in SITL, you can see how much "shaking" your drone can tolerate before it loses its position estimate or becomes unstable.

The Engineer's View

Defined in `SITL.cpp`.
The simulation physics engine calculates the "True" acceleration of the vehicle. This parameter sets the standard deviation (σ) of a Gaussian distribution that is added to each sample:

$$Accel_read = Accel_true + \mathcal{N}(0, \text{SIM_ACC1_RND})$$

Tuning & Behavior

- **Default Value:** 0 (Perfect, noise-free sensor).
- **Recommendation:** Set to **0.5 or 1.0** to simulate a typical frame with some vibration. Set to **5.0+** to simulate a "Worst-case" high-vibration scenario.

SIM_ACC1_SCAL (Simulation Accelerometer 1 Scale)

DEFAULT: 0	RANGE: -0.1 0.1

Simulation Accelerometer 1 Scale (SIM_ACC1_SCAL)

Description

`SIM_ACC1_SCAL` simulates a "Sensitivity Error."

If an accelerometer is perfectly calibrated, it should report exactly 9.81 m/s^2 when facing down. A scale error means it might report 9.90 or 9.70 instead.

- **0 (Default):** Perfect scale.
- **0.05:** 5% scale error.

Tuning & Behavior

- **Default Value:** 0.
- **Testing:** Use to verify the effectiveness of the 6-point Accel Calibration routine.

SIM_ACC2_BIAS (Simulation Accelerometer 2 Bias)

DEFAULT: 0	RANGE: -10 10	UNITS: m/s^2

Simulation Accelerometer 2 Bias (SIM_ACC2_BIAS)

Description

`SIM_ACC2_BIAS` simulates drift on the secondary accelerometer.

See SIM_ACC1_BIAS for details.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





SIM_ACC2_RND (Accel 2 motor noise factor)

DEFAULT: 0	RANGE: null	UNITS: null

Accel 2 motor noise factor

Note: This parameter functions identically to SIM_ACC1_RND.

SIM_ACC2_SCAL (Simulation Accelerometer 2 Scale)

DEFAULT: 0	RANGE: -0.1 0.1

Simulation Accelerometer 2 Scale (SIM_ACC2_SCAL)

Description

`SIM_ACC2_SCAL` simulates sensitivity errors on the secondary accelerometer.

See SIM_ACC1_SCAL for details.

SIM_ACC3_BIAS (Simulation Accelerometer 3 Bias)

DEFAULT: 0	RANGE: -10 10	UNITS: m/s^2

Simulation Accelerometer 3 Bias (SIM_ACC3_BIAS)

Description

`SIM_ACC3_BIAS` simulates drift on the tertiary IMU.

See SIM_ACC1_BIAS for details.

SIM_ACC3_RND (Accel 3 motor noise factor)

DEFAULT: 0	RANGE: null	UNITS: null

Accel 3 motor noise factor

Note: This parameter functions identically to SIM_ACC1_RND.

SIM_ACC3_SCAL (Accel 3 scaling factor)

DEFAULT: 0	RANGE: null

Accel 3 scaling factor

Note: This parameter configures instance 3. It functions identically to SIM_ACC1_SCAL.

SIM_ACC4_BIAS (Accel 4 bias)

DEFAULT: 0	RANGE: null

Accel 4 bias

Note: This parameter configures instance 4. It functions identically to SIM_ACC1_BIAS.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





SIM_ACC4_RND (Accel 4 motor noise factor)

DEFAULT: 0	RANGE: null	UNITS: null

Accel 4 motor noise factor

Note: This parameter functions identically to SIM_ACC1_RND.

SIM_ACC4_SCAL (Accel 4 scaling factor)

DEFAULT: 0	RANGE: null

Accel 4 scaling factor

Note: This parameter configures instance 4. It functions identically to SIM_ACC1_SCAL.

SIM_ACC5_BIAS (Accel 5 bias)

DEFAULT: 0	RANGE: null

Accel 5 bias

Note: This parameter configures instance 5. It functions identically to SIM_ACC1_BIAS.

SIM_ACC5_RND (Accel 5 motor noise factor)

DEFAULT: 0	RANGE: null	UNITS: null

Accel 5 motor noise factor

Note: This parameter functions identically to SIM_ACC1_RND.

SIM_ACC5_SCAL (Accel 4 scaling factor)

DEFAULT: 0	RANGE: null

Accel 4 scaling factor

Note: This parameter configures instance 5. It functions identically to SIM_ACC1_SCAL.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY

SIM_ACCEL1_FAIL (Simulated Accelerometer 1 Failure)

DEFAULT: 0	RANGE: 0 1

Simulated Accelerometer 1 Failure (SIM_ACCEL1_FAIL)

Description

`SIM_ACCEL1_FAIL` allows you to simulate a catastrophic hardware failure of your primary accelerometer.

When set to **1**, the sensor stops providing data. This is used by developers to verify that the "Lane Switching" logic in EKF3 correctly identifies the dead sensor and automatically switches the drone's control to Accelerometer 2 or 3 without crashing.

The Engineer's View

When this flag is set, the SITL backend for the IMU returns `false` on its `read()` calls or provides static/zero data. This triggers the EKF health monitoring system to mark the lane as unhealthy.

Tuning & Behavior

- **Default Value:** 0 (Sensor working normally).
- **Recommendation:** Only set to **1** while in flight in a safe simulation to verify your drone's redundant systems are working as expected.

SIM_ACCEL2_FAIL (ACCEL2 Failure)

DEFAULT: 0	RANGE: null	UNITS: null

ACCEL2 Failure

Note: This parameter functions identically to SIM_ACCEL1_FAIL.

SIM_ACCEL3_FAIL (ACCEL3 Failure)

DEFAULT: 0	RANGE: null	UNITS: null

ACCEL3 Failure

Note: This parameter functions identically to SIM_ACCEL1_FAIL.

SIM_ACCEL4_FAIL (ACCEL4 Failure)

DEFAULT: 0	RANGE: null	UNITS: null

ACCEL4 Failure

Note: This parameter functions identically to SIM_ACCEL1_FAIL.

SIM_ACCEL5_FAIL (ACCEL5 Failure)

DEFAULT: 0	RANGE: null	UNITS: null

ACCEL5 Failure

Note: This parameter functions identically to SIM_ACCEL1_FAIL.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY

SIM_ACC_FAIL_MSK (Simulated Accelerometer Failure Mask)

DEFAULT: 0	RANGE: 0 7

Simulated Accelerometer Failure Mask (SIM_ACC_FAIL_MSK)

Description

`SIM_ACC_FAIL_MSK` allows you to "kill" specific accelerometers in the simulator.

ArduPilot typically runs 3 IMUs (Inertial Measurement Units). By disabling one or two, you can verify that the EKF (Extended Kalman Filter) correctly switches to the remaining healthy sensors.

- **Bit 0 (1):** Fail Accel 1
- **Bit 1 (2):** Fail Accel 2
- **Bit 2 (4):** Fail Accel 3

Tuning & Behavior

- **Default Value:** 0 (All sensors active).
- **Testing:** Set to **1** to kill Accel 1. Watch the `EKF3.Lane` messages in the GCS console to see the switch occur.

SIM_ACC_FILE_RW (Simulated Accelerometer File Read/Write)

DEFAULT: 0	RANGE: 0 3

Simulated Accelerometer File Read/Write (SIM_ACC_FILE_RW)

Description

`SIM_ACC_FILE_RW` allows you to "record" the vibration and movement of a real flight and "replay" it in the simulator.

This is an incredibly powerful tool for tuning filters. You can fly your real drone, capture the raw accelerometer noise, and then feed that exact noise into the simulator to test if your notch filters or EKF settings can handle it.

- **0:** Disabled (Standard simulation).
- **1:** Read data from file (Replay).
- **2:** Write data to file (Record).

Tuning & Behavior

- **Default Value:** 0.
- **Recommendation:** Use **2** during a short hover to capture a noise profile, then **1** to replay it while tuning `INS_HNTCH_FREQ`.

SIM_ACC_TRIM (Simulation Accelerometer Trim)

DEFAULT: 0	RANGE: -5 5	UNITS: deg

Simulation Accelerometer Trim (SIM_ACC_TRIM)

Description

`SIM_ACC_TRIM` simulates a drone where the flight controller isn't mounted perfectly level.

It injects a constant tilt into the accelerometer data. This is used to test how robust the EKF and flight controllers are to slight misalignments and to verify that "Accel Calibration" or "AHRS Trim" procedures work correctly in SITL.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





SIM_ADSB_ALT (Simulated ADSB Altitude)

DEFAULT: 1000	RANGE: 0 10000	UNITS: m

Simulated ADSB Altitude (SIM_ADSB_ALT)

Description

`SIM_ADSB_ALT` sets the cruising altitude for the fake traffic generated by `SIM_ADSB_COUNT`.

Tuning & Behavior

- **Default Value:** 1000 m.
- **Testing:** Set this to your flying altitude (e.g., **50m**) to force "Near Miss" events.

SIM_ADSB_COUNT (Simulated ADSB Aircraft Count)

DEFAULT: -1	RANGE: -1 100

Simulated ADSB Aircraft Count (SIM_ADSB_COUNT)

Description

`SIM_ADSB_COUNT` populates the sky with fake airplanes.

This allows you to test the "ADSB Avoidance" logic (`AVOID_ADSB`) without needing a real ADSB receiver or real planes flying nearby.

- **-1:** Disabled.
- **0:** Enabled but 0 planes.
- **1+:** Generates X random aircraft flying near your home location.

Tuning & Behavior

- **Default Value:** -1 (Disabled).
- **Testing:** Set to **5** to see random traffic on your GCS map. Fly towards them to verify your drone triggers a failsafe or avoidance maneuver.

SIM_ADSB_RADIUS (Simulated ADSB Radius)

DEFAULT: 10000	RANGE: 1000 100000	UNITS: m

Simulated ADSB Radius (SIM_ADSB_RADIUS)

Description

`SIM_ADSB_RADIUS` sets the size of the "arena" for the simulated air traffic.

Simulated planes will spawn and fly within this distance from your home point.

Tuning & Behavior

- **Default Value:** 10000 m (10 km).
- **Recommendation:** Lower this to **1000 or 2000** if you want the traffic to be denser and closer to your drone for easier testing.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





SIM_ADSB_TX (Simulated ADSB Transmission)

DEFAULT: 0	RANGE: 0 1

Simulated ADSB Transmission (SIM_ADSB_TX)

Description

`SIM_ADSB_TX` simulates the presence of an ADSB "Out" unit on your drone.

When enabled, the simulator will act as if it is broadcasting your position to the world. This is primarily for checking that the `ADSB_RF_SELECT` and other transponder settings are configured correctly in the GCS.

Tuning & Behavior

- **Default Value:** 0 (Disabled).
- **1:** Enabled.

SIM_ADSB_TYPES (Simulated ADSB Emitter Types)

DEFAULT: 0	RANGE: 0 4294967295

Simulated ADSB Emitter Types (SIM_ADSB_TYPES)

Description

`SIM_ADSB_TYPES` filters the kind of "Fake Planes" the simulator generates.

If you only want to test how your drone reacts to other drones (UAVs), you can disable the "747 Jumbo Jet" simulation.

- **Bit 0:** Light (Cessna)
- **Bit 1:** Small (Learjet)
- **Bit 2:** Large (747)
- **Bit 14:** UAV

Tuning & Behavior

- **Default Value:** 0 (All types).
- **Recommendation:** Leave at **0** to test against a realistic mix of air traffic.

SIM_AIS_RADIUS (Simulation AIS Radius)

DEFAULT: 0	RANGE: 0 100000	UNITS: m

Simulation AIS Radius (SIM_AIS_RADIUS)

Description

`SIM_AIS_RADIUS` enables the simulation of other ships for maritime operations.

When set to a positive value, ArduPilot will generate "Fake" ships (AIS targets) within this distance. This allows you to test obstacle avoidance and traffic monitoring systems on boats and autonomous ships.

SIM_ARSPD2_FAIL (Airspeed sensor failure)

DEFAULT: 0	RANGE: null	UNITS: null

Airspeed sensor failure

Note: This parameter functions identically to SIM_ARSPD_FAIL.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





SIM_ARSPD2_FAILP (Airspeed sensor failure pressure)

DEFAULT: 0	RANGE: null	UNITS: Pa

Airspeed sensor failure pressure

Note: This parameter functions identically to SIM_ARSPD_FAILP.

SIM_ARSPD2_OFS (Compass offsets in milligauss on the Z axis)

DEFAULT: 0	RANGE: -400 400	UNITS: mGauss

Compass offsets in milligauss on the Z axis

Note: This parameter functions identically to SIM_ARSPD_OFS.

SIM_ARSPD2_PITOT (Airspeed pitot tube failure pressure)

DEFAULT: 0	RANGE: null	UNITS: Pa

Airspeed pitot tube failure pressure

Note: This parameter functions identically to SIM_ARSPD_PITOT.

SIM_ARSPD2_RATIO (Airspeed ratio)

DEFAULT: 2	RANGE: null	UNITS: null

Airspeed ratio

Note: This parameter functions identically to SIM_ARSPD_RATIO.

SIM_ARSPD2_RND (null)

DEFAULT: 2.0	RANGE: null	UNITS: null

None

Note: This parameter functions identically to SIM_ARSPD_RND.

SIM_ARSPD2_SIGN (Airspeed signflip)

DEFAULT: 0	RANGE: null	UNITS: null

Airspeed signflip

Note: This parameter functions identically to SIM_ARSPD_SIGN.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





SIM_ARSPD_FAIL (Simulated Airspeed Failure Value)

DEFAULT: 0	RANGE: 0 100	UNITS: m/s

Simulated Airspeed Failure Value (SIM_ARSPD_FAIL)

Description

`SIM_ARSPD_FAIL` allows you to simulate a "Stuck Sensor" or a "Clogged Pitot Tube."

By setting this to a static value (like 0 or 100), you can verify that the autopilot detects the failure and switches to "Synthetic Airspeed" (estimating speed from throttle and pitch) or triggers a failsafe.

The Engineer's View

If this value is non-zero (or if you toggle the failure flag in the SITL backend), the sensor driver will lock its output to this value.

- **0 m/s:** Simulates a pitot tube cover left on.
- **High Value:** Simulates a sensor blowing out or water ingress.

Tuning & Behavior

- **Default Value:** 0 (Disabled/Normal Operation).
- **Testing:** Set to **0** while flying to simulate a blockage. Watch if the EKF declares the sensor "Unhealthy."

SIM_ARSPD_FAILP (Simulated Airspeed Failure Pressure)

DEFAULT: 0	RANGE: 0 1000	UNITS: Pa

Simulated Airspeed Failure Pressure (SIM_ARSPD_FAILP)

Description

`SIM_ARSPD_FAILP` allows you to choose what "Broken" looks like for your Pitot tube.

If `SIM_ARSPD_FAIL` is active, the sensor will freeze at this pressure.

- **0 Pa:** Simulates a clogged tube (Zero airspeed). Useful for testing stall warnings.
- **High Value:** Simulates a blocked static port (High airspeed).

Tuning & Behavior

- **Default Value:** 0 Pa.

SIM_ARSPD_OFS (Simulated Airspeed Offset)

DEFAULT: 0	RANGE: 0 100	UNITS: Pa

Simulated Airspeed Offset (SIM_ARSPD_OFS)

Description

`SIM_ARSPD_OFS` simulates a calibration error.

Airspeed sensors measure minute pressure differences. If the sensor isn't zeroed correctly before flight (e.g., if it was windy when you plugged in the battery), it will report speed even when sitting still. This parameter allows you to simulate that error and test if your pre-flight checks catch it.

Tuning & Behavior

- **Default Value:** 0 Pa.
- **Testing:** Set to **100**. You should see the HUD report an airspeed of 5-10 m/s even while the drone is on the ground.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY

SIM_ARSPD_PITOT (Simulated Pitot Blockage Pressure)

DEFAULT: 0	RANGE: 0 1000	UNITS: Pa

Simulated Pitot Blockage Pressure (SIM_ARSPD_PITOT)

Description

`SIM_ARSPD_PITOT` simulates a blocked pitot tube (e.g., mud wasp nest).

If you set `SIM_ARSPD_FAIL` to enable failure, this is the pressure value the sensor will output.

Tuning & Behavior

- **Default Value:** 0 Pa.

SIM_ARSPD_RATIO (Simulated Airspeed Ratio)

DEFAULT: 2	RANGE: 1 4

Simulated Airspeed Ratio (SIM_ARSPD_RATIO)

Description

`SIM_ARSPD_RATIO` sets the sensitivity of the virtual pitot tube.

This corresponds to the `ARSPD_RATIO` parameter in the flight controller. If they don't match, your airspeed reading will be wrong.

Tuning & Behavior

- **Default Value:** 2.
- **Testing:** Change this to **3** and see if `ARSPD_AUTOCAL` can figure it out.

SIM_ARSPD_RND (Simulated Airspeed Noise)

DEFAULT: 2	RANGE: 0 20	UNITS: m/s

Simulated Airspeed Noise (SIM_ARSPD_RND)

Description

`SIM_ARSPD_RND` injects noise into the virtual pitot tube.

Airspeed sensors are notoriously noisy due to wind gusts and turbulence. This parameter helps you tune your TECS (Total Energy Control System) to ignore that noise.

- **0:** Perfect, clean airspeed data.
- **2 (Default):** Realistic turbulence.

Tuning & Behavior

- **Default Value:** 2.0 m/s.
- **Recommendation:** Keep at **2.0** to ensure your PID tuning is robust enough for real-world flight.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





SIM_ARSPD_SIGN (Simulated Airspeed Sign)

DEFAULT: 1	RANGE: -1 1

Simulated Airspeed Sign (SIM_ARSPD_SIGN)

Description

`SIM_ARSPD_SIGN` simulates a common plumbing error: swapping the Pitot and Static tubes.

If set to **-1**, the airspeed sensor will report negative pressure, which typically results in zero airspeed or an error message.

Tuning & Behavior

- **Default Value:** 1.

SIM_BAR2_DELAY (Pivot Delay)

DEFAULT: AR_PIVOT_DELAY_DEFAULT	RANGE: 0 60	UNITS: s

Pivot Delay

Note: This parameter functions identically to SIM_BARO_DELAY.

SIM_BAR2_DISABLE (Barometer disable)

DEFAULT: 0	RANGE: null	UNITS: null

Barometer disable

Note: This parameter functions identically to SIM_BARO_DISABLE.

SIM_BAR2_DRIFT (Barometer altitude drift)

DEFAULT: 0	RANGE: null	UNITS: m/s

Barometer altitude drift

Note: This parameter functions identically to SIM_BARO_DRIFT.

SIM_BAR2_FREEZE (Barometer freeze)

DEFAULT: 0	RANGE: null	UNITS: null

Barometer freeze

Note: This parameter functions identically to SIM_BARO_FREEZE.

SIM_BAR2_GLITCH (Barometer glitch)

DEFAULT: 0	RANGE: null	UNITS: m

Barometer glitch

Note: This parameter functions identically to SIM_BARO_GLITCH.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





SIM_BAR2_RND (null)

DEFAULT: 2.0	RANGE: null	UNITS: null

None

Note: This parameter functions identically to SIM_BARO_RND.

SIM_BAR2_WCF_BAK (Wind coefficient backward)

DEFAULT: 0.0	RANGE: null	UNITS: null

Wind coefficient backward

Note: This parameter functions identically to SIM_BARO_WCF_BAK.

SIM_BAR2_WCF_DN (Pressure error coefficient in negative Z direction (down))

DEFAULT: 0.0	RANGE: -1.0 1.0	UNITS: null

Pressure error coefficient in negative Z direction (down)

Note: This parameter functions identically to SIM_BARO_WCF_DN.

SIM_BAR2_WCF_FWD (Pressure error coefficient in positive X direction (forward))

DEFAULT: 0.0	RANGE: -1.0 1.0	UNITS: null

Pressure error coefficient in positive X direction (forward)

Note: This parameter functions identically to SIM_BARO_WCF_FWD.

SIM_BAR2_WCF_LFT (Pressure error coefficient in negative Y direction (left))

DEFAULT: 0.0	RANGE: -1.0 1.0	UNITS: null

Pressure error coefficient in negative Y direction (left)

Note: This parameter functions identically to SIM_BARO_WCF_LFT.

SIM_BAR2_WCF_RGT (Pressure error coefficient in positive Y direction (right))

DEFAULT: 0.0	RANGE: -1.0 1.0	UNITS: null

Pressure error coefficient in positive Y direction (right)

Note: This parameter functions identically to SIM_BARO_WCF_RGT.

SIM_BAR2_WCF_UP (Pressure error coefficient in positive Z direction (up))

DEFAULT: 0.0	RANGE: -1.0 1.0	UNITS: null

Pressure error coefficient in positive Z direction (up)

Note: This parameter functions identically to SIM_BARO_WCF_UP.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





SIM_BAR3_DELAY (Pivot Delay)

DEFAULT: AR_PIVOT_DELAY_DEFAULT	RANGE: 0 60	UNITS: s

Pivot Delay

Note: This parameter functions identically to SIM_BARO_DELAY.

SIM_BAR3_DISABLE (Barometer disable)

DEFAULT: 0	RANGE: null	UNITS: null

Barometer disable

Note: This parameter functions identically to SIM_BARO_DISABLE.

SIM_BAR3_DRIFT (Barometer altitude drift)

DEFAULT: 0	RANGE: null	UNITS: m/s

Barometer altitude drift

Note: This parameter functions identically to SIM_BARO_DRIFT.

SIM_BAR3_FREEZE (Barometer freeze)

DEFAULT: 0	RANGE: null	UNITS: null

Barometer freeze

Note: This parameter functions identically to SIM_BARO_FREEZE.

SIM_BAR3_GLITCH (Barometer glitch)

DEFAULT: 0	RANGE: null	UNITS: m

Barometer glitch

Note: This parameter functions identically to SIM_BARO_GLITCH.

SIM_BAR3_RND (null)

DEFAULT: 2.0	RANGE: null	UNITS: null

None

Note: This parameter functions identically to SIM_BARO_RND.

SIM_BAR3_WCF_BAK (Wind coefficient backward)

DEFAULT: 0.0	RANGE: null	UNITS: null

Wind coefficient backward

Note: This parameter functions identically to SIM_BARO_WCF_BAK.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





SIM_BAR3_WCF_DN (Pressure error coefficient in negative Z direction (down))

DEFAULT: 0.0	RANGE: -1.0 1.0	UNITS: null

Pressure error coefficient in negative Z direction (down)

Note: This parameter functions identically to SIM_BARO_WCF_DN.

SIM_BAR3_WCF_FWD (Pressure error coefficient in positive X direction (forward))

DEFAULT: 0.0	RANGE: -1.0 1.0	UNITS: null

Pressure error coefficient in positive X direction (forward)

Note: This parameter functions identically to SIM_BARO_WCF_FWD.

SIM_BAR3_WCF_LFT (Pressure error coefficient in negative Y direction (left))

DEFAULT: 0.0	RANGE: -1.0 1.0	UNITS: null

Pressure error coefficient in negative Y direction (left)

Note: This parameter functions identically to SIM_BARO_WCF_LFT.

SIM_BAR3_WCF_RGT (Pressure error coefficient in positive Y direction (right))

DEFAULT: 0.0	RANGE: -1.0 1.0	UNITS: null

Pressure error coefficient in positive Y direction (right)

Note: This parameter functions identically to SIM_BARO_WCF_RGT.

SIM_BAR3_WCF_UP (Pressure error coefficient in positive Z direction (up))

DEFAULT: 0.0	RANGE: -1.0 1.0	UNITS: null

Pressure error coefficient in positive Z direction (up)

Note: This parameter functions identically to SIM_BARO_WCF_UP.

SIM_BARO_COUNT (Simulated Barometer Count)

DEFAULT: 1	RANGE: 1 3

Simulated Barometer Count (SIM_BARO_COUNT)

Description

`SIM_BARO_COUNT` defines how many virtual atmospheric pressure sensors are available to the autopilot.

By increasing this to **2 or 3**, you can test ArduPilot's multi-baro redundancy and health monitoring. You can then use other `SIM_BARx_...` parameters to fail one sensor and watch the EKF switch to the backup.

Tuning & Behavior

- **Default Value:** 1.
- **Recommendation:** Set to **2** for a realistic simulation of standard flight controller hardware (which often has two baros).
- **Reboot Required:** Yes.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





SIM_BARO_DELAY (Simulated Barometer Delay)

DEFAULT: 0	RANGE: 0 1000	UNITS: ms

Simulated Barometer Delay (SIM_BARO_DELAY)

Description

`SIM_BARO_DELAY` adds lag to the altitude reading. High lag can cause the drone to "bounce" in AltHold because the controller is reacting to where the drone *was*, not where it *is*.

Tuning & Behavior

- **Default Value:** 0
- **Recommendation:** Use to test controller stability.

SIM_BARO_DISABLE (Disable Simulated Barometer)

DEFAULT: 0	RANGE: 0 1

Disable Simulated Barometer (SIM_BARO_DISABLE)

Description

`SIM_BARO_DISABLE` kills the primary barometer signal.

Use this to test your drone's ability to maintain altitude using ONLY the IMU (which will drift) or a secondary source (like a Rangefinder).

Tuning & Behavior

- **Default Value:** 0
- **Recommendation:** Use in flight to verify your "Secondary Altitude Source" logic.

SIM_BARO_DRIFT (Simulated Barometer Drift)

DEFAULT: 0	RANGE: 0 1.0	UNITS: m/s

Simulated Barometer Drift (SIM_BARO_DRIFT)

Description

`SIM_BARO_DRIFT` simulates a barometer that is slowly losing calibration or being affected by changing weather pressure.

Real barometers drift as the weather changes (e.g., a storm front moving in). This parameter adds a constant vertical velocity error to the sensor.

Tuning & Behavior

- **Default Value:** 0 m/s.
- **Recommendation:** Set to **0.1 m/s** to test if the EKF can use GPS vertical velocity to correct the baro drift.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





SIM_BARO_FREEZE (Simulated Barometer Freeze)

DEFAULT: 0	RANGE: 0 1

Simulated Barometer Freeze (SIM_BARO_FREEZE)

Description

`SIM_BARO_FREEZE` locks the barometer value.

This simulates a sensor driver crash or a clogged static port. If the drone climbs or descends, the barometer will report 0 vertical speed.

Tuning & Behavior

- **Default Value:** 0 (Disabled).
- **1:** Freeze enabled.

SIM_BARO_GLITCH (Simulated Barometer Glitch)

DEFAULT: 0	RANGE: 0 100	UNITS: m

Simulated Barometer Glitch (SIM_BARO_GLITCH)

Description

`SIM_BARO_GLITCH` applies an instant step-change to the altitude reading.

This simulates events like opening a window in a pressurized room or a sudden pressure wave from a passing truck.

Tuning & Behavior

- **Default Value:** 0 m.
- **Testing:** Set to **50** (meters) while hovering. Verify that the drone does NOT shoot up/down 50 meters instantly. The EKF should reject the jump or smooth it out.

SIM_BARO_RND (Simulated Barometer Noise)

DEFAULT: 0.2	RANGE: 0 10.0	UNITS: Pa

Simulated Barometer Noise (SIM_BARO_RND)

Description

`SIM_BARO_RND` simulates atmospheric pressure noise. Since ArduPilot uses the barometer for altitude, this noise directly affects how "bouncy" the altitude hold is in flight.

The Engineer's View

Adds Gaussian noise to the simulated pressure reading. Because 1 Pascal is roughly 8cm of altitude at sea level, a noise value of 0.2 represents about 1.6cm of altitude jitter.

Tuning & Behavior

- **Default Value:** 0.2 Pa
- **Effect of Increasing:** The drone will struggle to maintain a precise altitude, and the climb rate estimate will become noisier.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY

SIM_BARO_WCF_BAK (Simulation Barometer Wind Coefficient Backward)

DEFAULT: 0	RANGE: -0.1 0.1

Simulation Barometer Wind Coefficient Backward (SIM_BARO_WCF_BAK)

Description

`SIM_BARO_WCF_BAK` simulates pressure errors when the drone is flying backwards.

See SIM_BARO_WCF_FWD for more details.

SIM_BARO_WCF_DN (Simulation Baro Wind Coeff (Down))

DEFAULT: 0	RANGE: -1.0 1.0

Simulation Baro Wind Coeff (SIM_BARO_WCF_DN)

Description

`SIM_BARO_WCF_DN` simulates aerodynamic interference that makes the altitude reading glitch during a fast descent.

See BARO1_WCF_ENABLE for details on the real-world compensation model. This parameter allows you to inject those same errors into SITL to test your compensation tuning.

SIM_BARO_WCF_FWD (Simulation Barometer Wind Coefficient Forward)

DEFAULT: 0	RANGE: -0.1 0.1

Simulation Barometer Wind Coefficient Forward (SIM_BARO_WCF_FWD)

Description

`SIM_BARO_WCF_FWD` simulates "Static Port Error."

On real planes, as the airspeed increases, the pressure around the fuselage drops (Bernoulli's principle). If your barometer isn't perfectly vented, this can cause the drone to think it is climbing when it is actually just speeding up. This parameter allows you to simulate that effect to test compensation algorithms (`BARO_WCF_ENABLE`).

The Mathematics

The pressure error (ΔP) is calculated as:

$$\Delta P = \text{WCF} \cdot \frac{1}{2} \cdot \rho \cdot V^2$$

Where:

- V is airspeed in the forward direction.
- ρ is air density.

Tuning & Behavior

- **Default Value:** 0.
- **Testing:** Set to **0.02**. Fly the drone fast in a straight line. If you see the reported altitude change while you are at a constant height, the simulation is working.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY

SIM_BARO_WCF_LFT (Simulated Baro Wind Comp Factor Left)

DEFAULT: 0	RANGE: -10 10	UNITS: Pa/(m/s)^2

Simulated Baro Wind Comp Factor Left (SIM_BARO_WCF_LFT)

Description

Defines pressure change due to wind from the left. See `SIM_BARO_WCF_UP`.

SIM_BARO_WCF_RGT (Simulated Baro Wind Comp Factor Right)

DEFAULT: 0	RANGE: -10 10	UNITS: Pa/(m/s)^2

Simulated Baro Wind Comp Factor Right (SIM_BARO_WCF_RGT)

Description

Defines pressure change due to wind from the right. See `SIM_BARO_WCF_UP`.

SIM_BARO_WCF_UP (Simulated Baro Wind Comp Factor Up)

DEFAULT: 0	RANGE: -10 10	UNITS: Pa/(m/s)^2

Simulated Baro Wind Comp Factor Up (SIM_BARO_WCF_UP)

Description

`SIM_BARO_WCF_UP` simulates the "Bernoulli Effect" on your static port.

When wind blows across a hole (static port), it creates a pressure drop (or rise), causing the drone to think it is at a different altitude. This parameter defines how much the reported pressure changes when wind hits the drone from the top.

The Mathematics

$$\Delta P = C_up \times V_up^2$$

Tuning & Behavior

- **Default Value:** 0.
- **Recommendation:** Use **-0.05** to simulate a typical drone where propwash creates a low-pressure zone above the flight controller.

SIM_BATT_CAPACITY (Simulation Battery Capacity)

DEFAULT: 10000	RANGE: 0 50000	UNITS: mAh

Simulation Battery Capacity (SIM_BATT_CAPACITY)

Description

`SIM_BATT_CAPACITY` defines the size of the virtual fuel tank.

SITL calculates current draw based on motor thrust. This parameter allows you to test battery failsafe logic by simulating a realistic discharge curve.

Tuning & Behavior

- **Default Value:** 10000 mAh.
- **Testing:** Set to a small value (e.g. 500) to quickly trigger a "Low Battery" failsafe during a test flight.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





SIM_BATT_CAP_AH (Simulated Battery Capacity)

DEFAULT: 0	RANGE: 0 1000	UNITS: Ah

Simulated Battery Capacity (SIM_BATT_CAP_AH)

Description

`SIM_BATT_CAP_AH` defines how much "Fuel" the virtual battery has.

As you fly in SITL, the autopilot calculates current draw based on motor throttle. This parameter determines how fast the voltage drops. If set to 0, the battery is "Infinite" and will never drain.

SIM_BATT_VOLTAGE (Simulation Battery Voltage)

DEFAULT: 12.6	RANGE: 0 100	UNITS: V

Simulation Battery Voltage (SIM_BATT_VOLTAGE)

Description

`SIM_BATT_VOLTAGE` sets the initial charge state.

Tuning & Behavior

- **Default Value:** 12.6 V (3S Lipo).
- **Testing:** Set to **10.5** to test Battery Failsafe trigger on boot or takeoff.

SIM_BAUDLIMIT_EN (Simulated Baud Rate Limiting Enable)

DEFAULT: 0	RANGE: 0 1

Simulated Baud Rate Limiting Enable (SIM_BAUDLIMIT_EN)

Description

`SIM_BAUDLIMIT_EN` forces the simulator to respect the physics of serial data transfer speeds.

By default, SITL sends data as fast as the CPU allows. If you enable this, a simulated 57600 baud link will actually choke if you try to send too much data, causing packet loss and latency.

Tuning & Behavior

- **Default Value:** 0 (Unlimited speed).
- **Recommendation:** Enable this when testing telemetry radios or low-bandwidth links to verify your stream rates (`SRx_params`) are configured correctly.

SIM_BUOYANCY (Simulation Buoyancy)

DEFAULT: 1	RANGE: -100 100	UNITS: N

Simulation Buoyancy (SIM_BUOYANCY)

Description

`SIM_BUOYANCY` defines how much the simulated submarine wants to float or sink.

- **Positive Value:** Net upwards force (Positive buoyancy). The sub floats to the surface.
- **Negative Value:** Net downwards force (Negative buoyancy). The sub sinks to the bottom.
- **0:** Neutral buoyancy. The sub stays at its current depth (Ideal for ROVs).

Tuning & Behavior

- **Testing:** Essential for tuning the vertical depth controller on ArduSub.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





SIM_BZ_ENABLE (Simulation Buzzer Enable)

DEFAULT: 0	RANGE: 0 1

Simulation Buzzer Enable (SIM_BZ_ENABLE)

Description

`SIM_BZ_ENABLE` activates a virtual beeper.

When enabled, SITL will log and potentially play sound for arming beeps, failsafe warnings, and other audio cues produced by the autopilot. Useful for debugging new melody patterns or verifying failsafe alerts.

SIM_BZ_PIN (Simulation Buzzer Pin)

DEFAULT: -1	RANGE: -1 100

Simulation Buzzer Pin (SIM_BZ_PIN)

Description

`SIM_BZ_PIN` tells the simulator which virtual pin is connected to the beeper.

SIM_CAN_SRV_MSK (Simulated CAN Servo Mask)

DEFAULT: 0	RANGE: 0 4294967295

Simulated CAN Servo Mask (SIM_CAN_SRV_MSK)

Description

`SIM_CAN_SRV_MSK` tells the simulator which motors or servos are "Digital" (CAN).

Normally, SITL simulates servos as direct wires. If you are developing or testing DroneCAN hardware (using **AP_Periph**), you can set this mask. Any bit set to **1** will cause the simulator to expect that channel's data to come from a virtual CAN node rather than the main flight controller's internal logic.

Tuning & Behavior

- **0:** All servos simulated as direct internal connections.
- **Bit 0 (1):** Servo 1 is a CAN servo.
- **Bit 3 (8):** Servo 4 is a CAN servo.

SIM_CAN_TYPE1 (Simulated CAN 1 Transport Type)

DEFAULT: 1	RANGE: 0 2

Simulated CAN 1 Transport Type (SIM_CAN_TYPE1)

Description

`SIM_CAN_TYPE1` tells the simulator how to send "CAN Messages" between SITL and other simulated devices (like a DroneCAN GPS).

- **0: None.**
- **1: MulticastUDP (Default).** Sends messages over your computer's local network. This is the standard way SITL talks to the "CANDevices" or "UAVCAN GUI Tool."
- **2: SocketCAN.** Uses the Linux kernel's native CAN bus driver (Linux only).

Tuning & Behavior

- **Default Value:** 1.
- **Recommendation:** Leave at **1** for all standard DroneCAN testing.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





SIM_CAN_TYPE2 (transport type for second CAN interface)

DEFAULT: uint8_t(CANTransport::MulticastUDP	RANGE: null	UNITS: null

transport type for second CAN interface

Note: This parameter functions identically to SIM_CAN_TYPE1.

SIM_CLAMP_CH (Simulated Clamp Channel)

DEFAULT: 0	RANGE: 0 16

Simulated Clamp Channel (SIM_CLAMP_CH)

Description

`SIM_CLAMP_CH` creates a virtual "Mag-Lock" on the landing gear.

When the selected RC channel is High (PWM > 1800), the drone is physically stuck to the ground, regardless of throttle. This is useful for testing "Ground Idle" vibration or tuning PID loops without the drone taking off.

Tuning & Behavior

- **0:** Disabled.
- **1-16:** RC Channel number.

SIM_DRIFT_SPEED (Simulation GPS Drift Speed)

DEFAULT: 0.05	RANGE: 0 0.5	UNITS: m/s

Simulation GPS Drift Speed (SIM_DRIFT_SPEED)

Description

`SIM_DRIFT_SPEED` simulates the "Wandering" effect of GPS.

Real GPS position is never perfectly still; it slowly drifts around a central point due to atmospheric changes. This parameter controls how fast that drift moves.

Tuning & Behavior

- **Default Value:** 0.05 m/s.
- **Testing:** Increase to **0.2** to test if your Loiter PID tuning is tight enough to hold position against a "moving" GPS target.

SIM_DRIFT_TIME (Simulation GPS Drift Time)

DEFAULT: 5	RANGE: 1 50	UNITS: s

Simulation GPS Drift Time (SIM_DRIFT_TIME)

Description

`SIM_DRIFT_TIME` controls how often the GPS drift changes direction.

See SIM_DRIFT_SPEED.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





SIM_EFI_TYPE (Simulated EFI Type)

DEFAULT: 0	RANGE: 0 8

Simulated EFI Type (SIM_EFI_TYPE)

Description

`SIM_EFI_TYPE` adds a virtual engine to the simulation.

- **0:** Disabled.
- **1:** MegaSquirt.
- **2:** Loweheiser.
- **8:** Hirth.

Tuning & Behavior

- **Default Value:** 0.
- **Usage:** Enable this to test your `EFI_*` parameter configuration and OSD layout.

SIM_ENGINE_FAIL (Simulated Engine Failure)

DEFAULT: -1	RANGE: -1 8

Simulated Engine Failure (SIM_ENGINE_FAIL)

Description

`SIM_ENGINE_FAIL` selects the victim motor for failure testing.

- **-1 (Default):** No failure.
- **0:** Motor 1 fails.
- **1:** Motor 2 fails.

Tuning & Behavior

- **Default Value:** -1.
- **Procedure:**
    1. Take off and hover.
    2. Set `SIM_ENGINE_MUL` to 0.
    3. Set `SIM_ENGINE_FAIL` to 0 (Motor 1).
    4. Observe if the drone crashes or spins.

SIM_ENGINE_MUL (Simulation Engine Multiplier)

DEFAULT: 1.0	RANGE: 0 1.0

Simulation Engine Multiplier (SIM_ENGINE_MUL)

Description

`SIM_ENGINE_MUL` allows you to kill or weaken a motor in flight.

This is the primary tool for testing multirotor motor redundancy (e.g. Octocopter failure) or plane engine-out handling.

- **1.0 (Default):** Full power.
- **0.0:** Complete motor failure.
- **0.5:** 50% power loss (damaged prop or ESC).

Tuning & Behavior

- **Usage:** You must also set `SIM_ENGINE_FAIL` to the motor number (0-indexed) you want to affect.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





SIM_ESC_ARM_RPM (Simulated ESC Armed RPM)

DEFAULT: 0	RANGE: 0 5000	UNITS: RPM

Simulated ESC Armed RPM (SIM_ESC_ARM_RPM)

Description

`SIM_ESC_ARM_RPM` sets the "Idle Speed" for the virtual RPM feedback.

Tuning & Behavior

- **Default Value:** 0.
- **Recommendation:** Set to **500 or 1000** to simulate the low-end spinning of motors while the drone is on the ground but armed.

SIM_ESC_TELEM (Simulated ESC Telemetry Type)

DEFAULT: 0	RANGE: 0 2

Simulated ESC Telemetry Type (SIM_ESC_TELEM)

Description

`SIM_ESC_TELEM` enables the virtual return path for motor data.

If you are testing **DShot ESC Telemetry** or **RPM filtering**, you need this enabled so the autopilot "sees" the RPM and voltage data coming back from the virtual ESCs.

- **0:** Disabled.
- **1:** Enabled (Standard RPM + Voltage).
- **2:** Bidirectional DShot (RPM only, but high-speed).

Tuning & Behavior

- **Default Value:** 0.
- **Recommendation:** Set to **1** to verify your OSD battery display and motor fail detection.

SIM_FLOAT_EXCEPT (Simulated Floating Point Exceptions)

DEFAULT: 1	RANGE: 0 1

Simulated Floating Point Exceptions (SIM_FLOAT_EXCEPT)

Description

`SIM_FLOAT_EXCEPT` is a developer safety tool.

If enabled, the simulator will immediately "Hard Crash" if any part of the code performs an illegal math operation (like dividing by zero or taking the square root of a negative number). This is vital for catching bugs before they are ever flown on real hardware.

Tuning & Behavior

- **Default Value:** 1 (Enabled).
- **Recommendation:** Keep at **1** at all times unless you are debugging a known math issue that you need the simulator to bypass.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY

SIM_FLOW_DELAY (Simulated Optical Flow Delay)

DEFAULT: 0	RANGE: 0 500	UNITS: ms

Simulated Optical Flow Delay (SIM_FLOW_DELAY)

Description

`SIM_FLOW_DELAY` simulates the processing lag of a smart camera (like HereFlow or PX4Flow).

Tuning & Behavior

- **Default Value:** 0 ms.
- **Recommendation:** Set to **20** to match real hardware.

SIM_FLOW_ENABLE (Simulated Optical Flow Enable)

DEFAULT: 0	RANGE: 0 1

Simulated Optical Flow Enable (SIM_FLOW_ENABLE)

Description

`SIM_FLOW_ENABLE` adds a virtual Optical Flow camera to the drone.

This allows you to test non-GPS loiter modes (`FlowHold`) and indoor navigation logic.

Tuning & Behavior

- **Default Value:** 0 (Disabled).
- **1:** Enabled.

SIM_FLOW_POS (Simulation Optical Flow Position)

DEFAULT: 0	RANGE: 0 5	UNITS: m

Simulation Optical Flow Position (SIM_FLOW_POS)

Description

`SIM_FLOW_POS` defines where the optical flow sensor is mounted in the simulator.

Just like the IMU and GPS, if the optical flow sensor is away from the COG, it will experience extra movement during rotations. This parameter tests the flight controller's ability to use the FLOW_POS_X/Y/Z parameters to compensate for these "lever arm" effects.

SIM_FLOW_RATE (Simulated Optical Flow Rate)

DEFAULT: 10	RANGE: 1 50	UNITS: Hz

Simulated Optical Flow Rate (SIM_FLOW_RATE)

Description

`SIM_FLOW_RATE` sets the data frequency of the virtual flow sensor.

Tuning & Behavior

- **Default Value:** 10 Hz.
- **Recommendation:** Match your hardware specs (e.g., 50Hz for high-end sensors).

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





SIM_FLOW_RND (Simulated Optical Flow Noise)

DEFAULT: 0.05	RANGE: 0 0.5	UNITS: rad/s

Simulated Optical Flow Noise (SIM_FLOW_RND)

Description

`SIM_FLOW_RND` simulates the "jitter" of an optical flow sensor looking at a difficult surface (like grass or carpet).

Tuning & Behavior

- **Default Value:** 0.05 rad/s.
- **Recommendation:** Increase to **0.2** to test if your drone can hold position over low-contrast surfaces.

SIM_FTOWESC_ENA (Simulation FETtec OneWire ESC Enable)

DEFAULT: 0	RANGE: 0 1

Simulation FETtec OneWire ESC Enable (SIM_FTOWESC_ENA)

Description

`SIM_FTOWESC_ENA` activates the virtual FETtec OneWire interface.

This allows you to test the SERVO_FTW_MASK configuration and RPM telemetry feedback without having a physical FETtec ESC board connected.

SIM_FTOWESC_POW (Simulated TOWESC Power)

DEFAULT: 0	RANGE: 0 10000	UNITS: W

Simulated TOWESC Power (SIM_FTOWESC_POW)

Description

`SIM_FTOWESC_POW` defines the maximum power capacity of the heavy-lift motor simulation.

SIM_GLD_BLN_BRST (Simulation Balloon Burst)

DEFAULT: 0	RANGE: 0 1

Simulation Balloon Burst (SIM_GLD_BLN_BRST)

Description

`SIM_GLD_BLN_BRST` triggers the "Release" phase of a balloon glider test.

For drones that are carried to high altitudes by balloons, this parameter allows you to simulate the moment the balloon pops or the drone is released, instantly transitioning the physics model from buoyant ascent to aerodynamic glide.

SIM_GLD_BLN_RATE (Simulated Glider/Balloon Rate)

DEFAULT: 0	RANGE: 0 100	UNITS: Hz

Simulated Glider/Balloon Rate (SIM_GLD_BLN_RATE)

Description

`SIM_GLD_BLN_RATE` sets the physics resolution for slow-moving airframes like gliders and weather balloons.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





SIM_GND_BEHAV (Simulated Ground Behavior)

DEFAULT: -1	RANGE: -1 2

Simulated Ground Behavior (SIM_GND_BEHAV)

Description

`SIM_GND_BEHAV` modifies how the drone interacts with the virtual earth.

- **-1:** Automatic (standard).
- **0:** Non-Tailsitter (stays flat).
- **1:** Tailsitter (leans back onto its tail).
- **2:** Stay on ground (locked position).

Tuning & Behavior

- **Default Value:** -1.
- **Recommendation:** Leave at **-1** unless you are testing a tailsitter QuadPlane.

SIM_GPS1_ALT_OFS (Simulated GPS 1 Altitude Offset)

DEFAULT: 0	RANGE: -100 100	UNITS: m

Simulated GPS 1 Altitude Offset (SIM_GPS1_ALT_OFS)

Description

`SIM_GPS1_ALT_OFS` adds a constant error to the GPS altitude.

This is used to verify that the EKF can handle a disagreement between the Barometer (which reports correct altitude) and the GPS (which reports offset altitude).

Tuning & Behavior

- **Default Value:** 0 m.
- **Testing:** Set to **20**. The EKF should initially report a large height innovation (error), then slowly shift the GPS bias or reject the GPS.

SIM_GPS1_ENABLE (Simulated GPS 1 Enable)

DEFAULT: 1	RANGE: 0 1

Simulated GPS 1 Enable (SIM_GPS1_ENABLE)

Description

`SIM_GPS1_ENABLE` is the power switch for the virtual GPS.

If you want to test how your drone flies with ONLY the secondary GPS, or no GPS at all, you can use this parameter to turn off the primary unit.

Tuning & Behavior

- **Default Value:** 1 (Enabled).
- **0:** Disabled.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





SIM_GPS1_HDG_OFS (Simulated GPS 1 Heading Offset)

DEFAULT: 0	RANGE: 0 360	UNITS: deg

Simulated GPS 1 Heading Offset (SIM_GPS1_HDG_OFS)

Description

`SIM_GPS1_HDG_OFS` adds a constant error to the heading reported by the GPS.

This is only relevant if `SIM_GPS_HDG` is enabled (simulating a Dual Antenna GPS). It tests the EKF's ability to handle Compass/GPS yaw disagreement.

Tuning & Behavior

- **Default Value:** 0 deg.
- **Testing:** Set to **90**. The EKF should trigger a "Yaw Alignment" error.

SIM_GPS1_TYPE (Simulated GPS 1 Type)

DEFAULT: 1	RANGE: 0 19

Simulated GPS 1 Type (SIM_GPS1_TYPE)

Description

`SIM_GPS1_TYPE` tells the simulator which GPS language to speak to the autopilot.

- **1: UBlox (Default).** The standard binary protocol used by M8N/F9P modules.
- **5: NMEA.** The standard text-based protocol used by older or generic GPS units.
- **19: MSP.** MultiWii Serial Protocol, used by DJI or Walksnail video systems to pass GPS data.

Tuning & Behavior

- **Default Value:** 1.
- **Recommendation:** Keep at **1 (UBlox)** for the most realistic simulation of standard ArduPilot hardware.

SIM_GPS2_ACC (GPS 2 Accuracy)

DEFAULT: 0.3	RANGE: null	UNITS: null

GPS 2 Accuracy

Note: This parameter functions identically to SIM_GPS_ACC.

SIM_GPS2_ALT_OFS (Compass offsets in milligauss on the Z axis)

DEFAULT: 0	RANGE: -400 400	UNITS: mGauss

Compass offsets in milligauss on the Z axis

Note: This parameter configures instance 2. It functions identically to SIM_GPS_ALT_OFS.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY

SIM_GPS2_BYTELOS (Simulated GPS 2 Byte Loss)

DEFAULT: 0	RANGE: 0 100	UNITS: %

Simulated GPS 2 Byte Loss (SIM_GPS2_BYTELOS)

Description

`SIM_GPS2_BYTELOS` degrades the connection to the secondary GPS.

By dropping random bytes, you can simulate a loose cable or electrical noise on the UART line. This tests the GPS driver's ability to re-sync and the EKF's ability to handle packet loss.

Tuning & Behavior

- **Default Value:** 0%.

SIM_GPS2_DISABLE (GPS 2 disable)

DEFAULT: 1	RANGE: null

GPS 2 disable

Note: This parameter functions identically to SIM_GPS_DISABLE.

SIM_GPS2_DRFTALT (Simulated GPS 2 Altitude Drift)

DEFAULT: 0	RANGE: 0 5	UNITS: m/s

Simulated GPS 2 Altitude Drift (SIM_GPS2_DRFTALT)

Description

`SIM_GPS2_DRFTALT` makes the secondary GPS "Float" vertically.

GPS altitude is notoriously unreliable. This parameter adds a slow, continuous change to the reported height, allowing you to verify that the EKF (which usually trusts the Barometer for height) ignores the bad GPS altitude data.

Tuning & Behavior

- **Default Value:** 0.

SIM_GPS2_ENABLE (Enable pullup after altitude wait)

DEFAULT: 0	RANGE: null	UNITS: null

Enable pullup after altitude wait

Note: This parameter functions identically to SIM_GPS1_ENABLE.

SIM_GPS2_GLTCH (Simulation GPS 2 Glitch)

DEFAULT: 0	RANGE: -100 100	UNITS: m

Simulation GPS 2 Glitch (SIM_GPS2_GLTCH)

Description

`SIM_GPS2_GLTCH` allows you to test dual-GPS configurations by "breaking" the secondary receiver.

By injecting a glitch into only one GPS, you can verify that the EKF successfully switches to the healthy receiver (if configured for `GPS_AUTO_SWITCH = 1`).

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





SIM_GPS2_HDG (Simulated GPS 2 Heading)

DEFAULT: 0	RANGE: 0 360	UNITS: deg

Simulated GPS 2 Heading (SIM_GPS2_HDG)

Description

`SIM_GPS2_HDG` sets the "True North" for the secondary GPS if it is a "Moving Baseline" unit.

By setting this to a value different from the actual vehicle heading, you can test how the autopilot handles "GPS Yaw" errors or alignment issues.

Tuning & Behavior

- **Default Value:** 0 (Matches vehicle heading).

SIM_GPS2_HDG_OFS (Compass offsets in milligauss on the Z axis)

DEFAULT: 0	RANGE: -400 400	UNITS: mGauss

Compass offsets in milligauss on the Z axis

Note: This parameter functions identically to SIM_GPS1_HDG_OFS.

SIM_GPS2_HZ (GPS 2 Hz)

DEFAULT: 5	RANGE: null	UNITS: Hz

GPS 2 Hz

Note: This parameter functions identically to SIM_GPS_HZ.

SIM_GPS2_JAM (GPS jamming enable)

DEFAULT: 0	RANGE: null	UNITS: null

GPS jamming enable

Note: This parameter configures instance 2. It functions identically to SIM_GPS_JAM.

SIM_GPS2_LAG_MS (GPS 2 Lag)

DEFAULT: 100	RANGE: null	UNITS: ms

GPS 2 Lag

Note: This parameter functions identically to SIM_GPS_LAG_MS.

SIM_GPS2_LCKTIME (Simulation GPS 2 Lock Time)

DEFAULT: 0	RANGE: 0 120	UNITS: s

Simulation GPS 2 Lock Time (SIM_GPS2_LCKTIME)

Description

`SIM_GPS2_LCKTIME` allows you to simulate a scenario where one GPS gets a fix much later than the other, testing the autopilot's primary/secondary switchover logic.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





SIM_GPS2_NOISE (Simulated GPS 2 Noise)

DEFAULT: 0	RANGE: 0 10	UNITS: m

Simulated GPS 2 Noise (SIM_GPS2_NOISE)

Description

`SIM_GPS2_NOISE` adds jitter to the secondary GPS.

Use this to test GPS Blending (`GPS_AUTO_SWITCH` = 2). You can make GPS 2 noisier than GPS 1 and verify that the autopilot correctly weights GPS 1 higher.

Tuning & Behavior

- **Default Value:** 0.

SIM_GPS2_NUMSATS (GPS 2 Num Satellites)

DEFAULT: 10	RANGE: null	UNITS: null

GPS 2 Num Satellites

Note: This parameter functions identically to SIM_GPS_NUMSATS.

SIM_GPS2_POS (GPS 2 Position)

DEFAULT: 0	RANGE: null	UNITS: m

GPS 2 Position

Note: This parameter functions identically to SIM_GPS_POS.

SIM_GPS2_TYPE (Tiltrotor type)

DEFAULT: TILT_TYPE_CONTINUOUS	RANGE: null	UNITS: null

Tiltrotor type

Note: This parameter configures instance 2. It functions identically to SIM_GPS_TYPE.

SIM_GPS2_VERR (GPS 2 Velocity Error)

DEFAULT: 0	RANGE: null

GPS 2 Velocity Error

Note: This parameter configures instance 2. It functions identically to SIM_GPS_VERR.

SIM_GPS3_ALT_OFS (Compass offsets in milligauss on the Z axis)

DEFAULT: 0	RANGE: -400 400	UNITS: mGauss

Compass offsets in milligauss on the Z axis

Note: This parameter configures instance 3. It functions identically to SIM_GPS_ALT_OFS.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





SIM_GPS3_ENABLE (Enable pullup after altitude wait)

DEFAULT: 0	RANGE: null	UNITS: null

Enable pullup after altitude wait

Note: This parameter functions identically to SIM_GPS1_ENABLE.

SIM_GPS3_HDG_OFS (Compass offsets in milligauss on the Z axis)

DEFAULT: 0	RANGE: -400 400	UNITS: mGauss

Compass offsets in milligauss on the Z axis

Note: This parameter functions identically to SIM_GPS1_HDG_OFS.

SIM_GPS3_TYPE (Tiltrotor type)

DEFAULT: TILT_TYPE_CONTINUOUS	RANGE: null	UNITS: null

Tiltrotor type

Note: This parameter configures instance 3. It functions identically to SIM_GPS_TYPE.

SIM_GPS4_ALT_OFS (Compass offsets in milligauss on the Z axis)

DEFAULT: 0	RANGE: -400 400	UNITS: mGauss

Compass offsets in milligauss on the Z axis

Note: This parameter configures instance 4. It functions identically to SIM_GPS_ALT_OFS.

SIM_GPS4_ENABLE (Enable pullup after altitude wait)

DEFAULT: 0	RANGE: null	UNITS: null

Enable pullup after altitude wait

Note: This parameter functions identically to SIM_GPS1_ENABLE.

SIM_GPS4_HDG_OFS (Compass offsets in milligauss on the Z axis)

DEFAULT: 0	RANGE: -400 400	UNITS: mGauss

Compass offsets in milligauss on the Z axis

Note: This parameter functions identically to SIM_GPS1_HDG_OFS.

SIM_GPS4_TYPE (Tiltrotor type)

DEFAULT: TILT_TYPE_CONTINUOUS	RANGE: null	UNITS: null

Tiltrotor type

Note: This parameter configures instance 4. It functions identically to SIM_GPS_TYPE.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





SIM_GPS_ACC (Simulated GPS Accuracy)

DEFAULT: 0.3	RANGE: 0 10	UNITS: m

Simulated GPS Accuracy (SIM_GPS_ACC)

Description

`SIM_GPS_ACC` defines the "Radius of Confusion" for the virtual GPS.

It sets the value reported in the MAVLink `GPS_RAW_INT.h_acc` field. While it doesn't always add physical noise to the position (see `SIM_GPS_NOISE`), it tells the EKF how much to trust the position.

Tuning & Behavior

- **Default Value:** 0.3 m.
- **Recommendation:** Set to **0.1** for "Ideal" GPS or **2.0** to simulate poor signal conditions.

SIM_GPS_ALT_OFS (Simulation GPS Altitude Offset)

DEFAULT: 0	RANGE: -100 100	UNITS: m

Simulation GPS Altitude Offset (SIM_GPS_ALT_OFS)

Description

`SIM_GPS_ALT_OFS` injects a constant error into the GPS altitude. This is used to test if the EKF can handle significant differences between its primary altitude source (usually Barometer) and the GPS.

SIM_GPS_BYTELOSS (Simulation GPS Byte Loss)

DEFAULT: 0	RANGE: 0 100	UNITS: %

Simulation GPS Byte Loss (SIM_GPS_BYTELOSS)

Description

`SIM_GPS_BYTELOSS` simulates a flaky or noisy serial cable.

By randomly dropping a percentage of the data packets coming from the GPS, you can test how ArduPilot handles "Checksum Errors" and "Intermittent GPS" failures.

Tuning & Behavior

- **Default Value:** 0.
- **Testing:** Set to **10**. You should see "GPS Glitch" or "GPS 1: Checksum error" messages in your GCS.

SIM_GPS_DELAY (Simulation GPS Delay)

DEFAULT: 1	RANGE: 0 10

Simulation GPS Delay (SIM_GPS_DELAY)

Description

`SIM_GPS_DELAY` simulates the processing lag of a real GPS module.

Real GPS units don't report position instantly; there is a calculation delay.

Tuning & Behavior

- **Default Value:** 1.
- **Testing:** Increasing this can stress-test the EKF's ability to handle delayed measurements.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





SIM_GPS_DISABLE (Simulation GPS Disable)

DEFAULT: 0	RANGE: 0 1

Simulation GPS Disable (SIM_GPS_DISABLE)

Description

`SIM_GPS_DISABLE` kills the GPS signal.

- **0:** GPS Enabled.
- **1:** GPS Disabled (No Fix).

Tuning & Behavior

- **Default Value:** 0.
- **Testing:** Use to test Optical Flow, T265 Realsense, or just plain AltHold/Stabilize flight without GPS assistance.

SIM_GPS_DRIFTALT (Simulation GPS Altitude Drift)

DEFAULT: 0	RANGE: -1 1	UNITS: m/s

Simulation GPS Altitude Drift (SIM_GPS_DRIFTALT)

Description

`SIM_GPS_DRIFTALT` simulates the atmospheric drift often seen in low-cost GPS modules.

If the GPS reports that the drone is slowly "climbing" or "sinking" while it is actually sitting on the ground, the EKF must decide whether to believe the GPS or the Barometer. This parameter allows you to test that weighting logic.

SIM_GPS_GLITCH (Simulated GPS Glitch Offset)

DEFAULT: 0	RANGE: 0 100	UNITS: m

Simulated GPS Glitch Offset (SIM_GPS_GLITCH)

Description

`SIM_GPS_GLITCH` simulates a "GPS Jump."

It adds a sudden, constant offset to the GPS coordinates. This is the best way to test if your `EKF_CHECK_SCALE` and failsafe settings are tight enough to detect a position jump before the drone flies away.

Tuning & Behavior

- **Default Value:** 0.
- **Testing:** Set to **10** or **20** meters while in LOITER mode and verify the drone triggers a failsafe rather than banking hard to chase the glitched position.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





SIM_GPS_GLITCH_X (Simulation GPS Glitch X)

DEFAULT: 0	RANGE: -100 100	UNITS: m

Simulation GPS Glitch X (SIM_GPS_GLITCH_X)

Description

`SIM_GPS_GLITCH_X` allows you to inject a GPS failure mid-flight.

By changing this parameter via MAVLink while flying in SITL, you can instantly shift the reported GPS position.

Tuning & Behavior

- **Default Value:** 0.
- **Testing:** Set to **20** while Loitering. The drone should lean hard to "correct" its position (actually moving away from the true target), then EKF might reject the glitch. Useful for testing EKF failsafe logic.

SIM_GPS_HDG (Simulated GPS Heading Enable)

DEFAULT: 0	RANGE: 0 3

Simulated GPS Heading Enable (SIM_GPS_HDG)

Description

`SIM_GPS_HDG` allows the virtual GPS to report a "Moving Base" heading.

This is used for testing **GPS-for-Yaw** (Dual Antenna) setups in SITL. It tells the simulator to generate the relative position and heading messages that a real U-Blox or NMEA GPS would send when acting as a heading source.

Tuning & Behavior

- **0:** None.
- **1:** NMEA HDT sentence.
- **2:** U-Blox RELPOSNED.
- **3:** NMEA THS sentence.

SIM_GPS_HZ (Simulation GPS Update Rate)

DEFAULT: 5	RANGE: 1 50	UNITS: Hz

Simulation GPS Update Rate (SIM_GPS_HZ)

Description

`SIM_GPS_HZ` sets the "Refresh Rate" of the simulated GPS.

- **5 (Default):** Standard for most consumer GPS modules.
- **10:** High performance (u-blox M8N/M9N).
- **50:** Industrial RTK systems.

Tuning & Behavior

- **Default Value:** 5 Hz.
- **Testing:** Use this to verify that your `GPS_RATE_MS` and `EKF` settings can handle higher data rates without CPU overload.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





SIM_GPS_JAM (Simulated GPS Jamming)

DEFAULT: 0	RANGE: 0 1

Simulated GPS Jamming (SIM_GPS_JAM)

Description

`SIM_GPS_JAM` simulates a "Denial of Service" attack on your GPS.

When enabled, the GPS continues to report a "3D Fix" (usually), but the signal quality degrades, and the position may drift or freeze, mimicking the behavior of a jammed receiver.

Tuning & Behavior

- **0:** Disabled.
- **1:** Jamming Active.

SIM_GPS_JAM (Simulation GPS Jamming)

DEFAULT: 0	RANGE: 0 1

Simulation GPS Jamming (SIM_GPS_JAM)

Description

`SIM_GPS_JAM` simulates the drone flying into an area with a GPS jammer.

- **0: Normal.** GPS works as expected.
- **1: Jammed.** The autopilot instantly loses GPS fix, as if the antenna were disconnected. Use this to verify that your drone's EKF correctly fails over to dead-reckoning or triggers an immediate "Land" failsafe.

SIM_GPS_LAG_MS (Simulated GPS Lag)

DEFAULT: 120	RANGE: 0 500	UNITS: ms

Simulated GPS Lag (SIM_GPS_LAG_MS)

Description

`SIM_GPS_LAG_MS` simulates the time it takes for a GPS module to calculate your position and send it over the wire.

Real GPS units always have a delay (latency). This parameter is critical for EKF tuning; the EKF needs to know how "Old" the GPS data is so it can align it with the high-speed IMU data.

Tuning & Behavior

- **Default Value:** 120 ms.
- **Recommendation:** Set to match your hardware (e.g., U-Blox M8N is typically around **200ms**, M9N is closer to **80-120ms**).

SIM_GPS_LOCKTIME (Simulation GPS Lock Time)

DEFAULT: 0	RANGE: 0 120	UNITS: s

Simulation GPS Lock Time (SIM_GPS_LOCKTIME)

Description

`SIM_GPS_LOCKTIME` simulates a "Cold Start."

In SITL, the GPS usually gets a fix instantly. This parameter forces the autopilot to wait for a realistic duration before the GPS reports a valid position. Useful for testing pre-arm procedures.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





SIM_GPS_LOG_NUM (Simulated GPS Log Number)

DEFAULT: 0	RANGE: 0 100

Simulated GPS Log Number (SIM_GPS_LOG_NUM)

Description

`SIM_GPS_LOG_NUM` allows you to replay a specific GPS track from a previous flight log.

If you have a DataFlash log (e.g., `00000012.BIN`) where the GPS glitched or behaved weirdly, you can feed that exact GPS data into the simulator to see if the EKF can handle it better with different settings.

Tuning & Behavior

- **Default Value:** 0 (Disabled).
- **Usage:** Set `SIM_GPS_TYPE` to **7 (File)** and set this parameter to the log index you want to replay.

SIM_GPS_NOISE (Simulated GPS Noise)

DEFAULT: 0	RANGE: 0 10	UNITS: m

Simulated GPS Noise (SIM_GPS_NOISE)

Description

`SIM_GPS_NOISE` adds random position jitter to the virtual GPS.

Unlike `SIM_GPS_ACC` (which only *reports* accuracy), this parameter actually moves the reported latitude/longitude/altitude by a random amount in every frame. This is the primary tool for testing how much GPS "Fuzz" your EKF can handle before the drone starts to wander or failsafes.

Tuning & Behavior

- **Default Value:** 0 (Perfect GPS).
- **Recommendation:** Set to **0.5 or 1.0** to simulate a realistic low-cost GPS in an open field. Set to **3.0+** to simulate urban canyon environments.

SIM_GPS_NUMSATS (Simulation GPS Satellite Count)

DEFAULT: 10	RANGE: 0 30

Simulation GPS Satellite Count (SIM_GPS_NUMSATS)

Description

`SIM_GPS_NUMSATS` allows you to simulate "Signal Shading" or poor sky visibility.

ArduPilot's EKF requires a minimum number of satellites to maintain a high-quality position fix. By reducing this value in SITL, you can test how the drone behaves when it enters a "Degraded GPS" state.

- **10 (Default):** Healthy fix.
- **5:** Poor fix. EKF may report warnings.
- **3:** Fix lost. Drone will switch to non-GPS modes.

SIM_GPS_POS (Simulation GPS Position Offset)

DEFAULT: 0	RANGE: -5 5	UNITS: m

Simulation GPS Position Offset (SIM_GPS_POS)

Description

`SIM_GPS_POS` defines the mounting location of the virtual GPS antenna.

See GPS_POS for details on why antenna offsets are critical for flight stability.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





SIM_GPS_TYPE (Simulation GPS Type)

DEFAULT: 1	RANGE: 0 26

Simulation GPS Type (SIM_GPS_TYPE)

Description

`SIM_GPS_TYPE` tells the simulator which "Language" to use when sending data to the autopilot.

- **1: u-blox (Standard).** The most common GPS type.
- **5: NMEA.** Generic serial data.
- **9: DroneCAN.** Simulates a CAN-bus GPS.
- **14: MAVLink.**

SIM_GPS_VERR (Simulation GPS Velocity Error)

DEFAULT: 0	RANGE: 0 5	UNITS: m/s

Simulation GPS Velocity Error (SIM_GPS_VERR)

Description

`SIM_GPS_VERR` simulates a GPS receiver that reports incorrect speeds.

This is a common failure mode in real-world "Urban Canyons," where signal multipath can cause the GPS to think the drone is moving at 2 m/s even when it is perfectly still. This parameter helps you test the EKF's ability to reject bad velocity data and rely more on the accelerometers.

SIM_GRPE_ENABLE (Simulation Gripper Enable)

DEFAULT: 0	RANGE: 0 1

Simulation Gripper Enable (SIM_GRPE_ENABLE)

Description

`SIM_GRPE_ENABLE` activates a virtual robotic gripper.

This allows you to test "Pick and Place" missions in SITL. When enabled, the drone can physically attach to objects (like a virtual package) and carry them through the air, accounting for the added weight and inertia in the physics model.

SIM_GRPE_PIN (Simulated EPM Gripper Pin)

DEFAULT: 0	RANGE: 1 32

Simulated EPM Gripper Pin (SIM_GRPE_PIN)

Description

`SIM_GRPE_PIN` links the virtual EPM magnet to an output channel.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





SIM_GRPS_ENABLE (Gripper Servo Sim Enable)

DEFAULT: 0	RANGE: 0 1

Gripper Servo Sim Enable (SIM_GRPS_ENABLE)

Description

`SIM_GRPS_ENABLE` adds a functional virtual gripper to the drone.

This allows you to test "Cargo Hold" operations. The simulator models a servo-driven jaw that can pick up and drop a 1kg load when on the ground.

Tuning & Behavior

- **0:** Disabled.
- **1:** Enabled.

SIM_GRPS_GRAB (Gripper Grab PWM)

DEFAULT: 1100	RANGE: 1000 2000	UNITS: PWM

Gripper Grab PWM (SIM_GRPS_GRAB)

Description

`SIM_GRPS_GRAB` sets the "Closed" position for the virtual gripper.

SIM_GRPS_PIN (Gripper Servo Pin)

DEFAULT: -1	RANGE: -1 16

Gripper Servo Pin (SIM_GRPS_PIN)

Description

`SIM_GRPS_PIN` connects the virtual gripper to a flight controller output.

Set this to match your `SERVOx_FUNCTION = 28` (Gripper) channel. When ArduPilot moves that servo, the simulator will open or close the virtual jaws.

Tuning & Behavior

- **Default Value:** -1 (Disabled).

SIM_GRPS_RELEASE (Gripper Release PWM)

DEFAULT: 1900	RANGE: 1000 2000	UNITS: PWM

Gripper Release PWM (SIM_GRPS_RELEASE)

Description

`SIM_GRPS_RELEASE` sets the "Open" position for the virtual gripper.

SIM_GRPS_REVERSE (Gripper Reverse)

DEFAULT: 0	RANGE: 0 1

Gripper Reverse (SIM_GRPS_REVERSE)

Description

`SIM_GRPS_REVERSE` swaps the open and closed states of the virtual jaws.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





SIM_GYR1_BIAS (Simulation Gyro 1 Bias)

DEFAULT: 0	RANGE: -5 5	UNITS: deg/s

Simulation Gyro 1 Bias (SIM_GYR1_BIAS)

Description

`SIM_GYR1_BIAS` simulates "Gyro Drift."

Real-world gyroscopes are sensitive to temperature and time. If you leave a drone sitting on a table, the reported rotation rate might not be exactly zero. This drift is the primary cause of "Horizon Tilt" in flight. ArduPilot's EKF is designed to learn and cancel this drift.

Tuning & Behavior

- **Default Value:** 0.
- **Testing:** Set to **1.0** (1 degree per second). You can watch the EKF "innovations" in your logs as the autopilot detects and attempts to compensate for this fake drift.

SIM_GYR1_RND (Simulated Gyroscope 1 Noise)

DEFAULT: 0	RANGE: 0 10.0	UNITS: deg/s

Simulated Gyroscope 1 Noise (SIM_GYR1_RND)

Description

`SIM_GYR1_RND` injects artificial noise into the simulated primary gyroscope.

This is used to test how "twitchy" or unstable the drone's attitude control becomes when the gyro data is imperfect. High noise levels can cause the drone to oscillate or lose orientation entirely.

The Engineer's View

Adds Gaussian noise (σ) to the body-frame rotational rate:

$$\omega_read = \omega_true + \mathcal{N}(0, \text{SIM_GYR1_RND})$$

Tuning & Behavior

- **Default Value:** 0
- **Effect of Increasing:** The drone's attitude (roll/pitch/yaw) will appear to vibrate or "jitter" in the GCS HUD, and the EKF may trigger "high gyro noise" warnings.

SIM_GYR1_SCALE (Simulation Gyro 1 Scale)

DEFAULT: 0	RANGE: -0.1 0.1

Simulation Gyro 1 Scale (SIM_GYR1_SCALE)

Description

`SIM_GYR1_SCALE` simulates a sensitivity error on the rotation axis.

If you rotate the drone by exactly 360 degrees, a sensor with a scale error might report 365 or 355 degrees. This is important for verifying high-rate aerobatic performance.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





SIM_GYR2_BIAS (Simulation Gyro 2 Bias)

DEFAULT: 0	RANGE: -5 5	UNITS: deg/s

Simulation Gyro 2 Bias (SIM_GYR2_BIAS)

Description

`SIM_GYR2_BIAS` simulates drift on the secondary IMU.

See SIM_GYR1_BIAS for details.

SIM_GYR2_RND (Gyro 2 motor noise factor)

DEFAULT: 0	RANGE: null	UNITS: null

Gyro 2 motor noise factor

Note: This parameter functions identically to SIM_GYR1_RND.

SIM_GYR2_SCALE (Simulation Gyro 2 Scale)

DEFAULT: 0	RANGE: -0.1 0.1

Simulation Gyro 2 Scale (SIM_GYR2_SCALE)

Description

`SIM_GYR2_SCALE` simulates sensitivity errors on the secondary IMU.

See SIM_GYR1_SCALE for details.

SIM_GYR3_BIAS (Simulation Gyro 3 Bias)

DEFAULT: 0	RANGE: -5 5	UNITS: deg/s

Simulation Gyro 3 Bias (SIM_GYR3_BIAS)

Description

`SIM_GYR3_BIAS` simulates drift on the tertiary IMU.

See SIM_GYR1_BIAS for details.

SIM_GYR3_RND (Gyro 3 motor noise factor)

DEFAULT: 0	RANGE: null	UNITS: null

Gyro 3 motor noise factor

Note: This parameter functions identically to SIM_GYR1_RND.

SIM_GYR3_SCALE (Gyro 3 scaling factor)

DEFAULT: 0	RANGE: null

Gyro 3 scaling factor

Note: This parameter configures instance 3. It functions identically to SIM_GYR1_SCALE.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





SIM_GYR4_BIAS (Fourth Gyro bias on Z axis)

DEFAULT: 0	RANGE: null

Fourth Gyro bias on Z axis

Note: This parameter configures instance 4. It functions identically to SIM_GYR1_BIAS.

SIM_GYR4_RND (Gyro 4 motor noise factor)

DEFAULT: 0	RANGE: null	UNITS: null

Gyro 4 motor noise factor

Note: This parameter functions identically to SIM_GYR1_RND.

SIM_GYR4_SCALE (Gyro 4 scaling factor)

DEFAULT: 0	RANGE: null

Gyro 4 scaling factor

Note: This parameter configures instance 4. It functions identically to SIM_GYR1_SCALE.

SIM_GYR5_BIAS (Fifth Gyro bias on Z axis)

DEFAULT: 0	RANGE: null

Fifth Gyro bias on Z axis

Note: This parameter configures instance 5. It functions identically to SIM_GYR1_BIAS.

SIM_GYR5_RND (Gyro 5 motor noise factor)

DEFAULT: 0	RANGE: null	UNITS: null

Gyro 5 motor noise factor

Note: This parameter functions identically to SIM_GYR1_RND.

SIM_GYR5_SCALE (Gyro 5 scaling factor)

DEFAULT: 0	RANGE: null

Gyro 5 scaling factor

Note: This parameter configures instance 5. It functions identically to SIM_GYR1_SCALE.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





SIM_GYR_FAIL_MSK (Simulated Gyroscope Failure Mask)

DEFAULT: 0	RANGE: 0 7

Simulated Gyroscope Failure Mask (SIM_GYR_FAIL_MSK)

Description

`SIM_GYR_FAIL_MSK` kills specific gyroscopes in the simulator.

This is critical for testing the "vibration failsafe" and EKF lane switching logic.

- **Bit 0 (1):** Fail Gyro 1
- **Bit 1 (2):** Fail Gyro 2
- **Bit 2 (4):** Fail Gyro 3

Tuning & Behavior

- **Default Value:** 0
- **Recommendation:** Use in conjunction with `SIM_ACC_FAIL_MSK` to simulate a complete IMU failure.

SIM_GYR_FILE_RW (Simulated Gyroscope File Read/Write)

DEFAULT: 0	RANGE: 0 3

Simulated Gyroscope File Read/Write (SIM_GYR_FILE_RW)

Description

`SIM_GYR_FILE_RW` works exactly like `SIM_ACC_FILE_RW` but for the gyroscope.

Use this to replay real-world gyro noise (propeller vibration) into the simulator. This is the gold standard for testing Harmonic Notch Filter settings without risking a crash.

Tuning & Behavior

- **Default Value:** 0.
- **Procedure:**
    1. Set to **2** (Write).
    2. Fly/Hover for 1 minute.
    3. Copy the generated file to your SITL directory.
    4. Set to **1** (Read) and run SITL.

SIM_IE24_ENABLE (Simulated IE24 Fuel Cell Enable)

DEFAULT: 0	RANGE: 0 1

Simulated IE24 Fuel Cell Enable (SIM_IE24_ENABLE)

Description

`SIM_IE24_ENABLE` adds a virtual Hydrogen Fuel Cell to the simulation.

This allows you to test the telemetry feedback (tank pressure, stack voltage, error states) of an Intelligent Energy 2.4kW unit without needing the expensive hardware.

Tuning & Behavior

- **0:** Disabled.
- **1:** Enabled.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





SIM_IE24_ERROR (Simulated IE24 Error Code)

DEFAULT: 0	RANGE: 0 4294967295

Simulated IE24 Error Code (SIM_IE24_ERROR)

Description

`SIM_IE24_ERROR` simulates a hardware failure.

By setting this to a valid IE24 error bitmask, you can verify that the GCS displays the correct warning message (e.g. "Low Hydrogen Pressure" or "Stack Over-temp").

Tuning & Behavior

- **Default Value:** 0.

SIM_IE24_STATE (Simulated IE24 State)

DEFAULT: 0	RANGE: 0 10

Simulated IE24 State (SIM_IE24_STATE)

Description

`SIM_IE24_STATE` overrides the fuel cell logic.

- **0:** Auto (Normal operation).
- **1:** Starting.
- **2:** Running.
- **4:** Fault.

Tuning & Behavior

- **Default Value:** 0.

SIM_IMUT1_ENABLE (Simulated IMU 1 Thermal Enable)

DEFAULT: 0	RANGE: 0 1

Simulated IMU 1 Thermal Enable (SIM_IMUT1_ENABLE)

Description

`SIM_IMUT1_ENABLE` activates the simulated heat-up process for the first IMU.

When enabled, the IMU's reported temperature will follow the curve defined by `SIM_IMUT_START`, `END`, and `TCONST`. This is useful for verifying that your `INS_TCAL1_` parameters (learned during a real temp cal) correctly remove the bias drift in the simulator.

Tuning & Behavior

- **0:** Disabled (IMU stays at constant temperature).
- **1:** Enabled.

SIM_IMUT1_TMAX (Simulation IMU Temperature Max)

DEFAULT: 80	RANGE: -50 80	UNITS: degC

Simulation IMU Temperature Max (SIM_IMUT1_TMAX)

Description

`SIM_IMUT1_TMAX` defines the upper bound for the IMU thermal model.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





SIM_IMUT1_TMIN (Simulation IMU Temperature Min)

DEFAULT: -10	RANGE: -50 80	UNITS: degC

Simulation IMU Temperature Min (SIM_IMUT1_TMIN)

Description

`SIM_IMUT1_TMIN` defines the lower bound for simulating IMU thermal noise and bias shifts.

SIM_IMUT2_ENABLE (Enable pullup after altitude wait)

DEFAULT: 0	RANGE: null	UNITS: null

Enable pullup after altitude wait

Note: This parameter functions identically to SIM_IMUT1_ENABLE.

SIM_IMUT2_TMAX (Temperature calibration max)

DEFAULT: 70	RANGE: -70 80	UNITS: degC

Temperature calibration max

Note: This parameter functions identically to SIM_IMUT1_TMAX.

SIM_IMUT2_TMIN (Temperature calibration min)

DEFAULT: 0	RANGE: -70 80	UNITS: degC

Temperature calibration min

Note: This parameter functions identically to SIM_IMUT1_TMIN.

SIM_IMUT3_ENABLE (Enable pullup after altitude wait)

DEFAULT: 0	RANGE: null	UNITS: null

Enable pullup after altitude wait

Note: This parameter functions identically to SIM_IMUT1_ENABLE.

SIM_IMUT3_TMAX (Temperature calibration max)

DEFAULT: 70	RANGE: -70 80	UNITS: degC

Temperature calibration max

Note: This parameter functions identically to SIM_IMUT1_TMAX.

SIM_IMUT3_TMIN (Temperature calibration min)

DEFAULT: 0	RANGE: -70 80	UNITS: degC

Temperature calibration min

Note: This parameter functions identically to SIM_IMUT1_TMIN.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





SIM_IMUT4_ENABLE (Enable pullup after altitude wait)

DEFAULT: 0	RANGE: null	UNITS: null

Enable pullup after altitude wait

Note: This parameter functions identically to SIM_IMUT1_ENABLE.

SIM_IMUT4_TMAX (Temperature calibration max)

DEFAULT: 70	RANGE: -70 80	UNITS: degC

Temperature calibration max

Note: This parameter functions identically to SIM_IMUT1_TMAX.

SIM_IMUT4_TMIN (Temperature calibration min)

DEFAULT: 0	RANGE: -70 80	UNITS: degC

Temperature calibration min

Note: This parameter functions identically to SIM_IMUT1_TMIN.

SIM_IMUT5_ENABLE (Enable pullup after altitude wait)

DEFAULT: 0	RANGE: null	UNITS: null

Enable pullup after altitude wait

Note: This parameter functions identically to SIM_IMUT1_ENABLE.

SIM_IMUT5_TMAX (Temperature calibration max)

DEFAULT: 70	RANGE: -70 80	UNITS: degC

Temperature calibration max

Note: This parameter functions identically to SIM_IMUT1_TMAX.

SIM_IMUT5_TMIN (Temperature calibration min)

DEFAULT: 0	RANGE: -70 80	UNITS: degC

Temperature calibration min

Note: This parameter functions identically to SIM_IMUT1_TMIN.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





SIM_IMUT_END (Simulated IMU Final Temperature)

DEFAULT: 45	RANGE: -40 100	UNITS: degC

Simulated IMU Final Temperature (SIM_IMUT_END)

Description

`SIM_IMUT_END` sets the target temperature that the flight controller will eventually reach after it has been running for a while (due to CPU heat and internal electronics).

Tuning & Behavior

- **Default Value:** 45 °C.

SIM_IMUT_FIXED (Simulated IMU Fixed Temperature)

DEFAULT: 0	RANGE: -40 100	UNITS: degC

Simulated IMU Fixed Temperature (SIM_IMUT_FIXED)

Description

`SIM_IMUT_FIXED` forces the virtual IMU to stay at a specific temperature.

If set to a non-zero value, the IMU temperature will not follow the curve defined by `SIM_IMUT_START` and `SIM_IMUT_END`. This is useful for testing sensor bias at a specific operating point without waiting for the heat-up cycle.

Tuning & Behavior

- **0:** Disabled (Use dynamic thermal curve).
- **Non-Zero:** Fixed temperature in Celsius.

SIM_IMUT_START (Simulated IMU Start Temperature)

DEFAULT: 25	RANGE: -40 100	UNITS: degC

Simulated IMU Start Temperature (SIM_IMUT_START)

Description

`SIM_IMUT_START` defines the "Ambient Temperature" when the simulator starts.

SITL can simulate the heating of the flight controller over time. This parameter sets the baseline. If you want to test how your drone handles a sub-zero cold start, you can set this to **-10**.

Tuning & Behavior

- **Default Value:** 25 °C.
- **Usage:** Use this in conjunction with `SIM_IMUT_END` and `SIM_IMUT_TCONST` to create a thermal warm-up profile.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





SIM_IMUT_TCONST (Simulated IMU Thermal Time Constant)

DEFAULT: 300	RANGE: 1 3600	UNITS: s

Simulated IMU Thermal Time Constant (SIM_IMUT_TCONST)

Description

`SIM_IMUT_TCONST` determines how fast the flight controller heats up.

It defines the "Slope" of the temperature curve. A value of 300 seconds (5 minutes) means the drone will reach its steady-state temperature fairly slowly, mimicking a real flight controller on a workbench.

The Mathematics

The temperature $T(t)$ at time t is modeled as an exponential approach:

$$T(t) = T_start + (T_end - T_start) \times (1 - e^{-t/\tau})$$

Where τ is the `SIM_IMUT_TCONST`.

SIM_IMU_COUNT (Simulated IMU Count)

DEFAULT: 2	RANGE: 1 3

Simulated IMU Count (SIM_IMU_COUNT)

Description

`SIM_IMU_COUNT` sets the number of virtual internal sensors.

Modern flight controllers often have 2 or 3 IMUs for safety. This parameter allows SITL to mimic that redundancy. You can then use `SIM_ACCELx_FAIL` to kill one sensor and see how the autopilot handles it.

Tuning & Behavior

- **Default Value:** 2.
- **Recommendation:** Set to **2 or 3** to test EKF lane switching logic.
- **Reboot Required:** Yes.

SIM_IMU_ORIENT (Simulated IMU Orientation)

DEFAULT: 0	RANGE: 0 38

Simulated IMU Orientation (SIM_IMU_ORIENT)

Description

`SIM_IMU_ORIENT` sets the physical "Heading" of the virtual sensors.

By default (0), the IMU is aligned with the drone's nose. If you want to test how ArduPilot handles an IMU that is mounted upside down or rotated 90 degrees, you can change this parameter and then verify that you can still calibrate and fly successfully using `AHRS_ORIENTATION`.

Tuning & Behavior

- **Default Value:** 0 (None).
- **Recommendation:** Leave at **0** for standard tests. Use values like **8** (Roll 180) to test an inverted flight controller.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





SIM_IMU_POS (Simulation IMU Position Offset)

DEFAULT: 0	RANGE: 0 5	UNITS: m

Simulation IMU Position Offset (SIM_IMU_POS)

Description

`SIM_IMU_POS` simulates an IMU that is not mounted exactly at the center of gravity (COG).

When a drone rotates, any sensor away from the COG experiences "Centripetal Acceleration." If ArduPilot doesn't know about this offset (configured via `INS_POS_X/Y/Z`), the EKF will get confused and report incorrect velocity. This parameter allows you to test the compensation for these "Lever Arm" effects.

SIM_INIT_ALT_OFS (Simulation Initial Alt Offset)

DEFAULT: 0	RANGE: -100 100	UNITS: m

Simulation Initial Alt Offset (SIM_INIT_ALT_OFS)

Description

`SIM_INIT_ALT_OFS` allows you to start the simulation with the drone at a specific height above the ground. Useful for testing in-air resets or high-altitude drop launches.

SIM_INIT_LAT_OFS (Simulated Initial Latitude Offset)

DEFAULT: 0	RANGE: -1 1	UNITS: deg

Simulated Initial Latitude Offset (SIM_INIT_LAT_OFS)

Description

`SIM_INIT_LAT_OFS` allows you to test what happens if the "map" shifts under the drone.

It adds a constant offset to the GPS position reported by the simulator, effectively moving the world origin.

Tuning & Behavior

- **Default Value:** 0.

SIM_INIT_LON_OFS (Simulated Initial Longitude Offset)

DEFAULT: 0	RANGE: -1 1	UNITS: deg

Simulated Initial Longitude Offset (SIM_INIT_LON_OFS)

Description

`SIM_INIT_LON_OFS` shifts the GPS longitude origin. See `SIM_INIT_LAT_OFS`.

Tuning & Behavior

- **Default Value:** 0.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





SIM_INS_THR_MIN (Simulated INS Noise Throttle Min)

DEFAULT: 0.1	RANGE: 0 1	UNITS: %

Simulated INS Noise Throttle Min (SIM_INS_THR_MIN)

Description

`SIM_INS_THR_MIN` defines when the "Shaking" starts.

Real drones vibrate more when the motors are spinning fast. This parameter tells SITL to only add motor-induced noise when the throttle is above a certain percentage (e.g. 10%). On the ground at zero throttle, the IMUs will remain clean.

Tuning & Behavior

- **Default Value:** 0.1 (10% throttle).

SIM_JSON_MASTER (Simulated JSON Master)

DEFAULT: 0	RANGE: 0 4294967295

Simulated JSON Master (SIM_JSON_MASTER)

Description

`SIM_JSON_MASTER` enables the "External Physics" bridge.

Instead of SITL calculating the drone's movements, it can send the motor outputs to a high-fidelity simulator (like **Gazebo**, **AirSim**, or **Webots**) and get the position/sensor data back over a JSON socket. This parameter sets the IP address of that external simulator.

Tuning & Behavior

- **Default Value:** 127.0.0.1 (Localhost).
- **Usage:** Only active if the SITL build includes the JSON backend.

SIM_LED_LAYOUT (Simulation LED Layout)

DEFAULT: 0	RANGE: 0 10

Simulation LED Layout (SIM_LED_LAYOUT)

Description

`SIM_LED_LAYOUT` defines the number and position of virtual status LEDs in SITL. This is primarily used for verifying LED driver logic and patterns (like those for the SkyViper or ProfiLED) in a simulated environment.

SIM_LOOP_DELAY (Simulated Loop Delay)

DEFAULT: 0	RANGE: 0 100	UNITS: ms

Simulated Loop Delay (SIM_LOOP_DELAY)

Description

`SIM_LOOP_DELAY` simulates a "Slow Processor."

If you set this to 5ms, every single logic loop in the autopilot will take 5ms longer than usual. This is a great way to test the safety margins of your loop rate and see at what point the drone becomes unflyable due to latency.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





SIM_MAG1_DEVID (Simulated Magnetometer 1 Device ID)

DEFAULT: 0	RANGE: 0 16777216

Simulated Magnetometer 1 Device ID (SIM_MAG1_DEVID)

Description

`SIM_MAG1_DEVID` sets the unique hardware identifier for the virtual compass.

This is usually read-only or managed by `SIM_MAG_SAVE_IDS`. However, developers can manually set this to test how the OS handles specific sensor types (e.g. simulating a specific I2C address conflict).

Tuning & Behavior

- **Default Value:** 0 (Auto-generated).

SIM_MAG1_DIA (Simulation Magnetometer 1 Diagonal)

DEFAULT: 0	RANGE: -0.2 0.2

Simulation Magnetometer 1 Diagonal (SIM_MAG1_DIA)

Description

`SIM_MAG1_DIA` simulates "Soft Iron" interference.

Soft iron effects are caused by materials that distort magnetic fields but are not themselves magnets (like iron or nickel). This distorts the magnetic "sphere" into an "ellipsoid." ArduPilot's advanced calibration routine calculates these diagonal scaling factors to fix this distortion.

SIM_MAG1_FAIL (Simulation Compass 1 Failure)

DEFAULT: 0	RANGE: 0 1

Simulation Compass 1 Failure (SIM_MAG1_FAIL)

Description

`SIM_MAG1_FAIL` kills the first compass sensor.

- **0: Healthy.**
- **1: Failed.** The sensor stops reporting data.

Tuning & Behavior

- **Testing:** Use this to verify that your drone successfully fails over to the second compass (if available) or switches to GSF (GPS-based yaw) without crashing.

SIM_MAG1_ODI (Simulation Magnetometer 1 Off-Diagonal)

DEFAULT: 0	RANGE: -0.2 0.2

Simulation Magnetometer 1 Off-Diagonal (SIM_MAG1_ODI)

Description

`SIM_MAG1_ODI` simulates complex magnetic distortions.

While `SIM_MAG1_DIA` scales the X, Y, and Z axes independently, `ODI` (Off-Diagonal) terms simulate how a magnetic field in one axis (e.g. X) can bleed into another axis (e.g. Y). This represents a highly non-linear or asymmetric magnetic environment.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





SIM_MAG1_OFS (Simulation Magnetometer 1 Offset)

DEFAULT: 0	RANGE: -100 100	UNITS: mGauss

Simulation Magnetometer 1 Offset (SIM_MAG1_OFS)

Description

`SIM_MAG1_OFS` simulates "Hard Iron" interference.

Hard iron effects are caused by static magnetic fields on the drone (like a magnetized screw or the metal frame). This offsets the magnetic reading in a constant direction. This is what you are calibrating when you perform the "Compass Dance."

Tuning & Behavior

- **Default Value:** 0.
- **Testing:** Set to **50**. The drone will report "Compass Variance" until you perform a compass calibration in SITL.

SIM_MAG1_ORIENT (Simulated Magnetometer 1 Orientation)

DEFAULT: 0	RANGE: 0 38

Simulated Magnetometer 1 Orientation (SIM_MAG1_ORIENT)

Description

`SIM_MAG1_ORIENT` sets the physical rotation of the virtual compass.

If you simulate a GPS/Compass module mounted with the arrow pointing backwards (Yaw 180), you must set this parameter to match.

Tuning & Behavior

- **Default Value:** 0 (None).

SIM_MAG1_SCALING (Simulated Magnetometer 1 Scaling)

DEFAULT: 1.0	RANGE: 0.5 1.5

Simulated Magnetometer 1 Scaling (SIM_MAG1_SCALING)

Description

`SIM_MAG1_SCALING` simulates a mis-calibrated compass.

By setting this to 1.1 or 0.9, you can test if the "Compass Motel" (onboard calibration) or the EKF can learn and correct for the scale factor error.

Tuning & Behavior

- **Default Value:** 1.0.

SIM_MAG2_DEVID (Airspeed ID)

DEFAULT: 0	RANGE: null	UNITS: null

Airspeed ID

Note: This parameter functions identically to SIM_MAG1_DEVID.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





SIM_MAG2_DIA (Simulation Magnetometer 2 Diagonal)

DEFAULT: 0	RANGE: -0.2 0.2

Simulation Magnetometer 2 Diagonal (SIM_MAG2_DIA)

Description

`SIM_MAG2_DIA` simulates a distorted magnetic field on the secondary compass.

See SIM_MAG1_DIA for details.

SIM_MAG2_FAIL (MAG2 Failure)

DEFAULT: 0	RANGE: null	UNITS: null

MAG2 Failure

Note: This parameter functions identically to SIM_MAG1_FAIL.

SIM_MAG2_ODI (Simulation Magnetometer 2 Off-Diagonal)

DEFAULT: 0	RANGE: -0.2 0.2

Simulation Magnetometer 2 Off-Diagonal (SIM_MAG2_ODI)

Description

`SIM_MAG2_ODI` simulates complex magnetic twisting on the secondary compass.

See SIM_MAG1_ODI for details.

SIM_MAG2_OFS (Simulation Magnetometer 2 Offset)

DEFAULT: 0	RANGE: -100 100	UNITS: mGauss

Simulation Magnetometer 2 Offset (SIM_MAG2_OFS)

Description

`SIM_MAG2_OFS` simulates hard-iron interference on the secondary compass.

This is useful for testing the EKF's ability to handle inconsistent compass data. If Compass 1 is perfect and Compass 2 has a massive offset, the EKF should (ideally) reject Compass 2 or learn the offset.

Tuning & Behavior

- **Default Value:** 0.
- **Testing:** Set to **100** to simulate a magnetized screw near the second GPS puck.

SIM_MAG2_ORIENT (Camera Orientation)

DEFAULT: AC_PRECLAND_ORIENT_DEFAULT	RANGE: null	UNITS: null

Camera Orientation

Note: This parameter functions identically to SIM_MAG1_ORIENT.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





SIM_MAG2_SCALING (RPM scaling)

DEFAULT: 1.0f	RANGE: null	UNITS: null

RPM scaling

Note: This parameter functions identically to SIM_MAG1_SCALING.

SIM_MAG3_DEVID (Airspeed ID)

DEFAULT: 0	RANGE: null	UNITS: null

Airspeed ID

Note: This parameter functions identically to SIM_MAG1_DEVID.

SIM_MAG3_DIA ()

DEFAULT: 0	RANGE: null

Note: This parameter configures instance 3. It functions identically to SIM_MAG1_DIA.

SIM_MAG3_FAIL (MAG3 Failure)

DEFAULT: 0	RANGE: null	UNITS: null

MAG3 Failure

Note: This parameter functions identically to SIM_MAG1_FAIL.

SIM_MAG3_ODI ()

DEFAULT: 0	RANGE: null

Note: This parameter configures instance 3. It functions identically to SIM_MAG1_ODI.

SIM_MAG3_OFS (Simulation Magnetometer 3 Offset)

DEFAULT: 0	RANGE: -100 100	UNITS: mGauss

Simulation Magnetometer 3 Offset (SIM_MAG3_OFS)

Description

`SIM_MAG3_OFS` simulates hard-iron interference on the third compass.

See SIM_MAG1_OFS for details.

SIM_MAG3_ORIENT (Camera Orientation)

DEFAULT: AC_PRECLAND_ORIENT_DEFAULT	RANGE: null	UNITS: null

Camera Orientation

Note: This parameter functions identically to SIM_MAG1_ORIENT.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





SIM_MAG3_SCALING (RPM scaling)

DEFAULT: 1.0f	RANGE: null	UNITS: null

RPM scaling

Note: This parameter functions identically to SIM_MAG1_SCALING.

SIM_MAG4_DEVID (Airspeed ID)

DEFAULT: 0	RANGE: null	UNITS: null

Airspeed ID

Note: This parameter functions identically to SIM_MAG1_DEVID.

SIM_MAG5_DEVID (Airspeed ID)

DEFAULT: 0	RANGE: null	UNITS: null

Airspeed ID

Note: This parameter functions identically to SIM_MAG1_DEVID.

SIM_MAG6_DEVID (Airspeed ID)

DEFAULT: 0	RANGE: null	UNITS: null

Airspeed ID

Note: This parameter functions identically to SIM_MAG1_DEVID.

SIM_MAG7_DEVID (Airspeed ID)

DEFAULT: 0	RANGE: null	UNITS: null

Airspeed ID

Note: This parameter functions identically to SIM_MAG1_DEVID.

SIM_MAG8_DEVID (Airspeed ID)

DEFAULT: 0	RANGE: null	UNITS: null

Airspeed ID

Note: This parameter functions identically to SIM_MAG1_DEVID.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





SIM_MAG_ALY (Simulation Magnetic Anomaly)

DEFAULT: 0	RANGE: -100 100	UNITS: mGauss

Simulation Magnetic Anomaly (SIM_MAG_ALY)

Description

`SIM_MAG_ALY` simulates flying over a large iron object (like a bridge or underground pipe).

Unlike `MAG_OFS`, which is attached to the drone, `MAG_ALY` is a disturbance in the "World" that only affects the drone when it is in a specific location. Useful for testing the EKF's resilience to sudden, external heading shifts.

SIM_MAG_ALY_HGT (Simulated Magnetic Anomaly Height)

DEFAULT: 0	RANGE: 0 1000	UNITS: m

Simulated Magnetic Anomaly Height (SIM_MAG_ALY_HGT)

Description

`SIM_MAG_ALY_HGT` creates a virtual "Magnetic Distortion Field" at a specific altitude.

This is useful for simulating flying near a large steel structure (like a bridge or tower). You can test if the EKF correctly rejects the bad compass data or if the drone starts to toilet-bowl.

Tuning & Behavior

- **Default Value:** 0.

SIM_MAG_DELAY (Simulation Compass Delay)

DEFAULT: 0	RANGE: 0 100	UNITS: ms

Simulation Compass Delay (SIM_MAG_DELAY)

Description

`SIM_MAG_DELAY` simulates slow compass updates or high-latency I2C buses.

Real compasses often sample slower than the EKF's loop rate. This parameter tests if the EKF can correctly align delayed compass samples with the corresponding IMU poses.

SIM_MAG_MOT (Simulation Compass Motor Interference)

DEFAULT: 0	RANGE: 0 1

Simulation Compass Motor Interference (SIM_MAG_MOT)

Description

`SIM_MAG_MOT` simulates "CompassMot" effects.

In real drones, the high-current wires going to the motors produce magnetic fields that can bias the compass. As you increase throttle, the compass heading "twists." This parameter allows you to simulate this effect to test the efficacy of the CompassMot calibration.

Tuning & Behavior

- **Default Value:** 0.
- **Testing:** Set to **0.5**. Perform a full throttle punch in Loiter. The drone will likely rotate (yaw) unintentionally as the compass drifts under load.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





SIM_MAG_NOISE (Simulated Magnetometer Noise)

DEFAULT: 0	RANGE: 0 100	UNITS: mGauss

Simulated Magnetometer Noise (SIM_MAG_NOISE)

Description

`SIM_MAG_NOISE` adds random jitter to the compass readings.

Real compasses are very clean sensors, but they are affected by surrounding electronics. This parameter simulates that background electromagnetic noise.

Tuning & Behavior

- **Default Value:** 0.
- **Recommendation:** Set to **5-10 mGauss** for realistic testing.

SIM_MAG_RND (Simulated Magnetometer Noise)

DEFAULT: 0	RANGE: 0 1	UNITS: Gauss

Simulated Magnetometer Noise (SIM_MAG_RND)

Description

`SIM_MAG_RND` adds static fuzz to the compass readings.

Tuning & Behavior

- **Default Value:** 0.
- **Testing:** Increase this to see how much noise the EKF can tolerate before triggering a "Compass Variance" error.

SIM_MAG_SAVE_IDS (Simulated Magnetometer Save IDs)

DEFAULT: 0	RANGE: 0 1

Simulated Magnetometer Save IDs (SIM_MAG_SAVE_IDS)

Description

`SIM_MAG_SAVE_IDS` makes the virtual compasses persistent.

Normally, every time you restart SITL, it generates new random "Device IDs" for the sensors. This forces you to re-calibrate the compass every time. If you enable this, the simulator saves the IDs, allowing you to keep your calibration data.

Tuning & Behavior

- **0:** Disabled (New IDs on boot).
- **1:** Enabled (Save IDs).

SIM_ODOM_ENABLE (Simulation Odometry Enable)

DEFAULT: 0	RANGE: 0 1

Simulation Odometry Enable (SIM_ODOM_ENABLE)

Description

`SIM_ODOM_ENABLE` activates a virtual Visual Odometry (VO) sensor.

This simulates a camera-based system (like an Intel Realsense) that tracks the drone's position by looking at the ground. Useful for testing autonomous navigation in GPS-denied environments without needing a full Vicon setup.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





SIM_OH_MASK (SIM-on-Hardware Output Mask)

DEFAULT: 0	RANGE: 0 65535

SIM-on-Hardware Output Mask (SIM_OH_MASK)

Description

`SIM_OH_MASK` allows "Hardware-in-the-Loop" behavior without external software.

If you flash the SITL binary onto a real Pixhawk ("Sim on Hardware"), the physics run on the CPU, but the servo outputs are normally disabled. This mask enables specific physical pins so you can drive real servos or motors while flying a virtual drone.

Tuning & Behavior

- **Default Value:** 0 (All outputs disabled).
- **Bit 0 (1):** Enable Servo 1.
- **Bit 1 (2):** Enable Servo 2.

SIM_OH_RELAY_MSK (SIM-on-Hardware Relay Mask)

DEFAULT: 0	RANGE: 0 255

SIM-on-Hardware Relay Mask (SIM_OH_RELAY_MSK)

Description

`SIM_OH_RELAY_MSK` enables physical relay pins during "Sim on Hardware."

Tuning & Behavior

- **Default Value:** 0.

SIM_OPOS_ALT (Simulated Original Position (Altitude))

DEFAULT: 584.0	RANGE: 0 10000	UNITS: m

Simulated Original Position (Altitude) (SIM_OPOS_ALT)

Description

`SIM_OPOS_ALT` sets the spawn height Above Mean Sea Level (AMSL).

Tuning & Behavior

- **Default Value:** 584.0 m (Canberra elevation).
- **Important:** If you change Lat/Lng to a coastal city but leave this at 584m, your drone will spawn 584m in the air!

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





SIM_OPOS_HDG (Simulated Original Position (Heading))

DEFAULT: 353.0	RANGE: 0 360	UNITS: deg

Simulated Original Position (Heading) (SIM_OPOS_HDG)

Description

`SIM_OPOS_HDG` sets the direction the vehicle faces when it spawns.

- **0:** North
- **90:** East
- **180:** South
- **270:** West

Tuning & Behavior

- **Default Value:** 353.0 degrees.
- **Recommendation:** Align this with the "virtual runway" at your test site to make takeoff easier.

SIM_OPOS_LAT (Simulated Original Position (Latitude))

DEFAULT: -35.363261	RANGE: -90 90	UNITS: deg

Simulated Original Position (Latitude) (SIM_OPOS_LAT)

Description

`SIM_OPOS_LAT` sets the home location where your drone spawns in the simulator.

- **Default:** CMAC (Canberra Model Aircraft Club) in Australia.
- **Usage:** Change this to your local flying field's coordinates to test missions in a familiar environment.

Tuning & Behavior

- **Default Value:** -35.363261 (Canberra).
- **Recommendation:** Use a tool like Mission Planner's "Set Home" feature, which updates this parameter automatically.

SIM_OPOS_LNG (Simulated Original Position (Longitude))

DEFAULT: 149.165230	RANGE: -180 180	UNITS: deg

Simulated Original Position (Longitude) (SIM_OPOS_LNG)

Description

`SIM_OPOS_LNG` sets the starting longitude for the simulation.

Tuning & Behavior

- **Default Value:** 149.165230 (Canberra).

SIM_OSD_COLUMNS (Simulated OSD Columns)

DEFAULT: 30	RANGE: 1 50

Simulated OSD Columns (SIM_OSD_COLUMNS)

Description

`SIM_OSD_COLUMNS` defines the width of the virtual OSD screen.

Tuning & Behavior

- **Default Value:** 30 columns.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





SIM_OSD_ROWS (Simulated OSD Rows)

DEFAULT: 16	RANGE: 1 30

Simulated OSD Rows (SIM_OSD_ROWS)

Description

`SIM_OSD_ROWS` defines the height of the virtual OSD screen.

This is used for testing MSP (MultiWii Serial Protocol) OSD layouts in SITL. By changing this, you can simulate different goggle resolutions or display types.

Tuning & Behavior

- **Default Value:** 16 rows.
- **Common Settings:** 13 or 16 rows for standard analog/digital systems.

SIM_PARA_ENABLE (Simulated Parachute Enable)

DEFAULT: 0	RANGE: 0 1

Simulated Parachute Enable (SIM_PARA_ENABLE)

Description

`SIM_PARA_ENABLE` adds a drag-inducing chute to the physics model.

When the parachute is "deployed" (via `SIM_PARA_PIN`), the drone's drag coefficient increases massively, causing it to drift down slowly rather than crash. This is vital for testing `CHUTE_` parameters and failsafe triggers.

Tuning & Behavior

- **0:** Disabled.
- **1:** Enabled.

SIM_PARA_PIN (Simulated Parachute Pin)

DEFAULT: 0	RANGE: 0 100

Simulated Parachute Pin (SIM_PARA_PIN)

Description

`SIM_PARA_PIN` connects the virtual parachute to an output channel.

Set this to match your `CHUTE_SERVO_ON` value. When that pin goes high, the simulator deploys the chute.

SIM_PIN_MASK (Simulated GPIO Pin Mask)

DEFAULT: 0	RANGE: 0 65535

Simulated GPIO Pin Mask (SIM_PIN_MASK)

Description

`SIM_PIN_MASK` tells the simulator which "Digital Pins" it should pretend exist.

This allows you to test things like Relay control, Camera triggering, or specialized hardware switches without physical wires.

Tuning & Behavior

- **Default Value:** 0.
- **Recommendation:** Match the pins you have configured in your `RELAY_PIN` or `CAM_FEEDBACK_PIN` parameters.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





SIM_PLD_ALT_LMT (Simulation Precision Landing Max Altitude)

DEFAULT: 15	RANGE: 0 100	UNITS: m

Simulation Precision Landing Max Altitude (SIM_PLD_ALT_LMT)

Description

`SIM_PLD_ALT_LMT` simulates the "Range Limit" of your Precision Landing sensor (e.g. IR-Lock).

Real sensors have a limited effective range. If the drone is too high, the camera or sensor cannot see the beacon on the ground. This parameter tells the simulator to stop reporting valid target data when the drone climbs above this altitude.

Tuning & Behavior

- **Default:** 15 meters.
- **Usage:** Set this to match the datasheet of your real sensor (e.g. IR-Lock MarkOne is typically effective up to ~15m in daylight).
- **Testing:** Fly above this limit in SITL to verify that your failsafe or "Lost Target" logic engages correctly.

SIM_PLD_DIST_LMT (Simulation Precision Landing Lateral Range)

DEFAULT: 10	RANGE: 0 100	UNITS: m

Simulation Precision Landing Lateral Range (SIM_PLD_DIST_LMT)

Description

`SIM_PLD_DIST_LMT` simulates the "Field of View" (FOV) constraint of your Precision Landing sensor.

Cameras and IR sensors can only see a certain width. If the drone is too far to the side of the target, the beacon will fall out of the camera's frame. This parameter sets that lateral limit in meters.

Tuning & Behavior

- **Default:** 10 meters.
- **Physics:** This creates a virtual "cone" or cylinder of detection above the target. If the horizontal distance between the drone and the SIM_PLD_LAT/LON target is greater than this value, the sensor reports "No Target."

SIM_PLD_ENABLE (Simulated Precision Landing Enable)

DEFAULT: 0	RANGE: 0 1

Simulated Precision Landing Enable (SIM_PLD_ENABLE)

Description

`SIM_PLD_ENABLE` adds a virtual infrared beacon or visual target to the world.

This is the primary way to test "Precision Landing" (`PLND_ENABLE`) in the simulator. It simulates the data from an IR-Lock camera or a downward-facing visual system.

Tuning & Behavior

- **Default Value:** 0 (Disabled).
- **1:** Enabled. The target will spawn at the coordinates defined by `SIM_PLD_LAT/LON`.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





SIM_PLD_HEIGHT (Simulated Precision Landing Height)

DEFAULT: 0	RANGE: 0 100	UNITS: m

Simulated Precision Landing Height (SIM_PLD_HEIGHT)

Description

`SIM_PLD_HEIGHT` defines the vertical position of the landing target (e.g., a rooftop, a landing platform, or a pole-mounted beacon) in SITL.

This allows you to test precision landing algorithms on elevated platforms rather than just on the flat ground.

Tuning & Behavior

- **Default:** 0 (Target is at ground level).
- **Usage:** Set to the desired height in meters. The virtual sensor (IR-Lock) will report the target's position based on this height.

SIM_PLD_LAT (Simulated Precision Landing Latitude)

DEFAULT: 0	RANGE: -90 90	UNITS: deg

Simulated Precision Landing Latitude (SIM_PLD_LAT)

Description

`SIM_PLD_LAT` defines the north/south coordinate of the landing target beacon used in the **Precision Landing** simulation.

This allows you to place the virtual beacon at a specific location on the map to test how the drone detects and navigates toward it from different approach angles.

Tuning & Behavior

- **Default:** 0.
- **Usage:** Enter the exact latitude of the target.
- **Integration:** Must be used in conjunction with SIM_PLD_LON and SIM_PLD_HEIGHT.

SIM_PLD_LON (Simulated Precision Landing Longitude)

DEFAULT: 0	RANGE: -180 180	UNITS: deg

Simulated Precision Landing Longitude (SIM_PLD_LON)

Description

`SIM_PLD_LON` defines the east/west coordinate of the landing target beacon used in the **Precision Landing** simulation.

This allows you to place the virtual beacon at a specific location on the map to test how the drone detects and navigates toward it from different approach angles.

Tuning & Behavior

- **Default:** 0.
- **Usage:** Enter the exact longitude of the target.
- **Integration:** Must be used in conjunction with SIM_PLD_LAT and SIM_PLD_HEIGHT.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





SIM_PLD_OPTIONS (Simulation Precision Landing Options)

DEFAULT: 0	RANGE: 0 1

Simulation Precision Landing Options (SIM_PLD_OPTIONS)

Description

`SIM_PLD_OPTIONS` allows developers to inject realistic faults into the simulated Precision Landing sensor.

- **Bit 0 (1):** Enable Glitches. Randomly drops the target valid flag or shifts the target position momentarily to test the EKF's resilience to bad data.

Tuning & Behavior

- **Default:** 0 (Perfect sensor).
- **Recommendation:** Set to **1** when tuning your landing logic to ensure the drone doesn't make sudden, dangerous jerks if the sensor data becomes noisy.

SIM_PLD_ORIENT (Simulation Precision Landing Orientation)

DEFAULT: 25	RANGE: 0 38

Simulation Precision Landing Orientation (SIM_PLD_ORIENT)

Description

`SIM_PLD_ORIENT` tells the simulator which way the virtual camera is pointing.

By default, Precision Landing cameras (like IR-Lock) are mounted pointing straight down. However, some advanced setups use a gimballed camera or a fixed forward-facing camera for approach. This parameter allows you to test those configurations in SITL.

Tuning & Behavior

- **25 (Default):** Pitch 270 (Straight Down).
- **0:** None (Straight Forward).
- **Usage:** Must match the `PLND_ORIENT` parameter in your vehicle's configuration to ensure the EKF correctly interprets the target's position relative to the drone.

SIM_PLD_RATE (Simulation Precision Landing Update Rate)

DEFAULT: 100	RANGE: 0 200	UNITS: Hz

Simulation Precision Landing Update Rate (SIM_PLD_RATE)

Description

`SIM_PLD_RATE` controls how many times per second the virtual Precision Landing sensor (e.g., IR-Lock or MAVLink beacon) sends data to the autopilot in SITL.

A higher rate provides smoother tracking during the final descent, while a lower rate can be used to simulate the latency or jitter of real-world sensors.

Tuning & Behavior

- **Default:** 100 Hz.
- **Realistic Testing:** If you are using a sensor known to be slow (e.g. some vision systems), try lowering this to **10 Hz** or **20 Hz** to see how the landing algorithm handles the increased lag.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





SIM_PLD_SHIP (Simulation Landing Ship)

DEFAULT: 0	RANGE: 0 1

Simulation Landing Ship (SIM_PLD_SHIP)

Description

`SIM_PLD_SHIP` activates a moving landing platform.

This is the primary tool for testing Maritime operations. When enabled, SITL simulates a ship that pitches, rolls, and travels through the water. You can then test if your drone successfully performs a precision landing on a moving deck using the PLND_ENABLED logic.

SIM_PLD_TYPE (Simulated Precision Landing Type)

DEFAULT: 0	RANGE: 0 5

Simulated Precision Landing Type (SIM_PLD_TYPE)

Description

`SIM_PLD_TYPE` selects the protocol the virtual landing sensor uses.

- **1:** IR-Lock.
- **3:** MAVLink (for GCS-based landing targets).

Tuning & Behavior

- **Default Value:** 0.
- **Recommendation:** Use **1 (IR-Lock)** for most autonomous landing tests.

SIM_PLD_YAW (Simulation Precision Landing Target Yaw)

DEFAULT: 0	RANGE: -180 180	UNITS: deg

Simulation Precision Landing Target Yaw (SIM_PLD_YAW)

Description

`SIM_PLD_YAW` sets the heading of the virtual landing beacon or pad in SITL.

This is useful for testing sensors that report target orientation (like some 3D vision systems) or for verifying that the drone correctly aligns itself with the landing pad during the final touch-down phase.

Tuning & Behavior

- **Default:** 0 (Pointing North).
- **Usage:** Set to any heading from -180 to 180 degrees.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





SIM_RATE_HZ (Simulation Loop Rate)

DEFAULT: 1200	RANGE: 50 2000	UNITS: Hz

Simulation Loop Rate (SIM_RATE_HZ)

Description

`SIM_RATE_HZ` sets the "Frame Rate" of the physics world.

The flight controller runs its main loop at a certain speed (e.g. 400Hz for Copter, 50Hz for Plane). The simulator must run faster than the flight controller to provide smooth sensor data.

- **1200 (Default):** Good for most multirotors.
- **400:** Sufficient for Plane or Rover.

Tuning & Behavior

- **Constraint:** Must be a multiple of the flight controller loop rate to avoid aliasing artifacts.

SIM_RC_CHANCOUNT (Simulated RC Channel Count)

DEFAULT: 16	RANGE: 1 32

Simulated RC Channel Count (SIM_RC_CHANCOUNT)

Description

`SIM_RC_CHANCOUNT` sets the capacity of the virtual radio receiver.

Tuning & Behavior

- **Default Value:** 16.
- **Recommendation:** Keep at **16** for most standard setups. Use higher values (up to 32) for testing complex Lua scripts or multi-gimbal setups.

SIM_RC_FAIL (Simulated RC Failure)

DEFAULT: 0	RANGE: 0 2

Simulated RC Failure (SIM_RC_FAIL)

Description

`SIM_RC_FAIL` simulates your radio receiver losing power or signal.

- **0: Normal.** RC signal is healthy.
- **1: No Pulses.** The simulator stops sending any RC data to the autopilot. Mimics a disconnected SBus cable.
- **2: Failsafe Values.** Channels stay at neutral, except throttle which drops to 950µs.

Tuning & Behavior

- **Default Value:** 0.
- **Recommendation:** Use this in flight to verify that your `FS_THR_ENABLE` or `RC_FS_ACTION` is working correctly.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





SIM_RFL_OPTS (Simulation RealFlight Options)

DEFAULT: 1	RANGE: 0 7

Simulation RealFlight Options (SIM_RFL_OPTS)

Description

`SIM_RFL_OPTS` provides advanced control over the communication between ArduPilot SITL and the **RealFlight** simulator (FlightAxis).

- **Bit 0 (1): Reset Position.** If enabled, ArduPilot will command RealFlight to reset the aircraft position to the runway whenever the SITL instance is rebooted or reset.
- **Bit 1 (2): Use MAVLink State.**

Tuning & Behavior

- **Default:** 1 (Reset Position enabled).
- **Note:** This is only relevant if you are using ArduPilot SITL in "RealFlight mode" (e.g. running `sim_vehicle.py -v ArduPlane -f realflight`).

SIM_RICH_CTRL (RichenPower Control Pin)

DEFAULT: -1	RANGE: -1 16

SIM_RICH_CTRL: RichenPower Control Pin

Description

Pin RichenPower is connected to.

Values

- **Range:** -1 to 16
- **Default:** -1 (Disabled)

Description

This parameter configures the SITL (Simulator) to emulate a **RichenPower Hybrid Generator**.

- **Function:** It specifies the Sim-Pin used to simulate the control signal sent to the generator.
- **Usage:** Set to **8** (simulated UART or PWM pin) to enable the generator simulation logic, allowing you to test the generator drivers and failsafes without hardware.

SIM_RICH_ENABLE (Simulated Rich Environment Enable)

DEFAULT: 0	RANGE: 0 1

Simulated Rich Environment Enable (SIM_RICH_ENABLE)

Description

`SIM_RICH_ENABLE` turns on the "Chaos Monkey."

It enables a suite of randomized sensor glitches, timing jitters, and environmental disturbances designed to stress-test the autopilot's failsafe logic. It is primarily used by developers to find bugs in error handling code.

Tuning & Behavior

- **0:** Disabled.
- **1:** Enabled.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





SIM_SAIL_TYPE (Simulation Sailboat Type)

DEFAULT: 0	RANGE: 0 5

Simulation Sailboat Type (SIM_SAIL_TYPE)

Description

`SIM_SAIL_TYPE` tells the physics engine which kind of wind-powered boat to simulate.

- **0: Mono-hull.** Standard sailboat.
- **1: Catamaran.**
- **2: Wingsail.** High-efficiency rigid sail.

SIM_SB_ALT_TARG (Simulated StratoBlimp Target Altitude)

DEFAULT: 1000	RANGE: 0 40000	UNITS: m

Simulated StratoBlimp Target Altitude (SIM_SB_ALT_TARG)

Description

`SIM_SB_ALT_TARG` sets the "Zero Buoyancy" height for the StratoBlimp simulation. At this altitude, the blimp will naturally float without using its motors.

Tuning & Behavior

- **Default Value:** 1000 m.

SIM_SB_ARM_LEN (Simulation Blimp Arm Length)

DEFAULT: 3.6	RANGE: 0 100	UNITS: m

Simulation Blimp Arm Length (SIM_SB_ARM_LEN)

Description

`SIM_SB_ARM_LEN` defines the physical geometry of the simulated **StratoBlimp**.

It sets the length of the lever arm between the blimp's center of mass and its propulsion units. This distance determines how much torque is generated by the motors, directly affecting the blimp's turning rate (Yaw) and its ability to pitch.

Tuning & Behavior

- **Default:** 3.6 meters.
- **Physics:** A longer arm increases the rotational inertia but also increases the control authority (leverage).
- **Usage:** Match this to the physical dimensions of your blimp design for an accurate simulation.

SIM_SB_CLMB_RT (Simulation Blimp Target Climb Rate)

DEFAULT: 5	RANGE: 0 50	UNITS: m/s

Simulation Blimp Target Climb Rate (SIM_SB_CLMB_RT)

Description

`SIM_SB_CLMB_RT` defines the vertical speed the simulated **StratoBlimp** attempts to maintain while climbing toward its target altitude.

This parameter simulates the buoyancy-to-drag balance of the blimp's envelope. A higher value means the blimp has more lift or less aerodynamic drag in the vertical axis.

Tuning & Behavior

- **Default:** 5 m/s.
- **Physics:** This value is used by the simulator's simplified physics model to calculate the vertical force needed to achieve the target rate.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





SIM_SB_COL (Simulation Blimp Center of Lift)

DEFAULT: 2.54	RANGE: 0 10	UNITS: m

Simulation Blimp Center of Lift (SIM_SB_COL)

Description

`SIM_SB_COL` defines the physical offset between the buoyant center of the blimp's envelope (where it is pulled UP) and its center of gravity (where it is pulled DOWN).

In a stable blimp, the center of lift is usually high above the center of gravity (the gondola). This distance creates a "Pendulum Effect" that naturally keeps the blimp level. Increasing this value makes the blimp more stable but slower to pitch.

Tuning & Behavior

- **Default:** 2.54 meters.
- **Physics:** A larger offset results in a stronger restoring torque when the blimp is tilted.
- **Warning:** Setting this to 0 will make the blimp neutrally stable in pitch/roll, meaning it will stay at whatever angle it is pushed to.

SIM_SB_DRAG_FWD (Simulation Blimp Forward Drag)

DEFAULT: 0.27	RANGE: 0 5.0

Simulation Blimp Forward Drag (SIM_SB_DRAG_FWD)

Description

`SIM_SB_DRAG_FWD` defines how "Slippery" the blimp is when flying straight ahead.

A lower value means the blimp is more streamlined and will reach a higher top speed for a given motor thrust. A higher value simulates a blunter shape or more surface friction.

Tuning & Behavior

- **Default:** 0.27.
- **Physics:** Drag force is calculated as $F_d = \frac{1}{2}\rho v^2 C_dA$. This parameter represents the combined C_dA term (Coefficient of Drag $\times$ Frontal Area) for the forward direction.

SIM_SB_DRAG_SIDE (Simulation Blimp Lateral Drag)

DEFAULT: 0.5	RANGE: 0 5.0

Simulation Blimp Lateral Drag (SIM_SB_DRAG_SIDE)

Description

`SIM_SB_DRAG_SIDE` simulates the air resistance the blimp encounters when moving sideways or drifting.

Because blimps are long and cigar-shaped, their side profile (Lateral Area) is much larger than their front profile. This means the drag from the side should be significantly higher than the forward drag. This parameter is critical for modeling how the blimp handles crosswinds.

Tuning & Behavior

- **Default:** 0.5.
- **Physics:** This value is typically 2x to 5x higher than `SIM_SB_DRAG_FWD`.
- **Effect:** If this value is too low, the simulated blimp will "slide" sideways too easily in turns.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





SIM_SB_DRAG_UP (Simulation Blimp Vertical Drag)

DEFAULT: 0.4	RANGE: 0 5.0

Simulation Blimp Vertical Drag (SIM_SB_DRAG_UP)

Description

`SIM_SB_DRAG_UP` simulates the air resistance the blimp encounters when climbing or descending.

Because of the large horizontal surface area of a blimp's envelope (top and bottom), vertical drag is substantial. This parameter determines the terminal velocity for ascent (due to buoyancy) and descent.

Tuning & Behavior

- **Default:** 0.4.
- **Physics:** This value balances against `SIM_SB_FLR` (Free Lift Rate). A higher drag coefficient means the blimp will ascend slower for the same amount of positive buoyancy.

SIM_SB_FLR (Simulation Blimp Free Lift Rate)

DEFAULT: 0.12	RANGE: 0 1.0

Simulation Blimp Free Lift Rate (SIM_SB_FLR)

Description

`SIM_SB_FLR` simulates the "Positive Buoyancy Bias" often used in stratospheric balloon missions.

Many high-altitude airships are designed to be slightly lighter than air so they naturally ascend without using motor power. This parameter defines how much extra lift is available.

- **0.0:** Perfectly neutrally buoyant (will float at current altitude).
- **0.12 (Default):** 12% excess lift. The blimp will ascend automatically.

Tuning & Behavior

- **Default:** 0.12.
- **Physics:** This excess lift must be counteracted by downward thrust from the motors if the blimp needs to hold altitude or descend.

SIM_SB_HMASS (Simulation Blimp Helium Mass)

DEFAULT: 13.54	RANGE: 0 1000	UNITS: kg

Simulation Blimp Helium Mass (SIM_SB_HMASS)

Description

`SIM_SB_HMASS` simulates the weight of the lifting gas itself.

While helium is lighter than air, it still has mass. This parameter, combined with the envelope volume (implicit in the simulation's buoyancy logic), allows the physics engine to calculate the net lift and inertia of the airship.

Tuning & Behavior

- **Default:** 13.54 kg.
- **Physics:** A larger mass of helium implies a larger envelope volume, which increases both lift and aerodynamic drag.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





SIM_SB_MASS (Simulation Blimp Mass)

DEFAULT: 80	RANGE: 1 500	UNITS: kg

Simulation Blimp Mass (SIM_SB_MASS)

Description

`SIM_SB_MASS` defines the weight of the virtual blimp used in the **StratoBlimp** simulation.

This parameter directly affects the physics engine, influencing how much buoyancy (Helium) is needed to stay airborne and how the blimp reacts to motor thrust and wind.

SIM_SB_MOI_PITCH (Simulation Blimp Pitch Inertia)

DEFAULT: 3050	RANGE: 100 10000	UNITS: kg*m^2

Simulation Blimp Pitch Inertia (SIM_SB_MOI_PITCH)

Description

`SIM_SB_MOI_PITCH` defines how hard it is to tilt the blimp's nose up or down.

Because blimps are very long, their mass is distributed far from the center of rotation. This results in a very high pitch inertia. This parameter simulates that "heavy" feel, meaning it takes significant thrust to start pitching the nose up, and equally significant thrust to stop it once it's moving.

Tuning & Behavior

- **Default:** 3050 $kg \cdot m^2$.
- **Physics:** This value is typically much higher than the Roll Inertia (`SIM_SB_MOI_ROLL`) due to the elongated shape of the hull.
- **Effect:** If this value is too low, the simulated blimp will be able to perform impossible aerobatic loops.

SIM_SB_MOI_ROLL (Simulation Blimp Roll Inertia)

DEFAULT: 1400	RANGE: 100 10000	UNITS: kg*m^2

Simulation Blimp Roll Inertia (SIM_SB_MOI_ROLL)

Description

`SIM_SB_MOI_ROLL` defines how hard it is to tilt the blimp side-to-side.

A blimp with a high roll inertia is very stable and resists being rocked by turbulence, but it takes a lot of force to bank into a turn. A low roll inertia makes the blimp feel "twitchy" and unstable.

Tuning & Behavior

- **Default:** 1400 $kg \cdot m^2$.
- **Physics:** This value is derived from the mass distribution of the envelope and gondola. Since most of the mass is concentrated low (in the gondola), the roll inertia is typically lower than the pitch or yaw inertia.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





SIM_SB_MOI_YAW (Simulation Blimp Yaw Inertia)

DEFAULT: 2800	RANGE: 100 10000	UNITS: kg*m^2

Simulation Blimp Yaw Inertia (SIM_SB_MOI_YAW)

Description

`SIM_SB_MOI_YAW` defines how hard it is to turn the blimp left or right.

Like pitch inertia, yaw inertia is high for blimps because of their length. This parameter simulates the "resistance to turning" that pilot's feel. A high value means the blimp will be slow to start turning, but will also coast through the turn and be slow to stop spinning once the motors are centered.

Tuning & Behavior

- **Default:** 2800 $kg \cdot m^2$.
- **Physics:** This value dictates the responsiveness of the heading control loop.
- **Effect:** If this value is too high, the blimp will feel "sluggish" and may overshoot its target heading. If too low, it will turn unrealistically fast.

SIM_SB_MOT_ANG (Simulation Blimp Max Motor Angle)

DEFAULT: 20	RANGE: 0 90	UNITS: deg

Simulation Blimp Max Motor Angle (SIM_SB_MOT_ANG)

Description

`SIM_SB_MOT_ANG` defines the range of motion for the vectored thrust motors on the simulated **StratoBlimp**.

Unlike a standard drone, many blimps use motors that can tilt up or down to provide vertical lift or horizontal thrust. This parameter sets the maximum mechanical tilt angle that the simulator will model.

Tuning & Behavior

- **Default:** 20 degrees.
- **Usage:** Set this to match the physical tilt range of your blimp's motor mounts.
- **Physics:** This determines how much of the motor's thrust can be directed vertically for altitude control versus horizontally for forward speed.

SIM_SB_MOT_THST (Simulation Blimp Max Motor Thrust)

DEFAULT: 145	RANGE: 0 1000	UNITS: N

Simulation Blimp Max Motor Thrust (SIM_SB_MOT_THST)

Description

`SIM_SB_MOT_THST` defines the "Power" of the simulated propulsion system.

It represents the raw force (in Newtons) that a single motor/propeller unit generates when running at full throttle. This determines the blimp's maximum speed and its ability to fight against simulated wind.

Tuning & Behavior

- **Default:** 145 N.
- **Conversion:** 1 Newton $\approx$ 102 grams of thrust.
- **Usage:** Measure or calculate the peak thrust of your real motors and enter the equivalent Newtons here.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY



SIM_SB_WVANE (Simulation Blimp Weathervane Offset)

DEFAULT: 0.3	RANGE: -10 10	UNITS: m

Simulation Blimp Weathervane Offset (SIM_SB_WVANE)

Description

`SIM_SB_WVANE` determines how strongly the **StratoBlimp** naturally points into the wind.

It represents the horizontal distance between the Center of Gravity (CoG) and the Center of Pressure (CoP) for side winds.

- **Positive Value:** Center of Drag is behind the Center of Mass. The blimp is stable and will naturally turn its nose into the wind (like a weather vane).
- **Negative Value:** Center of Drag is in front of the Center of Mass. The blimp is unstable and will try to fly tail-first.

Tuning & Behavior

- **Default:** 0.3 meters.
- **Physics:** A larger positive value means stronger directional stability but makes it harder for the motors to turn the blimp out of the wind.

SIM_SB_YAW_RT (Simulation Blimp Max Yaw Rate)

DEFAULT: 60	RANGE: 0 360	UNITS: deg/s

Simulation Blimp Max Yaw Rate (SIM_SB_YAW_RT)

Description

`SIM_SB_YAW_RT` sets the "Turning Speed" limit for the simulated **StratoBlimp**.

It defines how many degrees per second the blimp will rotate when the motor outputs are fully differential (full left/right stick). This allows developers to simulate the sluggish turning characteristics of large blimps or the highly maneuverable response of small electric airships.

Tuning & Behavior

- **Default:** 60 deg/s.
- **Physics:** This parameter determines the effective yaw torque produced by the motors in the simulator's physics model.

SIM_SERVO_DELAY (Simulated Servo Delay)

DEFAULT: 0	RANGE: 0 0.1	UNITS: s

Simulated Servo Delay (SIM_SERVO_DELAY)

Description

`SIM_SERVO_DELAY` simulates the processing lag of digital servos or PWM signal propagation.

Tuning & Behavior

- **Default Value:** 0.
- **Testing:** Set to **0.05** (50ms). You will likely see the drone start to wobble as the control loop becomes unstable due to the lag.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





SIM_SERVO_FILTER (Simulation Servo Filter)

DEFAULT: 0	RANGE: 0 100	UNITS: Hz

Simulation Servo Filter (SIM_SERVO_FILTER)

Description

`SIM_SERVO_FILTER` adds damping to the simulated control surfaces.

Real servos and mechanical linkages don't snap to a new position instantly; they have physical inertia. This parameter simulates that "Smoothness" by applying a frequency-based filter to the output.

- **0: Disabled.**
- **Value (Hz):** The cutoff frequency. Lower values make the servos move slower and "mushier."

Tuning & Behavior

- **Default:** 0.
- **Usage:** Set to **20** or **30** to simulate a typical high-quality servo.
- **Integration:** Works in combination with SIM_SERVO_SPEED.

SIM_SERVO_SPEED (Simulated Servo Speed)

DEFAULT: 0.11	RANGE: 0.01 1.0	UNITS: s/60deg

Simulated Servo Speed (SIM_SERVO_SPEED)

Description

`SIM_SERVO_SPEED` defines how fast the virtual servos move.

Real servos aren't instant. If you have a slow servo on a racing quad, the PID loop will oscillate because the motor doesn't reach the target speed fast enough. This parameter allows you to simulate that physical delay.

Tuning & Behavior

- **Default Value:** 0.11 s/60° (Standard hobby servo).
- **High Performance:** Set to **0.05** for fast brushless servos.
- **Large Drones:** Set to **0.20** for heavy-duty, slow servos.

SIM_SF_JS_AXIS1 (Simulation SFML Joystick Axis 1)

DEFAULT: 0	RANGE: 0 7

Simulation SFML Joystick Axis 1 (SIM_SF_JS_AXIS1)

Description

`SIM_SF_JS_AXIS1` (and AXIS2 through AXIS8) allows you to rebind the physical axes of your controller to the expected inputs in the simulator.

By default, Axis 1 is typically mapped to the X-axis (Roll/Aileron) of the joystick.

Tuning & Behavior

- **Axis Mapping:** The value corresponds to the SFML `sf::Joystick::Axis` enum (X, Y, Z, R, U, V, PovX, PovY).
- **Configuration:** If your controls are swapped (e.g. throttle is on the wrong stick), use these parameters to realign them.

SIM_SF_JS_AXIS2 ()

DEFAULT: sf::Joystick::Axis::Y	RANGE: null

Note: This parameter functions identically to SIM_SF_JS_AXIS1.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





SIM_SF_JS_AXIS3 ()

DEFAULT: sf::Joystick::Axis::Z	RANGE: null

Note: This parameter functions identically to SIM_SF_JS_AXIS1.

SIM_SF_JS_AXIS4 ()

DEFAULT: sf::Joystick::Axis::U	RANGE: null

Note: This parameter functions identically to SIM_SF_JS_AXIS1.

SIM_SF_JS_AXIS5 ()

DEFAULT: sf::Joystick::Axis::V	RANGE: null

Note: This parameter functions identically to SIM_SF_JS_AXIS1.

SIM_SF_JS_AXIS6 ()

DEFAULT: sf::Joystick::Axis::R	RANGE: null

Note: This parameter functions identically to SIM_SF_JS_AXIS1.

SIM_SF_JS_AXIS7 ()

DEFAULT: sf::Joystick::Axis::PovX	RANGE: null

Note: This parameter functions identically to SIM_SF_JS_AXIS1.

SIM_SF_JS_AXIS8 ()

DEFAULT: sf::Joystick::Axis::PovY	RANGE: null

Note: This parameter functions identically to SIM_SF_JS_AXIS1.

SIM_SF_JS_STICK (Simulation SFML Joystick Index)

DEFAULT: 0	RANGE: 0 7

Simulation SFML Joystick Index (SIM_SF_JS_STICK)

Description

`SIM_SF_JS_STICK` selects which physical joystick or gamepad connected to your computer is used to control the simulated vehicle in SITL.

This is used by the SFML (Simple and Fast Multimedia Library) backend of SITL. If you have multiple controllers connected, you may need to change this ID to select the correct one.

Tuning & Behavior

- **Default:** 0.
- **Usage:** Set to 0 for the first detected controller, 1 for the second, and so on.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





SIM_SHIP_DSIZE (Simulated Ship Deck Size)

DEFAULT: 10	RANGE: 1 100	UNITS: m

Simulated Ship Deck Size (SIM_SHIP_DSIZE)

Description

`SIM_SHIP_DSIZE` defines the dimensions of the helipad.

This affects the visual model and the collision detection. If the drone lands outside this area, it will fall into the "Water."

Tuning & Behavior

- **Default Value:** 10 m.

SIM_SHIP_ENABLE (Simulated Ship Enable)

DEFAULT: 0	RANGE: 0 1

Simulated Ship Enable (SIM_SHIP_ENABLE)

Description

`SIM_SHIP_ENABLE` places a large, moving ship in the simulation.

This is primarily used for testing "Ship Landing" logic. The ship has a helipad deck and moves along a pre-defined path or at a constant velocity.

Tuning & Behavior

- **0:** Disabled.
- **1:** Enabled.

SIM_SHIP_OFS (Simulation Ship Position Offset)

DEFAULT: 0	UNITS: m

Simulation Ship Position Offset (SIM_SHIP_OFS)

Description

`SIM_SHIP_OFS` allows you to move the simulated ship's starting point.

When simulating a moving platform (like a ship or truck), it is useful to be able to place it at a specific distance from the drone's "Home" or takeoff location.

- **Format:** A vector (X, Y, Z) in meters.
- **X:** North.
- **Y:** East.
- **Z:** Altitude (Negative is down/below home).

Tuning & Behavior

- **Default:** 0,0,0.
- **Usage:** Set `Z` to a positive value (e.g. 5m) to simulate the ship's deck being higher than the water level or ground level.

SIM_SHIP_PSIZE (Simulated Ship Pad Size)

DEFAULT: 5	RANGE: 1 50	UNITS: m

Simulated Ship Pad Size (SIM_SHIP_PSIZE)

Description

`SIM_SHIP_PSIZE` controls the visual "H" or circle on the deck.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





SIM_SHIP_SPEED (Simulated Ship Speed)

DEFAULT: 0	RANGE: 0 15	UNITS: m/s

Simulated Ship Speed (SIM_SHIP_SPEED)

Description

`SIM_SHIP_SPEED` sets how fast the target vessel moves.

Tuning & Behavior

- **Default Value:** 0 m/s.
- **Recommendation:** Set to **5-10 m/s** to test the drone's ability to track and land on a moving deck.

SIM_SHIP_SYSID (Simulated Ship System ID)

DEFAULT: 0	RANGE: 0 255

Simulated Ship System ID (SIM_SHIP_SYSID)

Description

`SIM_SHIP_SYSID` gives the ship a "Voice."

If set to a non-zero value (e.g. 20), the ship will broadcast its position via MAVLink `GLOBAL_POSITION_INT` messages on the simulated network. This allows the drone to track the ship as a dynamic "Follow Me" target or a moving home point.

Tuning & Behavior

- **Default Value:** 0 (Silent).

SIM_SHOVE_TIME (Simulated Shove Time)

DEFAULT: 0	RANGE: 0 10000	UNITS: ms

Simulated Shove Time (SIM_SHOVE_TIME)

Description

`SIM_SHOVE_TIME` determines if the "Shove" is a punch or a push.

If set to 0, the velocity change is instantaneous (impulse). If set to a value (e.g. 2000 ms), the force is applied over that duration, simulating a sustained wind gust or thruster firing.

Tuning & Behavior

- **Default Value:** 0.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





SIM_SHOVE_X (Simulated Shove X)

DEFAULT: 0	RANGE: -30 30	UNITS: m/s

Simulated Shove X (SIM_SHOVE_X)

Description

`SIM_SHOVE_X` kicks the drone forward or backward.

When you change this parameter (e.g. via MAVProxy), the simulator instantly adds this velocity to the vehicle's state. It is used to test the position controller's ability to recover from a sudden disturbance.

Tuning & Behavior

- **Default Value:** 0.
- **Usage:** Set to **5** to simulate being hit by a 5 m/s gust or object from behind.

SIM_SHOVE_Y (Simulation Shove Y)

DEFAULT: 0	RANGE: -100 100	UNITS: m/s^2

Simulation Shove Y (SIM_SHOVE_Y)

Description

`SIM_SHOVE_Y` hits the drone from the side.

Unlike `SIM_WIND`, which is an aerodynamic force, `SHOVE` is a direct acceleration/force applied to the body.

Tuning & Behavior

- **Default Value:** 0.
- **Testing:** Set to **10** for 1 second, then back to **0**. This tests the drone's ability to recover from a hard collision or extreme gust.

SIM_SHOVE_Z (Simulation Shove Z)

DEFAULT: 0	RANGE: -100 100	UNITS: m/s^2

Simulation Shove Z (SIM_SHOVE_Z)

Description

`SIM_SHOVE_Z` simulates a sudden vertical impact or force.

- **Positive Value:** Pushes the drone **Down**.
- **Negative Value:** Pushes the drone **Up**.

Tuning & Behavior

- **Testing:** Set to **20** for a fraction of a second to simulate a heavy object being dropped on the drone or a sudden downburst. Test the position controller's ability to recover the target altitude.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





SIM_SLUP_DRAG (Slung Payload Drag Coefficient)

DEFAULT: 1.0	RANGE: 0 10

Slung Payload Drag Coefficient (SIM_SLUP_DRAG)

Description

`SIM_SLUP_DRAG` determines how "Aerodynamic" the payload is.

A high drag coefficient (e.g. 5.0) simulates a bulky object (like a large box) that will be pushed backward by the wind as the drone flies forward. A low value (e.g. 0.1) simulates a streamlined weight that hangs mostly vertical even at high speeds.

Tuning & Behavior

- **Default Value:** 1.0.

SIM_SLUP_ENABLE (Slung Payload Sim Enable)

DEFAULT: 0	RANGE: 0 1

Slung Payload Sim Enable (SIM_SLUP_ENABLE)

Description

`SIM_SLUP_ENABLE` adds a swinging physical weight beneath the drone.

This is used for testing helicopter or multirotor "Slung Load" operations. The simulator calculates the pendulum physics, including the mass of the payload, the length of the line, and the drag on the object. The swinging payload exerts forces back on the drone, which can cause significant instability if not tuned correctly.

Tuning & Behavior

- **0:** Disabled.
- **1:** Enabled.
- **Physics:** The payload is modeled as a point mass on a tether. It uses the vehicle's Earth-frame position and velocity to calculate tension and centripetal forces.

SIM_SLUP_LINELEN (Slung Payload Line Length)

DEFAULT: 30.0	RANGE: 1 100	UNITS: m

Slung Payload Line Length (SIM_SLUP_LINELEN)

Description

`SIM_SLUP_LINELEN` sets the distance between the drone and the payload.

Tuning & Behavior

- **Default Value:** 30 m.
- **Effect:** Longer lines result in slower pendulum oscillations but can lead to very large displacements. Shorter lines react much more quickly and can impart sharp "Jerk" forces to the flight controller.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY

SIM_SLUP_SYSID (Slung Payload MAVLink System ID)

DEFAULT: 2	RANGE: 0 255

Slung Payload MAVLink System ID (SIM_SLUP_SYSID)

Description

`SIM_SLUP_SYSID` enables the slung payload to report its own position to the GCS.

When enabled, the simulator spawns a second MAVLink "Vehicle" at the payload's location. This allows you to see the payload swinging on your GCS map or in a 3D visualizer (like Mission Planner's HUD).

Tuning & Behavior

- **Default Value:** 2.
- **Note:** The payload broadcasts heartbeats as a **MAV_TYPE_AIRSHIP** with component ID **MAV_COMP_ID_USER11**.

SIM_SLUP_WEIGHT (Slung Payload Weight)

DEFAULT: 1.0	RANGE: 0 15	UNITS: kg

Slung Payload Weight (SIM_SLUP_WEIGHT)

Description

`SIM_SLUP_WEIGHT` sets the mass of the underslung object.

Tuning & Behavior

- **Default Value:** 1.0 kg.
- **Effect:** Increasing the weight increases the "Pendulum Effect." A heavy payload can pull the drone into a state where it can no longer maintain attitude or altitude if it starts swinging excessively.

SIM_SONAR_GLITCH (Simulation Sonar Glitch)

DEFAULT: 0	RANGE: -5 5	UNITS: m

Simulation Sonar Glitch (SIM_SONAR_GLITCH)

Description

`SIM_SONAR_GLITCH` simulates a "bad bounce" or sensor error.

Rangefinders often report incorrect distances when flying over grass or water. This parameter allows you to inject sudden jumps in the reported altitude to test the EKF's rejection logic or the terrain following safety limits.

SIM_SONAR_POS (Simulation Sonar Position)

DEFAULT: 0	RANGE: 0 5	UNITS: m

Simulation Sonar Position (SIM_SONAR_POS)

Description

`SIM_SONAR_POS` defines the mounting location of the simulated rangefinder. Useful for testing terrain following and precise landing algorithms.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





SIM_SONAR_RND (Simulated Sonar Noise)

DEFAULT: 0	RANGE: 0 5	UNITS: m

Simulated Sonar Noise (SIM_SONAR_RND)

Description

`SIM_SONAR_RND` adds Gaussian noise to the rangefinder distance.

Tuning & Behavior

- **Default Value:** 0 m.
- **Recommendation:** Set to **0.05** (5cm) for realistic testing.

SIM_SONAR_ROT (Simulation Sonar Rotation)

DEFAULT: 0	RANGE: 0 35

Simulation Sonar Rotation (SIM_SONAR_ROT)

Description

`SIM_SONAR_ROT` sets the direction the simulated rangefinder is pointing.

- **25:** Down (Standard for altimeters).
- **0:** Forward (Standard for obstacle avoidance).

SIM_SONAR_SCALE (Simulated Sonar Scale)

DEFAULT: 12.1212	RANGE: 0 100	UNITS: m/V

Simulated Sonar Scale (SIM_SONAR_SCALE)

Description

`SIM_SONAR_SCALE` simulates the calibration of an analog rangefinder (like the Maxbotix EZ series).

It defines how many meters of distance are represented by 1 Volt of output.

Tuning & Behavior

- **Default Value:** 12.1212 (Standard Maxbotix).
- **Testing:** Change this to simulate an uncalibrated sensor and verify your `RNGFND1_SCALING` parameter can correct it.

SIM_SPEEDUP (Simulation Speedup)

DEFAULT: -1	RANGE: -1 100

Simulation Speedup (SIM_SPEEDUP)

Description

`SIM_SPEEDUP` allows you to run the simulation faster than real-time.

- **1:** Real-time (1 second = 1 second).
- **10:** 10x speed. Useful for flying long missions quickly.
- **-1 (Default):** Run as fast as the computer CPU allows.

Tuning & Behavior

- **Default Value:** -1.
- **Note:** If your computer is slow, the simulation might run *slower* than real-time even if you set this to 10.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





SIM_SPR_ENABLE (Simulated Sprayer Enable)

DEFAULT: 0	RANGE: 0 1

Simulated Sprayer Enable (SIM_SPR_ENABLE)

Description

`SIM_SPR_ENABLE` adds a liquid tank to the drone.

When the sprayer is active, the simulator slowly decreases the total mass of the drone, mimicking the fluid being pumped out. This allows you to test if your tuning holds up as the payload weight changes.

Tuning & Behavior

- **0:** Disabled.
- **1:** Enabled.

SIM_SPR_PUMP (Simulated Sprayer Pump Rate)

DEFAULT: 0	RANGE: 0 10	UNITS: L/s

Simulated Sprayer Pump Rate (SIM_SPR_PUMP)

Description

`SIM_SPR_PUMP` sets how fast the tank empties.

Tuning & Behavior

- **Default Value:** 0.

SIM_SPR_SPIN (Simulated Sprayer Spinner Torque)

DEFAULT: 0	RANGE: 0 1	UNITS: Nm

Simulated Sprayer Spinner Torque (SIM_SPR_SPIN)

Description

`SIM_SPR_SPIN` simulates the reaction torque from a spinning atomizer disk.

Some crop sprayers use high-speed spinning disks to atomize the liquid. These act like small gyroscopes and can impart a yaw torque on the drone. This parameter helps verify that the yaw controller can handle the disturbance.

Tuning & Behavior

- **Default Value:** 0.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





SIM_TA_ENABLE (Simulated ToneAlarm Enable)

DEFAULT: 0	RANGE: 0 1

Simulated ToneAlarm Enable (SIM_TA_ENABLE)

Description

`SIM_TA_ENABLE` activates the simulated hardware buzzer.

In a real flight controller, the ToneAlarm is the speaker that plays the "Happy" arming tune, "Sad" battery low tones, and error chirps. Enabling this in SITL allows developers to hear these tones while testing in the simulator.

The Engineer's View

Defined in `SIM_ToneAlarm.cpp`.
When enabled, the SITL backend pipes the audio frequency commands to the host computer's audio system (or logs them for verification). It mimics the behavior of the `AP_Notify` ToneAlarm backend.

Tuning & Behavior

- **Default Value:** 0 (Silent).
- **1:** Tones are enabled.

SIM_TEMP_BFACTOR (Simulation Barometer Temperature Factor)

DEFAULT: 0.0	RANGE: 0 5

Simulation Barometer Temperature Factor (SIM_TEMP_BFACTOR)

Description

`SIM_TEMP_BFACTOR` simulates "Thermal Altitude Drift."

Real pressure sensors (barometers) are slightly affected by temperature. As the flight controller heats up, the reported altitude might "drift" by a few meters even if the drone is sitting still. This parameter simulates that error.

Tuning & Behavior

- **Default Value:** 0 (Perfect sensor).
- **Testing:** Set to **1.2** to simulate a typical ICM-20789 sensor.

SIM_TEMP_BRD_OFF (Simulation Board Temperature Offset)

DEFAULT: 20	RANGE: 0 50	UNITS: degC

Simulation Board Temperature Offset (SIM_TEMP_BRD_OFF)

Description

`SIM_TEMP_BRD_OFF` simulates the self-heating of the flight controller electronics.

In the real world, the inside of an autopilot case is hotter than the outside air due to the heat generated by the CPU and power regulators. This parameter adds a fixed offset to the simulated ambient temperature to create the "Board Temperature" sensor reading.

Tuning & Behavior

- **Default:** 20 °C. (If ambient is 25°C, the board reports 45°C).
- **Usage:** Useful for testing temperature calibration routines (`TCAL`) or verifying fan control logic without needing a heat gun.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





SIM_TEMP_START (Simulation Start Temperature)

DEFAULT: 25.0	RANGE: -20 60	UNITS: degC

Simulation Start Temperature (SIM_TEMP_START)

Description

`SIM_TEMP_START` sets the initial temperature for SITL sensor thermal models.

ArduPilot simulates "Thermal Warmup." When the simulation starts, the sensors (IMU, Baro) begin at this temperature and gradually heat up towards the internal operating temperature of the flight controller. This is used to test temperature compensation algorithms.

Tuning & Behavior

- **Default Value:** 25.0°C.
- **Testing:** Set to **0.0** to test how your drone handles a cold-start in winter conditions.

SIM_TEMP_TCONST (Simulation Temperature Time Constant)

DEFAULT: 30.0	RANGE: 1 1000	UNITS: s

Simulation Temperature Time Constant (SIM_TEMP_TCONST)

Description

`SIM_TEMP_TCONST` defines how fast the flight controller heats up in the simulator.

A lower value means the board reaches its operating temperature very quickly. A higher value simulates a large, heavy flight controller that takes a long time to stabilize.

SIM_TERRAIN (Simulation Terrain Enable)

DEFAULT: 1	RANGE: 0 1

Simulation Terrain Enable (SIM_TERRAIN)

Description

`SIM_TERRAIN` turns on the "Ground."

When enabled, SITL downloads SRTM terrain data (just like the real flight controller) and uses it to calculate the ground height at the drone's current location. This is essential for testing terrain following, rangefinder altimetry, and crash detection.

- **0: Disabled.** The world is flat at altitude 0.
- **1: Enabled.** The ground height varies.

SIM_TETH_ENABLE (Simulated Tether Enable)

DEFAULT: 0	RANGE: 0 1

Simulated Tether Enable (SIM_TETH_ENABLE)

Description

`SIM_TETH_ENABLE` simulates a physical cable attached to the drone.

This is used for testing tethered power systems. The simulator calculates the weight and drag of the cable as the drone flies higher, pulling it down and affecting its stability.

Tuning & Behavior

- **0:** Disabled.
- **1:** Enabled.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





SIM_TETH_LINELEN (Simulated Tether Length)

DEFAULT: 100	RANGE: 10 1000	UNITS: m

Simulated Tether Length (SIM_TETH_LINELEN)

Description

`SIM_TETH_LINELEN` sets the maximum reach of the cable.

If the drone flies further than this distance from the tether anchor point, it will be physically jerked back by the "Rope," potentially causing a crash or oscillation.

Tuning & Behavior

- **Default Value:** 100 m.

SIM_TETH_SYSID (Simulated Tether System ID)

DEFAULT: 0	RANGE: 0 255

Simulated Tether System ID (SIM_TETH_SYSID)

Description

`SIM_TETH_SYSID` allows the simulator to spoof messages from a "Smart Tether."

Real tether stations often report cable tension and power usage via MAVLink. This parameter sets the source ID for those simulated messages.

Tuning & Behavior

- **Default Value:** 0 (Disabled).

SIM_THML_SCENARI (Simulation Thermal Scenarios)

DEFAULT: 0	RANGE: 0 5

Simulation Thermal Scenarios (SIM_THML_SCENARI)

Description

`SIM_THML_SCENARI` defines the "Thermal Map" for autonomous glider testing.

Instead of just one random thermal, this parameter selects different distributions of rising air. Useful for verifying that ArduSoar can reliably find and track lift in varied atmospheric conditions.

SIM_TIDE_DIR (Simulated Tide Direction)

DEFAULT: 0	RANGE: 0 360	UNITS: deg

Simulated Tide Direction (SIM_TIDE_DIR)

Description

`SIM_TIDE_DIR` sets the compass bearing of the water current.

- **0:** North current (pushing the boat towards South).
- **180:** South current.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





SIM_TIDE_SPEED (Simulated Tide Speed)

DEFAULT: 0	RANGE: 0 5	UNITS: m/s

Simulated Tide Speed (SIM_TIDE_SPEED)

Description

`SIM_TIDE_SPEED` simulates the "Wind of the Water."

For ArduRover (Boats) and ArduSub, this parameter adds a constant water current that pushes the vehicle in a specific direction. This is the primary way to test "Position Hold" and navigation robustness against currents.

Tuning & Behavior

- **Default Value:** 0 m/s.

SIM_TIME_JITTER (Simulated Time Jitter)

DEFAULT: 0	RANGE: 0 10000	UNITS: us

Simulated Time Jitter (SIM_TIME_JITTER)

Description

`SIM_TIME_JITTER` adds "Instability" to the simulator's clock.

On a real flight controller, the main loop usually runs at exactly 400Hz or 1000Hz. However, CPU spikes can cause small delays. This parameter allows you to simulate that behavior in SITL to ensure the EKF and PID controllers don't crash or lose stability when the timing isn't perfect.

SIM_TWIST_TIME (Simulated Twist Time)

DEFAULT: 0	RANGE: 0 10000	UNITS: ms

Simulated Twist Time (SIM_TWIST_TIME)

Description

`SIM_TWIST_TIME` controls the duration of the rotational disturbance.

Tuning & Behavior

- **Default Value:** 0.

SIM_TWIST_X (Simulated Twist X)

DEFAULT: 0	RANGE: -360 360	UNITS: deg/s

Simulated Twist X (SIM_TWIST_X)

Description

`SIM_TWIST_X` spins the drone in Roll.

Use this to test the attitude controller's ability to stop a roll tumble.

Tuning & Behavior

- **Default Value:** 0.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





SIM_TWIST_Y (Simulation Twist Y)

DEFAULT: 0	RANGE: -100 100	UNITS: deg/s^2

Simulation Twist Y (SIM_TWIST_Y)

Description

`SIM_TWIST_Y` applies an external rotation force to the pitch axis.

This is useful for simulating a mechanical failure (like a loose motor mount) or an external snag (like a payload cable pulling on the frame). It tests the attitude controller's ability to maintain level flight under constant torque.

SIM_TWIST_Z (Simulation Twist Z)

DEFAULT: 0	RANGE: -100 100	UNITS: deg/s^2

Simulation Twist Z (SIM_TWIST_Z)

Description

`SIM_TWIST_Z` applies an external rotation force to the yaw axis.

Use this to simulate a "Broken Propeller" scenario where one motor is producing more torque than the others, forcing the drone to spin.

SIM_UART_LOSS (Simulated UART Data Loss)

DEFAULT: 0	RANGE: 0 100	UNITS: %

Simulated UART Data Loss (SIM_UART_LOSS)

Description

`SIM_UART_LOSS` simulates a "Bad Connection" or noisy telemetry link.

It randomly drops the specified percentage of serial bytes. This is perfect for testing how well your GCS handles packet loss or how resilient your Serial GPS driver is to data corruption.

Tuning & Behavior

- **Default Value:** 0.
- **Testing:** Set to **5% or 10%** and observe if your MAVLink connection remains stable.

SIM_VIB_FREQ (Simulation Vibration Frequency)

DEFAULT: 0	RANGE: 0 300	UNITS: Hz

Simulation Vibration Frequency (SIM_VIB_FREQ)

Description

`SIM_VIB_FREQ` adds realistic motor noise to the IMU.

- **0:** No vibration.
- **50-200:** Typical motor/propeller frequencies.

Tuning & Behavior

- **Default Value:** 0.
- **Testing:** Essential for testing **Harmonic Notch Filters**. Set `SIM_VIB_FREQ` to 80Hz, enable `INS_HNTCH_ENABLE`, and see if the FFT or Notch filters successfully remove it from the logs.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





SIM_VIB_MOT_HMNC (Simulated Motor Harmonics)

DEFAULT: 1	RANGE: 1 5

Simulated Motor Harmonics (SIM_VIB_MOT_HMNC)

Description

`SIM_VIB_MOT_HMNC` simulates the "Overtones" of motor vibration.

In the real world, a motor doesn't just vibrate at its base frequency; it also creates noise at 2x, 3x, and 4x that frequency. This parameter tells SITL how many of these harmonics to simulate, allowing you to test if your `INS_HNTCH_HMNCS` bitmask is set correctly.

Tuning & Behavior

- **Default Value:** 1.
- **Testing:** Set to **3** and verify your Harmonic Notch can clear all three spikes in an FFT graph.

SIM_VIB_MOT_MASK (Simulated Motor Vibration Mask)

DEFAULT: 0	RANGE: 0 4294967295

Simulated Motor Vibration Mask (SIM_VIB_MOT_MASK)

Description

`SIM_VIB_MOT_MASK` selects which virtual motors are "Noisy."

By default (0), all motors contribute equally to vibration. By setting this bitmask, you can simulate a single motor with a damaged propeller to see how the EKF handles asymmetrical noise.

Tuning & Behavior

- **Default Value:** 0 (All motors).
- **Bit 0 (1):** Motor 1 only.

SIM_VIB_MOT_MAX (Simulated Motor Vibration Max Frequency)

DEFAULT: 0	RANGE: 0 500	UNITS: Hz

Simulated Motor Vibration Max Frequency (SIM_VIB_MOT_MAX)

Description

`SIM_VIB_MOT_MAX` sets the upper bound of the vibration "Whine."

Real motors vibrate at different frequencies as they spin up. This parameter allows you to test your **Harmonic Notch Filter** (`INS_HNTCH_ENABLE`) by injecting a realistic vibration spike that moves between 0 and this maximum frequency.

Tuning & Behavior

- **Default Value:** 0 (Disabled).
- **Recommendation:** Set to **150 or 200 Hz** to test your notch filter's tracking ability.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





SIM_VIB_MOT_MULT (Simulation Vibration Motor Scale)

DEFAULT: 1.0	RANGE: 0 10

Simulation Vibration Motor Scale (SIM_VIB_MOT_MULT)

Description

`SIM_VIB_MOT_MULT` controls the "Volume" of the simulated motor vibration.

While `SIM_VIB_FREQ` sets the frequency of the noise, this parameter determines how strong those vibrations are. This is used to test the resilience of the EKF and the effectiveness of internal dampening filters.

- **1.0 (Default):** Standard vibration level.
- **5.0:** Extreme vibration. Good for testing "Vibe Failsafe" logic.

Tuning & Behavior

- **Testing:** Set to **5.0** and watch the `VIBE` message in your Ground Control Station. You can verify if the drone can still maintain its position under high-stress vibration conditions.

SIM_VICON_FAIL (Simulation Vicon Failure)

DEFAULT: 0	RANGE: 0 1

Simulation Vicon Failure (SIM_VICON_FAIL)

Description

`SIM_VICON_FAIL` cuts the data feed from the external motion capture system.

This is critical for testing indoor drones. If the Vicon system crashes or the drone flies behind an obstacle, the EKF loses its primary position source. This parameter allows you to verify that the drone successfully fails over to Optical Flow or enters a safe landing mode.

- **0: Healthy.**
- **1: Failed.**

SIM_VICON_GLIT (Simulation Vicon Position Glitch)

DEFAULT: 0	RANGE: 0 10	UNITS: m

Simulation Vicon Position Glitch (SIM_VICON_GLIT)

Description

`SIM_VICON_GLIT` injects a "Jump" into the external navigation data.

When using an external motion capture system (like Vicon or OptiTrack) for indoor flight, the data is usually perfect. This parameter allows you to simulate a tracking error where the system momentarily reports the drone is in a different location.

Tuning & Behavior

- **Default:** 0 (No glitch).
- **Usage:** Set to **1.0** or **2.0** to simulate a large tracking jump.
- **Goal:** Use this to verify that the EKF (Extended Kalman Filter) correctly rejects the bad data or that the Failsafe logic engages.
- **Note:** This applies the glitch to the **Down** (Vertical) axis.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





SIM_VICON_GLIT_X (Simulation Vicon Position Glitch X)

DEFAULT: 0	RANGE: -100 100	UNITS: m

Simulation Vicon Position Glitch X (SIM_VICON_GLIT_X)

Description

`SIM_VICON_GLIT_X` injects a position error into the external vision system.

This is useful for testing how robust your indoor navigation is to sudden jumps in reported position (e.g. if the Vicon cameras lose calibration or the subject marker is partially occluded).

SIM_VICON_GLIT_Y (Simulation Vicon Position Glitch Y)

DEFAULT: 0	RANGE: -100 100	UNITS: m

Simulation Vicon Position Glitch Y (SIM_VICON_GLIT_Y)

Description

`SIM_VICON_GLIT_Y` injects a lateral position error into the simulated external vision system.

See SIM_VICON_GLIT_X for details.

SIM_VICON_GLIT_Z (Simulation Vicon Position Glitch Z)

DEFAULT: 0	RANGE: -100 100	UNITS: m

Simulation Vicon Position Glitch Z (SIM_VICON_GLIT_Z)

Description

`SIM_VICON_GLIT_Z` injects a vertical position error into the simulated external vision system.

SIM_VICON_POS (Simulation Vicon Position Offset)

DEFAULT: 0	RANGE: 0 5	UNITS: m

Simulation Vicon Position Offset (SIM_VICON_POS)

Description

`SIM_VICON_POS` defines where the "Vicon Marker" is attached to the drone.

In high-precision indoor flight, the position is often reported by an external camera system (Vicon/Optitrack). This parameter allows you to test the compensation for markers that are not mounted at the COG.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





SIM_VICON_RATE (Simulated Vicon Update Rate)

DEFAULT: 50	RANGE: 1 200	UNITS: Hz

Simulated Vicon Update Rate (SIM_VICON_RATE)

Description

`SIM_VICON_RATE` sets the data frequency of the virtual motion capture system.

Standard Vicon systems run at 50Hz to 100Hz. A higher rate provides a smoother position estimate for the EKF, while a lower rate tests how the autopilot handles sparse tracking data.

Tuning & Behavior

- **Default Value:** 50 Hz.
- **Recommendation:** Set to **100** for ultra-stable indoor loiter testing.

SIM_VICON_TMASK (Simulation Vicon Type Mask)

DEFAULT: 3	RANGE: 0 31

Simulation Vicon Type Mask (SIM_VICON_TMASK)

Description

`SIM_VICON_TMASK` controls which "Languages" the virtual motion capture system uses to talk to the autopilot.

Different external systems (Vicon, Optitrack, Marvelmind) use different MAVLink packets to report position and speed.

- **Bit 0 (1):** VISION_POSITION_ESTIMATE
- **Bit 1 (2):** VISION_SPEED_ESTIMATE
- **Bit 2 (4):** VICON_POSITION_ESTIMATE
- **Bit 3 (8):** VISION_POSITION_DELTA
- **Bit 4 (16):** ODOMETRY

Tuning & Behavior

- **Default Value:** 3 (Position + Speed estimates).

SIM_VICON_VGLI (Simulation Vicon Velocity Glitch)

DEFAULT: 0	RANGE: 0 10	UNITS: m/s

Simulation Vicon Velocity Glitch (SIM_VICON_VGLI)

Description

`SIM_VICON_VGLI` injects a false "Speed Spike" into the external navigation data.

Even if the drone is hovering perfectly still, this parameter makes the simulated motion capture system report that the drone is suddenly moving vertically at high speed. This tests how the EKF handles conflicting data (accelerometers say "stopped," Vicon says "moving").

Tuning & Behavior

- **Default:** 0 m/s.
- **Usage:** Set to **5.0** to simulate a tracking error where the system momentarily loses lock and reports a massive velocity spike.
- **Note:** This applies the glitch to the **Down** (Vertical) velocity.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





SIM_VICON_VGLI_X (Simulation Vicon Velocity Glitch X)

DEFAULT: 0	RANGE: -10 10	UNITS: m/s

Simulation Vicon Velocity Glitch X (SIM_VICON_VGLI_X)

Description

`SIM_VICON_VGLI_X` injects a velocity error into the external vision system.

This tests if the EKF can handle a Vicon system that reports a constant drift in speed (e.g. due to lighting changes or high-speed blur).

SIM_VICON_VGLI_Y (Simulation Vicon Velocity Glitch Y)

DEFAULT: 0	RANGE: -10 10	UNITS: m/s

Simulation Vicon Velocity Glitch Y (SIM_VICON_VGLI_Y)

Description

`SIM_VICON_VGLI_Y` injects a lateral velocity error into the simulated external vision system.

SIM_VICON_VGLI_Z (Simulation Vicon Velocity Glitch Z)

DEFAULT: 0	RANGE: -10 10	UNITS: m/s

Simulation Vicon Velocity Glitch Z (SIM_VICON_VGLI_Z)

Description

`SIM_VICON_VGLI_Z` injects a vertical velocity error into the simulated external vision system.

SIM_VICON_YAW (Simulation Vicon Yaw Angle)

DEFAULT: 0	RANGE: 0 360	UNITS: deg

Simulation Vicon Yaw Angle (SIM_VICON_YAW)

Description

`SIM_VICON_YAW` defines the simulated orientation of the external position source. Useful for testing if the flight controller correctly integrates absolute heading data from a motion capture system.

SIM_VICON_YAWERR (Simulation Vicon Yaw Error)

DEFAULT: 0	RANGE: -180 180	UNITS: deg

Simulation Vicon Yaw Error (SIM_VICON_YAWERR)

Description

`SIM_VICON_YAWERR` simulates a misalignment between the Vicon system's north and the drone's IMU north. This tests the EKF's ability to handle rotational offsets in external position data.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





SIM_VOLZ_ENA (Simulation Volz Servo Enable)

DEFAULT: 0	RANGE: 0 1

Simulation Volz Servo Enable (SIM_VOLZ_ENA)

Description

`SIM_VOLZ_ENA` activates the simulated driver for Volz digital servos.

Normally, SITL assumes standard PWM servos. If you enable this, ArduPilot will communicate with virtual Volz servos via a serial protocol, allowing you to test the serial driver and telemetry feedback in a simulated environment.

SIM_WAVE_AMP (Simulated Wave Amplitude)

DEFAULT: 0.5	RANGE: 0 5	UNITS: m

Simulated Wave Amplitude (SIM_WAVE_AMP)

Description

`SIM_WAVE_AMP` sets the "Roughness" of the sea.

A value of 1.0 means the peaks of the waves are 1 meter high. Large waves will cause significantly more rocking and may wash over the deck of low-profile boats.

Tuning & Behavior

- **Default Value:** 0.5 m.

SIM_WAVE_DIR (Simulated Wave Direction)

DEFAULT: 0	RANGE: 0 360	UNITS: deg

Simulated Wave Direction (SIM_WAVE_DIR)

Description

`SIM_WAVE_DIR` sets the compass bearing of the ocean swell.

- **0:** North waves (moving towards South).
- **90:** East waves.

SIM_WAVE_ENABLE (Simulated Wave Enable)

DEFAULT: 0	RANGE: 0 1

Simulated Wave Enable (SIM_WAVE_ENABLE)

Description

`SIM_WAVE_ENABLE` adds "Ocean Swell" to the simulation.

When enabled, virtual boats will pitch and roll as they ride over simulated waves. This is essential for testing "Surface Stability" and verifying that the AHRS/EKF doesn't get disoriented by constant rocking.

Tuning & Behavior

- **0:** Disabled (Glassy water).
- **1:** Enabled.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





SIM_WAVE_LENGTH (Simulated Wave Length)

DEFAULT: 10	RANGE: 1 100	UNITS: m

Simulated Wave Length (SIM_WAVE_LENGTH)

Description

`SIM_WAVE_LENGTH` sets the "Frequency" of the ocean swell.

Short wavelengths (e.g. 5m) create choppy, rapid rocking. Long wavelengths (e.g. 50m) create slow, graceful rises and falls.

Tuning & Behavior

- **Default Value:** 10 m.

SIM_WAVE_SPEED (Simulated Wave Speed)

DEFAULT: 3	RANGE: 0 20	UNITS: m/s

Simulated Wave Speed (SIM_WAVE_SPEED)

Description

`SIM_WAVE_SPEED` defines how fast the wave patterns move.

Tuning & Behavior

- **Default Value:** 3 m/s.

SIM_WIND_DIR (Simulation Wind Direction)

DEFAULT: 180	RANGE: 0 360	UNITS: deg

Simulation Wind Direction (SIM_WIND_DIR)

Description

`SIM_WIND_DIR` sets the wind bearing.

- **0:** North wind (Blows South).
- **90:** East wind (Blows West).

Tuning & Behavior

- **Default Value:** 180 (South wind).

SIM_WIND_DIR_Z (Simulated Vertical Wind Direction)

DEFAULT: 0	RANGE: -90 90	UNITS: deg

Simulated Vertical Wind Direction (SIM_WIND_DIR_Z)

Description

`SIM_WIND_DIR_Z` simulates "Thermals" or "Downbursts."

- **0:** Horizontal wind only.
- **90:** Vertical updraft.
- **-90:** Vertical downdraft.

Tuning & Behavior

- **Default Value:** 0.
- **Testing:** Set to **10 or 20** to test your drone's altitude hold against a constant thermal rise.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





SIM_WIND_SPD (Simulation Wind Speed)

DEFAULT: 0	RANGE: 0 50	UNITS: m/s

Simulation Wind Speed (SIM_WIND_SPD)

Description

`SIM_WIND_SPD` adds a constant wind to the simulation.

- **0:** Calm day.
- **10:** 10 m/s (22 mph) wind.

Tuning & Behavior

- **Default Value:** 0.
- **Testing:** Essential for tuning `WPNAV_SPEED` and checking if your drone can hold position in high winds.

SIM_WIND_T (Simulated Wind Type)

DEFAULT: 0	RANGE: 0 2

Simulated Wind Type (SIM_WIND_T)

Description

`SIM_WIND_T` models how wind speed decreases as you get closer to the ground (due to ground friction).

- **0: Square Law (Realistic).** Wind speed drops off naturally near the surface.
- **1: None.** Constant wind speed at all altitudes.
- **2: Linear.** Constant decrease from `SIM_WIND_T_ALT` down to zero.

Tuning & Behavior

- **Default Value:** 0.
- **Recommendation:** Keep at **0** for realistic landing tests.

SIM_WIND_TC (Simulated Wind Time Constant)

DEFAULT: 5	RANGE: 1 100	UNITS: s

Simulated Wind Time Constant (SIM_WIND_TC)

Description

`SIM_WIND_TC` controls how quickly the wind "Wanders" between different speeds and directions.

A small value means the wind changes instantly (very gusty). A large value means the wind changes slowly and smoothly over several seconds.

Tuning & Behavior

- **Default Value:** 5 seconds.
- **Recommendation:** Leave at **5** for realistic atmospheric behavior.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





SIM_WIND_TURB (Simulation Wind Turbulence)

DEFAULT: 0	RANGE: 0 50	UNITS: m/s

Simulation Wind Turbulence (SIM_WIND_TURB)

Description

`SIM_WIND_TURB` makes the wind gusty.

- **0:** Steady laminar flow.
- **5:** Gusts +/- 5 m/s.

Tuning & Behavior

- **Default Value:** 0.
- **Testing:** Use this to tune your Rate D-Term and Position Controller responsiveness.

SIM_WIND_T_ALT (Simulated Wind Type Altitude)

DEFAULT: 60	RANGE: 0 1000	UNITS: m

Simulated Wind Type Altitude (SIM_WIND_T_ALT)

Description

`SIM_WIND_T_ALT` defines the "Boundary Layer" height.

Above this altitude, the wind speed is constant (`SIM_WIND_SPD`). Below this altitude, the wind slows down due to ground friction (if `SIM_WIND_T` is enabled).

Tuning & Behavior

- **Default Value:** 60 m.

SIM_WIND_T_COEF (Simulated Wind Type Coefficient)

DEFAULT: 0.01	RANGE: 0 1

Simulated Wind Type Coefficient (SIM_WIND_T_COEF)

Description

`SIM_WIND_T_COEF` defines the "Roughness" of the ground.

A higher value means the ground slows the wind down more, creating a steeper wind gradient (shear) near the surface.

Tuning & Behavior

- **Default Value:** 0.01.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





SIM_WOW_PIN (Simulated Weight on Wheels Pin)

DEFAULT: -1	RANGE: -1 100

Simulated Weight on Wheels Pin (SIM_WOW_PIN)

Description

`SIM_WOW_PIN` enables a virtual sensor that detects if the aircraft is physically touching the ground.

This is critical for testing automatic arming/disarming logic and "Landed" state detection. When the drone is on the ground, the virtual pin state is changed.

Tuning & Behavior

- **Default Value:** -1 (Disabled).

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **SIMPLE** parameter group (specifically `SIMPLE`) configures the **Simple and Super Simple** pilot assistance modes for ArduCopter.

These modes decouple the drone's movement from its heading, making it much easier for beginners to fly or for experts to maintain orientation at long distances.

Key Concepts

1. Simple Mode

In standard flight, pushing the stick "Forward" moves the drone towards its own nose. If the drone rotates 180°, "Forward" on the stick moves the drone toward the pilot.

- **Simple Mode Fix:** When enabled, "Forward" on the stick always moves the drone away from the pilot (based on the heading when it was armed), regardless of the drone's current rotation.

2. Super Simple Mode

- **Super Simple Mode Fix:** Decouples movement from both heading and initial arming orientation. "Forward" on the stick always moves the drone away from the **Home position** (GPS based). Even if the drone flies behind the pilot, pushing "Away" still moves it further away.

Parameter Breakdown

- `SIMPLE`: Bitmask of which flight mode switch positions have Simple mode enabled.

Integration Guide

- **Setup:** Use the "Flight Modes" screen in Mission Planner to check the "Simple" or "SSimple" boxes for specific switch positions.
- **Prerequisite:** Super Simple mode requires a valid GPS lock.

Developer Notes

- **Library:** `ArduCopter/control_simple.cpp`.

SIMPLE (Simple mode bitmask)

DEFAULT: 0	RANGE: 0 63	UNITS: Bitmask

Simple mode bitmask (SIMPLE)

Description

Simple Mode is a pilot assistance feature where the controls are relative to the copter's heading *when it was armed* (or last reset). This means "Forward" on the stick always moves the copter North (if you armed facing North), even if the copter rotates to face East. It is "Headless Mode" (orientation independent).

The Mathematics

The control inputs are rotated by the difference between the current heading and the initial heading ψ_0.

The Engineer's View

Defined in `ArduCopter/Parameters.cpp`.

- **Bit 0:** Flight Mode 1
- **Bit 1:** Flight Mode 2
- ...
- **Bit 5:** Flight Mode 6

Tuning & Behavior

- **Default Value:** 0 (Disabled)
- **Vs Super Simple:** Simple Mode only handles orientation. Super Simple handles orientation AND position relative to home.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





SIMPLE_MODE (Simple Mode Enable)

DEFAULT: 0	RANGE: 0 1

Simple Mode Enable (SIMPLE_MODE)

Description

`SIMPLE_MODE` is a pilot-assistance feature that removes the need to track which way the drone is "facing" (its heading).

In **Normal** flight, if you push the stick right, the drone rolls to *its* right. If the drone is facing toward you, pushing right makes it move to *your* left. In **Simple Mode**, pushing the stick right *always* makes the drone move to the right relative to your starting position, regardless of which way the nose is pointing.

Tuning & Behavior

- **Default:** 0 (Disabled).
- **Usage:** Excellent for beginners or for retrieving a drone that is so far away you can't tell which way it's pointing.
- **Prerequisite:** Requires a functional GPS and compass lock. The "Home" position is recorded at the moment of arming and used as the reference point.

SIMPLE_TYPE (Simple mode types)

DEFAULT: 0	RANGE: 0 1

SIMPLE_TYPE: Simple mode types

Description

This parameter selects the behavior of "Simple Mode" on a Rover.

Values

Value	Meaning
0	Initial Heading (Stick forward matches vehicle's orientation at arming)
1	Home Direction (Stick forward always moves the vehicle away from home)

- **Default:** 0 (Initial Heading)

Description

Simple Mode changes how the steering stick works to make it easier for beginners to fly/drive when the vehicle is facing them.

- **Initial Heading (0):** The autopilot remembers which way the rover was facing when it armed. If you push the stick forward, it always tries to drive in that direction, regardless of which way the nose is currently pointed.
- **Home Direction (1):** "Super Simple" mode. If you push the stick away from you, the rover always drives away from its home point. If you pull the stick back, it always drives towards home. This is perfect if you lose orientation.
- **Requirement:** Requires a good Compass and GPS lock to function correctly.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **SOAR** parameter group configures the **Autonomous Soaring** engine for **ArduPlane**. This allows gliders to automatically detect thermals (rising air), bank into them to gain altitude, and then continue on their mission, significantly extending flight endurance.

Key Concepts

1. Thermal Detection

The autopilot monitors the vertical velocity of the aircraft (via EKF and TE airspeed). If it detects a climb that cannot be explained by throttle or pitch changes, it assumes it has found a thermal.

- `SOAR_VSPEED`: The minimum lift (m/s) required to trigger soaring behavior.

2. Loitering in Thermals (`SOAR_THML_...`)

Once a thermal is found, the plane switches to a circular loiter to stay within the rising air column.

- `SOAR_THML_BANK`: The bank angle used while circling in the thermal.
- `SOAR_THML_ARSPD`: The target airspeed for efficient climbing.

3. Altitude Limits

- `SOAR_ALT_MAX`: The height at which the plane will exit the thermal and resume its mission.
- `SOAR_ALT_MIN`: The minimum altitude below which the plane will ignore thermals and use its motor to stay aloft.

Parameter Breakdown

- `SOAR_ENABLE`: Master switch.
- `SOAR_POLAR_K / B`: Aerodynamic "Polar" coefficients for the specific glider wing.
- `SOAR_ALT_CUTOFF`: Altitude at which to stop using the motor during a climb.

Integration Guide

1. **Requirement:** A glider with an airspeed sensor and high efficiency.
2. **Config:** Set your wing polar values (see Wiki for common airframes).
3. **Mode:** Switch to `AUTO` or `CRUISE` and enable soaring via an RC switch (`RCx_OPTION = 88`).

Developer Notes

- **Library:** `libraries/AP_Soaring`.
- **Algorithm:** Uses the **Total Energy** rate of change to estimate environmental air movement.

SOAR_ALT_CUTOFF (Soaring Cutoff Altitude)

DEFAULT: 250.0	RANGE: 0 5000.0	UNITS: m

Soaring Cutoff Altitude (SOAR_ALT_CUTOFF)

Description

`SOAR_ALT_CUTOFF` defines the "Start of Mission" altitude for autonomous gliders.

When the drone is climbing using its motor, it will continue until it hits this altitude. At that point, ArduSoar kills the motor and begins the search for lift.

Tuning & Behavior

- **Default Value:** 250.0 meters.
- **Recommendation:** Set high enough to ensure you find lift, but low enough to save battery for multiple climb cycles.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





SOAR_ALT_MAX (Maximum Soaring Altitude)

DEFAULT: 350.0	RANGE: 0 5000.0	UNITS: m

Maximum Soaring Altitude (SOAR_ALT_MAX)

Description

`SOAR_ALT_MAX` is the "Ceiling" for thermal hunting.

When the drone is circling in a thermal and reaches this altitude, it will automatically exit the circle and return to its mission. This is used for both legal compliance (staying below 400ft/120m) and performance (avoiding thin air where glide performance drops).

Tuning & Behavior

- **Default Value:** 350.0 meters.
- **Recommendation:** Set to your local legal altitude limit.

SOAR_ALT_MIN (Minimum Soaring Altitude)

DEFAULT: 50.0	RANGE: 0 1000.0	UNITS: m

Minimum Soaring Altitude (SOAR_ALT_MIN)

Description

`SOAR_ALT_MIN` is your "Safety Floor."

If the drone's altitude falls below this limit, ArduSoar stops trying to be clever and just turns on the motor to climb back to safety. This prevents the drone from getting stuck in a weak thermal so low that it can't safely return home.

Tuning & Behavior

- **Default Value:** 50.0 meters.
- **Recommendation:** Set to at least **2x the height of the tallest obstacles** in your area.

SOAR_CRSE_ARSPD (Soaring Cruise Airspeed)

DEFAULT: 0	RANGE: -1 50	UNITS: m/s

Soaring Cruise Airspeed (SOAR_CRSE_ARSPD)

Description

`SOAR_CRSE_ARSPD` determines the speed of travel when hunting for the next thermal.

- **0 (Default):** Use `AIRSPEED_CRUISE`.
- **-1: Speed-To-Fly (STF) Mode.** This is the "Genius Mode." The autopilot uses your drag polar to calculate the mathematically perfect speed to fly based on the current wind and expected strength of the *next* thermal.
- **Positive Value:** Hold this specific speed.

Tuning & Behavior

- **Default Value:** 0.
- **Recommendation:** Set to **-1** for the most advanced autonomous glider performance.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





SOAR_DIST_AHEAD (Soaring Thermal Entry Look-Ahead)

DEFAULT: 5.0	RANGE: 0 100	UNITS: m

Soaring Thermal Entry Look-Ahead (SOAR_DIST_AHEAD)

Description

`SOAR_DIST_AHEAD` defines the "Starting Guess" for the autopilot when it first detects a thermal.

When the Variometer (climb rate sensor) indicates lift, the autopilot knows it is near a thermal, but it doesn't know exactly where the center is. This parameter tells the aircraft to assume the thermal core is this many meters directly ahead. The aircraft will then begin a turn toward that estimated spot. As it circles, the EKF updates this estimate based on the actual lift experienced.

Tuning & Behavior

- **Default:** 5.0 meters.
- **Recommendation:** Leave at **5.0** for most small electric gliders. For larger, faster aircraft, you may need to increase this to **10.0 or 15.0** to account for the larger turning radius.
- **Significance:** Correct initialization leads to faster "centering" of the thermal and better climb rates.

SOAR_ENABLE (Soaring Enable)

DEFAULT: 0	RANGE: 0 1

Soaring Enable (SOAR_ENABLE)

Description

`SOAR_ENABLE` activates "ArduSoar" – a sophisticated suite of algorithms that allow a glider to autonomously find, track, and orbit inside thermals (rising columns of air) to gain altitude without using the motor.

When enabled, the autopilot monitors the Variometer (climb rate) and longitudinal acceleration. If it detects that the plane is being lifted by a thermal, it can automatically switch to `THERMAL` mode, bank into a circle, and climb. Once the thermal weakens or the altitude limit is reached, it returns to its mission.

- **0: Disabled (Default).**
- **1: Enabled.**

Tuning & Behavior

- **Default Value:** 0.
- **Operational Standard:** This is typically mapped to an RC switch using `RCx_OPTION = 88` (Soaring Enable). This allows the pilot to give the drone permission to "Go Hunting" for lift.
- **Prerequisite:** Requires a well-calibrated airspeed sensor for best performance.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





SOAR_MAX_DRIFT (Maximum drift distance to allow when thermalling)

DEFAULT: -1	RANGE: 0 1000	UNITS: m

SOAR_MAX_DRIFT: Maximum drift distance to allow when thermalling

Description

The previous mode will be restored if the horizontal distance to the thermalling start location exceeds this value. -1 to disable.

Values

- **Range:** 0 to 1000
- **Units:** m
- **Default:** -1 (Disabled)

Description

This parameter defines a "Geofence" for autonomous thermalling.

- **Function:** When the glider detects lift and enters `THERMAL` mode, it starts circling. The wind will naturally push the thermal (and the glider) downwind.
- **Safety:** If the glider drifts too far from the point where it started thermalling, it might get blown too far downwind to return home safely. `SOAR_MAX_DRIFT` sets the maximum allowable distance.
- **Trigger:** If exceeded, the glider exits `THERMAL` mode and resumes its previous navigation task (e.g., `AUTO` waypoints or `CRUISE`).

SOAR_MAX_RADIUS (Maximum thermal radius)

DEFAULT: 200	RANGE: 0 500	UNITS: m

SOAR_MAX_RADIUS: Maximum thermal radius

Description

Maximum radius to allow the thermal to grow to. If the radius estimate exceeds this value then thermalling is stopped.

Values

- **Range:** 0 to 500
- **Units:** m
- **Default:** 200

Description

This parameter limits the size of the circling pattern when soaring.

- **Function:** The Soaring EKF constantly estimates the size and strength of the thermal.
- **Trigger:** If the estimated size of the thermal core grows larger than this radius, the autopilot assumes the lift is too diffuse or the estimate is invalid, and it abandons the thermal to continue the mission.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY

SOAR_MIN_CRSE_S (Minimum Soaring Cruise Time)

DEFAULT: 10	RANGE: 0 600	UNITS: s

Minimum Soaring Cruise Time (SOAR_MIN_CRSE_S)

Description

`SOAR_MIN_CRSE_S` prevents the glider from "Hunting" or flickering between thermalling and cruising states.

When the glider leaves a thermal, it enters the **Cruise** (glide) mode to head toward the next waypoint. This parameter ensures it stays in that mode for a minimum amount of time before it is allowed to search for or enter a new thermal. This prevents the plane from circling in weak, broken lift immediately after exiting a good thermal.

Tuning & Behavior

- **Default:** 10 seconds.
- **Recommendation:** Leave at **10-15 seconds**.
- **Significance:** Ensures the aircraft gains enough horizontal distance from the previous thermal's turbulence before attempting to detect the next one.

SOAR_MIN_THML_S (Minimum Soaring Thermal Time)

DEFAULT: 20	RANGE: 0 600	UNITS: s

Minimum Soaring Thermal Time (SOAR_MIN_THML_S)

Description

`SOAR_MIN_THML_S` sets a "Timer" for how long the glider must commit to a thermal once it starts circling.

Thermals can be turbulent and noisy. Without this parameter, the autopilot might think it's found a thermal, circle once, and then immediately "Lose" it and exit because of a brief downdraft. This setting forces the plane to stay in the circling state for at least this many seconds, providing enough time for the climb rate filters to stabilize and confirm if the lift is genuine.

Tuning & Behavior

- **Default:** 20 seconds.
- **Recommendation:** Keep at **20-30 seconds**. This is typically enough for one or two full circles, allowing the EKF and Soaring logic to build a reliable estimate of the thermal's strength and center.

SOAR_POLAR_B (Induced Drag Coefficient)

DEFAULT: 0.031	RANGE: 0.005 0.05

Induced Drag Coefficient (SOAR_POLAR_B)

Description

`SOAR_POLAR_B` describes how much speed the glider loses when it pulls up or turns.

As an aircraft produces more lift (e.g., during a turn), it creates "Induced Drag." This parameter tells the soaring controller how much extra sink rate to expect when the plane is banking in a thermal.

Tuning & Behavior

- **Default Value:** 0.031.
- **Recommendation:** High aspect ratio wings (long, skinny wings) have a lower `B` value (more efficient). Stubby wings have a higher `B`.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY

SOAR_POLAR_CD0 (Zero Lift Drag Coefficient)

DEFAULT: 0.027	RANGE: 0.005 0.5

Zero Lift Drag Coefficient (SOAR_POLAR_CD0)

Description

`SOAR_POLAR_CD0` is a fundamental aerodynamic constant for your glider. It represents the "Parasite Drag" – the air resistance caused by the fuselage, tail, and skin friction when the wing is not producing lift.

ArduSoar uses this (along with `SOAR_POLAR_B` and `K`) to build a mathematical model of how fast your plane sinks at any given airspeed. Without an accurate model, the autopilot cannot distinguish between a "Good Thermal" and the plane simply gliding efficiently.

The Mathematics

The sink rate (V_z) is calculated using the drag polar equation:

$$V_z = \frac{\text{CD0} \cdot V^3}{K} + \frac{B \cdot K}{V \cdot \cos(\phi)^2}$$

Where:

- V is airspeed.
- K is the loading factor.
- B is the induced drag coefficient.

Tuning & Behavior

- **Default Value:** 0.027.
- **Recommendation:**
  - **High Performance Glider:** 0.015 - 0.025.
  - **Foam Glider (Bixler/Skywalker):** 0.030 - 0.045.
- **Calibration:** If the drone thinks it's in lift even when the air is perfectly still, your `CD0` is likely set too high (it expects more drag than the plane actually has).

SOAR_POLAR_K (Wing Loading Factor (K))

DEFAULT: 25.6	RANGE: 20 400	UNITS: m²/s²

Wing Loading Factor (SOAR_POLAR_K)

Description

`SOAR_POLAR_K` accounts for the weight and wing area of your glider.

It acts as the "Scaling Constant" for the entire drag polar. A heavy glider needs to fly faster to stay in the air, and this parameter tells the math model exactly how much faster.

The Mathematics

$$K = \frac{2 \cdot m \cdot g}{\rho \cdot S}$$

Where:

- m is mass (kg).
- g is gravity (9.81).
- ho is air density (~1.225 at sea level).
- S is wing area (m^2).

Tuning & Behavior

- **Default Value:** 25.6.
- **Recommendation:** If you add a larger battery (making the plane heavier), you must increase `K` proportionally to maintain accuracy.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





SOAR_Q1 (Soaring EKF Thermal Strength Noise)

DEFAULT: 0.001	RANGE: 0.0001 0.01

Soaring EKF Thermal Strength Noise (SOAR_Q1)

Description

`SOAR_Q1` is a low-level tuning parameter for the soaring Kalman Filter.

It defines how much the autopilot expects the strength of a thermal to change over time. A higher value tells the filter that thermals are turbulent and their strength can change rapidly. A lower value tells the filter that thermals are steady, resulting in a smoother but slower-reacting estimate of the climb rate.

Tuning & Behavior

- **Default:** 0.001.
- **Recommendation:** Leave at **0.001**. Only adjust if you are a developer testing the soaring estimator's response to artificial lift profiles.

SOAR_Q2 (Soaring EKF Position Noise)

DEFAULT: 0.001	RANGE: 0.0001 0.01

Soaring EKF Position Noise (SOAR_Q2)

Description

`SOAR_Q2` defines how much the autopilot expects the center of a thermal to "drift" or move.

SOAR_R (Soaring EKF Measurement Noise)

DEFAULT: 0.1	RANGE: 0.01 1.0

Soaring EKF Measurement Noise (SOAR_R)

Description

`SOAR_R` defines how much the autopilot "Trusts" the data coming from your Variometer/Barometer.

SOAR_THML_ARSPD (Soaring Thermal Airspeed)

DEFAULT: 0	RANGE: 0 50	UNITS: m/s

Soaring Thermal Airspeed (SOAR_THML_ARSPD)

Description

`SOAR_THML_ARSPD` is the "Best Glide" speed for circling.

When you are inside a thermal, you want to stay in the air as long as possible (minimum sink rate).

- **0 (Default):** Use `AIRSPEED_CRUISE`.
- **Value:** Hold this specific speed.

Tuning & Behavior

- **Default Value:** 0.
- **Recommendation:** Set to your plane's "Minimum Sink Speed" (usually slightly slower than your best glide speed).

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





SOAR_THML_BANK (Thermalling Bank Angle)

DEFAULT: 30.0	RANGE: 20 50	UNITS: deg

Thermalling Bank Angle (SOAR_THML_BANK)

Description

`SOAR_THML_BANK` determines the "Tightness" of your search circle.

To stay inside a narrow thermal, the glider must bank its wings.

- **Shallow Bank (20°):** Large circles. Good for wide, weak thermals, but easy to fall out of a small one.
- **Steep Bank (45°):** Tight circles. Good for centering on small, strong thermals, but increases the "Induced Drag," which makes the plane sink faster.

Tuning & Behavior

- **Default Value:** 30.0°.
- **Recommendation:** 30° to 35° is a good sweet spot for most foam gliders.

SOAR_THML_FLAP (Soaring Thermal Flap Setting)

DEFAULT: 0	RANGE: 0 100	UNITS: %

Soaring Thermal Flap Setting (SOAR_THML_FLAP)

Description

`SOAR_THML_FLAP` defines the specialized wing configuration used for maximum lift during the "Thermalling" phase.

When a glider is circling in a thermal, it typically flies at a much lower speed than during cross-country cruise. Deploying a specific amount of flap increases the wing's camber, providing more lift and allowing the plane to circle tightly and slowly without stalling. This parameter overrides the usual flight mode flap settings specifically for soaring.

Tuning & Behavior

- **Default:** 0%.
- **Recommendation:** Set this to the "Thermal" or "Launch" flap position recommended by your glider's manufacturer (typically **10% to 30%**).
- **Significance:** Higher flap settings improve the "Climb" efficiency but increase drag, which is fine while you are inside the lifting air of a thermal.

SOAR_VSPEED (Soaring Vertical Speed Trigger)

DEFAULT: 0.7	RANGE: 0 10	UNITS: m/s

Soaring Vertical Speed Trigger (SOAR_VSPEED)

Description

`SOAR_VSPEED` is the "Excitement Threshold" for your glider.

It defines how strong a thermal must be before the drone decides to stop its mission and start circling. If the "Nettorate" (the vertical speed of the air around the plane, with the plane's own sink rate removed) exceeds this value, the drone declares "Lift Detected!"

- **Low Value (e.g. 0.5):** Very sensitive. The drone will try to circle in almost any rising air.
- **High Value (e.g. 2.0):** Selective. The drone only stops for strong, reliable thermals.

Tuning & Behavior

- **Default Value:** 0.7 m/s.
- **Recommendation:** Match this to your glider's performance. A high-performance carbon glider might circle in **0.5m/s**, while a draggy foam glider might need **1.0m/s** to make any real progress.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

Key Concepts

SONAR_POS (Sonar position)

DEFAULT: 0	RANGE: -5 5	UNITS: m

SONAR_POS: Sonar position

Description

This parameter (SITL only) defines the physical location of the simulated sonar sensor relative to the vehicle's center of gravity (CG).

Values

- **Range:** -5 to 5
- **Units:** Meters (m)
- **Default:** 0.0, 0.0, 0.0

Description

`SONAR_POS` allows for realistic simulation of rangefinder offsets.

- **Function:** If the sonar is mounted 20cm forward of the CG, you would set the X component to **0.2**.
- **Effect:** The simulation will calculate the distance to the ground from that specific point rather than the center of the aircraft, which is important for accurate landing and terrain-following behavior when the drone is pitched or rolled.
- **Components:** This is a Vector3f parameter (X, Y, Z).

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **SPD** parameter group (specifically `SPD_SCA_BASE` or `MOT_SPD_SCA_BASE`) configures speed-based scaling for the attitude controllers.

Key Concepts

1. Dynamic Scaling

On high-speed vehicles, the effectiveness of the control surfaces (or motor differential) increases with speed. At high speeds, the drone may become "twitchy" or oscillate if the PID gains remain the same as they are at a hover.

- `SPD_SCA_BASE`: Defines the ground speed (m/s) at which the attitude PID gains are at their nominal value. As speed increases above this, the autopilot automatically "detunes" (reduces) the gains to maintain stability.

Developer Notes

- **Library:** `libraries/AP_Motors`.
- **Context:** Most common on Tailsitters and high-speed multirotors.

SPD_SCA_BASE (Motor speed scaling base speed)

DEFAULT: 1.0	RANGE: 0 10	UNITS: m/s

Motor speed scaling base speed (SPD_SCA_BASE)

Description

This parameter is used on Ground Vehicles (Rovers/Boats) with "Regular" steering (e.g., a front-wheel steering car). It defines the "Safe Speed" threshold for full steering authority.

As a rover drives faster, small steering inputs cause much more aggressive cornering, which can lead to rollovers or loss of traction. To prevent this, ArduPilot automatically scales down the steering output as the vehicle's speed increases. `SPD_SCA_BASE` is the speed at which this reduction begins.

The Mathematics

The steering command is scaled linearly based on the ratio of the base speed to the current ground speed (V_ground):

$$\text{IF } |V_ground| > SPD_SCA_BASE:$$

$$ \text{Steering}_{eff} = \text{Steering}_{cmd} \times \left(\frac{SPD_SCA_BASE}{|V_{ground}|} \right) $$

If the vehicle is moving slower than the base speed, the multiplier is **1.0** (no reduction).

The Engineer's View

In `AP_MotorsUGV::output_regular()` (`libraries/AR_Motors/AP_MotorsUGV.cpp`):

1. The code checks if `_speed_scale_base` is positive.
2. It compares the current ground speed to this base.
3. The inverse scaling is applied to the steering term.
4. This implementation is essentially an "Electronic Steering Rack" that becomes slower and more stable at high speeds, mirroring the behavior of modern high-performance automobiles.

Tuning & Behavior

- **Default Value:** 1.0 m/s.
- **Effect of Increasing:** The vehicle maintains full steering sensitivity at higher speeds. Useful for slow, high-torque crawlers.
- **Effect of Decreasing:** The steering becomes "stiffer" (less sensitive) sooner as you accelerate. Recommended for fast on-road rovers to prevent high-speed flipping.
- **0:** Disables speed scaling entirely. Use with caution on fast vehicles.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY

Overview

The **SPEC** parameter group (specifically `SPEC_GRAV` or `BARO_SPEC_GRAV`) configures the density of the medium for **ArduSub**.

Key Concepts

1. Water Density

Depth calculation for ROVs depends on the weight of the water above the sensor. Fresh water and salt water have different densities.

- `SPEC_GRAV`: The specific gravity of the liquid.
  - **1.00:** Pure Fresh Water.
  - **1.025:** Standard Seawater.

Developer Notes

- **Library:** `libraries/AP_Baro`.
- **Accuracy:** Setting this correctly is essential for precise depth hold in different environments (e.g., Lake vs. Ocean).

SPEC_GRAV (Specific Gravity (For water depth measurement))

DEFAULT: 1.0	RANGE: 1.0 1.1

SPEC_GRAV: Specific Gravity (For water depth measurement)

Description

This sets the specific gravity of the fluid when flying an underwater ROV (ArduSub). 1.0 for fresh water, 1.025 for salt water.

Values

- **Range:** 1.0 to 1.1
- **Default:** 1.0

Description

This parameter is critical for **ArduSub** (underwater ROVs) to accurately measure their depth using a pressure sensor.

- **Function:** Water pressure increases as you go deeper. However, salt water is denser than fresh water, so the pressure increases faster.
- **Accuracy:** If you dive in the ocean with this set to 1.0 (Fresh Water), your reported depth will be slightly incorrect (it will think you are deeper than you actually are).
- **Tuning:** Always set this to match the body of water you are currently operating in. For example, use **1.025** for typical sea water.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **SPEED** parameter group (specifically `SPEED_MAX`) configures the horizontal travel speed for **ArduRover**.

Key Concepts

1. Velocity Target

- `SPEED_MAX`: The physical maximum speed (m/s) that the rover is capable of at full throttle.
- **Estimation:** If set to 0, the autopilot will attempt to estimate the max speed based on `CRUISE_SPEED` and `CRUISE_THROTTLE`.

Developer Notes

- **Library:** `ArduRover/Parameters.cpp`.
- **Related:** Multicopter navigation speeds are typically in the `WPNAV_` group.

SPEED_MAX (Speed maximum)

DEFAULT: 0.0f	RANGE: 0 30	UNITS: m/s

Speed maximum (SPEED_MAX)

Description

This parameter defines the **Physical Maximum Speed** of the Rover at 100% throttle. The navigation controller uses this value to scale its throttle output. If set to 0, the autopilot will attempt to estimate it based on `CRUISE_SPEED` and `CRUISE_THROTTLE`, but setting it explicitly provides better performance.

The Mathematics

$$\text{Throttle_}demand \approx \frac{\text{Target Speed}}{\text{SPEED_MAX}}$$

The Engineer's View

Defined in `Rover/Parameters.cpp`. Used in `AR_AttitudeControl` (the Ackermann/Skid steering controller) to normalize speed demands to throttle percentages.

- **If 0:** `SPEED_MAX` is derived: $\text{SPEED_MAX} = \text{CRUISE_SPEED} / (\text{CRUISE_THROTTLE}/100)$.

Tuning & Behavior

- **Default Value:** 0 (Auto-calculate)
- **Set to:** The actual top speed of the rover in meters/second.
- **Effect:** Improving this value helps the speed controller achieve target speeds more accurately without overshooting (if too low) or undershooting (if too high).

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY

SPEED_TYPE (Wind speed sensor Type)

DEFAULT: 0	RANGE: 0 4

SPEED_TYPE: Wind speed sensor Type

Description

This parameter selects the type of hardware used to measure wind speed (anemometer) when using the WindVane library.

Values

Value	Meaning
0	None (Disabled)
1	Analog Pin (Voltage-based anemometer)
2	Modern Device (I2C or Serial sensor)
4	SITL (Simulation only)

- **Default:** 0

Description

`SPEED_TYPE` is used on autonomous sailboats and wind-monitoring systems to quantify the current wind velocity.

- **Function:** Knowing the exact wind speed allows the autopilot to make better decisions about sail trim and tacking maneuvers.
- **Requirement:** Once you select a type, you must also configure the corresponding hardware pins (e.g., `WNDVN_SPEED_PIN`) and scaling factors.
- **Usage:** Primarily used in **ArduRover** (Sailboat) and specialized ground stations.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **SPRAY** parameter group configures the **Agricultural Sprayer** subsystem. This allows the drone to autonomously control pumps and spinners for crop dusting or industrial spraying missions.

Key Concepts

1. Speed-Based Flow (`SPRAY_PUMP_RATE`)

To ensure even coverage, the pump speed must increase as the drone flies faster and decrease as it slows down.

- `SPRAY_PUMP_RATE`: Defines the relationship between ground speed and pump PWM.

2. Auto-Shutoff (`SPRAY_SPEED_MIN`)

Prevents the drone from dumping chemicals while hovering or waiting for a GPS lock.

- `SPRAY_SPEED_MIN`: The pump will only activate if the drone is moving faster than this speed (m/s).

Parameter Breakdown

- `SPRAY_ENABLE`: Master switch.
- `SPRAY_SPINNER`: Toggles the centrifugal spinner (used to atomize the liquid).

Integration Guide

1. **Pin:** Map a servo output to `Sprayer` (Function 40) for the pump and `SprayerSpinner` (Function 83).
2. **Config:** Calibrate the flow rate based on your nozzle specs.

Developer Notes

- **Library:** `libraries/AC_Sprayer`.

SPRAY_ENABLE (Sprayer enable/disable)

DEFAULT: 0	RANGE: 0 1

SPRAY_ENABLE: Sprayer enable/disable

Description

This parameter enables or disables the automatic crop sprayer functionality.

Values

Value	Meaning
0	Disabled
1	Enabled

- **Default:** 0

Description

The **Sprayer** module is designed for agricultural drones. It automatically controls a pump and a spinner based on the aircraft's ground speed and position.

- **Function:** When enabled, the autopilot will calculate the required pump speed (`SPRAY_PUMP_RATE`) to ensure an even distribution of liquid regardless of how fast the drone is flying.
- **Automation:** It can be configured to start and stop automatically when the drone enters or leaves a designated spray area in a mission.
- **Hardware:** Requires a servo or ESC-controlled pump assigned to `SERVOx_FUNCTION = 94` (SprayerPump) and optionally a spinner on `SERVOx_FUNCTION = 95` (SprayerSpinner).

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





SPRAY_PUMP_MIN (Sprayer Pump Speed Minimum)

DEFAULT: 0	RANGE: 0 100	UNITS: %

Sprayer Pump Speed Minimum (SPRAY_PUMP_MIN)

Description

`SPRAY_PUMP_MIN` sets the "Idle Speed" for your crop sprayer's pump.

Agricultural pumps often struggle to start or maintain pressure at very low throttle values. This parameter ensures that if the sprayer is active, the pump never drops below a certain speed, even if the drone is moving slowly.

The Engineer's View

Defined in `AC_Sprayer.cpp`.
The sprayer logic calculates the required pump speed based on ground speed (`SPRAY_PUMP_RATE`).

$$\text{Target Speed} = \text{Ground Speed} \times \text{SPRAY_PUMP_RATE}$$

This target is then clamped:

$$\text{Output} = \max(\text{Target Speed}, \text{SPRAY_PUMP_MIN})$$

This prevents the pump from stalling or pulsing when the drone slows down for a turn.

Tuning & Behavior

- **Default Value:** 0%.
- **Recommendation:** Set to the lowest percentage where your pump still produces a clean spray pattern (e.g., **10% or 15%**).

SPRAY_PUMP_RATE (Pump speed)

DEFAULT: 10	RANGE: 0 100	UNITS: %

SPRAY_PUMP_RATE: Pump speed

Description

Desired pump speed when travelling 1m/s expressed as a percentage.

Values

- **Range:** 0 to 100
- **Units:** %
- **Default:** 10

Description

This parameter defines the "Volume" of the spray.

- **Function:** It sets the baseline pump speed (PWM output) required to deliver the desired amount of liquid per square meter when the drone is flying at exactly 1 meter per second.
- **Scaling:** If the drone flies faster (e.g., 2m/s), the autopilot will automatically double the pump speed to maintain the same application rate. If it slows down, the pump slows down.
- **Tuning:** You must calibrate this based on your nozzle size and pump flow rate. If you find the drone is under-applying, increase this value.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





SPRAY_SPEED_MIN (Sprayer Minimum Speed)

DEFAULT: 100	RANGE: 0 1000	UNITS: cm/s

Sprayer Minimum Speed (SPRAY_SPEED_MIN)

Description

`SPRAY_SPEED_MIN` prevents "Chemical Dumping" while the drone is moving too slowly.

In agricultural spraying missions, you want a uniform application of fertilizer or pesticide. If the drone is hovering or just starting to move, the sprayer would apply too much chemical to a single spot. This parameter ensures the pump only turns on when the drone has reached a safe, productive ground speed.

Tuning & Behavior

- **Default:** 100 cm/s (1.0 m/s).
- **Recommendation:** Set this to your minimum effective spraying speed.
- **Physics:** This acts as a logical gate. Below this speed, the pump is OFF. Above this speed, the pump is ON (or modulated based on SPRAY_PUMP_RATE).

SPRAY_SPINNER (Sprayer Spinner PWM)

DEFAULT: 1500	RANGE: 1000 2000	UNITS: PWM

Sprayer Spinner PWM (SPRAY_SPINNER)

Description

`SPRAY_SPINNER` defines the speed of the centrifugal dispersal mechanism on your sprayer.

Instead of just a simple nozzle, many high-end agricultural drones use a spinning disk to fling the chemicals outward. A higher PWM value makes the disk spin faster, throwing the liquid in a wider swath.

- **1000:** Stopped.
- **2000:** Maximum dispersal width.

Tuning & Behavior

- **Calibration:** Set this based on the desired "Swath Width" for your mission. If you are flying 4m apart and the chemical is not reaching the edge, increase this value.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY

Overview

The **SRTL** parameter group configures the **Smart Return to Launch** (SmartRTL) mode. Unlike standard RTL, which flies a straight line home, SmartRTL **retraces the exact path** the vehicle took to get there.

This is ideal for flying in areas with obstacles (like indoors or in a forest) where a straight-line path home might lead to a collision.

Key Concepts

1. Breadcrumb Trail

As the drone flies, it continuously drops "breadcrumbs" in memory to record its path.

- `SRTL_POINTS`: The maximum number of points held in memory. Once full, the oldest points are discarded (or simplified).

2. Path Pruning (`SRTL_ACCURACY`)

To save memory, the system "prunes" (simplifies) the path. If three points are in a nearly straight line, the middle point is deleted.

- `SRTL_ACCURACY`: The maximum allowed deviation (meters) from the original path during pruning.

Parameter Breakdown

- `SRTL_OPTIONS`: Bitmask for specialized behaviors (e.g., face direction of flight during return).

Integration Guide

- **Prerequisite:** SmartRTL is always active in the background if enabled. You can switch to it manually via RC or let it be the target for a Failsafe action.

Developer Notes

- **Library:** `libraries/AP_SmartRTL`.
- **Safety:** If the breadcrumb buffer is empty (e.g., failsafe happens immediately after takeoff), the vehicle defaults to standard straight-line RTL.

SRTL_ACCURACY (SmartRTL Accuracy)

DEFAULT: 2.0	RANGE: 0 10.0	UNITS: m

SmartRTL Accuracy (SRTL_ACCURACY)

Description

`SRTL_ACCURACY` defines the resolution of your "Breadcrumb" trail when using **Smart RTL**.

Unlike a standard RTL (which flies a straight line home), Smart RTL records your exact flight path and "rewinds" it to fly you back the same way you came. This is critical for avoiding obstacles like trees or buildings in complex environments. This parameter sets how far you must travel before the autopilot drops a new breadcrumb.

Tuning & Behavior

- **Default:** 2.0 meters.
- **Effect of Decreasing:** Dropping points more often (e.g., every 0.5m) creates a very smooth, exact path, but it fills up the drone's memory much faster.
- **Effect of Increasing:** (e.g. 5m) saves memory, but the return path will be "choppier" and might cut corners.
- **Safety:** If the breadcrumb memory fills up, the drone will revert to standard (straight-line) RTL.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





SRTL_OPTIONS (SmartRTL options)

DEFAULT: 0	RANGE: 0 3	UNITS: Bitmask

SmartRTL options (SRTL_OPTIONS)

Description

This bitmask allows you to customize the behavior of the Smart RTL (Return To Launch) mode, which returns the vehicle home by retracing its outbound path.

The Mathematics

Bitmask.

The Engineer's View

Defined in `libraries/AP_SmartRTL/AP_SmartRTL.cpp`.

- **Bit 0:** Ignore Pilot Yaw. If set, the vehicle will face the direction of travel along the path rather than allowing the pilot to look around.
- **Bit 1:** Prune Path. If set, the algorithm aggressively simplifies the path to save memory.

Tuning & Behavior

- **Default Value:** 0
- **Recommendation:** Default is usually best.

SRTL_POINTS (SmartRTL maximum number of points on path)

DEFAULT: 100	RANGE: 0 500

SRTL_POINTS: SmartRTL maximum number of points on path

Description

This parameter sets the memory limit for the "Smart RTL" (Return to Launch) path tracking.

Values

- **Range:** 0 to 500
- **Default:** 100
- **Memory Cost:** ~3 KB per 100 points.

Description

Smart RTL allows a drone to return home by replaying its outgoing path (like a breadcrumb trail), which is essential if you are flying in an area with obstacles (like buildings or trees) that would block a straight-line RTL.

- **Function:** As you fly, the autopilot saves your position as a series of "Points." This parameter limits how many points can be stored.
- **Out of Space:** If you fly a very long, complex path and run out of points, Smart RTL will simplify the older parts of the path by deleting intermediate breadcrumbs.
- **Disable:** Setting this to **0** completely disables the Smart RTL feature to save RAM on memory-constrained boards.
- **Reboot Required:** Changes to this parameter require a reboot to reallocate memory.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **STAB** parameter group (specifically `STAB_PITCH_DOWN`) configures the assisted flight behaviors in **Stabilize** and **FBWA** modes.

Key Concepts

1. Low Throttle Pitch Trim

In a manual airplane, reducing throttle typically causes the nose to drop as airspeed decreases. In an autopilot-stabilized mode, the autopilot might try to hold the nose up, which can lead to a stall.

- `STAB_PITCH_DOWN`: Automatically adds a small amount of down-elevator (pitch down) when the throttle is at zero or idle.
  - **Purpose:** Keeps the nose down in a glide to maintain airspeed and prevent stall.
  - **Typical Value:** 2 to 5 degrees.

Developer Notes

- **Library:** `ArduPlane/Parameters.cpp`.
- **Context:** Only relevant for **ArduPlane**.

STAB_PITCH_DOWN (Low throttle pitch down trim)

DEFAULT: 2.0f	RANGE: 0 15	UNITS: deg

Low throttle pitch down trim (STAB_PITCH_DOWN)

Description

This parameter improves flight safety in stabilized modes (like FBWA) by automatically pitching the nose down when the pilot reduces the throttle. This mimics the natural behavior of a well-trimmed aircraft and helps maintain airspeed during glides or landing approaches, preventing accidental stalls.

The Mathematics

$$ \text{Pitch}_{\text{bias}} = \text{STAB_PITCH_DOWN} \times \frac{\text{TRIM_THROTTLE} - \text{Throttle}}{\text{TRIM_THROTTLE} - \text{THR_MIN}} $$
(Applied only when Throttle < TRIM_THROTTLE)

The Engineer's View

Defined in `ArduPlane/Parameters.cpp`. This bias is added to the pitch controller's setpoint.

Tuning & Behavior

- **Default Value:** 2.0 degrees
- **Increase:** If the aircraft floats too much or slows down dangerously when you cut the throttle.
- **Decrease:** If the nose drops too aggressively.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **START** parameter group (specifically `START_LATITUDE` and `START_LONGITUDE`) configures the default starting location for the **SITL** simulator.

Key Concepts

1. Simulated World Origin

When you launch the SITL binary without a specific location flag, it defaults to the coordinates defined here.

- **Purpose:** Allows developers to always start their simulation at a specific airfield or test track.

Developer Notes

- **Library:** `libraries/SITL`.
- **SITL Only:** These parameters do not exist on real flight controller hardware.

START_LATITUDE (Initial Tracker Latitude)

DEFAULT: 0	RANGE: -90 90	UNITS: deg

Initial Tracker Latitude (START_LATITUDE)

Description

`START_LATITUDE` (working with START_LONGITUDE) allows you to manually tell an **Antenna Tracker** where it is physically located.

This is primarily used for trackers that do not have their own internal GPS. By providing these coordinates, the tracker can calculate the correct pointing angle toward the drone based on the drone's GPS telemetry.

Tuning & Behavior

- **Default:** 0.
- **Usage:** Enter the exact latitude of your ground station. If the tracker *does* have a GPS lock, this parameter is overridden by the real sensor data.

START_LONGITUDE (Initial Tracker Longitude)

DEFAULT: 0	RANGE: -180 180	UNITS: deg

Initial Tracker Longitude (START_LONGITUDE)

Description

Provides the manual longitude coordinate for an Antenna Tracker without GPS.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **STARTUP** parameter group (specifically `STARTUP_DELAY`) manages the initialization timing for the **Antenna Tracker**.

Key Concepts

1. Servo Warm-up

Some high-power servos or mechanical systems can draw massive current spikes or jitter violently if commanded immediately upon power-on.

- `STARTUP_DELAY`: Introduces a wait time (in seconds) during which the servos are held at their neutral trim position before active tracking begins.

Developer Notes

- **Library:** `AntennaTracker/Parameters.cpp`.

STARTUP_DELAY (Delay before first servo movement from trim)

DEFAULT: 0	RANGE: 0 10	UNITS: s

Delay before first servo movement from trim (STARTUP_DELAY)

Description

This parameter introduces a pause at system startup during which the Antenna Tracker servos are held at their trim (neutral) positions. This delay allows the system to stabilize, sensors to initialize, or the operator to stand clear before active tracking movement begins.

The Mathematics

Timer wait loop.

The Engineer's View

Defined in `AntennaTracker/Parameters.cpp`.

- **0:** No delay. Servos move immediately upon initialization.
- **>0:** Wait time in seconds.

Tuning & Behavior

- **Default Value:** 0 s
- **Usage:** Set to 2-5 seconds if your tracker tends to jerk violently on power-up.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





VEHICLE STATISTICS CONFIGURATION (STAT)

Overview

The **STAT** parameter group configures the **Flight Statistics** tracking system. This system acts like an odometer for your drone, recording total flight time, number of boots, and total runtime across the life of the vehicle.

These values are stored in non-volatile memory and persist across firmware updates.

Key Concepts

1. Lifetime Tracking

- `STAT_FLTTIME`: Total accumulated time the vehicle has been in the air (seconds).
- `STAT_RUNTIME`: Total time the flight controller has been powered on.
- `STAT_BOOTCNT`: Number of times the flight controller has been rebooted.

2. Resets (`STAT_RESET`)

Allows the user to clear the statistics (e.g., after a major rebuild or when selling the airframe).

- **Magic Number:** Usually requires a specific value to be written to prevent accidental resets.

Parameter Breakdown

- `STAT_FLTTIME`: Total flight time (seconds).
- `STAT_BOOTCNT`: Boot counter.

Developer Notes

- **Library:** `libraries/AP_Stats`.
- **Persistence:** Statistics are saved to the "Parameter Storage" area periodically and on disarm.

Boot Count ()

STAT_BOOTCNT: Boot Count

Description

Number of times board has been booted.

Values

- **Default:** 0

Description

This parameter serves as a persistent odometer for the flight controller hardware.

- **Function:** Every time the board powers up or reboots, this counter increments by 1.
- **Usage:** Useful for tracking the age of the board and identifying if uncommanded reboots (watchdog resets) have occurred during a mission or between flights.
- **Reset:** Can be reset by setting `STAT_RESET` to 0.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





STAT_FLTTIME (Total FlightTime)

DEFAULT: 0	UNITS: s

STAT_FLTTIME: Total FlightTime

Description

Total FlightTime (seconds).

Values

- **Units:** s
- **Default:** 0

Description

This parameter tracks the cumulative time the vehicle has been "flying" (armed and active).

- **Function:** It increments whenever the vehicle is armed and in a flying state.
- **Persistence:** This value is saved to storage and persists across reboots.
- **Maintenance:** Useful for scheduling maintenance intervals (e.g., motor bearing checks every 50 hours).

STAT_RESET (Statistics Reset Time)

DEFAULT: 1	UNITS: s

STAT_RESET: Statistics Reset Time

Description

Seconds since January 1st 2016 (Unix epoch+1451606400) since statistics reset (set to 0 to reset statistics, other set values will be ignored).

Values

- **Units:** s
- **Default:** 1

Description

This parameter holds the timestamp of the last time the flight statistics (`STAT_BOOTCNT`, `STAT_FLTTIME`, `STAT_RUNTIME`) were reset.

- **How to Reset:** To clear all accumulated statistics, manually set this parameter to **0**. The autopilot will then reset the counters and update this parameter to the current system time.
- **Format:** The value is the number of seconds since Jan 1st, 2016.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





STAT_RUNTIME (Total RunTime)

DEFAULT: 0	UNITS: s

STAT_RUNTIME: Total RunTime

Description

Total time autopilot has run.

Values

- **Units:** s
- **Default:** 0

Description

This parameter tracks the total cumulative uptime of the flight controller, regardless of whether it was flying or just sitting on the bench.

- **Function:** Increments continuously while the board is powered on.
- **Persistence:** Saved to storage and persists across reboots.
- **Usage:** Can be used to estimate total system life or debug "time since first power on" issues.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





STEERING CONTROLLER CONFIGURATION (STEER2SRV)

Overview

The **STEER2SRV** parameter group (often appearing as `STEER2SRV_` or `STR_`) configures the **Lateral Steering Controller** for **ArduRover** and for ground steering in **ArduPlane**.

This library is responsible for maintaining the requested heading or following a navigation line by driving the steering servo or motor differential.

Key Concepts

1. Steering-to-Servo (`STEER2SRV_...`)

The core PID controller for ground steering.

- `STEER2SRV_P/I/D`: Standard PID gains.
- `STEER2SRV_FF`: Feedforward gain. Critical for Ackermann (car-like) steering.
- `STEER2SRV_TCONST`: Time constant (seconds) for the response.

2. Speed Derating (`STEER2SRV_DRTFCT`)

At high speeds, a car becomes very sensitive to steering inputs.

- **Derating:** The autopilot automatically reduces the maximum steering angle as speed increases to prevent roll-overs.
- `STEER2SRV_DRTSPD`: The speed at which derating starts.

Parameter Breakdown

- `STEER2SRV_P`: Proportional gain (steering stiffness).
- `STEER2SRV_DRTMIN`: Minimum steering angle floor at high speed.

Integration Guide

- **Tuning:** Use **AUTOTUNE** in ArduRover to set these gains.
- **Manual Setup:** Ensure `STEER2SRV_FF` is set correctly so that the wheels turn to the correct angle to achieve the commanded turn rate at cruise speed.

Developer Notes

- **Library:** `libraries/APM_Control/AP_SteerController.cpp`.

STEER2SRV_D (Steering D Gain)

DEFAULT: 0.005	RANGE: 0 0.1

Steering D Gain (STEER2SRV_D)

Description

`STEER2SRV_D` stops the Rover from "Wagging its tail."

As the Rover reaches its target heading, it has momentum. `D` predicts that the error is closing and applies counter-steering to stop the turn smoothly.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





STEER2SRV_DRTFCT (Ground Steering Derating Factor)

DEFAULT: 10	RANGE: 0 50.0	UNITS: deg/m/s

Ground Steering Derating Factor (STEER2SRV_DRTFCT)

Description

`STEER2SRV_DRTFCT` is a safety parameter for fixed-wing aircraft with a steerable nose or tail wheel.

At very low speeds (taxiing), you need full steering authority to navigate around obstacles. However, at high speeds (takeoff roll), a tiny steering movement can cause the plane to veer wildly or even roll over. This parameter automatically "numbs" the steering wheel as the plane speeds up.

Tuning & Behavior

- **Default:** 10 deg/m/s.
- **Requirement:** Only takes effect when the speed exceeds STEER2SRV_DRTSPD.
- **Effect:** For every additional meter per second of speed above the threshold, the maximum steering angle is reduced by this many degrees.
- **Recommendation:** Increase this value if your plane is "twitchy" or difficult to keep straight on the runway during takeoff.

STEER2SRV_DRTMIN (Minimum angle of wheel)

DEFAULT: 4500	RANGE: 0 4500	UNITS: cdeg

Minimum angle of wheel (STEER2SRV_DRTMIN)

Description

This parameter defines the minimum steering angle (in centi-degrees) that the vehicle will maintain even at maximum speed. While the steering controller generally reduces (derates) the steering angle as speed increases to prevent rollovers, this lower limit ensures the vehicle retains some ability to turn rather than being locked to straight-ahead at full throttle.

The Mathematics

$$\theta_\text{limit} = \max(\theta_\text{derated}, \text{STEER2SRV_DRTMIN})$$

The Engineer's View

Defined in `libraries/APM_Control/AP_SteerController.cpp`.

- **Safety:** Prevents "Loss of Steering" at high speed.

Tuning & Behavior

- **Default Value:** 4500 cdeg (45 degrees). This is quite high and often means no derating floor is applied.
- **Adjustment:** Decrease to e.g., 500 cdeg (5 degrees) if you want to enforce a very strict limit at high speed.

STEER2SRV_DRTSPD (Ground Steering Derating Start Speed)

DEFAULT: 0	RANGE: 0 50	UNITS: m/s

Ground Steering Derating Start Speed (STEER2SRV_DRTSPD)

Description

`STEER2SRV_DRTSPD` defines the speed at which the autopilot begins to apply the "Speed-Based Numbing" for ground steering.

Below this speed, you have 100% steering authority. Above this speed, the authority is reduced based on STEER2SRV_DRTFCT.

Tuning & Behavior

- **Default:** 0 (Derating starts immediately from 0 speed).
- **Recommendation:** Set to your typical "Slow Taxi" speed (e.g. **2.0 or 3.0 m/s**). This ensures you have full control for parking but automatic protection as soon as you start your takeoff run.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





STEER2SRV_FF (Steering Feed-Forward)

DEFAULT: 0	RANGE: 0 5.0

Steering Feed-Forward (STEER2SRV_FF)

Description

`STEER2SRV_FF` is the "Guess" gain.

It tells the autopilot: "I know that to turn this fast, I need exactly this much steering angle." This provides the bulk of the steering command directly from the physics of the vehicle, leaving the P and I terms to only fix small errors.

Tuning & Behavior

- **Default Value:** 0.
- **Recommendation:** Tune this first. Adjust until the Rover follows a circular path at the correct radius for the given speed.

STEER2SRV_I (Steering I Gain)

DEFAULT: 0.2	RANGE: 0 1.0

Steering I Gain (STEER2SRV_I)

Description

`STEER2SRV_I` handles "Drift" on Rovers.

If your Rover's wheels are slightly misaligned, or if you are driving on a slope, the Rover will naturally pull to one side. The I-term builds up steering pressure to cancel out this bias and keep the Rover driving straight.

STEER2SRV_P (Steering P Gain)

DEFAULT: 1.8	RANGE: 0.1 5.0

Steering P Gain (STEER2SRV_P)

Description

`STEER2SRV_P` controls how aggressively a Rover turns to fix a heading error.

When the Rover is not pointed exactly at its target waypoint, the P-gain determines how much "Steering Angle" is applied immediately.

- **Too High:** The Rover will "Snake" or weave back and forth down the path.
- **Too Low:** The Rover will take a very long time to turn and may miss its waypoint.

STEER2SRV_TCONST (Steering Time Constant)

DEFAULT: 0.75	RANGE: 0.1 2.0	UNITS: s

Steering Time Constant (STEER2SRV_TCONST)

Description

`STEER2SRV_TCONST` defines the "Responsiveness" of the Rover.

- **Small Value (0.2s):** Very snappy, aggressive steering. Best for high-performance RC cars.
- **Large Value (1.5s):** Smooth, lazy steering. Best for large, slow tractors or boats.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





STICK MIXING CONFIGURATION (STICK)

Overview

The **STICK** parameter group (specifically `STICK_MIXING`) configures the pilot's ability to "nudge" or override the autopilot's path while in **Autonomous** flight modes (Auto, Guided, RTL).

This is a critical safety feature that allows the pilot to steer around unexpected obstacles (like a person walking onto a landing pad) without needing to switch into a fully manual flight mode.

Key Concepts

1. Manual Nudging

When stick mixing is enabled, moving the RC transmitter sticks adds an offset to the autopilot's target position or attitude.

- **Action:** If the drone is flying a straight line in Auto and you push the roll stick right, the drone will bank right and move off the line. When you release the stick, it will steer back to the original mission path.

2. Mixing Modes (`STICK_MIXING`)

- **0 (Disabled):** Sticks are completely ignored in Auto modes.
- **1 (FBW):** Recommended. The sticks provide a stabilized override.
- **3 (VTOL Yaw):** (QuadPlane) Allows the pilot to control heading during VTOL takeoff/landing while the autopilot handles position.

Developer Notes

- **Library:** `ArduPlane/Attitude.cpp`, `ArduCopter/control_auto.cpp`.
- **Safety:** Ensures the pilot always has "ultimate authority" over the vehicle's trajectory.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





STICK_MIXING (Stick Mixing)

DEFAULT: 1	RANGE: 0 3	UNITS: Option

Stick Mixing (STICK_MIXING)

Description

This parameter enables "Manual Nudging" during autonomous flight modes (like AUTO, GUIDED, or RTL). It allows the pilot to use the transmitter sticks to temporarily override the autopilot's path without actually taking full manual control.

This is a vital safety and utility feature. For example, if a drone is landing autonomously but the pilot sees an obstacle on the ground, they can nudge the stick to shift the landing spot slightly while the autopilot continues to handle the descent and stabilization.

The Mathematics

The autopilot combines the navigation command (Cmd_nav) with the pilot's stick input ($Input_pilot$):

FBW Mixing (Mode 1):
The stick input is treated as a delta to the target attitude.
$$ \text{TargetAngle} = \text{TargetAngle}*{nav} + (Input*{pilot} \times \text{MaxAngle}) $$

VTOL Yaw (Mode 3):
Specifically applies only to the Yaw axis during QuadPlane VTOL phases.
$$ \text{YawRate} = \text{YawRate}*{nav} + (Input*{yaw_stick} \times \text{MaxYawRate}) $$

The Engineer's View

In `ArduPlane/Attitude.cpp`, the function `Plane::stick_mixing_enabled()` checks this parameter.

- **Mode 0 (None):** Sticks are ignored in Auto modes. **Danger:** If the GPS glitches, the pilot has no way to "steer" the plane without first flipping a switch to a manual mode.
- **Mode 1 (FBW):** Uses "Fly-By-Wire" logic. This is the recommended setting. It ensures that the pilot can never command an unsafe bank or pitch angle, even while nudging.
- **Mode 3 (VTOL Yaw):** Used for QuadPlanes to allow the pilot to point the nose of the aircraft during an automatic VTOL takeoff or landing while the autopilot maintains position.

Tuning & Behavior

- **Default Value:** 1 (FBW).
- **Recommendation:** Keep at **1** for most users. It provides the best balance of autonomous precision and pilot authority.
- **Rover:** On Rovers, this allows the pilot to steer around obstacles on a mission path. The rover will return to the direct line to the waypoint once the pilot releases the stick.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **STILL** parameter group (specifically `STILL_THRESH` or `INS_STILL_THRESH`) defines the sensitivity of the autopilot's **Motion Detection** logic.

The system needs to know if the vehicle is physically stationary for critical operations like gyro calibration on boot or performing pre-arm checks.

Key Concepts

1. Stillness Detection

The autopilot monitors the raw variance of the accelerometers.

- **Stationary:** If the variance is below `STILL_THRESH`, the system assumes it is on the ground and still.
- **Moving:** If vibrations or movement exceed this value, the system assumes it is being handled or is on a moving platform.

Parameter Breakdown

- `STILL_THRESH`: Acceleration variance threshold.
  - **Typical:** 0.1 for fixed-wing, 2.5 for multicopters (to account for motor vibration).

Integration Guide

- **Boat Starts:** If you are launching from a moving boat, the autopilot may refuse to calibrate its gyros because it thinks it is moving. In this case, you must **increase** `STILL_THRESH` to allow the calibration to finish despite the deck's motion.

Developer Notes

- **Library:** `libraries/AP_InertialSensor`.

Stillness threshold for detecting if we are moving ()

STILL_THRESH: Stillness threshold for detecting if we are moving

Description

Threshold to tolerate vibration to determine if vehicle is motionless.

Values

- **Range:** 0.05 to 50
- **Default:** 0.1 (Plane/Rover), 2.5 (Copter/Heli)

Description

This parameter sets the sensitivity of the "Motion Detection" logic used by the EKF (Extended Kalman Filter) and the calibration routines.

- **Function:** The autopilot monitors the accelerometer noise. If the total acceleration variance is below this threshold, it assumes the vehicle is sitting still on the ground.
- **Usage:**
  - **Calibration:** When calibrating the gyros on boot, the vehicle must be "still." If you are on a boat or a windy platform, you may need to **increase** this threshold so the calibration can complete.
  - **Arming:** Some arming checks require the vehicle to be motionless.
- **Tuning:** Increase if you have trouble passing pre-arm checks on a moving platform. Decrease for high-precision calibration on a solid bench.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **STOP** parameter group (specifically `STOP_DIST` or `DOCK_STOP_DIST`) configures the terminal phase of the **Autonomous Docking** maneuver for ArduRover.

Key Concepts

1. Docking Termination

When a rover or boat is docking (e.g., for automated charging), it must stop precisely.

- `STOP_DIST`: Defines the safety gap (in meters) between the autopilot's target location and the physical stopping point.
- **Purpose:** Ensures the vehicle doesn't "over-ram" the dock, which could damage the charging pins or the airframe.

Developer Notes

- **Library:** `Rover/mode_dock.cpp`.
- **Context:** Only relevant for **ArduRover**.

STOP_DIST (Distance from docking target when we should stop)

DEFAULT: 0.3	RANGE: 0 2	UNITS: m

STOP_DIST: Distance from docking target when we should stop

Description

The vehicle starts stopping when it is this distance away from docking target.

Values

- **Range:** 0 to 2
- **Units:** m
- **Default:** 0.3

Description

This parameter is used in the **Docking Mode** for Rovers and Boats.

- **Function:** It defines the "Stopping Point" offset from the target. The vehicle will aim to come to a complete halt at this distance *before* hitting the physical dock or charging station.
- **Safety:** Prevents collision with the docking infrastructure.
- **Tuning:** Adjust based on the physical size of your vehicle and the tolerance of your docking mechanism.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY

Overview

The **STR** parameter group (often appearing as `ATC_STR_` in newer versions) provides the core tuning for the **Lateral Steering Controller** in ArduRover.

Key Concepts

1. Steering Limits

- `STR_RAT_MAX`: Maximum rotation rate (deg/s) the vehicle is allowed to request during a turn.
- `STR_ACC_MAX`: Maximum lateral acceleration (m/s²) allowed. This is critical for preventing rovers from tipping over in high-speed sharp turns.

2. Throttle-to-Steering Mix (`STR_THR_MIX`)

Used on **Skid-Steer** vehicles (tracked or differential drive).

- **Function:** Defines how much throttle is reduced when turning. This ensures the vehicle has enough power to pivot without stalling the motors.

Developer Notes

- **Library:** `libraries/AR_AttitudeControl`.
- **See Also:** STEER2SRV for related servo gains.

STR_ACC_MAX (Steering control angular acceleration maximum)

DEFAULT: 120	RANGE: 0 1000	UNITS: deg/s^2

Steering control angular acceleration maximum (STR_ACC_MAX)

Description

This parameter limits the angular acceleration (rotational jerk) of the steering control loop. It essentially smoothens the steering response by preventing the vehicle from trying to change its turn rate too instantly.

This is particularly useful for preventing "twitchy" behavior or mechanical stress on steering servos/linkages during aggressive maneuvers.

The Mathematics

The controller limits the *rate of change* of the desired turn rate.

$$|\Delta \text{Rate}| \leq \text{ACC_MAX} \times dt$$

If the desired change in turn rate exceeds this limit, the requested rate is clamped to the maximum allowable step for that time step.

The Engineer's View

In `AR_AttitudeControl::get_steering_out_rate`, `_steer_accel_max` is used to calculate `change_max`.
The current `_desired_turn_rate` is constrained to be within `+/- change_max` of the previous time step's rate.
This acts as a slew rate limiter on the target yaw rate.

Tuning & Behavior

- **Default Value:** 120 deg/s^2 (Rover default, may vary by frame).
- **Range:** 0 to 1000.
- **0:** Disabled (Unlimited acceleration).
- **Lower Value:** Smoother, lazier steering starts.
- **Higher Value:** Snappier response, potentially causing overshoot or mechanical strain.
- **Dependency:** This works in conjunction with `STR_RAT_MAX` (which limits the *speed* of the turn) and `STR_ANG_P` (which determines the *aggressiveness* of the correction).

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY

STR_RAT_MAX (Steering control rotation rate maximum)

DEFAULT: 120	RANGE: 0 1000	UNITS: deg/s

Steering control rotation rate maximum (STR_RAT_MAX)

Description

This parameter sets the hard limit on how fast the vehicle is allowed to turn (yaw) in degrees per second. Ideally, this should match the physical capability of the vehicle to turn without sliding or rolling over.

This acts as a "speed limit" for the steering controller. Even if the navigation system demands a sharp turn, the controller will not request a turn rate higher than this value.

The Mathematics

The controller clamps the desired turn rate.

$$|\text{Rate}_cmd| \leq STR_RAT_MAX$$

If `STR_RAT_MAX` is 0, this check is skipped.

The Engineer's View

In `AR_AttitudeControl::get_steering_out_rate`:

1. The code converts `_steer_rate_max` to radians.
2. It checks if `_desired_turn_rate` exceeds this limit.
3. If it does, it clamps the value and sets the `_steering_limit_left` or `_steering_limit_right` flags, which tells higher-level navigators (like the L1 controller) that the steering is saturated.

Tuning & Behavior

- **Default Value:** 120 deg/s
- **Range:** 0 to 1000 deg/s
- **Effect of Decreasing:** Makes the vehicle turn slower. Useful for preventing rollovers on high-CG rovers.
- **Effect of Increasing:** Allows faster turns.
- **0:** Disabled. The vehicle will try to turn as fast as the steering geometry and P-gain allow.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





STR_THR_MIX (Motor steering vs throttle prioritisation)

DEFAULT: 0.5	RANGE: 0.2 1.0	UNITS: null

Motor steering vs throttle prioritisation (STR_THR_MIX)

Description

This parameter manages the "Saturation Mixer" logic for skid-steer Rovers.
In a skid-steer vehicle (like a tank), turning is achieved by speeding up one side and slowing down the other. If you are already at 100% Throttle, you cannot speed up one side further to turn.

`STR_THR_MIX` decides what to sacrifice when the driver requests both full speed and a turn:

- **High Value (> 0.5):** Sacrifice Speed to ensure Turning (Slows down to turn).
- **Low Value (< 0.5):** Sacrifice Turning to maintain Speed (Turns less sharp to keep moving fast).

The Mathematics

The mixer checks if `Throttle + |Steering| > 1.0` (Saturation).
If saturated, it calculates a scaling factor based on `STR_THR_MIX` (α).

If $\alpha \geq 0.5$ (Prioritize Steering):
$$ \text{Steering} = \text{Steering}*{cmd} |times |text{FairScaler} $$
$$ |text{Throttle} = |text{Throttle}*{cmd} \times \text{interpolate}(\text{FairScaler}, 1.0, 0.5 - \alpha) $$
(Effectively reduces throttle to make "headroom" for the steering command)

If $\alpha < 0.5$ (Prioritize Throttle):
$$ \text{Throttle} = \text{Throttle}*{cmd} |times |text{FairScaler} $$
$$ |text{Steering} = |text{Steering}*{cmd} \times \text{interpolate}(\text{FairScaler}, 1.0, \alpha) $$
(Effectively reduces steering command)

The Engineer's View

In `AP_MotorsUGV::output_skid_steering`:
The code calculates `max_output = throttle + |steering|`.
If `max_output > 1.0`, it enters the saturation handling logic described above.
This ensures that the output to the motors never exceeds the physical limits while attempting to respect the user's priority preference.

Tuning & Behavior

- **Default Value:** 0.5 (Balanced)
- **Range:** 0.2 to 1.0
- **1.0 (Steering Priority):** The vehicle will slow down significantly to ensure it hits the requested turn rate. Recommended for autonomous missions where path tracking is critical.
- **0.2 (Throttle Priority):** The vehicle will prioritize speed, potentially missing a turn radius if moving fast. Useful for racing or manual FPV driving where flow is more important than precision.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **SUPER** parameter group (specifically `SUPER_SIMPLE`) configures the GPS-based headless mode for multicopters.

For full documentation, please see the SIMPLE group.

SUPER_SIMPLE (Super Simple Mode)

DEFAULT: 0	RANGE: 0 63	UNITS: Bitmask

Super Simple Mode (SUPER_SIMPLE)

Description

Super Simple Mode is a pilot assistance feature where the controls are always relative to the pilot's position (Home), regardless of the copter's heading or its position relative to the pilot. This differs from standard "Simple Mode" (which is just relative to heading). In Super Simple Mode, pushing the stick "Away" always makes the copter fly further away from Home, and pulling "Back" brings it home, even if it flies behind you.

The Mathematics

The control inputs are rotated based on the bearing from Home to the Vehicle.

$$\theta_\text{pilot} = \text{atan2}(P_\text{veh} - P_\text{home})$$

The Engineer's View

Defined in `ArduCopter/Parameters.cpp`. The parameter is a bitmask corresponding to the 6 Flight Mode switch positions (PWM ranges) on the RC transmitter.

- **Bit 0:** Flight Mode 1
- **Bit 1:** Flight Mode 2
- ...
- **Bit 5:** Flight Mode 6

Tuning & Behavior

- **Default Value:** 0 (Disabled)
- **Setup:** Enable specific bits to activate Super Simple Mode only when that flight mode is selected.
- **Warning:** Requires a valid GPS lock and Home position.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **SURFACE** parameter group (specifically `SURFACE_DEPTH` or `BARO_SURFACE_DEPTH`) calibrates the depth reference for **ArduSub**.

Key Concepts

1. Water Surface Reference

Submarines use pressure to calculate depth. However, atmospheric pressure changes.

- `SURFACE_DEPTH`: The reading the pressure sensor gives when the ROV is at the water's surface.
- **Calibration:** Usually set automatically by the GCS during the "Calibrate Pressure" step.

Developer Notes

- **Library:** `libraries/AP_Baro`.

SURFACE_DEPTH (Depth reading at surface)

DEFAULT: -10	RANGE: -100 0	UNITS: cm

Depth reading at surface (SURFACE_DEPTH)

Description

This parameter calibrates the external pressure sensor for the ROV. It represents the depth reading (in cm) that the sensor outputs when the vehicle is actually at the surface. This accounts for atmospheric pressure variations and sensor offset drift.

The Mathematics

$$ \text{Depth}*{\text{displayed}} = \text{Depth}*{\text{raw}} - \text{SURFACE_DEPTH} $$

The Engineer's View

Defined in `ArduSub/Parameters.cpp`.

- **Calibration:** Usually set automatically when you click "Calibrate Pressure" in QGroundControl while the ROV is on the surface.

Tuning & Behavior

- **Default Value:** -10 cm (Assumes sensor is slightly submerged or offset)

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **SURFTRAK** parameter group configures the **Surface Tracking** mode for **ArduSub**.

In this mode, the ROV maintains a constant distance from either the surface of the water or the seafloor (if a downward-facing rangefinder is used).

Key Concepts

1. Relative Depth Control

- `SURFTRAK_DEPTH`: The target distance (cm) to maintain from the reference (surface or bottom).
- `SURFTRAK_MODE`: Selects between Tracking the Surface or Tracking the Bottom.

Developer Notes

- **Library:** `ArduSub/Parameters.cpp`.
- **Requirement:** Requires a valid external pressure sensor or rangefinder.

SURFTRAK_DEPTH (SURFTRAK minimum depth)

DEFAULT: 50	RANGE: 0 1000	UNITS: cm

SURFTRAK minimum depth (SURFTRAK_DEPTH)

Description

This parameter sets the target depth for the ROV when in **Surface Tracking** mode. Unlike standard Depth Hold which locks the absolute pressure/depth, Surface Tracking tries to maintain a constant distance *below the water's surface* (often using a rangefinder or inverted logic).

The Mathematics

$$Z_\text{target} = Z_\text{surface} + \text{SURFTRAK_DEPTH}$$

The Engineer's View

Defined in `ArduSub/Parameters.cpp`.

Tuning & Behavior

- **Default Value:** 50 cm
- **Range:** Ensure this is deep enough to prevent the thrusters from breaching the surface and cavitating.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY

SURFTRAK_MODE (Surface Tracking Mode)

DEFAULT: 1	RANGE: 0 2

SURFTRAK_MODE: Surface Tracking Mode

Description

Set which surface to track in surface tracking.

Values

Value	Meaning
0	Do not track
1	Ground (Track distance to ground)
2	Ceiling (Track distance to ceiling)

- **Default:** 1 (Ground)

Description

This parameter configures the **Surface Tracking** library, which allows the vehicle to maintain a set distance from a surface using a rangefinder (Lidar/Sonar).

- **Ground (1):** Standard Terrain Following. The vehicle adjusts its altitude to maintain a constant height AGL (Above Ground Level).
- **Ceiling (2):** Used for indoor inspection or tunnel flight. The vehicle uses an upward-facing rangefinder to maintain a constant distance from the roof.
- **Do not track (0):** Disables surface tracking features.

SURFTRAK_TC (Surface Tracking Filter Time Constant)

DEFAULT: 1.0	RANGE: 0 5	UNITS: s

SURFTRAK_TC: Surface Tracking Filter Time Constant

Description

Time to achieve 63.2% of the surface altitude measurement change. If 0 filtering is disabled.

Values

- **Range:** 0 to 5
- **Units:** s
- **Default:** 1.0

Description

This parameter smooths the data coming from the rangefinder before it is used for Surface Tracking flight modes.

- **Function:** Rangefinders (especially Lidar over grass or Sonar over uneven ground) can be noisy. This Low-Pass Filter (LPF) smooths out spikes to prevent the drone from twitching vertically as it flies over small obstacles like bushes or rocks.
- **Trade-off:**
  - **Higher Value (More Filtering):** Smoother flight, but the drone reacts slower to terrain changes. Dangerous if flying fast towards a steep hill.
  - **Lower Value (Less Filtering):** Faster reaction to terrain, but "jumpier" flight characteristics.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **SW** parameter group (specifically `SW_METHOD` or `OSD_SW_METHOD`) defines how the pilot toggles through different **OSD Screens**.

Key Concepts

1. Screen Selection

ArduPilot supports multiple OSD layouts. This parameter selects the trigger for switching between them.

- `SW_METHOD`:
  - **0:** Fixed (Only Screen 1).
  - **1:** Use an RC Channel (defined in `OSD_CHAN`).
  - **2:** Toggle on a switch change.
  - **3:** Automatic (Switch based on flight stage: Arming/Failsafe/Mission).

Developer Notes

- **Library:** `libraries/AP_OSD`.

SW_METHOD (Screen switch method)

DEFAULT: 0	RANGE: 0 2

SW_METHOD: Screen switch method

Description

This sets the method used to switch different OSD screens.

Values

Value	Meaning
0	Toggle (Click to cycle through screens)
1	PWM Range (Switch selects screen based on PWM band)
2	Auto Switch (Switch screens automatically at a set interval)

- **Default:** 0

Description

This parameter (`OSD_SW_METHOD`) configures how the pilot changes between the multiple OSD layouts (Screen 1, Screen 2, etc.) using the RC channel assigned to `OSD_SW_CHAN`.

- **Toggle (0):** Each time you flip the switch High and back to Low, the OSD cycles to the next screen. Good for momentary switches.
- **PWM Range (1):** The position of a 3-position or rotary switch directly selects the screen. Low = Screen 1, Mid = Screen 2, High = Screen 3. Best for 3-pos switches.
- **Auto Switch (2):** The screens cycle automatically every few seconds. Not commonly used.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **SYSID** parameter group configures the identifiers used in MAVLink communication. These IDs ensure that multiple vehicles and ground stations can operate on the same radio network without conflicting.

Key Concepts

1. This Vehicle (`SYSID_THISMAV`)

Assigns a unique ID to the autopilot.

- **Default:** 1.
- **Recommendation:** If flying in a swarm or group, every drone must have a unique `SYSID_THISMAV` (e.g., 1, 2, 3...).

2. My GCS (`SYSID_MYGCS`)

Defines the ID of the Ground Control Station that the autopilot should trust for critical commands (like arming or parameter changes).

- **Default:** 255.

3. Enforce Identity (`SYSID_ENFORCE`)

If enabled, the autopilot will only accept commands from the specific GCS defined in `SYSID_MYGCS`.

Developer Notes

- **Library:** `libraries/GCS_MAVLink`.
- **Range:** 1 to 255.

`SYSID_ENFORCE` (GCS sysid enforcement)

DEFAULT: 0	RANGE: 0 1

SYSID_ENFORCE: GCS sysid enforcement

Description

This controls whether packets from other than the expected GCS system ID will be accepted.

Values

Value	Meaning
0	Not Enforced (Accept packets from any GCS)
1	Enforced (Only accept packets from `SYSID_MYGCS`)

- **Default:** 0

Description

This parameter adds a layer of security to the MAVLink communication link.

- **Function:** When enabled (1), the vehicle ignores all uplink commands (like mode changes or waypoint uploads) unless they come from a Ground Control Station whose System ID matches the value stored in `SYSID_MYGCS`.
- **Usage:** Useful if multiple pilots are flying in the same area on the same frequency/network to prevent cross-control.
- **Warning:** If you enable this and your GCS changes its System ID (or you use a different computer), you will be locked out of controlling the vehicle until you reset this parameter (e.g., via USB).

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





SYSID_MYGCS (MAVLink GCS System ID)

DEFAULT: 255	RANGE: 1 255

MAVLink GCS System ID (SYSID_MYGCS)

Description

`SYSID_MYGCS` sets the "Authorized" address for your ground control station (GCS).

In a MAVLink network, every device has a System ID. Most GCS software (Mission Planner, QGC) uses **255** as its default ID. This parameter tells the autopilot: "Only listen to commands from a GCS with this specific ID."

Tuning & Behavior

- **Default:** 255.
- **Usage:** For most users, leave at **255**. Only change this if you are using a custom GCS or are operating in a multi-pilot environment where you want to lock your drone to a specific laptop/tablet to prevent accidental interference.
- **Safety:** If you change this value and your GCS ID doesn't match, you may lose the ability to send commands to the drone via telemetry.

SYSID_TARGET (Target vehicle's MAVLink system ID)

DEFAULT: 0	RANGE: 0 255

Target vehicle's MAVLink system ID (SYSID_TARGET)

Description

This parameter tells the Antenna Tracker which MAVLink stream to listen to. In a swarm environment or at a flying field with multiple drones, this ensures your tracker points at *your* drone, not someone else's.

The Mathematics

MAVLink Message Filter:

$$\text{Accept Packet} \iff (\text{Packet.SysID} == \text{SYSID_TARGET}) \vee (\text{SYSID_TARGET} == 0)$$

The Engineer's View

Defined in `AntennaTracker/Parameters.cpp`.

- **0 (Auto):** The tracker will lock onto the first vehicle it receives a heartbeat from. Convenient for solo operations.
- **1-255:** The tracker will strictly ignore all MAVLink traffic except that from the specified System ID.

Tuning & Behavior

- **Default Value:** 0
- **Recommendation:** Set to match your vehicle's `SYSID_THISMAV` (usually 1) to prevent accidental tracking of other aircraft.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





SYSID_THISMAV (MAVLink System ID)

DEFAULT: 1	RANGE: 1 255

MAVLink System ID (SYSID_THISMAV)

Description

`SYSID_THISMAV` sets the "Address" of your drone on the MAVLink network.

If you are flying multiple drones at the same time with the same Ground Control Station (GCS), each drone **must** have a unique `SYSID_THISMAV`. If two drones have the same ID, the GCS will get confused and may display erratic data or fail to send commands correctly.

- **Default:** 1.
- **Recommended Range:** 1 to 255.

Tuning & Behavior

- **Default:** 1.
- **Action:** Change this to any unique number (e.g., Drone 1 = 1, Drone 2 = 2) when performing multi-vehicle operations.
- **Compatibility:** Most MAVLink systems expect the GCS to be ID 255. Avoid setting your drone to 255.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **TARG** parameter group (specifically `TARG_SP`) configures the target setpoints for specialized navigation modes.

Key Concepts

1. Autorotation Glide Speed

- `TARG_SP`: Defines the target ground speed (cm/s) for the helicopter autorotation controller to maintain during the glide phase.

Developer Notes

- **Library:** `libraries/AC_Autorotation`.

TARG_SP (Target Glide Ground Speed)

DEFAULT: 1500	RANGE: 800 2000	UNITS: cm/s

Target Glide Ground Speed (TARG_SP)

Description

This parameter sets the target ground speed for the autonomous autorotation controller during the glide phase. Maintaining the correct forward speed is critical for keeping the rotor RPM in the green arc and reaching the designated landing spot during an engine failure scenario.

The Mathematics

PID Setpoint.

The Engineer's View

Defined in `libraries/AC_Autorotation/AC_Autorotation.cpp`.

Tuning & Behavior

- **Default Value:** 1500 cm/s (15 m/s)
- **Tuning:** Adjust based on the specific best-glide speed of your helicopter.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





TEMPERATURE CALIBRATION (TCAL)

Overview

The **TCAL** parameter group configures the **Sensor Temperature Calibration** system. All MEMS sensors (Accelerometers, Gyroscopes, Barometers) suffer from "Thermal Drift"—their zero-point and scale changes as the board heats up.

TCAL uses mathematical models to compensate for this drift in real-time, ensuring a stable horizon and altitude reference from a cold startup through to high-power flight.

Key Concepts

1. Calibration Models

The autopilot records sensor behavior across a range of temperatures (defined by `TCAL_TEMP_MIN` and `MAX`) and fits a polynomial curve to the data.

- `TCAL_BARO_EXP`: The learned exponent for barometer temperature correction.

2. Auto-Learning

Modern ArduPilot can "learn" these coefficients automatically during the warm-up period while the drone is disarmed.

Parameter Breakdown

- `TCAL_ENABLED`: Master switch.
- `TCAL_OPTIONS`: Bitmask for learning behaviors.
- `TCAL_TEMP_MIN / MAX`: The valid temperature range for the calibration model.

Integration Guide

- **Recommendation:** Leave `TCAL_ENABLED = 1` for most modern hardware. If you see a drifting horizon after takeoff, perform a factory temperature calibration (requires a freezer and a heat source—see Wiki).

Developer Notes

- **Library:** `libraries/AP_TempCalibration`.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





TCAL_BARO_EXP (Temperature Calibration barometer exponent)

DEFAULT: 0

Temperature Calibration barometer exponent (TCAL_BARO_EXP)

Description

This parameter represents the **Learned Exponent** for the barometer's temperature calibration curve. It is automatically adjusted by the `AP_TempCalibration` library when the vehicle is disarmed and undergoing temperature changes (e.g., self-heating after boot). It ensures that the altitude reading remains stable even as the autopilot's temperature fluctuates.

The Mathematics

The calibration model corrects the raw pressure reading based on the temperature difference from a reference point:

$$P_corrected \approx P_raw + \text{TCAL_BARO_EXP} \cdot (T - T_cal)$$

(Note: This is a simplified representation of the internal polynomial fit).

The Engineer's View

Defined in `libraries/AP_TempCalibration/AP_TempCalibration.cpp`. It is part of the `AP_TempCalibration` system which monitors IMU and Baro drift vs. temperature.

- **Do Not Edit:** This value is learned. Manually changing it will invalidate the calibration.
- **Reset:** Setting to 0 resets the learning for this term.

Tuning & Behavior

- **Default Value:** 0
- **Behavior:** Should converge to a non-zero value over time if the board has significant temperature sensitivity.

TCAL_ENABLED (Temperature calibration enable)

DEFAULT: 0	RANGE: 0 2	UNITS: Option

Temperature calibration enable (TCAL_ENABLED)

Description

This parameter enables the advanced temperature calibration system. Many MEMS sensors (especially barometers like the ICM-20789) suffer from "thermal drift," where the reported values change significantly as the electronics heat up during operation.

Setting this parameter allows the autopilot to either "Learn" a correction model for your specific hardware or "Apply" previously learned corrections to ensure stable altitude and attitude estimates across varying temperatures.

The Mathematics

The system operates as a simple state machine:

- **0: Disabled.** No thermal corrections are applied.
- **1: Enabled (Use Learned).** The autopilot applies the learned exponent (e.g., `BARO_EXP`) to the raw sensor data.
- **2: Enable and Learn.** While the vehicle is stationary and disarmed, the system records sensor drift vs. temperature and uses a least-squares fit to calculate the correction model. Once a sufficient temperature range is covered, it updates the calibration parameters.

The Engineer's View

In `AP_TempCalibration.cpp`, the `update()` function runs at 10Hz:

1. If set to `TC_ENABLE_LEARN` (2), it calls `learn_calibration()`. This requires the vehicle to be perfectly still (`AP::ins().is_still()`) and disarmed.
2. If set to `TC_ENABLE_USE` (1) or `TC_ENABLE_LEARN` (2), it calls `apply_calibration()`.
3. The actual correction is applied to the barometer driver via `baro.set_pressure_correction()`.

Tuning & Behavior

- **Learning Procedure:** To calibrate a new board, set `TCAL_ENABLED=2` and let the vehicle sit stationary for 10-15 minutes while it warms up.
- **Verification:** Check the `TCAL_TEMP_MIN` and `TCAL_TEMP_MAX` parameters to see the range covered during learning.
- **Final Step:** Once you are happy with the stability, you can set this to **1** to stop the learning process and just use the saved model.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





TCAL_OPTIONS (Temperature Calibration Options)

DEFAULT: 0	RANGE: null	UNITS: Bitmask

Temperature Calibration Options (TCAL_OPTIONS)

Description

This parameter provides advanced control over how thermal calibration data is stored and managed. It specifically enables the "Persistence" feature, which allows the autopilot to save calibrated sensor values into a special protected area of memory (the bootloader sector).

Saving values here ensures that your high-fidelity temperature calibration remains intact even if you perform a "Reset to Defaults" on all other parameters. It is highly recommended for industrial-grade vehicles where re-performing a 20-minute thermal calibration in the field is not practical.

The Mathematics

This is a bitmask (B):

- **Bit 0 (Value 1): PersistTemps.** Saves the temperature-related calibration constants to the bootloader sector.
- **Bit 1 (Value 2): PersistAccels.** Saves the accelerometer-related thermal constants to the bootloader sector.

The Engineer's View

In `AP_InertialSensor.cpp`, this maps to `tcal_options`.

- **Safety:** The bootloader sector is used because it is rarely erased and exists outside the standard parameter storage area.
- **Transfer:** During the next bootloader update or specific GCS command, the autopilot will copy the volatile RAM-based calibration into this persistent storage.
- **Recall:** On every subsequent boot, ArduPilot checks the bootloader sector first. If valid calibration is found there, it is loaded into the active sensor drivers automatically.

Tuning & Behavior

- **Default Value:** 0 (Persistence disabled; values stored in standard EEPROM only).
- **Recommended:** Set to **3 (Bit 0 + Bit 1)** once you have achieved a "Perfect" thermal calibration.
- **Industrial Use:** This is standard for factory-calibrated aircraft where the sensor drift profiles are fixed for the life of the airframe.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY

TCAL_TEMP_MAX (Calibration temperature maximum)

DEFAULT: 0	RANGE: null	UNITS: degC

Calibration temperature maximum (TCAL_TEMP_MAX)

Description

This parameter records the maximum temperature reached during a successful "Enable and Learn" session for temperature calibration.

Thermal calibration works by watching how a sensor drifts as the flight controller heats up from its starting temperature to its steady-state operating temperature. `TCAL_TEMP_MAX` tells you the "Top End" of the data set used to create your correction model.

The Mathematics

The system tracks the highest temperature (T_max) encountered while the vehicle was stationary and learning:

$$TCAL_TEMP_MAX = T_start + (\text{LastValidIndex} \times \text{StepSize})$$

The `StepSize` is typically **0.25°C**. This value defines the upper boundary of the "Trusted Zone" for the correction exponent.

The Engineer's View

In `AP_TempCalibration::calculate_calibration()` (`libraries/AP_TempCalibration/AP_TempCalibration.cpp`):

1. The code determines the current progress of the learning buffer (`learn_i`).
2. It calculates the maximum temperature point reached in the data array.
3. It updates `temp_max` and saves it to EEPROM along with the new calibration exponent.
4. This provides a clear audit trail for the quality of the calibration.

Tuning & Behavior

- **Default Value:** 0.
- **Automatic:** Set by the system. Do not edit manually.
- **Troubleshooting:** If this value is very close to `TCAL_TEMP_MIN`, it means the vehicle didn't have enough time to warm up (or the environment was too cold), and the resulting calibration might be inaccurate. Aim for at least a **10-15°C** difference between MIN and MAX for a high-fidelity calibration.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





TCAL_TEMP_MIN (Calibration temperature minimum)

DEFAULT: 0	RANGE: null	UNITS: degC

Calibration temperature minimum (TCAL_TEMP_MIN)

Description

This parameter records the starting (lowest) temperature of the range over which the temperature calibration model was learned.

For a thermal correction model (like `BARO_EXP`) to be valid, it needs to be calculated based on data from a meaningful temperature change. This parameter, along with `TCAL_TEMP_MAX`, allows the pilot to verify that the learning process covered a wide enough range of temperatures to accurately model the sensor's drift.

The Mathematics

During the learning phase (`TCAL_ENABLED=2`), the system records the initial temperature (T_start). Once the least-squares fit is calculated and the calibration is saved:

$$TCAL_TEMP_MIN = T_start$$

This serves as a geofence for the calibration validity; corrections applied far outside the `MIN`/`MAX` range may be less accurate.

The Engineer's View

In `AP_TempCalibration::calculate_calibration()` (`libraries/AP_TempCalibration/AP_TempCalibration.cpp`):

1. The code calculates the optimal `baro_exponent`.
2. Upon a successful calculation, it saves the `learn_temp_start` value into `temp_min`.
3. The value is marked as `Volatile` in the parameter table, meaning it is expected to change during runtime as learning progresses.

Tuning & Behavior

- **Default Value:** 0.
- **Automatic:** This value is set automatically by the autopilot. You should not edit it manually.
- **Verification:** If `TCAL_TEMP_MAX` minus `TCAL_TEMP_MIN` is less than 10 degrees, the calibration might not be robust enough for all seasons.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





TOTAL ENERGY CONTROL SYSTEM (TECS)

Overview

The **TECS** parameter group configures the **Total Energy Control System**. This is the core altitude and airspeed management algorithm for **ArduPlane** (and QuadPlane).

TECS is a sophisticated "Energy Balancer." It treats the aircraft's energy as the sum of its Potential Energy (Altitude) and Kinetic Energy (Airspeed). It then uses the Throttle to change the *total* energy and the Elevator to *exchange* energy between altitude and airspeed.

Key Concepts

1. Energy Balancing

If the plane is too slow and too low, TECS increases throttle (add energy).
If the plane is too fast but too low, TECS pulls the nose up (exchange kinetic energy for potential energy).

2. Limits (`TECS_CLMB_MAX`, `TECS_SINK_MAX`)

Defines the physical performance envelope of the aircraft.

- `CLMB_MAX`: The steepest sustained climb rate the motor can provide (m/s).
- `SINK_MAX`: The fastest safe descent rate (m/s).

3. Weighting (`TECS_SPDWEIGHT`)

Defines the priority between speed and altitude.

- **1.0 (Default):** Equal priority.
- **2.0:** Prioritize Airspeed (essential for preventing stalls).
- **0.0:** Prioritize Altitude (glide slope tracking).

4. Synthetic Airspeed (`TECS_SYNAIRSPEED`)

Allows the plane to fly without a pitot tube by estimating airspeed using GPS ground speed and a wind model.

Parameter Breakdown

- `TECS_TIME_CONST`: The overall responsiveness of the controller (seconds). Smaller = Aggressive.
- `TECS_PTCH_DAMP`: Damping for pitch movements during energy exchange.
- `TECS_FLARE_HGT`: Altitude to begin the landing flare.

Integration Guide

1. **Preparation:** Ensure your `AIRSPEED_MIN/MAX` and `THROTTLE_CRUISE` are set correctly.
2. **Tuning:** If the plane "porpoises" (oscillates altitude) during a mission, **increase** `TECS_TIME_CONST`.
3. **No Pitot:** If you don't have an airspeed sensor, set `TECS_SYNAIRSPEED = 1`.

Developer Notes

- **Library:** `libraries/AP_TECS`.
- **Mathematics:** Based on the principle of work-energy. The controller minimizes the error in total energy and energy balance.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





TECS_APPR_SMAX (Sink rate max for landing approach stage)

DEFAULT: 0	RANGE: 0.0 20.0	UNITS: m/s

Sink rate max for landing approach stage (TECS_APPR_SMAX)

Description

This parameter allows for a steeper maximum descent rate specifically during the auto-landing approach phase, overriding the general `TECS_SINK_MAX` limit. This is useful for aircraft that need to descend rapidly to clear obstacles before the runway threshold.

The Mathematics

$$\dot{h}_\max = (\text{Land Approach})?\text{TECS_APPR_SMAX} : \text{TECS_SINK_MAX}$$

The Engineer's View

Defined in `libraries/AP_TECS/AP_TECS.cpp`.

- **0:** Disabled. Uses `TECS_SINK_MAX` for all flight phases.
- **>0:** Sets the limit in m/s.

Tuning & Behavior

- **Default Value:** 0
- **Warning:** Ensure your airframe can handle the airspeed buildup or has enough drag (flaps/reverse thrust) to descend this fast without overspeeding.

TECS_CLMB_MAX (Maximum Climb Rate)

DEFAULT: 5.0	RANGE: 1 15	UNITS: m/s

Maximum Climb Rate (TECS_CLMB_MAX)

Description

`TECS_CLMB_MAX` sets the "Speedometer Limit" for going up.

Even if you command a 100-meter climb, the drone will limit its vertical ascent speed to this value. This ensures the climb is steady and doesn't exceed the motor's power limits or cause a high-pitch stall.

Tuning & Behavior

- **Default Value:** 5.0 m/s.
- **Recommendation:** Set this to roughly **75%** of your plane's physical maximum climb rate at full throttle.

TECS_FLARE_HGT (Flare Height)

DEFAULT: 1.0	RANGE: 0 15	UNITS: m

Flare Height (TECS_FLARE_HGT)

Description

`TECS_FLARE_HGT` defines the start of the "Touchdown" phase during an autonomous landing.

When the drone's altitude above the ground (via rangefinder) drops below this value, the autopilot enters the "Flare" state. In this state, it stops following the steep glide slope and instead tries to maintain a very gentle descent rate (`TECS_LAND_SINK`) while raising the nose to bleed off airspeed.

- **Standard Setting:** 1.0 to 2.0 meters.

Tuning & Behavior

- **Default:** 1.0.
- **Too Low:** The plane might hit the ground too hard before it has a chance to slow down.
- **Too High:** The plane will "float" along the runway for a long time, potentially running out of space.
- **Dependencies:** Requires an enabled and functional rangefinder (RNGFND_LANDING).

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





TECS_HDEM_TCONST (Height Demand Time Constant)

DEFAULT: 3.0	RANGE: 1.0 5.0	UNITS: s

Height Demand Time Constant (TECS_HDEM_TCONST)

Description

`TECS_HDEM_TCONST` defines how "Softly" the autopilot reacts when a new altitude is requested.

When a mission or a pilot (in FBWB mode) changes the target altitude, the autopilot doesn't try to reach it instantly. Instead, it passes the request through a low-pass filter to create a smooth climb or descent path. This parameter sets the "Slowness" of that filter.

- **Higher Value (e.g. 5.0):** Very smooth, gradual altitude changes.
- **Lower Value (e.g. 1.0):** More aggressive reaction to altitude commands.

Tuning & Behavior

- **Default:** 3.0 seconds.
- **Recommendation:** Leave at **3.0** for standard transport or mapping drones. Reduce to **1.5 or 2.0** for high-performance aircraft that need to react quickly to terrain changes.
- **Effect:** Increasing this value reduces "Pitch pumping" (oscillations in pitch) during altitude transitions.

TECS_HGT_OMEGA (Height Estimator Bandwidth)

DEFAULT: 3.0	RANGE: 1.0 10.0	UNITS: rad/s

Height Estimator Bandwidth (TECS_HGT_OMEGA)

Description

`TECS_HGT_OMEGA` determines how much the autopilot trusts its sensors for altitude.

ArduPlane calculates its climb rate by blending the **Barometer** (slow but stable) with the **Accelerometers** (fast but drifty).

- **High Value (e.g. 5.0):** Trusts the Barometer more. Good if your IMU is vibey.
- **Low Value (e.g. 1.0):** Trusts the IMU more. Provides instantaneous climb rate detection. Better for high-performance acrobatics.

Tuning & Behavior

- **Default Value:** 3.0 rad/s.
- **Recommendation:** Leave at **3.0**. Only increase if you see the plane "ballooning" (pitching up) during rapid acceleration due to EKF height spikes.

TECS_INTEG_GAIN (TECS Integrator Gain)

DEFAULT: 0.1	RANGE: 0 0.5

TECS Integrator Gain (TECS_INTEG_GAIN)

Description

`TECS_INTEG_GAIN` handles "Persistence."

If the plane is consistently flying 5 meters below the target altitude, the I-term builds up more throttle or pitch to force it up to the correct height.

Tuning & Behavior

- **Default Value:** 0.1.
- **Too High:** The plane will overshoot its target and then oscillate up and down slowly.
- **Too Low:** The plane will stay slightly below its target altitude or above its target speed indefinitely.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





TECS_LAND_ARSPD (TECS Landing Airspeed)

DEFAULT: -1	RANGE: -1 40	UNITS: m/s

TECS Landing Airspeed (TECS_LAND_ARSPD)

Description

`TECS_LAND_ARSPD` overrides the cruise speed during the final approach.

Normally, the plane flies at `AIRSPEED_CRUISE`. But for landing, you usually want to fly slower (closer to stall speed) to reduce energy.

Tuning & Behavior

- **-1 (Default):** Use `AIRSPEED_CRUISE`.
- **Value:** Set to **1.3 x Stall Speed** for a safe, slow approach.

TECS_LAND_DAMP (TECS Landing Damping)

DEFAULT: 0.5	RANGE: 0.1 1.0

TECS Landing Damping (TECS_LAND_DAMP)

Description

`TECS_LAND_DAMP` smoothes out the pitch commands during the landing phase.

If the plane "Pumps" the elevator (rapidly pitching up and down) while trying to hold the glide slope, increasing this value can help dampen those oscillations.

Tuning & Behavior

- **Default Value:** 0.5.

TECS_LAND_IGAIN (TECS Land Integrator Gain)

DEFAULT: 0	RANGE: 0 0.5

TECS Land Integrator Gain (TECS_LAND_IGAIN)

Description

`TECS_LAND_IGAIN` allows you to have a different "Error Memory" for the final landing approach.

Normally, the flight controller uses `TECS_INTEG_GAIN`. However, during landing, ground effect and changing air density near the floor can cause persistent height errors. This parameter lets you set a dedicated integrator to trim out these errors for a more precise touchdown.

- **0 (Default):** Use the standard `TECS_INTEG_GAIN`.

Tuning & Behavior

- **Default Value:** 0.
- **Recommendation:** If your plane consistently lands too short or too long due to wind/drag offsets, try setting this to **0.2**.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





TECS_LAND_PDAMP (Landing Pitch Damping)

DEFAULT: 0	RANGE: 0 1.0

Landing Pitch Damping (TECS_LAND_PDAMP)

Description

`TECS_LAND_PDAMP` allows for a more "Solid" pitch response during the critical landing approach.

Landing often happens in ground effect or turbulent air near the runway, which can cause the plane's nose to bounce or oscillate. This parameter allows you to add extra damping specifically for the landing phase without affecting the feel of the aircraft during normal cruise.

- **0 (Default):** Uses the standard TECS_PTCH_DAMP value.
- **Non-Zero:** Overrides the standard damping with this value.

Tuning & Behavior

- **Default:** 0.
- **Recommendation:** If your plane "Pitch pumps" (constantly moves the nose up and down) while on the landing glide slope, set this to **0.5** or **0.8**.
- **Effect:** Higher values make the pitch response more sluggish but significantly more stable against external disturbances.

TECS_LAND_PMAX (TECS Landing Pitch Max)

DEFAULT: 0	RANGE: 0 45	UNITS: deg

TECS Landing Pitch Max (TECS_LAND_PMAX)

Description

`TECS_LAND_PMAX` prevents the autopilot from pulling the nose up too high during flare.

If the plane flares too aggressively, it might balloon up or strike its tail on the runway.

Tuning & Behavior

- **0 (Default):** Use normal pitch limits (`LIM_PITCH_MAX`).
- **Recommendation:** Set to **10 or 15 degrees** to ensure a safe flare attitude.

TECS_LAND_SINK (Sink Rate for Final Landing Stage)

DEFAULT: 0.25	RANGE: 0.0 2.0	UNITS: m/s

Sink Rate for Final Landing Stage (TECS_LAND_SINK)

Description

`TECS_LAND_SINK` is the "Soft Landing" target.

When the plane is just a few meters above the ground, it enters the **Flare** stage. Instead of following a steep glide slope, it aims for this very gentle descent rate to ensure it touches down without bouncing or breaking the landing gear.

- **0.25 (Default):** A very soft touchdown (0.5 knots descent).
- **0.5:** A firm landing. Good for rough grass where you want the wheels to dig in.

Tuning & Behavior

- **Default Value:** 0.25 m/s.
- **Recommendation:** If your plane consistently "Floats" and refuses to touch down, increase this to **0.5**. If your plane "Slam-Lands" and bounces, decrease to **0.15**.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY

TECS_LAND_SPDWGT (TECS Landing Speed Weight)

DEFAULT: -1.0	RANGE: -1.0 2.0

TECS Landing Speed Weight (TECS_LAND_SPDWGT)

Description

`TECS_LAND_SPDWGT` is the "Safety vs. Accuracy" slider.

- **2.0 (Speed Priority):** The plane will aggressively pitch down to regain airspeed if it gets too slow, even if that means diving below the glide slope. Prevents stalls.
- **0.0 (Height Priority):** The plane will prioritize staying on the glide slope, even if airspeed drops dangerously low.
- **-1.0 (Default):** Uses the standard `TECS_SPDWEIGHT` value.

Tuning & Behavior

- **Default Value:** -1.0.
- **Recommendation:** Set to **2.0** if you are landing in gusty winds and stall prevention is critical.

TECS_LAND_SRC (TECS Landing Height Source)

DEFAULT: 0	RANGE: 0 2

TECS Landing Height Source (TECS_LAND_SRC)

Description

`TECS_LAND_SRC` tells the plane what "Ground" is.

- **0 (Baro/GPS mix):** Standard. Lands at the altitude of the Home waypoint (Relative 0). Good for flat fields.
- **1 (Rangefinder):** Uses Lidar/Sonar. Required for precision auto-landing. The plane will follow the actual contour of the ground.
- **2 (Terrain DB):** Uses map data stored on the SD card. Useful for landing in hilly areas if you don't have a Lidar.

Tuning & Behavior

- **Default Value:** 0.
- **Recommendation:** Use **1 (Rangefinder)** if you have a Lidar installed (`RNGFND_LANDING` enabled).

TECS_LAND_TCONST (TECS Landing Time Constant)

DEFAULT: 2.0	RANGE: 0 5	UNITS: s

TECS Landing Time Constant (TECS_LAND_TCONST)

Description

`TECS_LAND_TCONST` controls how aggressively the plane corrects altitude errors during landing.

- **Higher Value:** Slower, smoother corrections. Better for calm weather.
- **Lower Value:** Faster corrections. Better for turbulent conditions where the plane gets knocked off the glide slope.

Tuning & Behavior

- **Default Value:** 2.0 seconds.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY

TECS_LAND_TDAMP (Landing Throttle Damping)

DEFAULT: 0	RANGE: 0 1.0

Landing Throttle Damping (TECS_LAND_TDAMP)

Description

`TECS_LAND_TDAMP` allows for more aggressive throttle corrections during an autonomous landing.

During an approach, the plane must stay precisely on the glide slope while maintaining airspeed. If the throttle response is too slow, the plane might fall behind the power curve. This parameter allows you to add damping (which actually increases throttle activity to counteract speed/height errors) specifically for the landing phase.

- **0 (Default):** Uses the standard TECS_THR_DAMP value.
- **Non-Zero:** Overrides the standard damping with this value.

Tuning & Behavior

- **Default:** 0.
- **Recommendation:** If your plane's airspeed varies wildly during the approach, or if it struggles to stay on the glide slope in wind, try setting this to **0.5**.
- **Effect:** Higher values result in more frequent and faster throttle movements to maintain the energy state.

TECS_LAND_THR (Cruise Throttle During Landing)

DEFAULT: -1	RANGE: -1 100	UNITS: %

Cruise Throttle During Landing (TECS_LAND_THR)

Description

`TECS_LAND_THR` is for planes **without** an airspeed sensor.

Normally, TECS calculates the throttle needed to maintain landing speed. If there is no speed sensor, it can't do that. This parameter allows you to hard-code a fixed throttle percentage (e.g. 40%) that you know is safe for a slow descent on your specific airframe.

- **-1 (Default):** Disabled (Use speed-based calculation).
- **45:** Hold 45% throttle during the entire landing approach.

Tuning & Behavior

- **Default Value:** -1.
- **Recommendation:** If you don't have an airspeed sensor, set this to the lowest throttle that keeps your plane flying safely above stall speed (e.g. 35-45%).

TECS_OPTIONS (Extra TECS Options)

DEFAULT: 0	RANGE: 0 3

Extra TECS Options (TECS_OPTIONS)

Description

`TECS_OPTIONS` unlocks advanced cruise behaviors for the Total Energy Control System.

- **Bit 0 (1): Glider Only.**
  If set, ArduPilot treats the motor as a "Launcher" only. Once at altitude, it optimizes everything for zero-thrust glide performance.
- **Bit 1 (2): Allow Descent Speedup.**
  Normally, ArduPilot tries to maintain your cruise speed during a descent. If this bit is set, and the drone needs to descend rapidly, it is allowed to increase its airspeed to help bleed off potential energy faster. This makes descents much quicker but carries a risk of overspeeding if the airframe is fragile.

Tuning & Behavior

- **Default Value:** 0.
- **Recommendation:**
  - **Long-Range Mapping:** Enable **Bit 1** to save time on mission returns (descending from survey altitude).
  - **Soaring Glider:** Enable **Bit 0**.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





TECS_PITCH_MAX (Maximum Pitch in Auto Flight)

DEFAULT: 15	RANGE: 0 45	UNITS: deg

Maximum Pitch in Auto Flight (TECS_PITCH_MAX)

Description

`TECS_PITCH_MAX` acts as a "Nose Limiter" for autonomous flight.

While the global `PTCH_LIM_MAX` sets the hard physical limit for the whole aircraft, `TECS_PITCH_MAX` is used specifically by the climb controller. It ensures that the drone doesn't point its nose so steeply into the sky that it loses all its airspeed, even if the flight controller thinks it needs to climb faster.

Tuning & Behavior

- **Default Value:** 15°.
- **Recommendation:** Set to the angle your plane can comfortably sustain at full throttle without losing airspeed.
- **Gliders:** Can be set higher (25°) for aggressive climbing.
- **Heavy Cargo:** Set lower (10°) to ensure the plane always keeps the nose down enough for speed.

TECS_PITCH_MIN (Minimum Pitch in Auto Flight)

DEFAULT: 0	RANGE: -45 0	UNITS: deg

Minimum Pitch in Auto Flight (TECS_PITCH_MIN)

Description

`TECS_PITCH_MIN` prevents the drone from "Diving" too steeply when it needs to lose altitude.

- **0 (Default):** The drone will not point its nose down during autonomous flight. It will only descend by reducing throttle.
- **-10:** The drone can point its nose 10° down to help reach its target altitude faster.

Tuning & Behavior

- **Default Value:** 0.
- **Recommendation:** Use **-5 or -10** if you find the drone "Over-climbs" and takes too long to get back down to its target altitude.

TECS_PTCH_DAMP (Pitch Damping Gain)

DEFAULT: 0.3	RANGE: 0.1 1.0

Pitch Damping Gain (TECS_PTCH_DAMP)

Description

`TECS_PTCH_DAMP` prevents the plane from "Nodding" its head.

When the autopilot commands a pitch change to maintain height, it needs to stop that movement at the right time. Damping provides the "Brakes" for the pitch maneuver.

- **Higher Value:** Less oscillation, but too high can cause the pitch to be sluggish and unresponsive.
- **Lower Value:** More active pitch, but can lead to a "Porpoising" motion where the drone constantly bobs up and down.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





TECS_PTCH_FF_K (Pitch Feed-Forward Gain)

DEFAULT: 0.0	RANGE: -5.0 0.0	UNITS: rad/(m/s)

Pitch Feed-Forward Gain (TECS_PTCH_FF_K)

Description

`TECS_PTCH_FF_K` allows the autopilot to "Anticipate" how much pitch is needed to achieve a target airspeed, rather than waiting for an airspeed error to build up.

When you demand a higher airspeed, the plane naturally needs to point its nose down. This feed-forward gain applies that pitch command immediately.

- **0.0 (Default):** Feature is disabled.
- **Negative Values:** Apply downward pitch for increased speed demand.

Tuning & Behavior

- **Default:** 0.
- **Recommendation (Gliders):** Start with **-0.04**.
- **Recommendation (Draggy planes):** Start with **-0.08**.
- **Effect:** Reduces the "Lag" in airspeed tracking during altitude changes. If the nose drops too aggressively when you speed up, reduce the magnitude (make it less negative).

TECS_PTCH_FF_V0 (Pitch Feed-Forward Reference Speed)

DEFAULT: 12.0	RANGE: 5.0 50.0	UNITS: m/s

Pitch Feed-Forward Reference Speed (TECS_PTCH_FF_V0)

Description

`TECS_PTCH_FF_V0` provides the baseline for the TECS_PTCH_FF_K feature.

It represents the natural "Neutral Speed" of the aircraft – the speed at which it glides with a neutral pitch command. When the autopilot demands exactly this speed, no feed-forward correction is applied. If it demands more speed, it pitches down. If it demands less, it pitches up.

Tuning & Behavior

- **Default:** 12.0 m/s.
- **Calibration:** Fly the plane in Manual or FBWA with neutral elevator trim and observe the resulting steady-state airspeed. Enter that value here.
- **Requirement:** This parameter is only used if `TECS_PTCH_FF_K` is non-zero.

TECS_RLL2THR (Roll to Throttle Compensation)

DEFAULT: 10	RANGE: 0 50

Roll to Throttle Compensation (TECS_RLL2THR)

Description

`TECS_RLL2THR` is the "Turn Gas."

When a plane banks into a turn, it naturally slows down and loses lift. This parameter automatically "Pushes the Gas" when the wings are tilted, ensuring the plane maintains its airspeed and altitude through the maneuver.

Tuning & Behavior

- **Default Value:** 10.
- **Recommendation:** If the plane loses too much speed in sharp turns, increase this value. If it speeds up excessively while turning, decrease it.
- **Physics:** This is a feed-forward term based on the bank angle.

$$\text{Throttle_}extra = k \cdot (1 - \cos(\text{roll}))$$

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





TECS_SINK_MAX (Maximum Descent Rate)

DEFAULT: 5.0	RANGE: 1 15	UNITS: m/s

Maximum Descent Rate (TECS_SINK_MAX)

Description

`TECS_SINK_MAX` prevents the drone from diving too steeply.

While gravity is free, diving too fast can lead to structural failure or exceed the plane's "Vne" (Never Exceed Speed). This parameter caps the rate of descent to a safe, controlled speed.

TECS_SINK_MIN (Minimum Glide Sink Rate)

DEFAULT: 2.0	RANGE: 0.1 10.0	UNITS: m/s

Minimum Glide Sink Rate (TECS_SINK_MIN)

Description

`TECS_SINK_MIN` defines the "Glide Profile" of your aircraft.

It represents the vertical descent rate the plane naturally experiences when flying at its cruise speed with the engine at idle or turned off. This is a fundamental physical constant for the TECS (Total Energy Control System) to understand how much potential energy (altitude) is lost simply to overcome aerodynamic drag.

Tuning & Behavior

- **Default:** 2.0 m/s.
- **Calibration:** Fly the plane in Manual or FBWA mode, cut the throttle, and maintain a level cruise speed. Note your rate of descent on the GCS. Enter that value here.
- **Gliders:** Typically have a very low sink rate (**0.5 to 1.0 m/s**).
- **Draggy Planes:** (Like a Cub or a loaded cargo plane) may have a sink rate of **3.0 to 5.0 m/s**.
- **Effect:** If this value is too high, the drone will undershoot its glide path during autonomous landings. If too low, it will overshoot (float).

TECS_SPDWEIGHT (Speed vs Altitude Weight)

DEFAULT: 1.0	RANGE: 0 2.0

Speed vs Altitude Weight (TECS_SPDWEIGHT)

Description

`TECS_SPDWEIGHT` is the "Priority Slider" for Plane flight control.

TECS (Total Energy Control System) views the plane as a battery. Altitude is "Potential Energy" and Airspeed is "Kinetic Energy." This parameter tells the autopilot which one is more important to maintain.

- **1.0 (Balanced):** The autopilot treats speed and altitude errors equally. It will use both pitch and throttle to fix errors.
- **2.0 (Speed Priority):** The autopilot will prioritize maintaining airspeed at all costs. If the plane slows down, it will immediately trade altitude (pitch down) to regain speed. (Safest for preventing stalls).
- **0.0 (Altitude Priority):** The autopilot will prioritize holding its height. It will trade airspeed to stay at the target altitude. (Riskier, as it could lead to a stall if the motor is weak).

Tuning & Behavior

- **Default Value:** 1.0.
- **Recommendation:** Use **2.0** for takeoffs and landings to ensure you never stall near the ground. Use **1.0** for general cruising.
- **Physics:**
  - $$\text{Total Energy} = \text{Height} + \frac{\text{Velocity}^2}{2g}$$
  - `SPDWEIGHT` adjusts the weighting of the Kinetic Energy term in the pitch and throttle demand calculations.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





TECS_SPD_OMEGA (Speed Estimator Bandwidth)

DEFAULT: 2.0	RANGE: 0.5 5.0	UNITS: rad/s

Speed Estimator Bandwidth (TECS_SPD_OMEGA)

Description

`TECS_SPD_OMEGA` determines how fast the autopilot reacts to speed changes.

It blends the **Airspeed Sensor** with the **longitudinal Accelerometer**.

- **High Value:** Trust the Airspeed Sensor more. Fast reaction to gusts, but noisy if the sensor isn't well-protected.
- **Low Value:** Trust the Inertial navigation more. Smooths out "jumps" in airspeed caused by turbulence.

Tuning & Behavior

- **Default Value:** 2.0 rad/s.
- **Recommendation:** Leave at **2.0**.

TECS_SYNAIRSPEED (Synthetic Airspeed Enable)

DEFAULT: 0	RANGE: 0 1

Synthetic Airspeed Enable (TECS_SYNAIRSPEED)

Description

`TECS_SYNAIRSPEED` allows the autopilot to "Guess" the airspeed using internal sensors (accelerometers, gyros, and GPS) instead of a physical pitot tube.

This is extremely useful for small or budget drones where mounting a pitot tube is difficult. However, it is **significantly less accurate** than a real sensor, especially in gusty conditions or during aggressive maneuvers, as it relies heavily on a perfect wind estimate.

Tuning & Behavior

- **0: Disabled (Default).** The autopilot will use its internal "Ground Speed to Airspeed" mapping or a physical sensor if available.
- **1: Enabled.** The autopilot uses the EKF's synthetic airspeed estimate for longitudinal control.
- **Warning:** Do not enable this unless you have verified that your drone has a stable and accurate wind estimate in flight. If the wind estimate is wrong, the synthetic airspeed will be wrong, which can lead to a stall or over-speed condition.

TECS_THR_DAMP (Throttle Damping)

DEFAULT: 0.5	RANGE: 0.1 1.0

Throttle Damping (TECS_THR_DAMP)

Description

`TECS_THR_DAMP` prevents the motor from "Pulsing" or hunting for the correct speed.

It smooths out the throttle commands sent to the ESC. If you hear your motor constantly changing pitch (speeding up and slowing down) while in a steady cruise, your damping might be too low.

Tuning & Behavior

- **Default Value:** 0.5.
- **Recommendation:** If the motor "Surges," increase this value by 0.1.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





TECS_TIME_CONST (TECS Time Constant)

DEFAULT: 5.0	RANGE: 3.0 10.0	UNITS: s

TECS Time Constant (TECS_TIME_CONST)

Description

`TECS_TIME_CONST` defines how fast the plane tries to fix altitude and speed errors.

It is effectively the "Damping" of the whole navigation system.

- **Small Value (3.0s):** Fast, aggressive control. The plane will snap back to the target altitude and speed quickly. Can cause the motor to pulse and the nose to bob.
- **Large Value (10.0s):** Smooth, relaxed control. The plane takes its time to reach the target. Good for long-range efficient cruising.

Tuning & Behavior

- **Default Value:** 5.0 seconds.
- **Recommendation:** Leave at **5.0** for most planes. Increase to **7.0** for very large, slow UAVs to prevent the autopilot from over-correcting.

TECS_TKOFF_IGAIN (TECS Takeoff Integrator Gain)

DEFAULT: 0	RANGE: 0 0.5

TECS Takeoff Integrator Gain (TECS_TKOFF_IGAIN)

Description

`TECS_TKOFF_IGAIN` allows you to have a different "Memory" setting for takeoff.

Normally, `TECS_INTEG_GAIN` handles long-term errors. However, during takeoff, the dynamics change rapidly (ground effect, rapid acceleration). This parameter overrides the standard gain to ensure the plane tracks the climb rate accurately during this critical phase.

- **0 (Default):** Use `TECS_INTEG_GAIN`.
- **> 0:** Use this specific gain for takeoff.

Tuning & Behavior

- **Default Value:** 0.
- **Recommendation:** If your plane tends to overshoot or undershoot the climb angle right after launch, try setting this to **0.4**.

TECS_VERT_ACC (Vertical Acceleration Limit)

DEFAULT: 7.0	RANGE: 1.0 10.0	UNITS: m/s^2

Vertical Acceleration Limit (TECS_VERT_ACC)

Description

`TECS_VERT_ACC` defines the "G-Limit" for vertical maneuvers.

When the drone needs to climb or descend suddenly, it will limit the force of the pull-up or push-down to this value. This protects the wings from structural damage and prevents the EKF from being overwhelmed by high G-loads.

- **7.0 (Default):** Standard for most planes.
- **10.0:** For high-performance, stiff airframes.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **TEMP** parameter group configures the **Temperature Sensing** subsystem. This library manages sensors for monitoring battery temperature, motor temperature, or ambient air temperature.

Key Concepts

1. Sensor Types (`TEMPn_TYPE`)

ArduPilot supports multiple temperature sensor instances.

- **1 (Analog):** Using a thermistor or active analog sensor on an ADC pin.
- **2 (I2C-MCP9808):** High-precision digital temperature sensor.
- **3 (DroneCAN):** CAN-based temperature reports.

2. Monitoring and Logging

- `TEMP_LOG` : Toggles recording of temperature data to the DataFlash log.
- `TEMP_MSG_RATE` : Defines how often temperature updates are sent over MAVLink.

Parameter Breakdown

- `TEMP1_TYPE` : Driver selection.
- `TEMP1_SRC` : Selects what the sensor is measuring (Battery, Internal, Motor).
- `TEMP_PIN` : (Analog only) ADC pin number.

Integration Guide

1. **Hardware:** Connect your sensor (e.g., an NTC thermistor to an ADC pin with a pull-up resistor).
2. **Enable:** Set `TEMP1_TYPE` and `TEMP1_SRC` .
3. **Calibrate:** For analog sensors, use the `TEMP1_A0` ... `A5` coefficients to define the thermistor's Beta-curve.

Developer Notes

- **Library:** `libraries/AP_TempSensor` .

TEMP1_A0 (Temperature Polynomial Coefficient A0 (Sensor 1))

`DEFAULT: 0`

Temperature Polynomial Coefficient A0 (TEMP1_A0)

Description

`TEMP1_A0` is the first part of a 5th-order polynomial used to translate raw analog voltage from a thermistor or probe into a human-readable temperature in degrees Celsius.

In its simplest form (linear), `A0` acts as the offset. In a complex curve, it works with `A1` through `A5` to fit the specific response curve of your sensor.

The Mathematics

The autopilot calculates the temperature (T) from the measured voltage (V) using this formula:

$$T = A_0 + (A_1 \cdot V) + (A_2 \cdot V^2) + (A_3 \cdot V^3) + (A_4 \cdot V^4) + (A_5 \cdot V^5)$$

Tuning & Behavior

- **Default:** 0.
- **Usage:** For most standard analog sensors, you will find these coefficients in the manufacturer's data sheet or by performing a curve-fitting calibration in a spreadsheet.
- **Note:** This only applies if `TEMP1_TYPE` is set to **5 (Analog)**.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY

TEMP1_A1 (Temperature Polynomial Coefficient A1 (Sensor 1))

DEFAULT: 0

Temperature Polynomial Coefficient A1 (TEMP1_A1)

Description

`TEMP1_A1` is the linear part of the 5th-order polynomial used to calculate temperature from analog voltage.

- **Linear Scale:** It defines how many degrees the temperature changes for every 1 volt increase.
- **Formula:** $T = A_0 + (A_1 \cdot V) + (A_2 \cdot V^2) + \dots$

Tuning & Behavior

- **Default:** 0.
- **Usage:** Consult your sensor's datasheet for the "Slope" or "Scale" value.
- **Note:** Only used if TEMP1_TYPE is set to **5 (Analog)**.

TEMP1_A2 (Temperature Polynomial Coefficient A2 (Sensor 1))

DEFAULT: 0

Temperature Polynomial Coefficient A2 (TEMP1_A2)

Description

`TEMP1_A2` is the second-order (square) part of the 5th-order polynomial used to calculate temperature from analog voltage.

- **Role:** Defines the curvature of the sensor's response. Most low-cost thermistors are non-linear and require this coefficient to provide accurate readings across a wide temperature range.
- **Formula:** $T = A_0 + (A_1 \cdot V) + (A_2 \cdot V^2) + \dots$

Tuning & Behavior

- **Default:** 0.
- **Usage:** Consult your sensor's datasheet or a calibration table.
- **Note:** Only used if TEMP1_TYPE is set to **5 (Analog)**.

TEMP1_A3 (Temperature Polynomial Coefficient A3 (Sensor 1))

DEFAULT: 0

Temperature Polynomial Coefficient A3 (TEMP1_A3)

Description

`TEMP1_A3` is the third-order (cubic) part of the 5th-order polynomial used to calculate temperature from analog voltage.

- **Role:** Refines the accuracy of non-linear sensors.
- **Formula:** $T = A_0 + (A_1 \cdot V) + (A_2 \cdot V^2) + (A_3 \cdot V^3) + \dots$

TEMP1_A4 (Temperature Polynomial Coefficient A4 (Sensor 1))

DEFAULT: 0

Temperature Polynomial Coefficient A4 (TEMP1_A4)

Description

`TEMP1_A4` is the fourth-order part of the 5th-order polynomial used to calculate temperature from analog voltage.

- **Role:** Used for extremely precise curve fitting on high-end specialized analog probes.
- **Formula:** $T = A_0 + (A_1 \cdot V) + (A_linear \cdot V^2) + (A_3 \cdot V^3) + (A_4 \cdot V^4) + \dots$

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





TEMP1_A5 (Temperature Polynomial Coefficient A5 (Sensor 1))

DEFAULT: 0

Temperature Polynomial Coefficient A5 (TEMP1_A5)

Description

`TEMP1_A5` is the fifth-order part of the 5th-order polynomial used to calculate temperature from analog voltage.

- **Role:** The highest order term for complex curve fitting.
- **Formula:** $T = A_0 + (A_1 \cdot V) + (A_linear \cdot V^2) + (A_3 \cdot V^3) + (A_4 \cdot V^4) + (A_5 \cdot V^5)$

TEMP1_ADDR (Temperature Sensor Address (Sensor 1))

DEFAULT: 0	RANGE: 0 127

Temperature Sensor Address (TEMP1_ADDR)

Description

`TEMP1_ADDR` sets the 7-bit I2C address for digital temperature sensors (Types 1-4, 7).

Most sensors have a fixed or jumper-selectable address. If you have multiple sensors of the same type on the same bus, you must configure them with unique addresses and set those addresses here.

Tuning & Behavior

- **Default:** 0 (Use the driver's default address for the selected type).
- **Recommendation:** Only change this if your hardware documentation specifies a non-standard I2C address.

TEMP1_BUS (Temperature Sensor Bus)

DEFAULT: 0	RANGE: 0 3

Temperature Sensor Bus (TEMP1_BUS)

Description

`TEMP1_BUS` specifies which physical I2C port on the flight controller the temperature sensor is plugged into.

Most flight controllers have multiple I2C buses (e.g. internal for compass/baro, and external for GPS/Peripherals).

- **0:** Bus 0.
- **1:** Bus 1 (Often the primary external port).
- **etc.**

Tuning & Behavior

- **Default:** 0.
- **Action:** If your I2C temperature sensor is not being detected, check your flight controller's hardware specification to find the correct bus number for the port you have used.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





TEMP1_MSG_ID (Temperature DroneCAN Message ID)

DEFAULT: 0	RANGE: 0 65535

Temperature DroneCAN Message ID (TEMP1_MSG_ID)

Description

`TEMP1_MSG_ID` identifies which specific piece of hardware on the DroneCAN bus is providing the temperature data for this instance.

This allows you to map multiple DroneCAN temperature sensors (e.g., from different ESCs or battery monitors) to specific slots in the autopilot's telemetry system.

Tuning & Behavior

- **Default:** 0 (Auto-detection).
- **Usage:** Only set this if you have multiple sensors of the same type and need to force a specific one to Instance 1.
- **Requirement:** Only used if TEMP1_TYPE is set to **1 (DroneCAN)**.

TEMP1_SRC (Temperature Sensor Source)

DEFAULT: 0	RANGE: 0 6

Temperature Sensor Source (TEMP1_SRC)

Description

`TEMP1_SRC` tells the autopilot what the temperature sensor is actually touching.

By default, temperature data is just recorded in the logs. However, if you set a source here, the autopilot will "link" this sensor to a specific component. For example, if you set this to **1 (ESC)**, the temperature reported for ESC 1 will come from this physical sensor instead of the ESC's built-in (often inaccurate) telemetry.

- **0: None.** Data is logged but not displayed as a component temperature.
- **1: ESC.**
- **2: Motor.**
- **3: Battery.** (Uses TEMP1_SRC_ID to select which battery).
- **5: Pitot Tube.**

Tuning & Behavior

- **Default:** 0.
- **Integration:** Use this to enable thermal protection for motors or to monitor battery health more closely.

TEMP1_SRC_ID (Sensor Source Instance ID)

DEFAULT: 0	RANGE: 0 255

Sensor Source Instance ID (TEMP1_SRC_ID)

Description

`TEMP1_SRC_ID` tells the autopilot which specific system component this temperature sensor is "Replacement" data for.

For example, if you have a standalone temperature probe mounted to your motor, and you want that probe's data to show up as "ESC 1 Temperature" in your logs and GCS, you would use this parameter.

- **Mapping:** This works with TEMP1_SRC.
  - If `TEMP1_SRC` = 1 (ESC), and `TEMP1_SRC_ID` = 1, this sensor's data replaces the temperature report for **ESC 1**.
  - If `TEMP1_SRC` = 3 (Battery), and `TEMP1_SRC_ID` = 2, it replaces **Battery 2**.

Tuning & Behavior

- **Default:** 0.
- **Significance:** This is a powerful integration tool for customizing your vehicle's telemetry dashboard without needing custom firmware.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





TEMP1_TYPE (Temperature Sensor Type (Sensor 1))

DEFAULT: 0	RANGE: 0 7

Temperature Sensor Type (TEMP1_TYPE)

Description

TEMP1_TYPE enables the primary external temperature sensor.

This is often used to monitor the heat of critical components like motors, ESCs, or batteries that do not have built-in telemetry.

- **0: Disabled.**
- **1: TSYS01.** (I2C).
- **2: MCP9600.** (I2C Thermocouple).
- **5: Analog.** (Voltage-divider based thermistor).
- **6: DroneCAN.**

Tuning & Behavior

- **Reboot Required:** Yes.
- **Integration:** Once enabled, use TEMP1_SRC to tell ArduPilot which system component this sensor is measuring.

TEMP2_ADDR (Temperature sensor address)

DEFAULT: AP_TEMPERATURE_SENSOR_I2C_ADDR_DEFAULT	RANGE: 0 127

Temperature sensor address

Note: This parameter functions identically to TEMP1_ADDR.

TEMP2_BUS (Temperature sensor bus)

DEFAULT: AP_TEMPERATURE_SENSOR_I2C_BUS_DEFAULT	RANGE: 0 3

Temperature sensor bus

Note: This parameter functions identically to TEMP1_BUS.

TEMP2_SRC (Sensor Source)

DEFAULT: (float	RANGE: null

Sensor Source

Note: This parameter functions identically to TEMP1_SRC.

TEMP2_SRC_ID (Sensor Source Identification)

DEFAULT: AP_TEMPERATURE_SENSOR_SOURCE_ID_DEFAULT	RANGE: null

Sensor Source Identification

Note: This parameter functions identically to TEMP1_SRC_ID.

TEMP2_TYPE (Temperature Sensor Type)

DEFAULT: (float	RANGE: null

Temperature Sensor Type

Note: This parameter functions identically to TEMP1_TYPE.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





TEMP3_ADDR (Temperature sensor address)

DEFAULT: AP_TEMPERATURE_SENSOR_I2C_ADDR_DEFAULT	RANGE: 0 127

Temperature sensor address

Note: This parameter functions identically to TEMP1_ADDR.

TEMP3_BUS (Temperature sensor bus)

DEFAULT: AP_TEMPERATURE_SENSOR_I2C_BUS_DEFAULT	RANGE: 0 3

Temperature sensor bus

Note: This parameter functions identically to TEMP1_BUS.

TEMP3_SRC (Sensor Source)

DEFAULT: (float	RANGE: null

Sensor Source

Note: This parameter functions identically to TEMP1_SRC.

TEMP3_SRC_ID (Sensor Source Identification)

DEFAULT: AP_TEMPERATURE_SENSOR_SOURCE_ID_DEFAULT	RANGE: null

Sensor Source Identification

Note: This parameter functions identically to TEMP1_SRC_ID.

TEMP3_TYPE (Temperature Sensor Type)

DEFAULT: (float	RANGE: null

Temperature Sensor Type

Note: This parameter functions identically to TEMP1_TYPE.

TEMP4_ADDR (Temperature sensor address)

DEFAULT: AP_TEMPERATURE_SENSOR_I2C_ADDR_DEFAULT	RANGE: 0 127

Temperature sensor address

Note: This parameter functions identically to TEMP1_ADDR.

TEMP4_BUS (Temperature sensor bus)

DEFAULT: AP_TEMPERATURE_SENSOR_I2C_BUS_DEFAULT	RANGE: 0 3

Temperature sensor bus

Note: This parameter functions identically to TEMP1_BUS.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





TEMP4_SRC (Sensor Source)

DEFAULT: (float	RANGE: null

Sensor Source

Note: This parameter functions identically to TEMP1_SRC.

TEMP4_SRC_ID (Sensor Source Identification)

DEFAULT: AP_TEMPERATURE_SENSOR_SOURCE_ID_DEFAULT	RANGE: null

Sensor Source Identification

Note: This parameter functions identically to TEMP1_SRC_ID.

TEMP4_TYPE (Temperature Sensor Type)

DEFAULT: (float	RANGE: null

Temperature Sensor Type

Note: This parameter functions identically to TEMP1_TYPE.

TEMP5_ADDR (Temperature sensor address)

DEFAULT: AP_TEMPERATURE_SENSOR_I2C_ADDR_DEFAULT	RANGE: 0 127

Temperature sensor address

Note: This parameter functions identically to TEMP1_ADDR.

TEMP5_BUS (Temperature sensor bus)

DEFAULT: AP_TEMPERATURE_SENSOR_I2C_BUS_DEFAULT	RANGE: 0 3

Temperature sensor bus

Note: This parameter functions identically to TEMP1_BUS.

TEMP5_SRC (Sensor Source)

DEFAULT: (float	RANGE: null

Sensor Source

Note: This parameter functions identically to TEMP1_SRC.

TEMP5_SRC_ID (Sensor Source Identification)

DEFAULT: AP_TEMPERATURE_SENSOR_SOURCE_ID_DEFAULT	RANGE: null

Sensor Source Identification

Note: This parameter functions identically to TEMP1_SRC_ID.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





TEMP5_TYPE (Temperature Sensor Type)

DEFAULT: (float	RANGE: null

Temperature Sensor Type

Note: This parameter functions identically to TEMP1_TYPE.

TEMP6_ADDR (Temperature sensor address)

DEFAULT: AP_TEMPERATURE_SENSOR_I2C_ADDR_DEFAULT	RANGE: 0 127

Temperature sensor address

Note: This parameter functions identically to TEMP1_ADDR.

TEMP6_BUS (Temperature sensor bus)

DEFAULT: AP_TEMPERATURE_SENSOR_I2C_BUS_DEFAULT	RANGE: 0 3

Temperature sensor bus

Note: This parameter functions identically to TEMP1_BUS.

TEMP6_SRC (Sensor Source)

DEFAULT: (float	RANGE: null

Sensor Source

Note: This parameter functions identically to TEMP1_SRC.

TEMP6_SRC_ID (Sensor Source Identification)

DEFAULT: AP_TEMPERATURE_SENSOR_SOURCE_ID_DEFAULT	RANGE: null

Sensor Source Identification

Note: This parameter functions identically to TEMP1_SRC_ID.

TEMP6_TYPE (Temperature Sensor Type)

DEFAULT: (float	RANGE: null

Temperature Sensor Type

Note: This parameter functions identically to TEMP1_TYPE.

TEMP7_ADDR (Temperature sensor address)

DEFAULT: AP_TEMPERATURE_SENSOR_I2C_ADDR_DEFAULT	RANGE: 0 127

Temperature sensor address

Note: This parameter functions identically to TEMP1_ADDR.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





TEMP7_BUS (Temperature sensor bus)

DEFAULT: AP_TEMPERATURE_SENSOR_I2C_BUS_DEFAULT	RANGE: 0 3

Temperature sensor bus

Note: This parameter functions identically to TEMP1_BUS.

TEMP7_SRC (Sensor Source)

DEFAULT: (float	RANGE: null

Sensor Source

Note: This parameter functions identically to TEMP1_SRC.

TEMP7_SRC_ID (Sensor Source Identification)

DEFAULT: AP_TEMPERATURE_SENSOR_SOURCE_ID_DEFAULT	RANGE: null

Sensor Source Identification

Note: This parameter functions identically to TEMP1_SRC_ID.

TEMP7_TYPE (Temperature Sensor Type)

DEFAULT: (float	RANGE: null

Temperature Sensor Type

Note: This parameter functions identically to TEMP1_TYPE.

TEMP8_ADDR (Temperature sensor address)

DEFAULT: AP_TEMPERATURE_SENSOR_I2C_ADDR_DEFAULT	RANGE: 0 127

Temperature sensor address

Note: This parameter functions identically to TEMP1_ADDR.

TEMP8_BUS (Temperature sensor bus)

DEFAULT: AP_TEMPERATURE_SENSOR_I2C_BUS_DEFAULT	RANGE: 0 3

Temperature sensor bus

Note: This parameter functions identically to TEMP1_BUS.

TEMP8_SRC (Sensor Source)

DEFAULT: (float	RANGE: null

Sensor Source

Note: This parameter functions identically to TEMP1_SRC.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





TEMP8_SRC_ID (Sensor Source Identification)

DEFAULT: AP_TEMPERATURE_SENSOR_SOURCE_ID_DEFAULT	RANGE: null

Sensor Source Identification

Note: This parameter functions identically to TEMP1_SRC_ID.

TEMP8_TYPE (Temperature Sensor Type)

DEFAULT: (float	RANGE: null

Temperature Sensor Type

Note: This parameter functions identically to TEMP1_TYPE.

TEMP9_ADDR (Temperature sensor address)

DEFAULT: AP_TEMPERATURE_SENSOR_I2C_ADDR_DEFAULT	RANGE: 0 127

Temperature sensor address

Note: This parameter functions identically to TEMP1_ADDR.

TEMP9_BUS (Temperature sensor bus)

DEFAULT: AP_TEMPERATURE_SENSOR_I2C_BUS_DEFAULT	RANGE: 0 3

Temperature sensor bus

Note: This parameter functions identically to TEMP1_BUS.

TEMP9_SRC (Sensor Source)

DEFAULT: (float	RANGE: null

Sensor Source

Note: This parameter functions identically to TEMP1_SRC.

TEMP9_SRC_ID (Sensor Source Identification)

DEFAULT: AP_TEMPERATURE_SENSOR_SOURCE_ID_DEFAULT	RANGE: null

Sensor Source Identification

Note: This parameter functions identically to TEMP1_SRC_ID.

TEMP9_TYPE (Temperature Sensor Type)

DEFAULT: (float	RANGE: null

Temperature Sensor Type

Note: This parameter functions identically to TEMP1_TYPE.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





TEMP_LOG (Temperature Sensor Logging)

DEFAULT: 0	RANGE: 0 2

Temperature Sensor Logging (TEMP_LOG)

Description

`TEMP_LOG` enables or disables the recording of temperature sensor data to the drone's internal dataflash log (`.BIN` files).

- **0: Disabled.** No temperature data is logged.
- **1: Log All.** Data from every enabled temperature sensor is recorded.
- **2: Log Only Source None.** Only logs sensors that are NOT assigned to a specific component (like an ESC or Battery). This avoids duplicating data already being recorded in other messages.

Tuning & Behavior

- **Default Value:** 0.
- **Use Case:** Enable this if you have added a dedicated sensor for monitoring ambient air temperature or a custom payload.

TEMP_PIN (Wind vane speed sensor analog temp pin)

DEFAULT: WINDSPEED_DEFAULT_TEMP_PIN	RANGE: null	UNITS: null

Wind vane speed sensor analog temp pin

Note: This parameter functions identically to TEMP1_PIN.

Temperature sensor message rate ()

TEMP_MSG_RATE: Temperature sensor message rate

Description

This is the rate Temperature sensor data is sent in Hz. Zero means no send.

Values

- **Range:** 0 to 200
- **Units:** Hz
- **Default:** 0

Description

This parameter controls how frequently the AP_Periph device broadcasts temperature data over the DroneCAN bus.

- **0 (Default):** Disabled. No temperature messages are sent.
- **1-200:** The frequency in Hertz.
- **Usage:** Set this to a positive value (e.g., 10) if you have connected an external temperature sensor to your CAN peripheral and want the flight controller to receive the data.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





TERRAIN FOLLOWING CONFIGURATION (TERRAIN)

Overview

The **TERRAIN** parameter group configures the **Terrain Following** system. This library allows the vehicle to maintain a constant height above the ground during autonomous missions by referencing a global terrain database stored on the SD card.

Unlike Lidar-based avoidance, Terrain Following uses pre-recorded topographic maps (SRTM data) and does not require an active sensor to see the ground (though it can be used together).

Key Concepts

1. The Terrain Database

The GCS (Mission Planner) downloads terrain data for your mission area and uploads it to the autopilot's SD card.

- `TERRAIN_SPACING`: The resolution (meters) of the grid points. Smaller values provide more detail but consume more memory.

2. Following Logic (`TERRAIN_FOLLOW`)

Determines if the vehicle uses terrain data for altitude control.

- **Disabled:** All altitudes are relative to the home/takeoff point.
- **Enabled:** Waypoint altitudes are treated as "Above Ground Level" (AGL). The drone will climb and descend as the terrain rises and falls.

3. Lookahead (`TERRAIN_LOOKAHD`)

The autopilot calculates the terrain height ahead of the vehicle to start climbing *before* it hits a hill, rather than reacting only when it arrives.

Parameter Breakdown

- `TERRAIN_ENABLE`: Master switch.
- `TERRAIN_MARGIN`: Vertical safety buffer.
- `TERRAIN_CACHE_SZ`: RAM allocated for holding terrain tiles.

Integration Guide

1. **Preparation:** Ensure your flight controller has an SD card.
2. **Enable:** Set `TERRAIN_ENABLE = 1`.
3. **Download:** Use Mission Planner to "Pre-fetch" terrain for your intended flight area while your laptop has internet.
4. **Planning:** In your mission, set the altitude type to "Terrain."

Developer Notes

- **Library:** `libraries/AP_Terrain`.
- **Safety:** Always set `FENCE_ALT_MAX` as a global backup. Terrain data can have errors (e.g., missing trees or buildings).

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





TERRAIN_CACHE_SZ (Terrain Cache Size)

DEFAULT: 0	RANGE: 0 128	UNITS: blocks

Terrain Cache Size (TERRAIN_CACHE_SZ)

Description

`TERRAIN_CACHE_SZ` allocates memory for the terrain map.

The autopilot cannot hold the entire world's map in memory. Instead, it reads small "blocks" (tiles) from the SD card as needed. This parameter determines how many of these blocks can be kept in RAM at once for fast access.

- **0 (Default):** Use the board-specific default (usually scaled to available RAM).
- **Value:** Number of blocks (each block is approx. 2KB).

Tuning & Behavior

- **Default:** 0.
- **Recommendation:** Leave at **0**.
- **Significance:** If you fly very fast over varied terrain, a larger cache might prevent "Data Starvation" where the autopilot is waiting for the SD card to load the next hill.

TERRAIN_FOLLOW (Enable Terrain Following)

DEFAULT: 0	RANGE: 0 1

Enable Terrain Following (TERRAIN_FOLLOW)

Description

`TERRAIN_FOLLOW` switches the autopilot's altitude reference from "Above Home" to "Above Ground Level (AGL)" for semi-autonomous modes.

When enabled, modes like **CRUISE** and **FBWB** (Fly-By-Wire B) will attempt to maintain a constant height above the terrain rather than a constant barometric altitude. This is essential for low-level flying in hilly environments.

- **0: Disabled.** Altitude is relative to the home takeoff point.
- **1: Enabled.** Altitude is relative to the terrain database or rangefinder reading.

Tuning & Behavior

- **Default:** 0.
- **Safety:** You **MUST** have valid terrain data (on SD card) or a long-range lidar for this to work safely. If the terrain data is missing, the plane may climb unexpectedly (failsafe) or fly into a hill.
- **RTL:** This also affects Return-to-Launch. The plane will climb over hills on its way home.

TERRAIN_LOOKAHD (Terrain Look-Ahead Distance)

DEFAULT: 2000	RANGE: 0 5000	UNITS: m

Terrain Look-Ahead Distance (TERRAIN_LOOKAHD)

Description

`TERRAIN_LOOKAHD` allows the drone to see a mountain coming and start climbing *before* it gets there.

Without look-ahead, the drone would only react to the ground directly beneath it. If it flew toward a steep cliff, it might not have enough time to climb over it. This parameter tells the navigation controller to scan the terrain map ahead of the current position. If it sees a hill coming, it will begin the climb early to ensure a safe and smooth clearance.

Tuning & Behavior

- **Default:** 2000 meters.
- **Recommendation:**
  - **Fast Planes:** Keep at **2000m** or more. You need time to climb.
  - **Slow Copters:** You can reduce this to **500m** or **1000m**, as copters can climb vertically if needed.
- **Note:** The look-ahead never extends past the *next* waypoint.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





TERRAIN_MARGIN (Terrain Data Acceptance Margin)

DEFAULT: 0.05	RANGE: 0 5.0	UNITS: m

Terrain Data Acceptance Margin (TERRAIN_MARGIN)

Description

`TERRAIN_MARGIN` is a compatibility setting for older terrain databases.

ArduPilot expects terrain grid points to be aligned perfectly to specific latitude/longitude intervals. If the GCS sends data that is slightly shifted (due to floating point rounding errors in older software), this margin allows the autopilot to accept it anyway.

Tuning & Behavior

- **Default:** 0.05 m (5 cm).
- **Recommendation:** Leave at default. Modern Mission Planner and MAVProxy versions do not require adjustment.

TERRAIN_OFS_MAX (Terrain Reference Offset Maximum)

DEFAULT: 30	RANGE: 0 50	UNITS: m

Terrain Reference Offset Maximum (TERRAIN_OFS_MAX)

Description

`TERRAIN_OFS_MAX` is a sanity check for your map data.

When you arm the drone on the ground, ArduPilot checks the terrain database for the current location. Often, the map says the ground is at 100m, but the GPS/Baro says 105m. ArduPilot assumes the drone is on the ground and calculates an "Offset" (e.g. -5m) to shift the map to match reality. This parameter limits how large that shift can be.

Tuning & Behavior

- **Default:** 30 meters.
- **Scenario:** If the map says you are at 100m elevation, but the GPS says 150m (a 50m error), and `TERRAIN_OFS_MAX` is 30, the system will reject the calibration because the map is likely wrong or the GPS glitch is too large.
- **0:** Disables automatic terrain offset calibration.

TERRAIN_OPTIONS (Terrain System Options)

DEFAULT: 0	RANGE: 0 1

Terrain System Options (TERRAIN_OPTIONS)

Description

`TERRAIN_OPTIONS` provides fine-grained control over how the terrain database is queried.

- **Bit 0 (1): Disable Interpolation.** By default, the system smooths out the steps between grid points. Setting this bit forces the system to return the raw grid height, which is faster but "blocky."

Tuning & Behavior

- **Default:** 0 (Interpolation enabled).
- **Recommendation:** Leave at **0** for smoother flight paths.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





TERRAIN_SPACING (Terrain Grid Spacing)

DEFAULT: 100	RANGE: 30 1000	UNITS: m

Terrain Grid Spacing (TERRAIN_SPACING)

Description

`TERRAIN_SPACING` defines the "Resolution" of the map used by the autopilot.

When the drone flies into a new area, it asks the Ground Control Station (GCS) for terrain height data. This parameter tells the GCS how dense that data should be.

- **100m (Default):** A good balance. The drone requests the height of the ground every 100 meters. This matches the standard SRTM database resolution used by Mission Planner.
- **30m:** High resolution. Useful for very detailed low-level flying, but uses more SD card space and telemetry bandwidth.

Tuning & Behavior

- **Default:** 100.
- **Recommendation:** Leave at **100**. Setting it lower than your GCS's actual map data (e.g. asking for 10m spacing when the map is only 90m resolution) is wasteful and provides no extra accuracy.

Terrain data enable ()

TERRAIN_ENABLE: Terrain data enable

Description

Enable terrain data.

Values

Value	Meaning
0	Disabled
1	Enabled

- **Default:** 1

Description

This parameter enables the autopilot to download, store, and use digital elevation maps (DEM) for Terrain Following.

- **Function:** When enabled, the drone will request terrain data tiles from the Ground Control Station (Mission Planner or QGroundControl) or download them from the internet if the companion computer has access. These tiles are stored on the SD card.
- **Usage:** Essential for safe low-altitude flight over hills and mountains. Without this data, the drone assumes the world is flat at Home altitude (unless using a rangefinder).

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **THR** parameter group configures the global **Throttle** behaviors for the vehicle. It defines safety limits, deadzones, and automated management logic.

Key Concepts

1. Failsafe (`THR_FAILSAFE`)

One of the most important safety settings. If the signal from the RC receiver drops below a certain threshold (`THR_FS_VALUE`), the autopilot triggers a Radio Failsafe.

2. Limits (`THR_MIN`, `THR_MAX`)

Defines the allowable range of motor power.

- `THR_MIN`: The lowest throttle level allowed in flight (ensures motors don't stop).
- `THR_MAX`: The absolute ceiling for throttle output.

3. Slew Rate (`THR_SLEWRATE`)

Limits how fast the throttle can change (percentage per second).

- **Purpose:** Prevents mechanical stress and battery current spikes from instantaneous throttle jumps.

Developer Notes

- **Library:** `libraries/AP_Motors`, `libraries/AP_Vehicle`.
- **Context:** `THR_MIN` is critical for ArduPlane to ensure the motor stays running during an auto-landing or glide if an ICE engine is used.

Minimum Throttle ()

THR_MIN: Minimum Throttle

Description

Minimum throttle percentage used in all modes except manual, provided THR_PASS_STAB is not set.

Values

- **Range:** -100 to 100
- **Units:** %
- **Default:** 0

Description

This parameter sets the "Floor" for the autopilot's throttle output.

- **Plane:**
  - **0:** Motor stops when throttle is cut.
  - **> 0 (e.g. 10):** Motor idles at 10% whenever armed, preventing the engine from quitting in flight (common for gas engines).
  - **< 0:** Allows Reverse Thrust (if supported by ESC/Prop).
- **Rover:**
  - Determines the minimum speed for autonomous driving.
- **Safety:** Ensure this is not set so high that the vehicle moves when armed on the ground.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





THR_DZ (Throttle Deadzone)

DEFAULT: 30	RANGE: 0 300	UNITS: PWM

Throttle Deadzone (THR_DZ)

Description

`THR_DZ` defines the "Idle Area" in the middle of your throttle stick when flying in modes like **AltHold**, **Loiter**, or **PosHold**.

In these modes, the autopilot tries to maintain a steady altitude. When the stick is near the center, the drone holds its height. If you push the stick above or below this deadzone, the drone starts to climb or descend. `THR_DZ` ensures that tiny movements or "jitter" in your transmitter sticks don't cause the drone to slowly drift up or down.

Tuning & Behavior

- **Default:** 30 PWM (e.g. if center is 1500, the deadzone is 1470 to 1530).
- **Recommendation:** If your drone starts climbing or falling the moment you let go of the stick, increase this value to **40** or **50**.
- **Physics:** This parameter only affects stick input interpretation; it does not change the motor's physical response.

THR_FAILSAFE (Throttle Failsafe Enable (Plane))

DEFAULT: 1	RANGE: 0 2

Throttle Failsafe Enable (THR_FAILSAFE)

Description

`THR_FAILSAFE` is the master switch for RC signal loss protection on ArduPlane.

- **0: Disabled.** The plane will not react to a loss of radio signal. (Highly Dangerous).
- **1: Enabled (Standard).** Triggers a failsafe action (like RTL) if the throttle signal drops below `THR_FS_VALUE` or if the receiver reports a lost link.
- **2: Enabled (No Action).** Ignores RC inputs if a failsafe is detected but does not trigger an autonomous return-to-launch.

Tuning & Behavior

- **Default:** 1 (Enabled).
- **Requirement:** Requires THR_FS_VALUE to be calibrated correctly.
- **Safety:** Always test your failsafe behavior on the ground (with propellers removed) by turning off your transmitter and verifying the flight mode changes to RTL.

THR_FS_VALUE (Throttle Failsafe PWM Threshold (Plane))

DEFAULT: 950	RANGE: 925 1100	UNITS: PWM

Throttle Failsafe PWM Threshold (THR_FS_VALUE)

Description

`THR_FS_VALUE` defines the "Signal Lost" voltage for your radio receiver.

Most receivers are configured to pull the throttle channel to a very low value (e.g. 900) if the link is broken. This parameter tells the autopilot: "If the throttle drops below 950, assume the radio is off and start the failsafe routine."

Tuning & Behavior

- **Default:** 950.
- **Setup:** Turn off your transmitter and see what the throttle PWM value is in the GCS. Set this parameter to about **10-20 units above** that value, but well below your normal idle throttle (usually 1100).

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





THR_PASS_STAB (Throttle Pass-Through in Stabilize)

DEFAULT: 0	RANGE: 0 1

Throttle Pass-Through in Stabilize (THR_PASS_STAB)

Description

`THR_PASS_STAB` is a specialized setting for ArduPlane, primarily used for combustion (gas/nitro) engines.

Normally, ArduPilot clamps the throttle between THR_MIN and THR_MAX. For a gas engine, your "Min" throttle is usually set to a fast idle. If you wanted to kill the engine using a switch on your transmitter, the autopilot's clamping would normally prevent you from reaching the "cutoff" PWM. Enabling this parameter allows the autopilot to ignore its own limits and pass the raw stick signal directly to the motor in non-GPS modes.

- **0: Disabled (Default).** Autopilot limits apply in all modes.
- **1: Enabled.** direct control in Stabilize, FBWA, and Acro.

Tuning & Behavior

- **Default:** 0.
- **Usage:** Enable this if you have a "Throttle Cut" safety switch configured on your transmitter that needs to pull the PWM below the idle threshold.

THR_SLEWRATE (Throttle Slew Rate)

DEFAULT: 100	RANGE: 0 500	UNITS: %/s

Throttle Slew Rate (THR_SLEWRATE)

Description

`THR_SLEWRATE` prevents the autopilot from "slamming" the throttle.

If you are using a large gas engine or a high-torque helicopter motor, changing from 0% to 100% throttle instantly can cause mechanical failure or strip gears. This parameter forces the throttle to ramp up or down smoothly over a specific time.

- **100:** Allows the throttle to move from 0% to 100% in exactly one second.
- **0:** Unlimited (Instant response).

Tuning & Behavior

- **Default:** 100 (Heli) or 0 (Multicopter).
- **Recommendation:** For large electric motors or combustion engines, set to **50 or 100** to ensure a smooth power transition.
- **Note:** This is separate from the `MOT_SPOOL_TIME` which only applies at the moment of arming. `THR_SLEWRATE` applies throughout the entire flight.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





THR_SUPP_MAN (Throttle suppress manual passthru)

DEFAULT: 0	RANGE: 0 1

Throttle suppress manual passthru (THR_SUPP_MAN)

Description

This parameter changes the behavior of the "Throttle Suppression" feature in ArduPlane. Normally, when the autopilot suppresses the throttle (e.g., because the aircraft is on the ground or in a landing flare), the motor output is forced to 0 (Idle/Off). Enabling this parameter allows the pilot's manual throttle stick to override this suppression, keeping the engine running at the stick-commanded speed.

The Mathematics

$$ \text{Throttle}*{|\text{out}} = (|\text{Suppressed}) ? |\text{Throttle}*{\text{stick}} : \text{Throttle}_{\text{auto}} $$

The Engineer's View

Defined in `ArduPlane/Parameters.cpp`.

- **Target Audience:** Gas/Nitro planes where 0 throttle might kill the engine, or for pilots who want manual idle control during auto-takeoff waits.

Tuning & Behavior

- **Default Value:** 0 (Disabled)
- **1:** Enable manual pass-through during suppression.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

Key Concepts

THROTTLE_NUDGE (Throttle nudge enable)

DEFAULT: 1	RANGE: 0 1

Throttle nudge enable (THROTTLE_NUDGE)

Description

This feature allows the pilot to "nudge" (override) the target airspeed or throttle setting during autonomous flight (Auto, Cruise, Loiter) simply by moving the throttle stick. It gives the pilot an intuitive way to speed up or slow down the aircraft without switching modes or changing parameters.

The Mathematics

If `Throttle Stick` > 50%:

$$\text{Target Airspeed} = \text{AIRSPEED_CRUISE} + (\text{AIRSPEED_MAX} - \text{AIRSPEED_CRUISE}) \times \frac{\text{Stick\%} - 50}{50}$$

If no airspeed sensor:

$$\text{Target Throttle} = \text{Base Throttle} + \text{Nudge Factor}$$

The Engineer's View

Defined in `ArduPlane/Parameters.cpp`. Logic resides in `ArduPlane/Events.cpp` or similar throttle loop.

- **Enabled (1):** Pilot has authority to increase energy/speed.
- **Disabled (0):** Stick position is ignored in auto-throttle modes.

Tuning & Behavior

- **Default Value:** 1 (Enabled)
- **Safety:** Useful for flying into strong headwinds where the cruise airspeed might be insufficient.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY

Overview

The **THROW** parameter group configures the **Throw Mode** for multicopters. This specialized mode allows the pilot to literally throw the drone into the air; the autopilot detects the freefall and autonomously stabilizes the vehicle and powers up the motors.

Key Concepts

1. Detection Type (`THROW_TYPE`)

Determines how the "throw" is identified.

- **Upward Throw:** Drone is tossed up.
- **Drop:** Drone is released from height.

2. Startup Sequence

- `THROW_MOT_START`: Defines when the motors should start spinning (e.g., at the peak of the arc or after a delay).

Parameter Breakdown

- `THROW_NEXTMODE`: The flight mode the vehicle automatically switches to after a successful throw stabilization (usually `LOITER`).
- `THROW_ALT_MIN`: Safety altitude floor for activation.

Integration Guide

1. **Arm:** Arm the vehicle while holding it.
2. **Switch:** Change mode to `THROW`.
3. **Toss:** Throw the drone clear of your body.
4. **Auto-Start:** The autopilot will detect the freefall, stabilize level, and then switch to the mode defined in `THROW_NEXTMODE`.

Developer Notes

- **Library:** `ArduCopter/mode_throw.cpp`.
- **Safety: EXTREMELY DANGEROUS.** Only use with small drones and ensure you have a clear area.

THROW_ALT_MAX (Throw Mode Max Altitude)

DEFAULT: 0	RANGE: 0 500.0	UNITS: m

Throw Mode Max Altitude (THROW_ALT_MAX)

Description

`THROW_ALT_MAX` provides an upper safety boundary for Throw Mode.

This prevents the drone from attempting a launch if it is already at an extreme altitude (e.g., if you are trying to release it from a high-altitude platform and it has already drifted past your safe zone).

- **0 (Default):** Check is disabled.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





THROW_ALT_MIN (Throw Mode Min Altitude)

DEFAULT: 0	RANGE: 0 50.0	UNITS: m

Throw Mode Min Altitude (THROW_ALT_MIN)

Description

`THROW_ALT_MIN` is a safety filter for **Throw Mode**.

It prevents the motors from starting if the drone thinks it is too close to the ground. This protects the pilot's hands and the drone if it is accidentally bumped while sitting on the floor.

- **0 (Default):** Check is disabled.
- **Value (m):** The drone must be at least this high (e.g. 1.5 meters) before the "Throw" detection logic becomes active.

THROW_MOT_START (Throw Mode Motor Start Behavior)

DEFAULT: 0	RANGE: 0 1

Throw Mode Motor Start Behavior (THROW_MOT_START)

Description

`THROW_MOT_START` determines if the propellers are spinning while the drone is in your hand.

- **0: Stopped.** The motors remain completely off after arming. They only start spinning once the autopilot detects that you have thrown the drone. This is the **Safest** for your hands.
- **1: Spinning.** The motors start spinning at MOT_SPIN_MIN as soon as you arm. This ensures the motors are "Ready" and synced, but increases the risk of finger injury during the toss.

Tuning & Behavior

- **Default:** 0.
- **Recommendation:** Keep at **0** for safety. Only use **1** if your motors or ESCs have trouble starting reliably in free-fall.

THROW_NEXTMODE (Throw Mode Target Mode)

DEFAULT: 18	RANGE: 0 25

Throw Mode Target Mode (THROW_NEXTMODE)

Description

`THROW_NEXTMODE` defines what the drone does after it successfully "catches itself" in the air.

- **18 (Default):** Stay in Throw Mode. The drone will hover in place and wait for further commands.
- **5: Loiter.** Transition immediately to GPS-based position hold.
- **10: Auto.** Start the pre-programmed mission immediately after the toss.

Tuning & Behavior

- **Default:** 18.
- **Recommendation:** Set to **5 (Loiter)** if you want the drone to hold its position perfectly after you throw it.
- **Significance:** Allows for fully autonomous "toss and go" mission starts.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





THROW_TYPE (Throw Mode Launch Type)

DEFAULT: 0	RANGE: 0 1

Throw Mode Launch Type (THROW_TYPE)

Description

`THROW_TYPE` defines the initial motion the drone expects when using **Throw Mode**.

Throw mode allows a pilot to start the drone without using a transmitter or ground station – the drone senses it is "falling" or "flying" and starts its motors automatically to stabilize.

- **0: Upward Throw.** The drone expects to be tossed up into the air. It waits for the peak of the arc (near zero vertical velocity) before starting the motors.
- **1: Drop.** The drone expects to be released from a high point (e.g. from a balloon or a hand held above the head). It starts the motors as soon as it detects free-fall.

Tuning & Behavior

- **Default:** 0.
- **Safety: NEVER** use Throw mode indoors. Ensure you toss the drone at least 3-5 meters away from you.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

Key Concepts

TIMEOUT_MS (DDS ping timeout)

DEFAULT: 1000	RANGE: 1 10000	UNITS: ms

TIMEOUT_MS: DDS ping timeout

Description

The time in milliseconds the DDS client will wait for a response from the XRCE agent before reattempting.

Values

- **Range:** 1 to 10000
- **Units:** ms
- **Default:** 1000

Description

This parameter manages the connection timeout for the **DDS (Data Distribution Service)** interface, typically used for ROS 2 integration.

- **Function:** The autopilot (DDS Client) periodically pings the companion computer (XRCE Agent). If no response is received within this time limit, the client assumes the connection is lost and attempts to reconnect.
- **Usage:** Adjust this based on the latency and reliability of your serial or Ethernet link to the companion computer.
- **Tuning:** If you have a slow or congested network, increase this to prevent spurious disconnects.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **TKOFF** parameter group configures the **Autonomous Takeoff** sequence for ArduPlane and QuadPlane.

Automating the takeoff ensures a consistent climb-out and prevents pilot errors during the most vulnerable stage of flight.

Key Concepts

1. Launch Acceleration (`TKOFF_THR_MINACC`)

(Plane) For hand-launched or bungee-launched planes, the motor shouldn't start until the plane is physically thrown.

- **Threshold:** The motor starts only after the IMU detects an acceleration spike (m/s^2) exceeding this value.

2. Takeoff Pitch (`TKOFF_LVL_PITCH`)

The target pitch angle during the initial climb-out.

- **Recommendation:** Usually 10-15 degrees for most airframes.

3. Transition Altitude (`TKOFF_ALT`)

The height at which the takeoff sequence is considered "Complete," and the autopilot switches to the first mission waypoint or its standard navigation logic.

Parameter Breakdown

- `TKOFF_THR_MAX`: Maximum throttle allowed during takeoff.
- `TKOFF_DIST`: (Rover/Plane) Distance from home to consider the takeoff complete.
- `TKOFF_FLAP_PCNT`: Percentage of flaps to deploy during takeoff.

Integration Guide

- **Hand Launch:** Set `TKOFF_THR_MINACC = 15` and `TKOFF_THR_DELAY = 2` (0.2s) to ensure the motor doesn't bite your hand as you throw.

Developer Notes

- **Library:** `ArduPlane/mode_takeoff.cpp`.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





TKOFF_ACCEL_CNT (Takeoff Throttle Acceleration Count)

DEFAULT: 1	RANGE: 1 10

Takeoff Throttle Acceleration Count (TKOFF_ACCEL_CNT)

Description

`TKOFF_ACCEL_CNT` is the "Confirmation Code" for your launch trigger. It enables a feature often called **"Shake-to-Arm."**

Instead of a single throw triggering the motor, you can require multiple forward and backward jolts. This is extremely useful for preventing accidental motor starts caused by bumping the drone or walking with it in windy conditions.

- **1 (Default):** A single forward throw triggers the motor.
- **2+:** Requires multiple deliberate forward/backward shakes to start the motor.

The Engineer's View

Defined in `ArduPlane/takeoff.cpp`.
The code maintains a counter of "acceleration events." An event is recorded every time the X-axis acceleration exceeds `TKOFF_THR_MINACC`. To prevent false counts from vibration, events must alternate direction (Forward/Backward) or happen within a specific time window.

```
// ArduPlane/takeoff.cpp
if (g2.takeoff_throttle_accel_count ≤ 1) {
    if (xaccel < g.takeoff_throttle_min_accel) {
        goto no_launch;
    }
} else {
    // Advanced shake detection logic
    bool odd_event = ((takeoff_state.accel_event_counter & 1) ≠ 0);
    // ...
}
```

Tuning & Behavior

- **Default Value:** 1.
- **Recommendation:** Set to **2 or 3** if you are launching in high-vibration environments or from a moving vehicle.
- **Usage:** If set to **2**, you must give the plane a firm forward "pump" or shake before the final throw to arm the throttle.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





TKOFF_ALT (Takeoff Target Altitude)

DEFAULT: 50	RANGE: 10 1000	UNITS: m

Takeoff Target Altitude (TKOFF_ALT)

Description

`TKOFF_ALT` defines the "Destination Height" for an automatic plane takeoff.

When you trigger a takeoff mission (e.g., via a catapult or hand-launch in AUTO mode), the plane will climb at a steady angle until it reaches this specific height. Once this altitude is reached, the takeoff is considered complete, and the autopilot will move to the next waypoint in your mission.

- **Default Value:** 50 meters.
- **Units:** Meters above the takeoff point.

The Engineer's View

Defined in `ArduPlane/mode_takeoff.cpp`.
This parameter is used by the `ModeTakeoff` state machine. While the altitude is below `TKOFF_ALT`, the system enforces specific pitch and roll limits (`TKOFF_LVL_ALT`, `TKOFF_LVL_PITCH`) to ensure a safe, stable climb-out.

```
// ArduPlane/mode_takeoff.cpp
AP_GROUPINFO("ALT", 1, ModeTakeoff, target_alt, 50),
```

The climb logic is typically handled by the **TECS (Total Energy Control System)**, which manages throttle and pitch together to maintain the requested climb rate and airspeed.

Tuning & Behavior

- **Default Value:** 50 m.
- **Effect of Increasing:** The drone spends more time in the "Takeoff" protected state. Safer for clearing tall trees or buildings around your launch site.
- **Effect of Decreasing:** The drone transitions to mission navigation sooner.

Use Case Recommendations

- **Small Field (Tight Space): Increase to 100m.** Ensure you have plenty of height before the drone tries to bank and turn toward the first waypoint.
- **Large Open Area: Keep at 50m.** Standard value provides a safe buffer for most airframes.
- **Long-Range Mapping: Set to 50m.** Allows the drone to get on track as quickly as possible to maximize efficiency.

TKOFF_DIST (Takeoff Completion Distance)

DEFAULT: 200	RANGE: 0 1000	UNITS: m

Takeoff Completion Distance (TKOFF_DIST)

Description

`TKOFF_DIST` is a secondary exit condition for a takeoff. It ensures the drone transitions to its normal mission path even if it hasn't reached its target altitude (`TKOFF_ALT`) yet, but has flown a safe distance away from the launch site.

This is useful for prevent the drone from climbing endlessly in a straight line if it's having trouble reaching a high `TKOFF_ALT`.

- **Default Value:** 200 meters.

The Engineer's View

Used by the `ModeTakeoff` state machine.
The takeoff is marked as "Complete" if **either** of these conditions are met:

1. Current Altitude $\geq$ `TKOFF_ALT`.
2. Horizontal Distance from Start $\geq$ `TKOFF_DIST`.

Tuning & Behavior

- **Default Value:** 200 m.
- **Recommendation:** Set this to roughly the size of your takeoff field. If you have a 500m long runway, set this to **300 or 400**.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





TKOFF_FLAP_PCNT (Takeoff Flap Percentage)

DEFAULT: 0	RANGE: 0 100	UNITS: %

Takeoff Flap Percentage (TKOFF_FLAP_PCNT)

Description

`TKOFF_FLAP_PCNT` sets the specialized wing configuration for the "Roll-out" and "Climb-out" phase of an autonomous takeoff.

Deploying flaps increases the wing's lift coefficient, allowing the plane to rotate and become airborne at a lower ground speed. This is essential for short-field takeoffs or for heavy aircraft.

Tuning & Behavior

- **Default:** 0%.
- **Recommendation:** Set to the "Takeoff Flap" position recommended by your airframe's manufacturer (typically **15% to 40%**).
- **Automation:** The autopilot will automatically retract the flaps to their normal cruise position once the aircraft reaches a safe altitude and airspeed after takeoff.

TKOFF_GND_PITCH (Takeoff Ground Pitch)

DEFAULT: 5	RANGE: 0 15	UNITS: deg

Takeoff Ground Pitch (TKOFF_GND_PITCH)

Description

`TKOFF_GND_PITCH` is used for **Runway Takeoffs**. It defines the attitude of the aircraft while it is accelerating along the ground, before it has reached enough speed to lift its nose and fly.

For tricycle gear aircraft (nose wheel), this is usually set to **0 or 5 degrees** to keep the plane level on its wheels. For taildraggers, this setting helps the autopilot manage the transition from "tail on ground" to "tail in air" (level roll).

- **Default Value:** 5 degrees.

The Engineer's View

Defined in `ArduPlane/mode_takeoff.cpp`.
This parameter is active whenever the airspeed is below `TKOFF_ROTATE_SPD`. It ensures the autopilot doesn't try to pull up too early, which would cause the tail to strike the ground or the plane to "jump" off the runway prematurely.

```
// ArduPlane/takeoff.cpp
if (auto_state.highest_airspeed < g.takeoff_rotate_speed) {
    nav_pitch_cd = int32_t(100.0f * mode_takeoff.ground_pitch);
    // ...
}
```

Tuning & Behavior

- **Default Value:** 5°.
- **Recommendation:** Match the "Natural" sitting angle of your drone on its wheels.
- **Tricycle Gear:** Use **0 or 2 degrees**.
- **Taildragger:** Use **5 or 8 degrees** to help lift the tail wheel off the ground during the takeoff run.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY

TKOFF_LVL_ALT (Takeoff Level-Flight Altitude)

DEFAULT: 10	RANGE: 0 50	UNITS: m

Takeoff Level-Flight Altitude (TKOFF_LVL_ALT)

Description

`TKOFF_LVL_ALT` is a safety buffer that prevents your airplane from banking too steeply when it is close to the ground.

When a plane first takes off, it is often flying slowly and lacks full stability. If the drone tried to make a sharp turn while only 2 meters high, it could strike a wing on the ground. `TKOFF_LVL_ALT` keeps the wings level (or nearly level) until the drone has reached a safe "Staging Altitude."

- **Default Value:** 10 meters.
- **Units:** Meters above the takeoff point.

The Mathematics

ArduPilot uses a dual-threshold scaling logic for the maximum allowed bank angle (ϕ_max) during takeoff:

1. **Below** `TKOFF_LVL_ALT`**:** Roll is restricted to `LEVEL_ROLL_LIMIT` (typically 5-10 degrees).
2. **Above** `TKOFF_LVL_ALT` **x 3:** Roll is allowed up to the full `ROLL_LIMIT_DEG` (typically 45 degrees).
3. **In Between:** The limit is linearly interpolated.

$$\text{Active_Roll_Limit} = \text{lerp}(\text{LEVEL_ROLL}, \text{ROLL_LIMIT}, \text{Alt}, \text{LVL_ALT}, \text{LVL_ALT} \times 3)$$

The Engineer's View

Defined in `ArduPlane/takeoff.cpp` inside `takeoff_calc_roll()`.

```
// ArduPlane/takeoff.cpp
const float lim1 = MAX(mode_takeoff.level_alt, 0);
const float lim2 = MIN(mode_takeoff.level_alt*3, mode_takeoff.target_alt);
const float current_baro_alt = barometer.get_altitude();

takeoff_roll_limit_cd = linear_interpolate(g.level_roll_limit*100, roll_limit_cd,
                                current_baro_alt,
                                auto_state.baro_takeoff_alt+lim1, auto_state.baro_takeoff_alt+lim2);
```

This logic ensures a smooth transition of control authority. As the plane gains altitude and clears ground obstacles, it is granted more freedom to maneuver toward its mission path.

Tuning & Behavior

- **Default Value:** 10 m.
- **Effect of Increasing:** Safer, but the drone will take longer to correct its heading if it was thrown off-track.
- **Effect of Decreasing:** Allows earlier turning. Only recommended for small, high-power drones in tight spaces.

Use Case Recommendations

- **Large Scale / Heavy Planes: Increase to 20m.** Heavy planes have more momentum and take longer to stabilize; keeping the wings level longer prevents tip-stalls.
- **Windy Sites: Increase to 15m.** Prevents a gust from flipping the plane into the ground before it has enough altitude to recover.
- **Standard build: Keep at 10m.**

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY



TKOFF_LVL_PITCH (Takeoff Level-Flight Pitch)

DEFAULT: 15	RANGE: 0 45	UNITS: deg

Takeoff Level-Flight Pitch (TKOFF_LVL_PITCH)

Description

`TKOFF_LVL_PITCH` is the "Launch Angle" for your airplane. It defines how steeply the plane should point its nose up immediately after being thrown or launched.

This initial pitch is maintained until the aircraft reaches the `TKOFF_LVL_ALT` (Stage 1 altitude). It ensures the drone gains altitude rapidly without stalling or wavering.

- **Default Value:** 15 degrees.
- **15 - 20 Degrees:** Standard for high-power park flyers and gliders.
- **5 - 10 Degrees:** For heavy, low-power aircraft that need to prioritize airspeed over climb rate.

The Engineer's View

Defined in `ArduPlane/mode_takeoff.cpp`.
During the initial "Phase 1" of a takeoff mission, the autopilot ignores the standard TECS (Total Energy Control System) pitch demands and forces the nose to this specific angle. This provides a predictable and repeatable launch behavior.

```
// ArduPlane/mode_takeoff.cpp
AP_GROUPINFO("LVL_PITCH", 3, ModeTakeoff, level_pitch, 15),
```

Tuning & Behavior

- **Default Value:** 15°.
- **Effect of Increasing:** Steeper climb. Better for clearing nearby obstacles, but carries a higher risk of a stall if the motor power isn't sufficient.
- **Effect of Decreasing:** Shallower climb. The plane gains airspeed faster but altitude slower.

Troubleshooting

- **Scenario:** The plane climbs steeply after I throw it, but then the wings wobble and it falls (Tip Stall).
  - **Diagnosis:** `TKOFF_LVL_PITCH` is too high for your plane's weight or power.
  - **Fix:** Reduce to 10 or 12 degrees.
- **Scenario:** I threw the plane perfectly, but it slowly drifted down into the tall grass.
  - **Diagnosis:** `TKOFF_LVL_PITCH` is too low, or `TKOFF_THR_MAX` is too low.
  - **Fix:** Increase to 18 degrees and ensure full throttle is allowed.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





TKOFF_OPTIONS (Takeoff Options)

DEFAULT: 0	RANGE: 0 1

Takeoff Options (TKOFF_OPTIONS)

Description

`TKOFF_OPTIONS` is a bitmask that selects how "Smart" the throttle is during takeoff.

Normally, ArduPilot just pins the throttle at `THR_MAX` during a takeoff to ensure the plane doesn't fall. However, high-power modern planes sometimes don't need full power for the entire climb-out. This parameter allows the plane to use its speed sensor to modulate the throttle.

Key Bits:

- **Bit 0 (1): Use TECS Variable Throttle.**
  If enabled, the plane will not just "Blast" the motors. Instead, it will use the Total Energy Control System (TECS) to balance speed and height. It will reduce throttle if the plane is already exceeding its target climb rate. **Only use this if you have a well-calibrated airspeed sensor.**

The Engineer's View

If Bit 0 is **Unset (0)**: The autopilot forces `THR_MAX` (or `TKOFF_THR_MAX`) for the duration of the takeoff. This is the safest default.
If Bit 0 is **Set (1)**: The `TECS` controller is allowed to calculate a throttle command between `THR_MIN` and `THR_MAX` based on the energy balance.

Tuning & Behavior

- **Default Value:** 0 (Full throttle takeoff).
- **Recommendation:** Keep at **0** for most users.
- **Advanced:** Use **1** for high-efficiency gliders that would overspeed if full power was applied during the entire climb-out.

TKOFF_PLIM_SEC (Takeoff Pitch Limit Lead Time)

DEFAULT: 2	RANGE: 0 10	UNITS: s

Takeoff Pitch Limit Lead Time (TKOFF_PLIM_SEC)

Description

`TKOFF_PLIM_SEC` prevents your plane from overshooting its target altitude at the end of a takeoff.

Normally, during takeoff, the autopilot enforces a "Minimum Pitch" to ensure the plane keeps climbing. If this limit is held all the way to the target altitude, the plane's momentum will cause it to fly past the target height before it can level off. This parameter tells the autopilot to start relaxing that pitch limit a few seconds early, allowing for a smooth, graceful transition to level flight exactly at the target altitude.

Tuning & Behavior

- **Default:** 2 seconds.
- **Recommendation:** Leave at **2**. If you find your plane consistently "pops" up 5-10 meters above the takeoff altitude before settling, increase this to **4 or 5**.
- **0:** Disables the lead time. The pitch limit is held until the altitude is reached.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





TKOFF_ROTATE_SPD (Takeoff Rotate Speed)

DEFAULT: 0	RANGE: 0 30	UNITS: m/s

Takeoff Rotate Speed (TKOFF_ROTATE_SPD)

Description

`TKOFF_ROTATE_SPD` is the "Fly Speed" for runway takeoffs. It defines the exact moment the autopilot pulls back on the elevator to lift the nose and start climbing.

If you set this too low, the plane might try to fly before it has enough speed, leading to a stall and a crash. If you set it too high, you will waste runway length and potentially damage your landing gear by rolling too fast.

- **0 (Default):** The plane will rotate as soon as it feels enough airspeed to move the control surfaces (not recommended for runway takeoffs).
- **Recommendation:** Set to **1.1x to 1.3x** your stall speed.

The Engineer's View

Defined in `ArduPlane/takeoff.cpp`.
The parameter acts as a state trigger. While `Airspeed < TKOFF_ROTATE_SPD`, the plane uses `TKOFF_GND_PITCH`. Once the threshold is crossed, the autopilot transitions to the mission climb pitch or `TKOFF_LVL_PITCH`.

```
// ArduPlane/takeoff.cpp
if (!auto_state.rotation_complete && g.takeoff_rotate_speed > 0) {
    if (auto_state.highest_airspeed < g.takeoff_rotate_speed) {
        // Still rolling
    } else {
        // Rotate!
    }
}
```

Tuning & Behavior

- **Default Value:** 0.
- **Effect of Increasing:** Safer takeoff with more lift energy. Requires a longer runway.
- **Effect of Decreasing:** Shorter takeoff run. Higher risk of stalling during the initial climb.

Use Case Recommendations

- **Standard Build:** If your plane stalls at 12 m/s, set `TKOFF_ROTATE_SPD` to **14 or 15 m/s**.
- **Hand/Catapult Launch:** Always set to **0**. You want the plane to enter climb mode immediately after it leaves your hand.

TKOFF_RPM_MAX (Takeoff RPM Maximum)

DEFAULT: 0	RANGE: 0 20000	UNITS: RPM

Takeoff RPM Maximum (TKOFF_RPM_MAX)

Description

`TKOFF_RPM_MAX` detects "Free-Spinning" motors.

If a propeller nut comes loose, the motor shaft will spin up to a very high RPM because it has no load. This parameter detects that condition and prevents the drone from attempting to fly with a loose prop.

The Engineer's View

Used in `ArduCopter/takeoff_check.cpp`.
If `RPM > TKOFF_RPM_MAX`, the takeoff is aborted.

Tuning & Behavior

- **Default Value:** 0 (Disabled).
- **Recommendation:** Set to a value slightly above your hover RPM, but below your max RPM.
  - **5" Quad:** Hover ~5000-8000 RPM. Max ~30k. Set check to **10000**.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY



TKOFF_RPM_MIN (Takeoff RPM Minimum)

DEFAULT: 0	RANGE: 0 5000	UNITS: RPM

Takeoff RPM Minimum (TKOFF_RPM_MIN)

Description

`TKOFF_RPM_MIN` is a safety gate for drones with ESC Telemetry (BLHeli_32/AM32).

Before the drone leaves the ground in Auto mode, it checks that all motors are actually spinning. If any motor reports an RPM lower than this value, the takeoff is aborted. This prevents "Flip on Takeoff" crashes caused by a dead motor or a desync.

The Engineer's View

Used in `ArduCopter/takeoff_check.cpp`.
The autopilot queries `AP_ESC_Telem` for the RPM of all active motors.
If `RPM < TKOFF_RPM_MIN`, the takeoff is held.

Tuning & Behavior

- **Default Value:** 0 (Disabled).
- **Recommendation:** Set to a value slightly below your idle RPM.
  - **5" Quad:** Idle is ~1500 RPM. Set to **1000**.
  - **Large Quad:** Idle is ~800 RPM. Set to **500**.

TKOFF_SLEW_TIME (Takeoff Slew Time)

DEFAULT: 2.0	RANGE: 0 10	UNITS: s

Takeoff Slew Time (TKOFF_SLEW_TIME)

Description

`TKOFF_SLEW_TIME` makes automatic takeoffs smoother.

When you switch to **Auto** or **Guided** and command a takeoff, the drone doesn't just jump to full climb speed instantly. Instead, it smoothly ramps up the throttle target over this duration. This prevents the drone from "launching like a rocket" and stressing the battery or motors.

- **Low Value (0.5s):** Aggressive takeoff.
- **High Value (5.0s):** Very gentle lift-off.

Tuning & Behavior

- **Default Value:** 2.0 seconds.
- **Cinematography:** Increase to **4.0s** for a buttery smooth start to the shot.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





TKOFF_TDRAG_ELEV (Takeoff Tail-Dragger Elevator)

DEFAULT: 0	RANGE: -100 100	UNITS: %

Takeoff Tail-Dragger Elevator (TKOFF_TDRAG_ELEV)

Description

`TKOFF_TDRAG_ELEV` is an essential setting for planes with a **Tail Wheel**.

When a tail-dragger begins its takeoff run, the tail wheel provides the steering. However, as the motor pulls the plane forward, the tail naturally wants to lift into the air. If it lifts too soon, you lose steering control before the rudder becomes effective, causing the plane to "ground loop" (spin out).

This parameter forces the elevator to pull the tail down, keeping the steering wheel firmly on the ground until the plane has enough speed to steer with the rudder.

- **100 (Full Up):** Recommended for tail-draggers. Holds the tail wheel down.
- **0 (Neutral):** Standard for hand-launches.
- **-20 to -30 (Down):** Used for some tricycle-gear planes to hold the nose wheel down for better grip.

The Engineer's View

This parameter provides a direct override to the elevator channel during the "Ground Run" phase of takeoff. It is active from the start of the roll until the speed reaches `TKOFF_TDRAG_SPD1`.

```
// ArduPlane/takeoff.cpp
// Apply g.takeoff_tdrag_elevator to elevator output during initial roll.
```

Tuning & Behavior

- **Default Value:** 0.
- **Recommendation for Tail-draggers:** Set to **100**. As the plane reaches `TKOFF_TDRAG_SPD1`, the elevator will return to neutral, allowing the tail to rise into a level flight attitude.
- **Recommendation for Tricycle (Nose wheel):** Set to **-20** if your nose wheel tends to skitter or lose traction during high-power takeoffs.

Troubleshooting

- **Scenario:** My tail-dragger always veers sharply to the left as soon as I start the takeoff roll.
  - **Diagnosis:** The tail is lifting too early, and you are losing steering.
  - **Fix:** Ensure `TKOFF_TDRAG_ELEV` is 100 and `TKOFF_TDRAG_SPD1` is set to about 70% of your stall speed.

TKOFF_TDRAG_SPD1 (Tail-Dragger Tail-Lift Airspeed)

DEFAULT: 0	RANGE: 0 30.0	UNITS: m/s

Tail-Dragger Tail-Lift Airspeed (TKOFF_TDRAG_SPD1)

Description

`TKOFF_TDRAG_SPD1` automates the specific takeoff physics of a **Tail-Dragger** airplane.

Tail-draggers require the pilot to hold "Full Up" elevator early in the takeoff roll to keep the tailwheel pinned for steering. Once enough speed is gained for the rudder to become effective, the tail must be allowed to lift so the plane can accelerate in a level attitude. This parameter defines that transition speed.

- **0:** Disabled. Use for tricycle gear or hand-launches.
- **Value (m/s):** The airspeed at which the autopilot relaxes the "Up" elevator and levels the fuselage.

Tuning & Behavior

- **Recommendation:** Set this to slightly below the stall speed of your aircraft.
- **Sequence:**
  1. Ground roll starts with tail held down.
  2. At `TKOFF_TDRAG_SPD1`, tail lifts and plane rolls on main wheels only.
  3. At `TKOFF_ROTATE_SPD`, the plane pitches up and climbs.
- **Benefit:** Ensures stable, straight-line takeoff rolls without the aircraft "nosing over" or bouncing.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





TKOFF_THR_DELAY (Takeoff Throttle Delay)

DEFAULT: 2	RANGE: 0 127	UNITS: ds

Takeoff Throttle Delay (TKOFF_THR_DELAY)

Description

`TKOFF_THR_DELAY` is a critical safety timer. It provides a brief pause between when you **throw** the plane and when the **propeller starts spinning**.

This delay is essential for hand-launches with pusher-propellers (motors at the back). It gives you enough time to move your hand away from the fuselage so that your fingers are not struck by the rotating blades.

- **Units:** Deci-seconds (1/10th of a second).
- **2 (Default):** 0.2 seconds.
- **Recommendation:** Use at least **2** for all hand launches. Use **20-30** (2-3 seconds) for bungee/catapult launches.

The Mathematics

The delay is a hard timer (T_wait) applied after the acceleration check passes:

$$T_wait = \text{TKOFF_THR_DELAY} \times 0.1 \text{ seconds}$$

The throttle is held at 0% until this timer expires. After expiration, the `TKOFF_THR_MINSPD` check is performed.

The Engineer's View

Defined in `ArduPlane/takeoff.cpp`.

```
// ArduPlane/takeoff.cpp
uint16_t wait_time_ms = MIN(uint16_t(g.takeoff_throttle_delay)*100,12700);
```

The maximum allowed delay is 12.7 seconds. This is sufficient for even the longest catapult rails or complex bungee setups.

Tuning & Behavior

- **Default Value:** 2 (0.2s).
- **Effect of Increasing:** The motor starts later. Safe for your hands, but the plane will lose more altitude before the motor takes over.
- **Effect of Decreasing:** The motor starts sooner. High risk of prop-striking your hand during launch.

Use Case Recommendations

- **Bungee Launch:** Set to **30 (3.0s)**. This ensures the bungee ring has completely detached from the plane's hook before the propeller starts spinning, preventing the line from tangling in the motor.
- **Hand Launch (Front Prop):** Set to **0 or 1**. Your hand is behind the motor, so risk is low.
- **Hand Launch (Pusher Prop):** Set to **2 or 5**. You need to be fast, but safe.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





TKOFF_THR_MAX (Maximum Throttle for Takeoff)

DEFAULT: 0	RANGE: 0 100	UNITS: %

Maximum Throttle for Takeoff (TKOFF_THR_MAX)

Description

`TKOFF_THR_MAX` allows you to give your plane an "Extra Boost" for the launch.

In normal cruise, you might want to limit your throttle to 75% to save battery. However, during a hand-launch or a steep climb-out, you need 100% of the available power to avoid hitting the ground. This parameter lets you set a different power limit specifically for the takeoff phase.

- **0 (Default):** Use the global `THR_MAX` setting for takeoff as well.
- **100:** Allow full motor power during takeoff, even if the normal limit is lower.

The Engineer's View

Used by the **TECS (Total Energy Control System)**.
During the takeoff flight stage, the `aparm.throttle_max` value is replaced by `g.takeoff_throttle_max` if it is non-zero. This ensures the energy controller has the maximum possible "Energy Add" capability to maintain airspeed and climb angle simultaneously.

Tuning & Behavior

- **Default Value:** 0.
- **Recommendation:** Set to **100** for all hand-launches to ensure the best possible climb rate.
- **Constraint:** This value is still subject to your ESC's physical limits and the `MOT_PWM_MAX` settings.

TKOFF_THR_MIN (Takeoff Minimum Throttle)

DEFAULT: 0	RANGE: 0 100	UNITS: %

Takeoff Minimum Throttle (TKOFF_THR_MIN)

Description

`TKOFF_THR_MIN` forces the motor to keep pulling during the takeoff climb.

Normally, the TECS controller manages throttle. If the plane climbs too steeply and airspeed drops, TECS adds power. If the plane is fast and climbing well, TECS might reduce power. This parameter sets a "Floor" to prevent the throttle from dropping too low, ensuring a strong, positive climb-out.

The Engineer's View

Used when `TKOFF_OPTIONS` bit 0 is set (Allow variable throttle).
The throttle demand is clamped: `output = max(output, TKOFF_THR_MIN)`.

Tuning & Behavior

- **Default Value:** 0 (Use `THR_MIN`).
- **Recommendation:** Set to **50% or 75%** for hand-launched planes to guarantee they have energy to clear obstacles, even if the airspeed sensor thinks the plane is already fast enough.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





TKOFF_THR_MINACC (Takeoff Throttle Min Acceleration)

DEFAULT: 0	RANGE: 0 30	UNITS: m/s^2

Takeoff Throttle Min Acceleration (TKOFF_THR_MINACC)

Description

`TKOFF_THR_MINACC` is the "Starting Gun" for an automatic plane launch. It detects the physical throw of your hand or the pull of a bungee cord to tell the autopilot: "The mission has begun."

Without this parameter, the drone might try to start its motor the moment you switch to AUTO mode. With it, the drone sits silently until it feels a sharp forward jolt.

- **0 (Default):** Acceleration check is disabled. The takeoff sequence starts immediately upon entering AUTO mode.
- **15.0 m/s²:** Recommended for hand-launches. This requires a firm, confident throw to trigger.

The Mathematics

The autopilot uses the forward-axis accelerometer (X-axis) and removes the effects of gravity and tilt. It looks for a peak value:

$$\text{Forward Acceleration} > \text{TKOFF_THR_MINACC}$$

Once this threshold is crossed, the takeoff timer starts. For even greater safety, you can require multiple "jolts" using the `TKOFF_ACCEL_CNT` parameter.

The Engineer's View

The value is sourced from the `TECS_controller.get_VXdot()` method, which provides a highly filtered and compensated acceleration estimate.

```
// ArduPlane/takeoff.cpp
float xaccel = TECS_controller.get_VXdot();
if (xaccel < g.takeoff_throttle_min_accel) {
    goto no_launch;
}
```

This prevents the takeoff from being triggered by wind gusts or simple tilting of the aircraft, as only true longitudinal acceleration is considered.

Tuning & Behavior

- **Default Value:** 0
- **Range:** 0 to 30 m/s^2 (0 to ~3G).
- **Effect of Increasing:** Requires a harder throw. Safer against accidental triggers while walking with the drone.
- **Effect of Decreasing:** Easier to trigger. Good for weak throwers or lightweight planes.

Troubleshooting

- **Scenario:** I threw the plane but the motor never started!
  - **Diagnosis:** Your throw was too weak, or `TKOFF_THR_MINACC` is set too high.
  - **Fix:** Check your logs for `TECS.VXdot`. Reduce the parameter to match 70% of your typical throw peak.
- **Scenario:** The motor started while I was just walking to the launch site!
  - **Diagnosis:** `TKOFF_THR_MINACC` is too low or 0.
  - **Fix:** Set to at least **10.0** to require a deliberate motion.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY

TKOFF_THR_MINSPD (Takeoff Throttle Min Speed)

DEFAULT: 0	RANGE: 0 30	UNITS: m/s

Takeoff Throttle Min Speed (TKOFF_THR_MINSPD)

Description

`TKOFF_THR_MINSPD` is a safety gate for your airplane's motor. It ensures that the propeller does not start spinning until the aircraft is already moving at a safe speed.

This is primarily used for **Hand Launches** and **Catapult Launches**.

- For **Hand Launches**, it prevents the motor from starting while the drone is still in your hand, protecting your fingers from a pusher-prop.
- For **Catapults**, it ensures the motor only engages after the plane has gained some initial velocity from the bungee or rail.
- **0 (Default):** Speed check is disabled. The motor starts as soon as the acceleration and delay checks pass.
- **4.0 m/s:** Recommended for hand-launched pusher-prop planes.

The Mathematics

The takeoff state machine evaluates three conditions in sequence:

1. **Acceleration:** Longitudinal acceleration $> \text{TKOFF_THR_MINACC}$.
2. **Delay:** Wait for $T_delay = \text{TKOFF_THR_DELAY} \times 0.1$ seconds.
3. **Speed:** GPS Ground Speed $> \text{TKOFF_THR_MINSPD}$.

The throttle remains at 0% until:

$$(V_gps > \text{TKOFF_THR_MINSPD}) \vee (\text{TKOFF_THR_MINSPD} = 0)$$

The Engineer's View

Defined in `ArduPlane/takeoff.cpp` inside `auto_takeoff_check()`.

The logic is designed to be robust against GPS noise. The ground speed check only arms *after* the acceleration check passes. This prevents a "GPS Jump" (where the coordinates jump 10 meters while sitting still) from accidentally starting the motor.

```
// ArduPlane/takeoff.cpp
if (((gps.ground_speed() > g.takeoff_throttle_min_speed || is_zero(g.takeoff_throttle_min_speed))) &&
    ((now - takeoff_state.last_tkoff_arm_time) >= wait_time_ms)) {
    // Motor unsuppressed
}
```

Tuning & Behavior

- **Default Value:** 0
- **Effect of Increasing:** Safer launches. The motor will start later in the throw/launch.
- **Effect of Decreasing:** Faster motor response. The motor starts sooner.
- **Crucial Note:** GPS speed usually lags real speed by about **0.5 seconds**. Account for this lag when setting the speed threshold.

Use Case Recommendations

- **Bungee Launch:** Set to **0** and rely on `TKOFF_THR_DELAY` (e.g., 3.0s) to let the bungee fall away before the prop starts.
- **Hand Launch (Pusher):** Set to **4.0 m/s**. This ensures the plane is already out of your hand before the blades spin up.
- **Rolling Takeoff (Runway):** Set to **0**. You want the motor to start immediately to begin the roll.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





TKOFF_THR_SLEW (Takeoff Throttle Slew Rate)

DEFAULT: 0	RANGE: -1 100	UNITS: %/s

Takeoff Throttle Slew Rate (TKOFF_THR_SLEW)

Description

`TKOFF_THR_SLEW` prevents your motor from jumping from 0% to 100% instantly. It forces a smooth, gradual "ramp up" of power.

This is critical for **Runway Takeoffs** where a sudden burst of power could cause the plane to veer sideways or "torque roll" before the wings are generating lift. It is also easier on your battery and ESC, as it avoids massive instantaneous current spikes.

- **0 (Default):** Use the global `THR_SLEWRATE` value (typically 100%/s).
- **-1:** No limit. The throttle jumps instantly. (Dangerous for large props!).
- **20:** Ramps the throttle from 0 to 100 over **5 seconds**.

The Engineer's View

This parameter provides a dedicated slew rate limit specifically for the `AUTO` and `TAKEOFF` flight stages. It helps stabilize the initial ground run where aerodynamic surfaces like the rudder are not yet effective.

```
// ArduPlane/takeoff.cpp
// Limits the rate of change of the internal _throttle_output variable.
```

Tuning & Behavior

- **Default Value:** 0.
- **Recommendation for Runway:** Set to **20 or 30**. This provides a scale-like, controllable acceleration.
- **Recommendation for Hand-Launch:** Keep at **0 or set to 100**. You want the power as fast as possible once the plane has left your hand.

Troubleshooting

- **Scenario:** My plane always spins out on the runway as soon as the motor starts.
  - **Diagnosis:** The sudden torque of the propeller is overcoming the steering traction.
  - **Fix:** Reduce `TKOFF_THR_SLEW` to 20.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





TKOFF_TIMEOUT (Takeoff Safety Timeout)

DEFAULT: 0	RANGE: 0 120	UNITS: s

Takeoff Safety Timeout (TKOFF_TIMEOUT)

Description

`TKOFF_TIMEOUT` is a vital "Dead-Man's Switch" for automatic launches. It monitors your aircraft's progress and aborts the mission if the plane isn't speeding up as expected.

If you launch your plane but the motor fails, or if a wheel is stuck in a rut on the runway, the drone might continue to sit there with the throttle active, or it might stumble into the grass. `TKOFF_TIMEOUT` ensures that if the plane doesn't reach at least 4 m/s (roughly 9 mph) within this time limit, the autopilot kills the motor and disarms the drone.

- **0 (Default):** Timeout is disabled. The drone will keep trying to take off until the battery dies or it is manually disarmed.
- **Recommendation:** Set to **5 or 10 seconds**.

The Engineer's View

Defined in `ArduPlane/takeoff.cpp`.
The timer starts the moment the "Triggered AUTO" message appears (after the acceleration and delay checks pass). The logic checks the GPS ground speed against a fixed internal threshold (4.0 m/s).

```
// ArduPlane/takeoff.cpp
if (g2.takeoff_timeout > 0 &&
    !takeoff_state.launchTimerStarted &&
    now - takeoff_state.start_time_ms > (uint32_t)g2.takeoff_timeout*1000U &&
    gps.ground_speed() < 4.0f) {
    gcs().send_text(MAV_SEVERITY_CRITICAL, "Takeoff timeout");
    disarm_motors();
}
```

Tuning & Behavior

- **Default Value:** 0 (Disabled).
- **Effect of Setting:** Provides a crucial safety backup for autonomous operations.
- **Usage Tip:** If you have a very large, slow-accelerating plane, ensure your timeout is long enough to account for the slow build-up of speed.

Troubleshooting

- **Scenario:** My drone disarms immediately after a hand-launch!
  - **Diagnosis:** `TKOFF_TIMEOUT` is set too low, or your `TKOFF_THR_DELAY` is so long that the timeout expires before the motor even starts.
  - **Fix:** Increase `TKOFF_TIMEOUT` to 10.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **TMODE** parameter group configures the **Toy Mode** engine. This is a specialized simplified interface used for consumer drones (like the SkyViper) or for integrating ArduPilot with cheap gamepad-style controllers.

Toy Mode simplifies flight control and abstracts many complex settings into easy-to-use button mappings.

Key Concepts

1. Button Mapping (`TMODE_ACTIONx`)

Maps specific gamepad buttons to ArduPilot actions.

- **Actions:** Flip, Take Photo, Change Mode, Land.

2. Simplified Tuning (`TMODE_TMAX` / `VMAX`)

Toy Mode often uses simplified limits for speed and tilt to make the drone easier for children or beginners to fly.

Parameter Breakdown

- `TMODE_ENABLE`: Master switch.
- `TMODE_TRIM_AUTO`: Automatically calculates trim from stick positions.

Developer Notes

- **Library:** `ArduCopter/toy_mode.cpp`.

TMODE_ACTION1 (Toy Mode Action 1)

DEFAULT: 2	RANGE: 0 24

Toy Mode Action 1 (TMODE_ACTION1)

Description

`TMODE_ACTION1` maps a specific function to a button on the transmitter.

Values

- **1:** Take Photo.
- **2:** Toggle Video.
- **7:** Mode RTL.
- **18:** Disarm.

TMODE_ACTION2 (Toy Mode Action 2)

DEFAULT: 1	RANGE: 0 24

Toy Mode Action 2 (TMODE_ACTION2)

Description

`TMODE_ACTION2` maps a specific function to the secondary action button on a consumer transmitter.

Values

- **1:** Take Photo (Default).
- **2:** Toggle Video.
- **8:** Mode Circle.
- **16:** Flip.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





TMODE_ACTION3 (Toy Mode Action 3)

DEFAULT: 18	RANGE: 0 24

Toy Mode Action 3 (TMODE_ACTION3)

Description

TMODE_ACTION3 defines the behavior of the third utility button on the transmitter.

Values

- **18: Disarm (Default).** Instantly kills the motors.
- **7: Mode RTL.** Return to home.
- **9: Mode Land.**

TMODE_ACTION4 (Toy Mode Action 4)

DEFAULT: 0	RANGE: 0 24

Toy Mode Action 4 (TMODE_ACTION4)

Description

TMODE_ACTION4 enables a "Shift Key" functionality for the transmitter buttons. By holding down the Mode button (Left), the other buttons can perform secondary tasks.

Values

- **0: None (Default).**
- **23: Motor Load Test.**
- **15: Mode Throw.**

TMODE_ACTION5 (Toy Mode Action 5)

DEFAULT: 0	RANGE: 0 24

Toy Mode Action 5 (TMODE_ACTION5)

Description

TMODE_ACTION5 maps a secondary function to the right action button while in the "Shifted" state.

TMODE_ACTION6 (Toy Mode Action 6)

DEFAULT: 0	RANGE: 0 24

Toy Mode Action 6 (TMODE_ACTION6)

Description

TMODE_ACTION6 maps a secondary function to the power button while in the "Shifted" state.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





TMODE_ENABLE (Toy Mode Enable)

DEFAULT: 0	RANGE: 0 2

Toy Mode Enable (TMODE_ENABLE)

Description

`TMODE_ENABLE` activates a simplified control scheme often used in consumer drones.

Instead of the standard ArduPilot stick behaviors and button mappings, Toy Mode reconfigures the flight controller to respond to specific button layouts (like those found on SkyViper transmitters) and enables "easy" features like auto-takeoff on high throttle.

- **0: Disabled (Default).** Standard ArduPilot behavior.
- **1: Version 1 (SkyViper V2450GPS).**
- **2: Version 2 (F412-based boards).**

Tuning & Behavior

- **Default Value:** 0.
- **Warning:** Do not enable this on a custom-built drone unless you are specifically using a SkyViper transmitter, as it will drastically change how your sticks and buttons work.
- **Reboot Required:** Yes.

TMODE_FLAGS (Toy Mode Flags)

DEFAULT: 0	RANGE: 0 15

Toy Mode Flags (TMODE_FLAGS)

Description

`TMODE_FLAGS` toggles specific "Ease of Use" behaviors.

- **Bit 0 (1): Disarm on Low Throttle.** Automatically disarms the drone if the throttle is held at minimum for a few seconds.
- **Bit 1 (2): Arm on High Throttle.** Allows arming the motors by simply pushing the throttle stick up (Auto-Takeoff style).
- **Bit 2 (4): Upgrade to Loiter.** Automatically switches from `AltHold` to `Loiter` once a solid GPS fix is acquired.
- **Bit 3 (8): RTL Cancel.** Cancels an automatic Return-to-Launch if the pilot provides a large stick input.

TMODE_LEFT (Toy Mode Left Action)

DEFAULT: 19	RANGE: 0 24

Toy Mode Left Action (TMODE_LEFT)

Description

`TMODE_LEFT` handles the primary "Mode" button.

- **19: Toggle Mode (Default).** Switches between the primary modes configured in `TMODE_MODE1` and `TMODE_MODE2` (typically swapping between AltHold and Loiter).

TMODE_LEFT_LONG (Toy Mode Left Long Action)

DEFAULT: 0	RANGE: 0 24

Toy Mode Left Long Action (TMODE_LEFT_LONG)

Description

`TMODE_LEFT_LONG` defines the behavior for a sustained press of the Mode button.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





TMODE_LOAD_FILT (Toy Mode Load Test Filter)

DEFAULT: 1	RANGE: 0 100

Toy Mode Load Test Filter (TMODE_LOAD_FILT)

Description

`TMODE_LOAD_FILT` controls the temporal resolution of thrust commands during a Toy Mode Load Test.

Instead of applying a new thrust value on every loop, this parameter "stretches" each value.

- **1:** No filtering. Commands are applied at the full loop rate.
- **X:** Each thrust value from the test sequence is repeated X times.

Tuning & Behavior

- **Default:** 1.
- **Usage:** Used to slow down the rate of change during battery stress tests to see how the voltage recovers between steps.
- **Context:** Only applicable when TMODE_LOAD_TYPE is set to a Log Replay mode.

TMODE_LOAD_MUL (Load test multiplier)

DEFAULT: 1.0	RANGE: 0 1

TMODE_LOAD_MUL: Load test multiplier

Description

This scales the load test output, as a value between 0 and 1.

Values

- **Range:** 0 to 1
- **Default:** 1.0

Description

This parameter is part of the **Toy Mode** factory test suite (`ENABLE_LOAD_TEST` must be compiled in).

- **Function:** It multiplies the motor PWM output during the automated load test sequence.
- **Usage:** Used to run the motors at a reduced power level (e.g., **0.5**) during endurance testing to verify battery life or thermal performance without running at full hover power.

TMODE_LOAD_TYPE (Toy Mode Load Test Type)

DEFAULT: 1	RANGE: 0 2

Toy Mode Load Test Type (TMODE_LOAD_TYPE)

Description

`TMODE_LOAD_TYPE` is used for factory testing and battery performance analysis on consumer drones using Toy Mode.

When a Load Test is active (triggered via specialized commands), this parameter determines the thrust profile:

- **0: Constant Thrust.** The drone applies a steady, fixed amount of power.
- **1: Log Replay 1.** Follows a pre-defined sequence of thrust variations to simulate a real flight profile.
- **2: Log Replay 2.** An alternative test sequence.

Tuning & Behavior

- **Default:** 1.
- **Warning:** This is an engineering/diagnostic feature. Activating a load test while the drone is on the ground can cause it to flip or fly away if propellers are attached.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





TMODE_MODE1 (Toy Mode Primary Mode 1)

DEFAULT: 2	RANGE: 0 25

Toy Mode Primary Mode 1 (TMODE_MODE1)

Description

`TMODE_MODE1` sets the default flight mode that the drone starts in when powered on. In Toy Mode, this is typically a mode that does not require GPS (like `AltHold`).

Tuning & Behavior

- **Default Value:** 2 (AltHold).

TMODE_MODE2 (Toy Mode Primary Mode 2)

DEFAULT: 5	RANGE: 0 25

Toy Mode Primary Mode 2 (TMODE_MODE2)

Description

`TMODE_MODE2` is the "Advanced" mode for Toy Mode drones, typically requiring a GPS fix (like `Loiter`).

TMODE_RIGHT (Tmode right action)

DEFAULT: 2	RANGE: 0 5

Tmode right action (TMODE_RIGHT)

Description

This parameter defines the function of the "Right" shoulder button (often labeled "Return") on Toy-grade controllers supported by ArduCopter's Toy Mode.

The Mathematics

Enum Mapping.

The Engineer's View

Defined in `ArduCopter/toy_mode.cpp`.

- **0:** None
- **1:** Safety (Toggle Motor Interlock)
- **2:** Arm / Land / RTL (Context sensitive)
- **3:** Mode Toggle
- **4:** Stunt

Tuning & Behavior

- **Default Value:** 2 (Arm/Land/RTL)
- **Behavior:** Press to Arm. Press in flight to RTL. Press near ground to Land.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





TMODE_TMAX (Toy Mode Max Thrust Multiplier)

DEFAULT: 1.0	RANGE: 0.1 1.0

Toy Mode Max Thrust Multiplier (TMODE_TMAX)

Description

`TMODE_TMAX` is the low-voltage counterpart to `TMIN`. It defines the scaling factor applied when the battery is near empty.

Typically, this is kept at **1.0** (full available power) to compensate for the naturally lower voltage of a drained battery.

Tuning & Behavior

- **Default:** 1.0.
- **Physics:** As voltage drops, RPM drops. By setting `TMIN` < 1.0 and `TMAX` = 1.0, the autopilot "unlocks" more signal range as the battery drains to maintain constant lift.

TMODE_TMIN (Toy Mode Min Thrust Multiplier)

DEFAULT: 1.0	RANGE: 0.1 1.0

Toy Mode Min Thrust Multiplier (TMODE_TMIN)

Description

`TMODE_TMIN` is used in **Toy Mode** to ensure the drone's throttle feel remains consistent even as the battery drains.

A fully charged battery produces more power than a low battery. To prevent the drone from feeling "hyper" at the start of a flight and "sluggish" at the end, Toy Mode scales the thrust. `TMIN` defines the scaling factor applied at the **Maximum** voltage.

- **1.0 (Default):** No scaling.
- **0.8:** Reduces the motor power to 80% when the battery is full.

Tuning & Behavior

- **Default:** 1.0.
- **Recommendation:** Leave at 1.0 unless you find the drone is too difficult to control with a fresh battery.
- **Integration:** Works in conjunction with TMODE_TMAX.

TMODE_TRIM_AUTO (Toy Mode Auto Stick Trim)

DEFAULT: 50	RANGE: 0 100	UNITS: PWM

Toy Mode Auto Stick Trim (TMODE_TRIM_AUTO)

Description

`TMODE_TRIM_AUTO` is a convenience feature for consumer drones using low-cost transmitters (RC controllers).

Cheap transmitters often have "Drift" in their gimbal sticks – the center point might not be exactly 1500 PWM every time. In Toy Mode, while the vehicle is **Disarmed**, the autopilot monitors the sticks. If they are near the center and remain steady for several seconds, the autopilot assumes that position is the intended "Neutral" and applies a small trim offset automatically.

Tuning & Behavior

- **Default:** 50 PWM.
- **Safety:** This only happens while disarmed. If the stick is too far from center (more than this parameter's value), the autopilot will NOT apply the trim, assuming you are intentionally holding the stick.
- **Benefit:** Prevents the drone from slowly drifting in one direction just because your transmitter's potentiometers are slightly aged or imprecise.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





TMODE_VMAX (Toy Mode Max Voltage Threshold)

DEFAULT: 3.8	RANGE: 3.0 4.2	UNITS: V

Toy Mode Max Voltage Threshold (TMODE_VMAX)

Description

`TMODE_VMAX` sets the "Ceiling" for battery compensation in Toy Mode.

When the battery is above this voltage, the autopilot applies the minimum scaling set in TMODE_TMIN.

- **Default:** 3.8V.
- **Physics:** Defines the X_2 point in a linear interpolation for thrust scaling. Between `VMIN` and `VMAX`, the autopilot smoothly varies the motor power to keep the drone's vertical performance constant.

TMODE_VMIN (Toy Mode Min Voltage Threshold)

DEFAULT: 3.5	RANGE: 3.0 4.2	UNITS: V

Toy Mode Min Voltage Threshold (TMODE_VMIN)

Description

`TMODE_VMIN` sets the "Floor" for battery compensation in Toy Mode.

When the battery voltage drops to this level, the autopilot applies the maximum scaling set in TMODE_TMAX. This is calibrated for single-cell (1S) voltages, as Toy Mode is primarily used on small consumer drones.

Tuning & Behavior

- **Default:** 3.5V.
- **Physics:** Defines the X_1 point in a linear interpolation for thrust scaling.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY

Overview

The **TRIM** parameter group (specifically `TRIM_THROTTLE` or `THROTTLE_CRUISE`) defines the steady-state cruise values for the propulsion system.

Key Concepts

1. Cruise Throttle

- `TRIM_THROTTLE`: (ArduPlane/Rover) The percentage of throttle required to maintain `CRUISE_SPEED` in level flight.
- **Usage:** Provides the FeedForward baseline for the speed controller. If set correctly, the drone will maintain speed perfectly when you switch from Manual to an autonomous mode.

Developer Notes

- **Library:** `libraries/AP_TECS`, `libraries/AR_AttitudeControl`.
- **See Also:** CRUISE for related speed settings.

Throttle cruise percentage ()

TRIM_THROTTLE: Throttle cruise percentage

Description

Target percentage of throttle to apply for flight in automatic throttle modes and throttle percentage that maintains `AIRSPEED_CRUISE`.

Values

- **Range:** 0 to 100
- **Units:** %
- **Default:** 45

Description

This parameter sets the baseline throttle level that the autopilot will use to maintain the target cruise airspeed in level flight.

- **Function:** It is the "Feed-Forward" term for the throttle controller. If `TRIM_THROTTLE` is set correctly, the drone will fly at `AIRSPEED_CRUISE` without the throttle PID loop having to do much work.
- **Tuning:** Fly in FBWB or Cruise mode at your desired speed. Note the throttle percentage on your OSD or logs. Set `TRIM_THROTTLE` to this value.
- **Important:** If you don't have an airspeed sensor, this parameter *defines* your cruise speed.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **TRQD** parameter group configures the driver for **Torqeedo Electric Boat Motors**.

Torqeedo is a leading manufacturer of electric outboards for boats. ArduPilot can communicate with these motors via a serial link (RS485) to read battery level, power consumption, and to provide precise throttle control.

Key Concepts

1. Communication Type (`TRQD_TYPE`)

Defines how the autopilot talks to the motor.

- **0:** Disabled.
- **1:** Serial (RS485).

2. Output Scaling (`TRQD_POWER`)

Sets the maximum power limit for the motor, allowing the pilot to cap the speed for trolling or to extend battery life.

Parameter Breakdown

- `TRQD_TYPE`: Driver selection.
- `TRQD_DE_PIN`: (RS485) Data Enable pin for the UART adapter.
- `TRQD_SLEW_TIME`: Limits how fast the motor power can ramp up/down.

Integration Guide

1. **Hardware:** Requires a UART-to-RS485 adapter. Connect the adapter to an autopilot UART and the motor's diagnostic port.
2. **Config:** Set `TRQD_TYPE = 1` and `SERIALx_PROTOCOL = 41` (Torqeedo).

Developer Notes

- **Library:** `libraries/AP_Torqeedo`.
- **Context:** Only used in **ArduRover** (Boats).

TRQD_DE_PIN (Torqeedo RS485 Data Enable Pin)

DEFAULT: -1	RANGE: -1 103

Torqeedo RS485 Data Enable Pin (TRQD_DE_PIN)

Description

`TRQD_DE_PIN` is required when using a generic RS485-to-Serial converter module to talk to the Torqeedo motor.

RS485 is a half-duplex protocol. The converter needs a signal to know when to transmit and when to listen. This pin provides that "Push-to-Talk" signal.

Tuning & Behavior

- **-1 (Default):** Use the serial port's CTS pin (if hardware flow control is supported).
- **0-103:** Physical GPIO pin number connected to the DE/RE pin on the RS485 module.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





TRQD_DIR_DELAY (Torqeedo Direction Change Delay)

DEFAULT: 0.5	RANGE: 0 2.0	UNITS: s

Torqeedo Direction Change Delay (TRQD_DIR_DELAY)

Description

`TRQD_DIR_DELAY` protects the motor's gearbox and electronics from high-torque reversals.

Electric boat motors generate massive torque. Attempting to switch from full-forward to full-reverse instantly can strip gears or blow electrical fuses. This parameter introduces a mandatory "neutral" pause to allow the propeller to slow down before it is powered in the opposite direction.

Tuning & Behavior

- **Default:** 0.5 seconds.
- **Recommendation:** Set to **1.0 second** for larger propellers to ensure a smooth and safe transition.

TRQD_ONOFF_PIN (Torqeedo Power Control Pin)

DEFAULT: -1	RANGE: -1 103

Torqeedo Power Control Pin (TRQD_ONOFF_PIN)

Description

`TRQD_ONOFF_PIN` allows ArduPilot to electronically "Push the Power Button" on your Torqeedo motor.

Torqeedo motors require a specific startup sequence. By connecting a GPIO pin to the motor's power control line, the autopilot can automatically wake the motor up when the boat is armed and put it to sleep when disarmed, saving battery and ensuring readiness.

- **-1:** Disabled (Manual power control).
- **0-103:** Physical GPIO pin number.

Tuning & Behavior

- **Recommendation:** Use the serial port's RTS pin if your hardware supports it, as it provides a clean, dedicated signal line for this purpose.

TRQD_OPTIONS (Torqeedo Integration Options)

DEFAULT: 1	RANGE: 0 1

Torqeedo Integration Options (TRQD_OPTIONS)

Description

`TRQD_OPTIONS` is a bitmask to enable specific communication behaviors with the Torqeedo Tiller or motor controller.

- **Bit 0 (1): Reverse Direction.** If set, the forward/reverse logic of the Torqeedo motor is swapped at the protocol level.

Tuning & Behavior

- **Default:** 1.
- **Usage:** Enable Bit 0 if your boat moves backward when the autopilot commands forward speed.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY

TRQD_POWER (Torqeedo Motor Power Limit)

DEFAULT: 100	RANGE: 0 100	UNITS: %

Torqeedo Motor Power Limit (TRQD_POWER)

Description

`TRQD_POWER` allows you to cap the maximum output of a Torqeedo electric boat motor.

Since Torqeedo motors are very powerful, you may want to limit the maximum energy consumption to extend your mission time or to prevent the motor from overwhelming a small hull. This parameter scales the autopilot's full-throttle request to this percentage of the motor's actual capability.

Tuning & Behavior

- **Default:** 100%.
- **Usage:** Set to **50** or **75** to improve energy efficiency for long-range autonomous boating.
- **Dependency:** Only active if TRQD_TYPE is set to a motor driver mode.

TRQD_SERVO_FN (Torqeedo Servo Function)

DEFAULT: 0	RANGE: 0 100

Torqeedo Servo Function (TRQD_SERVO_FN)

Description

`TRQD_SERVO_FN` maps the Torqeedo driver to a specific motor output channel logic.

- **73:** Throttle Left.
- **74:** Throttle Right.
- **0:** Disabled.

TRQD_SLEW_TIME (Torqeedo Throttle Slew Time)

DEFAULT: 2.0	RANGE: 0 5.0	UNITS: s

Torqeedo Throttle Slew Time (TRQD_SLEW_TIME)

Description

`TRQD_SLEW_TIME` ensures smooth power delivery to the **Torqeedo** motor.

Large electric boat motors generate massive torque. If the autopilot "slams" the throttle from 0 to 100% instantly, it can cause the boat to lurch, put excessive strain on the transom, or even trigger a fuse/breaker due to the massive current surge. This parameter limits how fast the throttle can increase.

- **Higher Value (e.g. 5.0):** Very slow, majestic ramp-up.
- **Lower Value (e.g. 0.5):** Very aggressive, snappy response.
- **0:** (Not Recommended). Instantaneous response.

Tuning & Behavior

- **Default:** 2.0 seconds.
- **Recommendation:** Leave at **2.0** or higher for large hulls to ensure mechanical longevity.
- **Note:** This only limits the *increase* in throttle; for safety, throttle can always be *decreased* much faster.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY



TRQD_TYPE (Torqeedo Connection Type)

DEFAULT: 0	RANGE: 0 2

Torqeedo Connection Type (TRQD_TYPE)

Description

`TRQD_TYPE` enables the autopilot to communicate with a **Torqeedo** electric outboard motor (common on large autonomous boats).

The autopilot communicates with the motor's internal controller over a serial link (usually via an RS485 converter). This allows ArduPilot to command thrust precisely and read back live telemetry like battery status and power consumption from the motor itself.

- **0: Disabled.**
- **1: Tiller.** Commands the motor as if the autopilot were the physical Torqeedo tiller.
- **2: Motor.** Direct motor control mode.

Tuning & Behavior

- **Reboot Required:** Yes.
- **Integration:** You must also configure the ON/OFF pin (TRQD_ONOFF_PIN) and the Serial port protocol (`SERIALx_PROTOCOL = 39`).

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

Key Concepts

`True speed and direction low pass filter frequency ()`

TRUE_FILT: True speed and direction low pass filter frequency

Description

True speed and direction low pass filter frequency. A value of -1 disables the filter.

Values

- **Range:** -1 to 10
- **Units:** Hz
- **Default:** 0.05

Description

This parameter smooths the calculated "True Wind" vector on autonomous sailboats.

- **Function:** The autopilot calculates True Wind by subtracting the boat's own velocity (from GPS) from the Apparent Wind (from the Wind Vane). Since both GPS and Vane data can be noisy, this Low-Pass Filter smooths the result.
- **Tuning:**
  - **0.05 Hz (Default):** Very slow filter (20 second period). Good for average wind conditions.
  - **Higher:** Faster response to gusts, but noisier steering.
  - **-1:** Disabled.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





IN-FLIGHT TUNING CONFIGURATION (TUNE)

Overview

The **TUNE** parameter group configures the **Transmitter Tuning** feature. This allows the pilot to map a PID gain or other parameter (like `ATC_RAT_RLL_P`) to a knob or slider on their RC transmitter.

This is a powerful tool for finding the "Sweet Spot" of a gain while the vehicle is in the air.

Key Concepts

1. Parameter Selection (`TUNE_PARAM`)

Defines which parameter is currently mapped to the transmitter knob.

- **0:** Disabled.
- **1:** Roll Rate P.
- **2:** Roll Rate I.
- **...** (See Wiki for full list).

2. Scaling (`TUNE_MIN` / `MAX`)

Defines the range of the knob.

- `TUNE_MIN`: The value of the parameter when the knob is at its minimum position.
- `TUNE_MAX`: The value at maximum position.

Parameter Breakdown

- `TUNE_CHAN`: The RC input channel assigned to the tuning knob.
- `TUNE_SELECTOR`: Allows assigning a 3-position switch to toggle *what* parameter is being tuned (e.g., Switch Low = P, Mid = I, High = D).

Integration Guide

1. **Map Channel:** Set `TUNE_CHAN` to your slider channel.
2. **Select Param:** Set `TUNE_PARAM = 1` (Roll P).
3. **Set Range:** If your current P is 0.1, set `MIN = 0.05` and `MAX = 0.2`.
4. **Fly:** Move the slider until the drone feels stable but crisp. Land and check the value.

Developer Notes

- **Library:** `libraries/AP_Tuning`.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





TUNE (Channel 6 Tuning (Copter))

DEFAULT: 0	RANGE: 0 100

Channel 6 Tuning (TUNE)

Description

`TUNE` is the master selector for in-flight parameter adjustment on ArduCopter.

By assigning a knob on your transmitter to Channel 6, you can tune gains while the drone is in the air. This is the fastest way to achieve a perfect tune.

Values

- **1:** Roll/Pitch P.
- **3:** Roll/Pitch I.
- **4:** Throttle P.
- **50:** Loiter P.
- *Refer to the Full List for all 50+ options.*

Tuning & Behavior

- **Default:** 0 (Disabled).
- **Requirement:** Requires TUNE_MIN and TUNE_MAX to be set to the safe range for the parameter you are tuning.
- **Safety:** Always start with the knob in the center (current known good value) and move it slowly.

TUNE_CHAN (Transmitter Tuning Channel)

DEFAULT: 0	RANGE: 0 16

Transmitter Tuning Channel (TUNE_CHAN)

Description

`TUNE_CHAN` enables you to tune your drone's PIDs or other parameters while you are actually flying, using a knob or slider on your transmitter.

Instead of landing, changing a parameter, and taking off again, you can map a knob to a parameter (e.g., Roll P) and find the perfect value by turning the knob in mid-air.

- **0:** Disabled.
- **5-16:** RC Channel number.

Tuning & Behavior

- **Prerequisite:** You must also set TUNE_CHAN_MIN and TUNE_CHAN_MAX to match the output of your knob.
- **Safety:** Always start with a small TUNE_RANGE to avoid making the drone unstable with a small turn of the knob.

TUNE_CHAN_MAX (Tuning Channel Maximum PWM)

DEFAULT: 2000	RANGE: 900 2100	UNITS: PWM

Tuning Channel Maximum PWM (TUNE_CHAN_MAX)

Description

`TUNE_CHAN_MAX` defines the "Full Scale" position for the tuning channel.

When your transmitter knob is at its highest position and sends this PWM value, the autopilot sets the parameter being tuned to TUNE_MAX.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





TUNE_CHAN_MIN (Tuning Channel Minimum PWM)

DEFAULT: 1000	RANGE: 900 2100	UNITS: PWM

Tuning Channel Minimum PWM (TUNE_CHAN_MIN)

Description

`TUNE_CHAN_MIN` defines the "Zero Position" for the RC channel used for in-flight tuning.

When your transmitter knob or slider is at its lowest position and sends this PWM value, the autopilot will set the parameter being tuned to TUNE_MIN.

Tuning & Behavior

- **Default:** 1000.
- **Calibration:** Ensure this matches the minimum PWM reported during your radio calibration for the channel assigned to tuning.

TUNE_ERR_THRESH (Tuning Instability Threshold)

DEFAULT: 0.15	RANGE: 0 1.0

Tuning Instability Threshold (TUNE_ERR_THRESH)

Description

`TUNE_ERR_THRESH` is an "Over-Gain Alarm" for in-flight tuning.

As you increase PID gains using your transmitter knob, the drone's movements will become sharper. However, if the gains go too high, the drone will start to vibrate or oscillate (instability). This parameter monitors the root-mean-square (RMS) of the PID corrections. If the corrections become too large and fast (indicating oscillation), the autopilot will sound a buzzer alarm and send a "Controller Instability" message to your Ground Control Station.

Tuning & Behavior

- **Default:** 0.15.
- **Recommendation:** Leave at **0.15**. If the alarm triggers, it means you have pushed your gains too far and should immediately lower the tuning knob.
- **-1:** Disables the alarm.

TUNE_MAX (Tuning maximum)

DEFAULT: 0

TUNE_MAX: Tuning maximum

Description

Transmitter Tuning maximum value. The parameter being tuned will have its value set to this maximum value when the tuning knob is at its highest position.

Values

- **Default:** 0

Description

This parameter works with the "In-Flight Tuning" feature.

- **Function:** When you rotate your transmitter's tuning knob to its maximum position, the parameter you have selected in `TUNE` will be set to this value (`TUNE_MAX`).
- **Example:** If tuning `PSC_POSXY_P` and you want to test P-gains up to 2.0, set `TUNE_MAX = 2.0`.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





TUNE_MIN (Tuning minimum)

DEFAULT: 0

TUNE_MIN: Tuning minimum

Description

Transmitter Tuning minimum value. The parameter being tuned will have its value set to this minimum value when the tuning knob is at its lowest position.

Values

- **Default:** 0

Description

This parameter works with the "In-Flight Tuning" feature (usually assigned to Channel 6).

- **Function:** When you rotate your transmitter's tuning knob to its minimum position, the parameter you have selected in `TUNE` will be set to this value (`TUNE_MIN`).
- **Example:** If you are tuning `PSC_POSXY_P`, and you set `TUNE_MIN = 0.5` and `TUNE_MAX = 2.0`, rotating the knob fully counter-clockwise sets P to 0.5.

TUNE_MODE_REVERT (Tuning Revert on Mode Change)

DEFAULT: 1	RANGE: 0 1

Tuning Revert on Mode Change (TUNE_MODE_REVERT)

Description

`TUNE_MODE_REVERT` is an important safety switch for in-flight tuning.

- **1: Enabled (Default/Safe).** If you are tuning your drone and it starts to oscillate dangerously, you can simply flick your flight mode switch to any other mode (e.g. from Stabilize to AltHold). The autopilot will immediately discard the current knob position and revert the parameter to the value it had when you started.
- **0: Disabled.** The tuned value is kept even if you change modes.

Tuning & Behavior

- **Recommendation:** Leave at **1 (Enabled)** until you are very comfortable with the tuning process. This provides a "Panic Button" to instantly recover from an unstable tune.
- **Usage:** Once you find a value you like, you must either "Save" it (using the selector switch or a MAVLink command) or manually enter it into the parameter list before power-cycling.

TUNE_RANGE (Transmitter Tuning Range)

DEFAULT: 2.0	RANGE: 1.1 5.0

Transmitter Tuning Range (TUNE_RANGE)

Description

`TUNE_RANGE` sets the "Sensitivity" of the tuning knob.

It defines the total spread of values the knob can command, centered around the value the parameter had when the tuning session started.

- **2.0 (Default):** The knob can adjust the parameter from half of its original value (at the low end) to double its original value (at the high end).
- **1.2:** A much smaller range, allowing for very fine control and reducing the risk of a crash.

Tuning & Behavior

- **Recommendation:** Start with **1.2 or 1.5** for high-risk parameters like PID gains. Use **2.0** for less sensitive settings.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





TUNE_SELECTOR (In-Flight Tuning Selector Channel)

DEFAULT: 0	RANGE: 0 16

In-Flight Tuning Selector Channel (TUNE_SELECTOR)

Description

`TUNE_SELECTOR` allows you to manage multiple tuning parameters from a single knob on your transmitter.

While TUNE_CHAN is the knob that actually changes the value, `TUNE_SELECTOR` is a separate switch (usually a 2-position spring-loaded switch) that tells the autopilot to:

1. **Toggle between parameters:** (If multiple parameters are in the set).
2. **Lock in the value:** Saves the current knob position to the parameter.
3. **Re-center:** Resets the knob's effect.

Tuning & Behavior

- **Default:** 0 (Disabled).
- **Recommendation:** Assign to a momentary switch on your transmitter. This allows you to "Dial in" a value with the knob, then "Click" the switch to save it before moving to the next parameter.

Transmitter tuning parameter or set of parameters ()

TUNE_PARAM: Transmitter tuning parameter or set of parameters

Description

This sets which parameter or set of parameters will be tuned.

Values

- **0:** None
- **1:** RLL_RATE_P
- **2:** RLL_RATE_I
- **3:** RLL_RATE_D
- **4:** PTCH_RATE_P
- **5:** PTCH_RATE_I
- **6:** PTCH_RATE_D
- **...**
- **101:** Roll P, I, and D simultaneously

Description

This parameter works in conjunction with `TUNE_CHAN` (Channel 6) to allow real-time adjustment of control gains while flying.

- **Values < 50:** Select a single specific parameter (e.g., `RLL_RATE_P`).
- **Values > 100:** Select a "Set" of parameters (e.g., 101 adjusts all Roll PID terms together, scaling them up or down proportionally).
- **QuadPlane:** Parameters less than 50 generally apply to the QuadPlane vertical lift motors if `Q_ENABLE` is active.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

Key Concepts

`Enable turbo mode ()`

TURBO_MODE: Enable turbo mode

Description

Enables double speed on high offset.

Values

Value	Meaning
0	Disabled
1	Enabled

- **Default:** 0

Description

This parameter is specific to the **Blimp** vehicle type when using flapping fin propulsion.

- **Function:** When enabled, if the fin offset is high (meaning a large steering or pitch command is applied) and the oscillation amplitude is low, the flapping frequency is doubled.
- **Effect:** This provides a burst of extra control authority ("Turbo") during aggressive maneuvers.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **TURN** parameter group configures the physical limits of the vehicle's turning capability.

Key Concepts

1. Lateral Acceleration (`TURN_MAX_G`)

(Rover/Plane) Defines the maximum lateral force (in Gs) the vehicle is allowed to pull in a turn.

- **Purpose:** Prevents high-speed tip-overs (Rover) or structural stress (Plane).

2. Turn Radius (`TURN_RADIUS`)

(Rover) The physical minimum turning circle of the vehicle at low speeds.

- **Importance:** Used by the path planner to ensure waypoints are achievable.

Developer Notes

- **Library:** `libraries/AP_L1_Control`, `libraries/AR_AttitudeControl`.

TURN_MAX_G (Turning maximum G force)

DEFAULT: 0.6	RANGE: 0.1 10	UNITS: G

TURN_MAX_G: Turning maximum G force

Description

The maximum turning acceleration (in units of gravities) that the rover can handle while remaining stable.

Values

- **Range:** 0.1 to 10
- **Units:** G (Gravities)
- **Default:** 0.6

Description

This parameter is a safety limit for **Rover**.

- **Function:** It tells the navigation controller the maximum lateral acceleration (cornering force) the vehicle can sustain without flipping over or skidding.
- **Effect:** If a waypoint requires a turn that would exceed this G-force at current speed, the rover will automatically slow down *before* entering the turn to keep the G-force below this limit.
- **Tuning:**
  - **Low CG / Wide vehicles:** Can handle higher values (e.g., 1.0G or more).
  - **High CG / Narrow vehicles:** Must be set lower (e.g., 0.3G) to prevent rollovers.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





TURN_RADIUS (Turn radius of vehicle)

DEFAULT: 0.9	RANGE: 0 10	UNITS: m

Turn radius of vehicle (TURN_RADIUS)

Description

This parameter defines the **Physical Minimum Turn Radius** of the rover at low speeds. It is a critical constraint for the navigation path planner (L1 controller). The autopilot uses this value to ensure it does not attempt turns sharper than the vehicle is mechanically capable of making, which would result in large crosstrack errors.

The Mathematics

Used to clamp lateral acceleration demand:

$$a_lat_max = \frac{V^2}{\text{TURN_RADIUS}}$$

(Simplified relation for limiting path curvature).

The Engineer's View

Defined in `Rover/Parameters.cpp`.

- **Skid Steering:** Even though skid steer vehicles can turn in place (Radius = 0), setting a non-zero value here (e.g., 0.5m) helps produce smoother, flowing turns during waypoint missions rather than stopping and pivoting.
- **Ackermann Steering:** Must be set to the actual mechanical limit (plus a safety margin).

Tuning & Behavior

- **Default Value:** 0.9 m
- **Procedure:** Drive the rover in a full lock circle manually, measure the radius, and enter that value here.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY

Overview

The **UART** parameter group (specifically within **AP_Periph**) configures the physical serial ports on peripheral hardware.

Key Concepts

1. Serial Mapping

On DroneCAN nodes, UARTs are used to connect GPS receivers or ADS-B units.

- `UARTx_OPTIONS`: Bitmask for signal inversion and swapping.
- `UARTx_RTSCTS`: Enables hardware flow control.

Developer Notes

- **Library:** `Tools/AP_Periph`.
- **See Also:** SERIAL for the main autopilot serial configuration.

UART0_OPTIONS (UART 0 Serial Options)

DEFAULT: 0	RANGE: 0 15

UART 0 Serial Options (UART0_OPTIONS)

Description

`UART0_OPTIONS` (usually appearing as `SERIAL0_OPTIONS`) provides low-level electrical control over the primary communication port.

This is often used when connecting the flight controller to non-standard hardware that requires specific signal logic.

- **Bit 0 (1): Invert RX.** Flips the high/low logic for receiving.
- **Bit 1 (2): Invert TX.** Flips the high/low logic for transmitting.
- **Bit 2 (4): Half-Duplex.** Combines TX and RX onto a single wire (One-Wire mode).
- **Bit 3 (8): Swap Pins.** Swaps the RX and TX functionality (Requires STM32 F7/H7).

Tuning & Behavior

- **Default:** 0.
- **Usage:** Only change if your external hardware (like an RC receiver or specialized sensor) explicitly requires it.
- **Reboot Required:** Yes.

UART0_RTSCTS (UART 0 Flow Control)

DEFAULT: 0	RANGE: 0 2

UART 0 Flow Control (UART0_RTSCTS)

Description

`UART0_RTSCTS` enables Hardware Flow Control for the primary serial/USB port.

It is functionally identical to UART1_RTSCTS.

UART1_OPTIONS (Serial options)

DEFAULT: 0	RANGE: null

Serial options

Note: This parameter functions identically to SERIAL1_OPTIONS.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





UART1_RTSCTS (UART 1 Flow Control)

DEFAULT: 2	RANGE: 0 3

UART 1 Flow Control (UART1_RTSCTS)

Description

`UART1_RTSCTS` enables Hardware Flow Control for the first telemetry port.

Flow control uses two extra wires (RTS and CTS) to signal when a device is ready to receive data. This prevents "Buffer Overflows" where one device sends data faster than the other can process it, which would otherwise result in corrupted or missing telemetry.

- **0: Disabled.**
- **1: Enabled.**
- **2: Auto (Default).** The autopilot will attempt to detect if the connected device supports flow control.

Tuning & Behavior

- **Recommendation:** Leave at **2 (Auto)**.
- **SiK Radios:** These radios support hardware flow control. If you have the RTS/CTS wires connected, this setting will ensure maximum data reliability.

UART2_OPTIONS (Serial options)

DEFAULT: 0	RANGE: null

Serial options

Note: This parameter functions identically to SERIAL1_OPTIONS.

UART2_RTSCTS (Serial1 flow control)

DEFAULT: float(AP_HAL::UARTDriver::FLOW_CONTROL_DISABLE	RANGE: null

Serial1 flow control

Note: This parameter functions identically to UART1_RTSCTS.

UART3_OPTIONS (Serial options)

DEFAULT: 0	RANGE: null

Serial options

Note: This parameter functions identically to SERIAL1_OPTIONS.

UART3_RTSCTS (Serial1 flow control)

DEFAULT: float(AP_HAL::UARTDriver::FLOW_CONTROL_DISABLE	RANGE: null

Serial1 flow control

Note: This parameter functions identically to UART1_RTSCTS.

UART4_OPTIONS (Serial options)

DEFAULT: 0	RANGE: null

Serial options

Note: This parameter functions identically to SERIAL1_OPTIONS.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





UART4_RTSCTS (Serial1 flow control)

DEFAULT: float(AP_HAL::UARTDriver::FLOW_CONTROL_DISABLE	RANGE: null

Serial1 flow control

Note: This parameter functions identically to UART1_RTSCTS.

UART5_OPTIONS (Serial options)

DEFAULT: 0	RANGE: null

Serial options

Note: This parameter functions identically to SERIAL1_OPTIONS.

UART5_RTSCTS (Serial1 flow control)

DEFAULT: float(AP_HAL::UARTDriver::FLOW_CONTROL_DISABLE	RANGE: null

Serial1 flow control

Note: This parameter functions identically to UART1_RTSCTS.

UART6_OPTIONS (Serial options)

DEFAULT: 0	RANGE: null

Serial options

Note: This parameter functions identically to SERIAL1_OPTIONS.

UART6_RTSCTS (Serial1 flow control)

DEFAULT: float(AP_HAL::UARTDriver::FLOW_CONTROL_DISABLE	RANGE: null

Serial1 flow control

Note: This parameter functions identically to UART1_RTSCTS.

UART7_OPTIONS (Serial options)

DEFAULT: 0	RANGE: null

Serial options

Note: This parameter functions identically to SERIAL1_OPTIONS.

UART7_RTSCTS (Serial1 flow control)

DEFAULT: float(AP_HAL::UARTDriver::FLOW_CONTROL_DISABLE	RANGE: null

Serial1 flow control

Note: This parameter functions identically to UART1_RTSCTS.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





UART8_OPTIONS (Serial options)

DEFAULT: 0	RANGE: null

Serial options

Note: This parameter functions identically to SERIAL1_OPTIONS.

UART8_RTSCTS (Serial1 flow control)

DEFAULT: float(AP_HAL::UARTDriver::FLOW_CONTROL_DISABLE	RANGE: null

Serial1 flow control

Note: This parameter functions identically to UART1_RTSCTS.

UART9_OPTIONS (Serial options)

DEFAULT: 0	RANGE: null

Serial options

Note: This parameter functions identically to SERIAL1_OPTIONS.

UART9_RTSCTS (Serial1 flow control)

DEFAULT: float(AP_HAL::UARTDriver::FLOW_CONTROL_DISABLE	RANGE: null

Serial1 flow control

Note: This parameter functions identically to UART1_RTSCTS.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

Key Concepts

UDP_PORT (DDS UDP port)

DEFAULT: 2019	RANGE: 1 65535

UDP_PORT: DDS UDP port

Description

UDP port number for DDS.

Values

- **Range:** 1 to 65535
- **Default:** 2019

Description

This parameter sets the destination UDP port for the Data Distribution Service (DDS) interface, which allows ArduPilot to communicate with ROS 2 (Robot Operating System).

- **Function:** The autopilot acts as a DDS client and sends telemetry to an external agent (like Micro-XRCE-DDS-Agent running on a companion computer).
- **Default (2019):** This is the standard port for Micro-XRCE-DDS.
- **Config:** Ensure your companion computer is listening on this same port.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY

Overview

Key Concepts

USE_REV_THRUST (Bitmask for when to allow negative reverse thrust)

DEFAULT: 2	RANGE: 0 8191	UNITS: Bitmask

Bitmask for when to allow negative reverse thrust (USE_REV_THRUST)

Description

This bitmask controls exactly when the autopilot is allowed to use reverse thrust (negative throttle) during autonomous flight modes. Reverse thrust is a powerful tool for steep descents and shortening landing rolls, but it can cause stalls or loss of control if engaged inappropriately (e.g., during normal cruise or loiter).

The Mathematics

The parameter is a bitmask where each bit represents a flight condition:

- **Bit 0:** Auto Always
- **Bit 1:** Auto Land Approach (Most Common)
- **Bit 2:** Auto Loiter to Alt
- **Bit 3:** Auto Loiter All
- **Bit 4:** Auto Waypoint
- **Bit 5:** Loiter
- **Bit 6:** RTL
- **Bit 7:** Circle
- **Bit 8:** Cruise
- **Bit 9:** FBWB
- **Bit 10:** Guided
- **Bit 11:** Auto Landing Pattern
- **Bit 12:** FBWA

The Engineer's View

Defined in `ArduPlane/Parameters.cpp` and used throughout the flight mode logic (e.g., `ArduPlane/reverse_thrust.cpp`).

- **Safeguards:** Reverse thrust usually requires `THR_MIN` to be negative.
- **Default (2):** `USE_REVERSE_THRUST_AUTO_LAND_APPROACH`. This is the safest default, allowing braking only when committed to a landing.

Tuning & Behavior

- **Default Value:** 2 (Auto Land Approach Only)
- **0:** Never use reverse thrust in auto modes.
- **2 (Bit 1):** Allow only during the landing sequence.
- **65 (Bit 0 + Bit 6):** Allow in all Auto modes + RTL.
- **Warning:** Enabling reverse thrust in modes like `FBWA` or `CRUISE` can lead to mid-air stops if the pilot pulls back on the throttle stick. Use with caution.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY

Overview

Key Concepts

V_OFFSET (OSD vertical offset)

DEFAULT: 16	RANGE: 0 31

V_OFFSET: OSD vertical offset

Description

Sets vertical offset of the osd inside image.

Values

- **Range:** 0 to 31
- **Default:** 16

Description

This parameter centers the OSD overlay vertically on your video screen.

- **Function:** Because different cameras (PAL vs NTSC) and video transmitters have different timing, the OSD text might appear too high or too low. This value shifts the entire text overlay up or down.
- **Tuning:** If the top row of text is cut off, increase this value. If the bottom row is cut off, decrease it.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

Key Concepts

VEC_ANGLEMAX (Vector thrust angle max)

DEFAULT: 0.0	RANGE: 0 90	UNITS: deg

VEC_ANGLEMAX: Vector thrust angle max

Description

The angle between steering's middle position and maximum position when using vectored thrust (boats only).

Values

- **Range:** 0 to 90
- **Units:** deg
- **Default:** 0.0

Description

This parameter is used for boats equipped with **Vectored Thrust** (e.g., an outboard motor or an airboat with rudders behind the prop).

- **Function:** It tells the autopilot the physical range of motion of the steering mechanism.
- **Usage:** If your outboard motor can turn 30 degrees left and 30 degrees right, set this parameter to **30**.
- **Effect:** The autopilot uses this angle to calculate the correct mixing between thrust and steering force. If set to 0, vectored thrust mixing is disabled.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **VEL** parameter group (specifically `VEL_M_NSE`) configures the expected noise level of the velocity measurements within the **EKF**.

Key Concepts

1. Velocity Noise

- `VEL_M_NSE`: Measurement noise for the velocity sensors (typically GPS).
  - **Higher values:** Tells the EKF to trust the GPS velocity less (good for noisy environments).
  - **Lower values:** Trusts the GPS velocity more (crisper position response but more susceptible to glitches).

Developer Notes

- **Library:** `libraries/AP_NavEKF3`.

VEL_M_NSE (Visual odometry velocity measurement noise)

DEFAULT: 0.1	RANGE: 0.05 5.0	UNITS: m/s

VEL_M_NSE: Visual odometry velocity measurement noise

Description

Visual odometry velocity measurement noise in m/s.

Values

- **Range:** 0.05 to 5.0
- **Units:** m/s
- **Default:** 0.1

Description

This parameter sets the "Trust" level for the velocity reported by a Visual Odometry (VO) system (like Intel RealSense T265 or OpenMV).

- **Function:** It tells the EKF how noisy the velocity data is.
- **Lower Value:** The EKF trusts the visual odometry velocity more than other sensors (like GPS or IMU integration). Use this if your VO system is very accurate and stable.
- **Higher Value:** The EKF trusts the visual odometry less. Use this if the VO system drifts or jumps frequently.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY

Overview

The **VIB** parameter group (specifically `VIB_FREQ`) configures the frequency of the vibration reporting system.

Key Concepts

1. Vibration Monitoring

ArduPilot tracks the health of the IMU by measuring vibration levels (clipping and variance).

- `VIB_FREQ`: The update rate for the internal vibration data messages.

Developer Notes

- **Library:** `libraries/AP_InertialSensor`.
- **Importance:** High vibration is the #1 cause of EKF failure and erratic flight behavior.

VIB_FREQ (Vibration Frequencies)

DEFAULT: 0	RANGE: 0 1000	UNITS: Hz

VIB_FREQ: Vibration Frequencies

Description

Vibration frequencies on each axis.

Values

- **Range:** 0 to 1000
- **Units:** Hz
- **Default:** 0

Description

This parameter injects simulated vibration noise into the SITL (Software In The Loop) physics engine.

- **Function:** It sets the base frequency of the vibration.
- **Components:** This is a Vector3f parameter (X, Y, Z). You can set different frequencies for each axis.
- **Usage:** Used by developers to test how well the autopilot's filters (`INS_GYRO_FILTER`, `INS_ACCEL_FILTER`) handle noise.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

Key Concepts

VICON_POS (SITL Vicon Position)

DEFAULT: 0	RANGE: -5 5	UNITS: m

VICON_POS: SITL Vicon Position

Description

Position of the Vicon/Optitrack marker on the vehicle body frame (X, Y, Z).

Values

- **Range:** -5 to 5
- **Units:** m
- **Default:** 0, 0, 0

Description

This parameter simulates the offset of an external motion capture target (like Vicon or Optitrack) in the SITL simulator.

- **Function:** If your simulated drone has the Vicon tracking balls mounted 10cm above the center of gravity, you would set the Z component of this parameter to **-0.1**.
- **Effect:** Allows testing of offset compensation logic in the EKF external navigation drivers.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY

Overview

The **VISO** parameter group configures the **Visual Odometry** (External Navigation) subsystem. This allows ArduPilot to use position and velocity data from an external camera-based system (like Intel RealSense or an OptiTrack mo-cap system) for navigation.

Visual Odometry is the primary method for high-performance non-GPS autonomous flight.

Key Concepts

1. Vision Types (`VISO_TYPE`)

- **1 (MAVLink):** Receives `VISION_POSITION_ESTIMATE` or `ODOMETRY` MAVLink messages from a companion computer.
- **2 (Intel RealSense):** Specialized driver for RealSense devices.

2. Offsets and Orientation

- `VISO_POS_X/Y/Z`: The physical distance from the camera lens to the vehicle's center of gravity.
- `VISO_ORIENT`: The rotation of the camera relative to the flight controller.

Parameter Breakdown

- `VISO_ENABLE`: Master switch.
- `VISO_DELAY_MS`: Time lag of the vision system (crucial for EKF synchronization).
- `VISO_QUAL_MIN`: Minimum confidence threshold for the vision solution.

Integration Guide

1. **Preparation:** Requires an H7 processor and a high-speed telemetry link (921k baud).
2. **Config:** Set `VISO_TYPE = 1`.
3. **EKF Source:** Set `EK3_SRC1_POSXY = 6` (External Nav).
4. **Verify:** Check the GCS map. The drone's position should move accurately as you walk it around a room.

Developer Notes

- **Library:** `libraries/AP_VisualOdom`.
- **Messages:** Primarily consumes the `MAVLINK_MSG_ID_ODOMETRY` message.

VISO_DELAY_MS (Visual Odometry Processing Delay)

DEFAULT: 10	RANGE: 0 250	UNITS: ms

Visual Odometry Processing Delay (VISO_DELAY_MS)

Description

`VISO_DELAY_MS` tells the autopilot how "Old" the vision-based position data is when it arrives.

Vision algorithms (like those in the Intel Realsense T265) take significant time to process images and calculate a pose. If the EKF (Extended Kalman Filter) thinks this data is happening "Now," but it actually represents where the drone was 50ms ago, the position estimate will be unstable. This parameter allows the autopilot to look back in its IMU history and align the vision data with the correct moment in time.

Tuning & Behavior

- **Default:** 10 ms.
- **Recommendation:** Consult your vision system's documentation for its typical latency. For the T265, **50ms to 100ms** is common.
- **Symptom of incorrect delay:** The drone may "Wobble" or circle slowly while trying to hold position, especially after a quick movement.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY

VISO_ORIENT (Visual Odometry Orientation)

DEFAULT: 0	RANGE: 0 35

Visual Odometry Orientation (VISO_ORIENT)

Description

`VISO_ORIENT` defines the mounting angle of your Visual Inertial Odometry (VIO) camera.

For the autopilot to correctly translate "Camera Motion" into "Vehicle Motion," it must know which way the camera is facing.

- **0: Rotation_None (Standard).** Camera is facing forward, right-side up.
- **Other Values:** Standard ArduPilot rotations (e.g. 25 for Downward).

Tuning & Behavior

- **Default:** 0.
- **Reboot Required:** Yes.
- **Validation:** View the `VISO` position data in your GCS while moving the drone. If moving forward causes the reported position to move backward or sideways, this orientation parameter is likely incorrect.

VISO_POS (Visual Odometry Position Offset)

DEFAULT: 0	RANGE: -5 5	UNITS: m

Visual Odometry Position Offset (VISO_POS)

Description

`VISO_POS` (often appearing as `VISO_POS_X`, `VISO_POS_Y`, `VISO_POS_Z`) defines the physical location of the camera lens relative to the drone's center of gravity (CoG).

Correctly setting these offsets allows the autopilot to perform "Lever Arm Compensation." This ensures that when the drone rotates (pitches or rolls), the resulting camera movement isn't mistaken for actual horizontal translation.

- **X:** Positive = Forward of CoG.
- **Y:** Positive = Right of CoG.
- **Z:** Positive = Below CoG.

Tuning & Behavior

- **Default:** 0.
- **Measurement:** Use a tape measure to find the distance from the center of the flight controller (or the vehicle's balance point) to the center of the camera lenses.
- **Accuracy:** Small errors (1-2cm) are usually acceptable, but errors > 5cm can cause instability in position hold during aggressive maneuvers.

VISO_POS_M_NSE (Visual Odometry Position Noise)

DEFAULT: 0.2	RANGE: 0.1 10.0	UNITS: m

Visual Odometry Position Noise (VISO_POS_M_NSE)

Description

`VISO_POS_M_NSE` tells the EKF how much to "Trust" the position data from your vision sensor.

If the vision sensor reports a noise value that is lower than this parameter, the autopilot overrides it with this value. This ensures the navigation filter remains conservative and doesn't over-react to potentially over-confident vision data, preventing the drone from "twitching" in high-gain vision scenarios.

Tuning & Behavior

- **Default:** 0.2 m.
- **Recommendation:** Leave at **0.2** for standard Intel T265 setups.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





VISO_QUAL_MIN (Visual Odometry Minimum Quality)

DEFAULT: 0	RANGE: -1 100	UNITS: %

Visual Odometry Minimum Quality (VISO_QUAL_MIN)

Description

`VISO_QUAL_MIN` defines the "Certainty Threshold" for your vision-based position data.

Computer vision systems (like the T265) report a "Confidence" score along with their position. In low light or over featureless surfaces (like smooth white floors), the system may become "lost" and start reporting random or drift-prone data. This parameter tells the autopilot to reject any vision data if the confidence falls below this level.

- **0 (Default):** Use data if the sensor thinks it's okay.
- **50:** Reject data if the sensor is less than 50% sure.
- **-1:** Trust all data, even if the sensor says it is failed (DANGEROUS).

Tuning & Behavior

- **Recommendation:** Set to **50** or higher for safety. If the vision system fails this quality check, the autopilot will fall back to GPS (if available) or trigger an EKF failsafe.

VISO_SCALE (Visual Odometry Scale Factor)

DEFAULT: 1.0	RANGE: 0.1 2.0

Visual Odometry Scale Factor (VISO_SCALE)

Description

`VISO_SCALE` allows you to calibrate the "Ruler" used by your vision sensor.

Sometimes, due to camera lens distortion or inaccuracies in the vision algorithm, the sensor might think it has moved 1.0 meters when it has actually moved 1.1 meters. This scaling factor allows you to correct that error so that simulated movement matches real-world distance.

- **1.0 (Default):** No scaling.
- **1.1:** Increases the reported distance by 10%.

Tuning & Behavior

- **Calibration:** Place the drone on a long measuring tape. Move it manually from 0 to 5 meters. Observe the reported `VISO` position in the GCS.
  - $$\text{Scale} = \frac{\text{Actual Distance}}{\text{Reported Distance}}$$
- **Note:** If your scale factor is far from 1.0 (e.g. 1.5 or 0.5), it usually indicates a more fundamental problem with the vision sensor's calibration or lens FOV settings.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





VISO_TYPE (Visual Odometry Type)

DEFAULT: 0	RANGE: 0 3

VISO_TYPE: Visual Odometry Type

Description

Selects the visual odometry camera driver.

Values

Value	Meaning
0	Disabled
1	MAVLink (OpenMV, T265 via MAVLink)
2	Intel T265 (USB) - Linux boards only
3	VOXL (Serial)

- **Default:** 0

Description

This parameter enables the **Visual Odometry** subsystem, allowing the drone to hold position and navigate without GPS by tracking features in the environment.

- **MAVLink (1):** Most common. Used when a companion computer (like a Raspberry Pi or Jetson) runs the VIO software (e.g., OpenVINS, ORB-SLAM) and sends the pose data to the flight controller via MAVLink.
- **T265 (2):** Only applicable for Linux-based flight controllers that can directly read the Intel Realsense T265 via USB.

VISO_VEL_M_NSE (Visual Odometry Velocity Noise)

DEFAULT: 0.1	RANGE: 0.05 2.0	UNITS: m/s

Visual Odometry Velocity Noise (VISO_VEL_M_NSE)

Description

`VISO_VEL_M_NSE` tells the EKF how much to trust the velocity (speed) reports from your vision sensor.

VISO_YAW_M_NSE (Visual Odometry Yaw Noise)

DEFAULT: 0.2	RANGE: 0.05 1.0	UNITS: rad

Visual Odometry Yaw Noise (VISO_YAW_M_NSE)

Description

`VISO_YAW_M_NSE` defines how much the autopilot trusts the heading data coming from a Visual Odometry (VO) sensor (like an Intel T265).

- **Low Value:** High trust. The EKF will rely heavily on the camera for heading.
- **High Value:** Low trust. The EKF will rely more on the IMU and Compass.

Tuning & Behavior

- **Default Value:** 0.2 rad.
- **Recommendation:** If the drone's heading "Glitchy" while using VO, increase this value to reduce the camera's influence.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **VTX** parameter group configures the **Video Transmitter** control interface. This allows the pilot to change the channel, band, and power level of their FPV transmitter directly through the ArduPilot OSD or GCS.

Key Concepts

1. SmartAudio and Tramp

ArduPilot supports the **SmartAudio** (TBS) and **Tramp** (IRC) protocols for VTX control.

- `VTX_ENABLE`: Master switch.

2. Frequency Control

- `VTX_BAND`: Selects the band (A, B, E, Fatshark, Raceband).
- `VTX_CHANNEL`: Selects the channel (1-8).
- `VTX_FREQ`: Displays the resulting frequency in MHz.

3. Power Control

- `VTX_POWER`: Sets the output power (e.g., 25mW, 200mW, 800mW).
- `VTX_MAX_POWER`: Limits the maximum power to ensure local legal compliance or prevent overheating.

Integration Guide

1. **Hardware:** Connect the SmartAudio/Tramp wire from the VTX to a UART TX pin.
2. **Serial:** Set `SERIALx_PROTOCOL = 37` (SmartAudio) or `42` (Tramp).
3. **Config:** Set `VTX_ENABLE = 1`.
4. **Control:** Access the VTX menu via the OSD to change settings in the field.

Developer Notes

- **Library:** `libraries/AP_VideoTX`.
- **Tables:** Supports "VTX Tables" in newer versions for custom frequency mapping.

Is the Video Transmitter enabled or not ()

VTX_ENABLE: Is the Video Transmitter enabled or not

Description

Toggles the Video Transmitter on and off.

Values

Value	Meaning
0	Disable
1	Enable

- **Default:** 0

Description

This parameter enables the **Video Transmitter (VTX) Control** subsystem.

- **Function:** When enabled, ArduPilot will attempt to communicate with the VTX (using SmartAudio, IRC Tramp, or CRSF) to set the channel, band, and power level based on the `VTX_` parameters.
- **Benefit:** Allows you to change your video channel from the OSD menu or Ground Control Station without pressing buttons on the VTX itself. It also enables features like "Pit Mode" to keep power low until you are ready to fly.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





VTX_BAND (Video Transmitter Band)

DEFAULT: 0	RANGE: 0 5

Video Transmitter Band (VTX_BAND)

Description

VTX_BAND defines the group of frequencies (Band) your video transmitter is using.

- **1: Band A.**
- **2: Band B.**
- **3: Band E.**
- **4: Airwave (Fatshark).**
- **5: Raceband.** (Most popular for multi-pilot racing).

Tuning & Behavior

- **Default:** 0.
- **Usage:** Set this to match your video goggles. If you are flying with others, coordinate your Band and Channel to prevent signal overlap.

VTX_CHANNEL (Video Transmitter Channel)

DEFAULT: 0	RANGE: 0 7

Video Transmitter Channel (VTX_CHANNEL)

Description

VTX_CHANNEL selects the specific frequency within your chosen VTX_BAND.

Most bands have 8 channels. This parameter uses an index from **0 to 7** (representing channels 1 to 8).

Tuning & Behavior

- **Default:** 0.
- **Usage:** If you are experiencing interference from another pilot, try switching to a different channel within your band.
- **Verification:** You should see the frequency update on your GCS status screen (if using SmartAudio/Tramp) and your goggles should lose and re-acquire the signal on the new frequency.

VTX_FREQ (Video Transmitter Frequency (Manual))

DEFAULT: 5800	RANGE: 1000 6000	UNITS: MHz

Video Transmitter Frequency (VTX_FREQ)

Description

VTX_FREQ allows you to specify a custom frequency for your video link, rather than using the standard VTX_BAND and VTX_CHANNEL presets.

Tuning & Behavior

- **Default:** 5800 MHz.
- **Note:** In most cases, it is better to use the Band and Channel parameters to ensure compatibility with standard video goggles.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





VTX_MAX_POWER (Video Transmitter Max Power Limit)

DEFAULT: 800	RANGE: 25 1000	UNITS: mW

Video Transmitter Max Power Limit (VTX_MAX_POWER)

Description

`VTX_MAX_POWER` acts as a safety "Cap" for your video transmitter.

If you have assigned an RC channel to change your VTX power while flying, this parameter prevents you from accidentally selecting a power level higher than your hardware can safely handle.

Tuning & Behavior

- **Default:** 800 mW.
- **Recommendation:** Set this to the maximum rated output of your physical VTX. If your VTX only supports up to 400mW, set this to **400** to avoid sending unsupported commands to the hardware.

VTX_POWER (Video Transmitter Power Level)

DEFAULT: 0	RANGE: 0 1000	UNITS: mW

Video Transmitter Power Level (VTX_POWER)

Description

`VTX_POWER` controls the strength of your video signal.

- **Standard Values:** 25, 100, 200, 400, 600, 800 (mW).
- **0:** Lowest available power (Pit mode).

Tuning & Behavior

- **Recommendation:** Use the lowest power necessary for your flight. High power settings (800mW+) generate significant heat and can damage the VTX if there is no airflow (e.g., sitting on the bench).
- **Protocol:** Requires a VTX that supports **SmartAudio** or **Tramp** protocol connected to a serial port.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

Key Concepts

W_AVGCELLV (AVGCELLV warn level)

DEFAULT: 3.6f	RANGE: 0 100	UNITS: null

AVGCELLV warn level

Note: This parameter configures instance AVGCELLV. It functions identically to OSD_W_AVGCELLV.

W_BATVOLT (BAT_VOLT warn level)

DEFAULT: 10.0f	RANGE: 0 100	UNITS: null

BAT_VOLT warn level

Note: This parameter configures instance BATVOLT. It functions identically to OSD_W_BATVOLT.

W_NSAT (NSAT warn level)

DEFAULT: 9	RANGE: 1 30	UNITS: null

NSAT warn level

Note: This parameter configures instance NSAT. It functions identically to OSD_W_NSAT.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **WENC** parameter group configures the **Wheel Encoders**. These sensors measure the rotation of the vehicle's wheels (or tracks) to provide an accurate estimate of ground velocity and distance traveled.

Wheel encoders are the primary method for high-precision navigation in GPS-denied environments for ground rovers.

Key Concepts

1. Encoder Types (`WENCn_TYPE`)

ArduPilot supports up to 2 wheel encoder instances.

- **1 (Quadrature):** Uses two digital pins (A and B) to detect both speed and direction.
- **2 (Single Pin):** Uses one pin to detect speed (cannot detect reverse automatically).

2. Resolution (`WENCn_CPR`)

Defines how many pulses the encoder generates for one full rotation of the motor or wheel.

- **Calculation:** `Pulses_per_revolution * Gear_Ratio`.

3. Physical Dimensions (`WENCn_RADIUS`)

The autopilot needs to know the radius (meters) of the wheel to convert "rotations" into "meters traveled."

Parameter Breakdown

- `WENC1_TYPE`: Driver selection.
- `WENC1_PINA` / `PINB`: GPIO pin assignments.
- `WENC1_POS_X/Y/Z`: Physical offset of the wheel from the vehicle center.

Integration Guide

1. **Hardware:** Mount encoders to the driven wheels.
2. **Enable:** Set `WENC1_TYPE = 1`.
3. **Calibrate:** Set the `CPR` and `RADIUS` accurately.
4. **Verify:** Push the rover forward 10 meters and verify the GCS reports exactly 10.0m traveled.

Developer Notes

- **Library:** `libraries/AP_WheelEncoder`.
- **EKF Fusion:** Wheel encoder data is fused into `AP_NavEKF3` to provide a "dead-reckoning" solution when GPS is lost.

WENC2_CPR (WheelEncoder 2 counts per revolution)

DEFAULT: WHEELENCODER_CPR_DEFAULT	RANGE: null

WheelEncoder 2 counts per revolution

Note: This parameter functions identically to WENC_CPR.

WENC2_PINA (Second Encoder Input Pin A)

DEFAULT: 53	RANGE: null

Second Encoder Input Pin A

Note: This parameter functions identically to WENC_PINA.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





WENC2_PINB (Second Encoder Input Pin B)

DEFAULT: 52	RANGE: null

Second Encoder Input Pin B

Note: This parameter functions identically to WENC_PINB.

WENC2_POS (Wheel2's Z position offset)

DEFAULT: 0.0f	RANGE: -5 5	UNITS: m

Wheel2's Z position offset

Note: This parameter functions identically to WENC_POS.

WENC2_RADIUS (Wheel2's radius)

DEFAULT: WHEELENCODER_RADIUS_DEFAULT	RANGE: null	UNITS: m

Wheel2's radius

Note: This parameter functions identically to WENC_RADIUS.

WENC2_TYPE (Second WheelEncoder type)

DEFAULT: 0	RANGE: null

Second WheelEncoder type

Note: This parameter functions identically to WENC_TYPE.

WENC_CPR (Wheel Encoder Counts Per Revolution)

DEFAULT: 0	RANGE: 0 100000

Wheel Encoder Counts Per Revolution (WENC_CPR)

Description

`WENC_CPR` calibrates the encoder resolution.

If your encoder has 1000 lines and uses quadrature (4x) decoding, the CPR is 4000. Correct setting is essential for accurate odometry.

Tuning & Behavior

- **Default Value:** 0 (or Driver Default).
- **Calculation:** `Lines * 4` for Quadrature.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





WENC_PINA (Wheel Encoder Pin A (Instance 1))

DEFAULT: -1	RANGE: -1 103

Wheel Encoder Pin A (WENC_PINA)

Description

`WENC_PINA` defines the input pin for Phase A of the primary wheel encoder.

Wheel encoders typically use two pins (A and B) to determine both speed and direction of rotation. Phase A provides the primary pulse count.

Tuning & Behavior

- **Default:** -1.
- **Setup:** Set this to the digital input pin connected to your encoder's A channel. If using an IOMCU-equipped board (like Pixhawk), these must typically be AUX pins configured as GPIO.
- **Integration:** You must also configure WENC_PINB for quadrature decoding to function.

WENC_PINB (Wheel Encoder Pin B (Instance 1))

DEFAULT: -1	RANGE: -1 103

Wheel Encoder Pin B (WENC_PINB)

Description

`WENC_PINB` defines the input pin for Phase B of the primary wheel encoder.

In a quadrature encoder, Phase B is offset from Phase A. By comparing the timing of the pulses on these two pins, the autopilot can detect if the wheel is spinning forward or backward.

Tuning & Behavior

- **Default:** -1.
- **Direction Fix:** If your Rover reports it is moving forward when it is actually reversing, swap the values of `WENC_PINA` and `WENC_PINB` in the parameters.

WENC_POS (Wheel Encoder Position Offset)

DEFAULT: 0	RANGE: -5 5	UNITS: m

Wheel Encoder Position Offset (WENC_POS)

Description

`WENC_POS` (often appearing as `WENC_POS_X/Y/Z`) tells the autopilot exactly where the wheel is located on your frame.

If you are using encoders for high-precision autonomous navigation (like indoor mapping), the autopilot must know if the wheel is at the front, back, or side of the drone. This allows the EKF to correctly translate the wheel's rotation into the movement of the vehicle's center of gravity.

- **X:** Positive = Forward.
- **Y:** Positive = Right.
- **Z:** Positive = Down.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





WENC_POS_X (Wheel Encoder Position X)

DEFAULT: 0	RANGE: -5 5	UNITS: m

Wheel Encoder Position X (WENC_POS_X)

Description

`WENC_POS_X` defines the forward/backward offset of the wheel relative to the flight controller (or COG).

Accurate positioning helps the EKF account for the "Lever Arm" effect when the vehicle turns.

Tuning & Behavior

- **Default Value:** 0 m.

WENC_POS_Y (Wheel Encoder Position Y)

DEFAULT: 0	RANGE: -5 5	UNITS: m

Wheel Encoder Position Y (WENC_POS_Y)

Description

`WENC_POS_Y` defines the left/right offset of the wheel.

Tuning & Behavior

- **Default Value:** 0 m.

WENC_POS_Z (Wheel Encoder Position Z)

DEFAULT: 0	RANGE: -5 5	UNITS: m

Wheel Encoder Position Z (WENC_POS_Z)

Description

`WENC_POS_Z` defines the vertical offset.

Tuning & Behavior

- **Default Value:** 0 m.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





WENC_RADIUS (Wheel Radius)

DEFAULT: 0.05	RANGE: 0.001 10.0	UNITS: m

Wheel Radius (WENC_RADIUS)

Description

`WENC_RADIUS` defines the physical size of the wheel being monitored.

This value is used by the autopilot to translate "Wheel Rotations" (from the encoder) into "Meters Traveled" (for the EKF). If this value is incorrect, the drone will have a massive error in its position and speed estimate when relying on odometry.

The Mathematics

$$\text{Distance} = \text{Rotations} \times 2\pi \times \text{WENC_RADIUS}$$

Tuning & Behavior

- **Default:** 0.05 m (5 cm).
- **Recommendation:** Measure the wheel carefully with calipers. Include the tire if applicable.
- **Note:** If your Rover is moving 10m in reality but the GCS says 11m, your radius parameter is likely 10% too large.

WENC_TYPE (Wheel Encoder Type (Instance 1))

DEFAULT: 0	RANGE: 0 10

Wheel Encoder Type (WENC_TYPE)

Description

`WENC_TYPE` enables the primary wheel encoder.

Wheel encoders provide high-precision distance and velocity measurements by counting the rotations of the vehicle's wheels. This is critical for Rovers operating in GPS-denied environments (Dead Reckoning) or for precise low-speed positioning.

- **0: None.** Disabled.
- **1: Quadrature.** For standard optical or magnetic encoders with A/B signal phases.
- **10: SITL Quadrature.** For simulation testing.

Tuning & Behavior

- **Reboot Required:** Yes.
- **Integration:** Once enabled, you must also configure the counts per revolution (WENC_CPR) and the wheel radius (WENC_RADIUS).

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **WINCH** parameter group configures the **Autonomous Winch** subsystem. This allows the vehicle to lower and raise a payload on a cable.

Key Concepts

1. Winch Types (`WINCH_TYPE`)

- **1 (Servo):** Standard PWM winch servo.
- **2 (MAVLink):** Winch controlled via an external MAVLink device.
- **3 (DroneCAN):** CAN-based winch.

2. Rate Limits (`WINCH_RATE_MAX`)

Limits the speed (m/s) at which the cable is deployed or retracted.

Developer Notes

- **Library:** `libraries/AP_Winch`.
- **Interaction:** Tightly coupled with the `PLDP` (Payload Place) library for automated cargo delivery.

`WINCH_RATE_MAX` (Winch deploy or retract rate maximum)

DEFAULT: 1.0f	RANGE: 0 10	UNITS: m/s

Winch deploy or retract rate maximum (WINCH_RATE_MAX)

Description

This parameter sets the hard limit for the winch motor's speed when deploying or retracting the tether. It protects the winch mechanism from over-speeding, particularly in "slack line" conditions where the load is minimal.

The Mathematics

$$\omega_\text{cmd} = \min(\omega_\text{req}, \text{WINCH_RATE_MAX})$$

The Engineer's View

Defined in `libraries/AP_Winch/AP_Winch.cpp`. The winch library calculates the required rate based on the target length or velocity request, then clamps it to this value before sending the output to the ESC or servo.

Tuning & Behavior

- **Default Value:** 1.0 m/s
- **Safety:** Ensure this value is within the safe operating limits of your physical winch hardware.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY

Winch Type ()

WINCH_TYPE: Winch Type

Description

Selects the type of winch hardware connected to the vehicle.

Values

Value	Meaning
0	None (Disabled)
1	PWM (Servo/ESC controlled)
2	Daiwa (Specific serial protocol)

- **Default:** 0

Description

This parameter enables the Winch library for package delivery or tether management.

- **PWM (1):** Allows control of a generic continuous rotation servo or ESC-driven motor using the `WINCH_` parameters.
- **Daiwa (2):** Enables support for the Daiwa winch protocol.

Source Code

ardupilot/libraries/AP_Winch/AP_Winch.cpp

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **WIND** parameter group (specifically `WIND_MAX`) configures the safety limits for wind speed.

Key Concepts

1. Wind Limit (`WIND_MAX`)

(ArduPlane) Defines the maximum wind speed (m/s) that the autopilot is allowed to attempt to fly in.

- **Action:** If the estimated environmental wind exceeds this value, the autopilot may trigger a warning or refuse to start a mission.

Developer Notes

- **Library:** `libraries/AP_AHRS`.
- **Estimation:** Wind is estimated by subtracting the Airspeed vector from the GPS Ground Speed vector.

Maximum airspeed and ground speed difference ()

WIND_MAX: Maximum airspeed and ground speed difference

Description

If the difference between airspeed and ground speed is greater than this value the sensor will be marked unhealthy.

Values

- **Range:** 0 to 100
- **Units:** m/s
- **Default:** 0

Description

This parameter sets a sanity check for the airspeed sensor.

- **Logic:** The autopilot assumes that `Airspeed - GroundSpeed` roughly equals `Wind Speed`. If this difference exceeds `WIND_MAX`, it implies the wind is impossibly strong (e.g., a hurricane) or, more likely, the airspeed sensor is giving bad data (e.g., pitot tube blocked).
- **Effect:** If this threshold is breached, the EKF may declare the airspeed sensor unhealthy and stop using it for flight control.
- **0:** Disabled (No check).

Source Code

ardupilot/libraries/AP_Airspeed/AP_Airspeed.cpp

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **WNDSPD** parameter group (specifically `WNDSPD_MIN`) configures the environmental thresholds for specialized vehicle behaviors.

Key Concepts

1. Sailing Threshold (`WNDSPD_MIN`)

(ArduRover Sailboat) Defines the minimum wind speed required to engage the sailing logic.

- **Behavior:** If the wind is below this threshold, the boat may rely on its motor (if equipped) rather than attempting to tack inefficiently.

Developer Notes

- **Library:** `Rover/sailboat.cpp`.

WNDSPD_MIN (Sailboat minimum wind speed to sail in)

DEFAULT: 0	RANGE: 0 5	UNITS: m/s

WNDSPD_MIN: Sailboat minimum wind speed to sail in

Description

Sailboat minimum wind speed to continue sail in, at lower wind speeds the sailboat will motor if one is fitted.

Values

- **Range:** 0 to 5
- **Units:** m/s
- **Default:** 0

Description

This parameter sets the threshold for **Hybrid Sailing**.

- **Function:** If the wind drops below this speed, the autopilot assumes sailing is ineffective and will start the motor (if equipped) to maintain progress towards the waypoint.
- **Use Case:** Cruising or long-endurance missions where you want to sail as much as possible but need to ensure you don't get becalmed.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **WNDVN** parameter group configures the **Wind Vane** (Anemometer) sensor. This is required for **Autonomous Sailing** and provides high-accuracy wind data for landing and takeoff on all vehicles.

Key Concepts

1. Sensor Types (`WNDVN_TYPE`)

Defines the hardware protocol.

- **1 (Analog):** Potentiometer-based vane (reads voltage).
- **2 (I2C):** Digital I2C sensor (e.g., AS5600).
- **3 (NMEA):** Standard marine NMEA 0183 wind sensor.
- **4 (DroneCAN):** CAN-based sensors.

2. Calibration (`WNDVN_DIR_OFS`)

Aligns the sensor's "zero" position with the vehicle's "Forward" axis.

Parameter Breakdown

- `WNDVN_SPEED_TYPE`: Selection of speed sensor (e.g., cup anemometer vs ultrasonic).
- `WNDVN_DIR_PIN`: (Analog) ADC pin number.

Integration Guide

1. **Hardware:** Mount the vane at the highest point of the vehicle, clear of air turbulence.
2. **Config:** Set `WNDVN_TYPE`.
3. **Calibrate:** Point the vane exactly forward and use `WNDVN_DIR_OFS` to zero the reading in the GCS.

Developer Notes

- **Library:** `libraries/AP_WindVane`.

`WNDVN_CAL` (Wind Vane Calibration Trigger)

DEFAULT: 0	RANGE: 0 2

Wind Vane Calibration Trigger (WNDVN_CAL)

Description

`WNDVN_CAL` initiates a specialized routine to find the voltage limits of your analog wind vane.

Instead of manually typing in the V_MIN and V_MAX voltages, you can use this trigger to let the autopilot learn them.

- **1: Start Calibration.** The autopilot begins recording the highest and lowest voltages it sees.
- **2: Finish and Save.** Saves the learned values to the `V_MIN/V_MAX` parameters and resets this trigger to 0.

Tuning & Behavior

- **Default:** 0.
- **Procedure:**
    1. Set `WNDVN_CAL` to 1.
    2. Manually rotate your wind vane through several full 360-degree circles.
    3. Set `WNDVN_CAL` to 2 to lock in the results.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





WNDVN_DIR_DZ (Wind Vane Analog Deadzone)

DEFAULT: 0	RANGE: 0 360	UNITS: deg

Wind Vane Analog Deadzone (WNDVN_DIR_DZ)

Description

`WNDVN_DIR_DZ` compensates for the "Mechanical Gap" found in many analog wind vanes.

Most analog vanes use a simple potentiometer to measure direction. In a full 360-degree rotation, there is usually a tiny physical area where the metal "wiper" inside the sensor isn't touching the resistive track. In this gap, the sensor may output a random or incorrect voltage. This parameter tells the autopilot how many degrees wide that gap is, so it can ignore the data and "Hold" the last known good heading while the vane passes through that sector.

Tuning & Behavior

- **Default:** 0.
- **Calibration:** Rotate the vane slowly. If you see the direction "Jump" wildly at one specific point in the circle, measure the width of that jump in degrees and enter it here.
- **Note:** Only applicable for **Analog** wind vanes.

WNDVN_DIR_FILT (Wind Vane Direction Filter)

DEFAULT: 0.5	RANGE: -1.0 10.0	UNITS: Hz

Wind Vane Direction Filter (WNDVN_DIR_FILT)

Description

`WNDVN_DIR_FILT` smooths the raw data coming from the wind vane.

Wind is naturally turbulent, and physical vanes tend to "jitter" or bounce rapidly. This filter removes that high-frequency noise, providing a stable "Apparent Wind" direction for the autopilot to use for sail trimming.

- **Higher Frequency (e.g. 2.0 Hz):** More responsive to quick wind shifts, but "nervous" sails.
- **Lower Frequency (e.g. 0.1 Hz):** Very smooth sail movement, but may be too slow to react to real gusts.
- **-1:** Disables the filter (Not recommended).

Tuning & Behavior

- **Default:** 0.5 Hz.
- **Recommendation:** Leave at **0.5 Hz**. If your sail servos are constantly twitching, reduce this value to **0.2 Hz**.

WNDVN_DIR_OFS (Wind Vane Direction Offset)

DEFAULT: 0	RANGE: 0 360	UNITS: deg

Wind Vane Direction Offset (WNDVN_DIR_OFS)

Description

`WNDVN_DIR_OFS` allows you to calibrate your wind vane without having to physically remount it. It defines the angle the sensor reports when the wind is coming directly from the front of the vehicle.

- **Units:** Degrees.
- **Calibration:** Point the vehicle directly into the wind and hold the wind vane at its "neutral" position. If the HUD reports 10 degrees, set the offset to -10 (or 350).

Tuning & Behavior

- **Default:** 0.
- **Usage:** Use this to align the digital "Front" of the sensor with the physical "Front" of your vehicle.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





WNDVN_DIR_PIN (Wind Vane Direction Pin)

DEFAULT: -1	RANGE: -1 103

Wind Vane Direction Pin (WNDVN_DIR_PIN)

Description

`WNDVN_DIR_PIN` sets the physical analog pin on the flight controller where the wind vane's direction signal is connected.

This parameter is only used if `WNDVN_TYPE` is set to 3 (Analog).

- **-1: Not used.**
- **0-9: Analog Pin Number.** (e.g., 0 for A0).
- **50-55: AUX Out Pins.** (When configured as analog inputs).
- **103: RSSI/SBUS pin.** (On some hardware).

Tuning & Behavior

- **Default:** -1.
- **Selection:** Consult your flight controller's hardware documentation (GPIO/Analog Pin mapping) to find the correct number for the pin you have used.

WNDVN_DIR_V_MAX (Wind Vane Max Voltage)

DEFAULT: 3.3	RANGE: 0 5.0	UNITS: V

Wind Vane Max Voltage (WNDVN_DIR_V_MAX)

Description

`WNDVN_DIR_V_MAX` calibrates the "End" of the rotation for an **Analog** wind vane.

It defines the voltage that the autopilot should interpret as a full 360-degree rotation. Together with `V_MIN`, it establishes the scaling for the entire rotation circle.

Tuning & Behavior

- **Default:** 3.3V.
- **Calibration:** Rotate the wind vane manually to its mechanical end point and observe the voltage in your GCS. Enter that value here.
- **Important:** Most analog vanes have a small "Dead Zone" between 359° and 0° where the wiper doesn't touch the resistive track. See WNDVN_DIR_DZ to handle this gap.

WNDVN_DIR_V_MIN (Wind Vane Min Voltage)

DEFAULT: 0	RANGE: 0 5.0	UNITS: V

Wind Vane Min Voltage (WNDVN_DIR_V_MIN)

Description

`WNDVN_DIR_V_MIN` calibrates the "Start" of the rotation for an **Analog** wind vane.

Most analog wind vanes use a potentiometer that maps 0-360 degrees of rotation to a voltage range (e.g. 0.0V to 3.3V). This parameter defines the voltage that the autopilot should interpret as the very beginning of that rotation.

Tuning & Behavior

- **Default:** 0V.
- **Calibration:** Rotate the wind vane manually to its mechanical start point (often marked on the sensor) and observe the voltage in your GCS. Enter that value here.
- **Constraint:** On most modern flight controllers, the maximum readable voltage is 3.3V.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY

WNDVN_SPEED_FILT (Wind Speed Low-Pass Filter)

DEFAULT: 0.5	RANGE: 0 20	UNITS: Hz

Wind Speed Low-Pass Filter (WNDVN_SPEED_FILT)

Description

`WNDVN_SPEED_FILT` smooths out the "gusty" or noisy data from your anemometer (wind speed sensor).

Wind speed is naturally turbulent, especially near the surface. Without a filter, the reported wind speed would jump around rapidly, which can make the sail trimming or mission logic twitchy. This parameter sets the cutoff frequency for a low-pass filter to provide a stable "Apparent Wind" speed.

- **Higher Frequency (e.g. 2.0 Hz):** More responsive to quick wind shifts, but more noise.
- **Lower Frequency (e.g. 0.1 Hz):** Very smooth data, but slow to react to real gusts.

Tuning & Behavior

- **Default:** 0.5 Hz.
- **Recommendation:** Leave at **0.5 Hz**. If your telemetry shows very jumpy wind speeds even in steady air, reduce this value to **0.2 Hz**.

WNDVN_SPEED_MIN (Wind Vane Cut-off Speed)

DEFAULT: 0	RANGE: 0 5	UNITS: m/s

Wind Vane Cut-off Speed (WNDVN_SPEED_MIN)

Description

`WNDVN_SPEED_MIN` prevents the autopilot from reacting to "Flutter" or "Dithering" when there is no wind.

Physical wind vanes require a certain amount of airflow to overcome friction and point accurately. In very light winds, a vane might just flop around or stay stuck in an old position. This parameter tells the autopilot to ignore the wind direction data unless the measured wind speed is above this threshold.

Tuning & Behavior

- **Default:** 0.
- **Recommendation:** Set to **0.5 m/s** to **1.0 m/s** depending on how free-moving your physical wind vane is.
- **Dependencies:** Requires a functional wind speed sensor (WNDVN_SPEED_TYPE).

WNDVN_SPEED_OFS (Wind Speed Voltage Offset)

DEFAULT: 0	RANGE: 0 3.3	UNITS: V

Wind Speed Voltage Offset (WNDVN_SPEED_OFS)

Description

`WNDVN_SPEED_OFS` calibrates the "Zero Point" for **Analog** wind speed sensors.

Many analog anemometers (like the Modern Devices sensor) output a small baseline voltage even when there is no wind. This parameter tells the autopilot to treat that specific voltage as "0.0 m/s," ensuring that subsequent speed readings are accurate.

Tuning & Behavior

- **Default:** 0.
- **Calibration:** Power on the vehicle in a room with perfectly still air. Observe the raw voltage from the sensor in the GCS status tab (or via logs) and enter that value here.
- **Effect:** If this value is too high, the drone will under-report wind speed. If too low, it will report a phantom wind even when stationary.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





WNDVN_SPEED_PIN (Wind Speed Pin)

DEFAULT: -1	RANGE: -1 103

Wind Speed Pin (WNDVN_SPEED_PIN)

Description

`WNDVN_SPEED_PIN` sets the physical analog pin on the flight controller where the wind speed sensor's signal is connected.

This parameter is only used if `WNDVN_SPEED_TYPE` is set to 2 (Modern Devices Analog).

- **-1: Not used.**
- **0-9: Analog Pin Number.** (e.g., 0 for A0).
- **50-55: AUX Out Pins.** (When configured as analog inputs).
- **103: RSSI/SBUS pin.**

Tuning & Behavior

- **Default:** -1.
- **Configuration:** You must also set WNDVN_SPEED_OFS to calibrate the voltage at zero wind speed.

WNDVN_SPEED_TYPE (Wind Speed Sensor Type)

DEFAULT: 0	RANGE: 0 11

Wind Speed Sensor Type (WNDVN_SPEED_TYPE)

Description

`WNDVN_SPEED_TYPE` enables the measurement of wind speed (Anemometer).

This is critical for sailboats to calculate "True Wind" and for any vehicle that needs to record local weather conditions. If enabled, the autopilot will use this data to improve navigation and sail control.

- **0: None.** No wind speed sensor.
- **1: Airspeed Library.** Uses the drone's existing pitot tube.
- **2: Modern Devices.** (Analog).
- **3: RPM Library.** Calculates wind speed based on the rotation rate of a cup-style anemometer.
- **4: NMEA.**

Tuning & Behavior

- **Reboot Required:** Yes.
- **Integration:** If using an analog sensor, you must also set WNDVN_SPEED_PIN.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





WNDVN_TEMP_PIN (Wind vane speed sensor analog temp pin)

DEFAULT: -1	RANGE: -1 100

Wind vane speed sensor analog temp pin (WNDVN_TEMP_PIN)

Description

This parameter specifies the microcontroller pin connected to the temperature output of the wind speed sensor (specifically designed for the Modern Devices Wind Sensor Rev. P). This analog temperature reading is critical for compensating the hot-wire anemometer data, ensuring accurate wind speed measurements across varying ambient temperatures.

The Mathematics

$$V_\text{wind_comp} = f(V_\text{raw}, V_\text{temp})$$

The Engineer's View

Defined in `libraries/AP_WindVane/AP_WindVane.cpp`.

- **-1:** Disabled.
- **Pin Number:** The ADC pin ID (e.g., 14 or 15).

Tuning & Behavior

- **Default Value:** -1 (Disabled)
- **Usage:** Only relevant for analog hot-wire sensors. Mechanical cups/vanes do not use this.

WNDVN_TRUE_FILT (True Wind Data Low-Pass Filter)

DEFAULT: 0.05	RANGE: 0 10	UNITS: Hz

True Wind Data Low-Pass Filter (WNDVN_TRUE_FILT)

Description

`WNDVN_TRUE_FILT` smooths the "Global" wind estimate.

Unlike apparent wind (what the drone feels while moving), **True Wind** is the actual wind speed and direction relative to the ground. It is calculated by combining apparent wind data with the drone's own velocity and heading. This calculation is prone to errors during turns. This very slow filter (default 0.05 Hz) ensures that the drone builds a stable long-term estimate of the real wind conditions.

Tuning & Behavior

- **Default:** 0.05 Hz (Very Slow).
- **Significance:** It takes about 20 seconds for a change in the real wind to be fully reflected in the "True Wind" estimate. This prevent's the drone's own movements from "polluting" the wind estimate.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





WNDVN_TYPE (Wind Vane Type)

DEFAULT: 0	RANGE: 0 11

Wind Vane Type (WNDVN_TYPE)

Description

`WNDVN_TYPE` enables support for measuring the wind direction relative to the vehicle. This is primarily used by Sailboats to trim their sails and by some autonomous vehicles for advanced weather logging or drift compensation.

- **0: None.** Wind vane support is disabled.
- **1: Heading when armed.** Assumes the wind is coming from the direction the vehicle is pointing when it arms.
- **3: Analog.** For physical wind vanes that output a voltage proportional to the angle.
- **4: NMEA.** For digital wind sensors (Anemometers) that output MAVLink or NMEA data strings.
- **10-11: SITL.** For simulation testing.

Tuning & Behavior

- **Reboot Required:** Yes.
- **Setup:** Once enabled, you must configure the input pin (WNDVN_DIR_PIN) if using an Analog type.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY

Overview

The **WP** parameter group configures the high-level **Waypoint Navigation** logic.

Key Concepts

1. Radius (`WP_RADIUS`)

Defines how close the vehicle must get to a waypoint before it is considered "Reached" and the vehicle moves to the next one.

- **Too Small:** Vehicle may circle the waypoint indefinitely if it can't quite hit the target (e.g., in high wind).
- **Too Large:** Vehicle will cut corners aggressively.

2. Facing Behavior (`WP_YAW_BEHAVIOR`)

Defines where the drone points its nose during autonomous flight.

- **0:** None.
- **1:** Face next waypoint.
- **2:** Face direction of flight.

Developer Notes

- **Library:** `libraries/AP_Mission`.
- **Related:** See WPNAV for multicopter-specific path tuning.

WP_ACCEL (Waypoint Acceleration)

DEFAULT: 0	RANGE: 0 100	UNITS: m/s^2

Waypoint Acceleration (WP_ACCEL)

Description

`WP_ACCEL` determines how aggressively the vehicle speeds up and slows down during an autonomous mission.

- **Higher Value:** Faster acceleration and more "snappy" starts/stops.
- **Lower Value:** Smoother, more gradual transitions.

Tuning & Behavior

- **Default:** 0.
- **Fallback:** If set to 0, the autopilot uses the value from the attitude controller (`ATC_ACCEL_MAX`).
- **Recommendation:** Use **2.0** to **5.0** for a smooth but responsive Rover.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY

WP_JERK (Waypoint Navigation Jerk Limit)

DEFAULT: 0	RANGE: 0 100	UNITS: m/s^3

Waypoint Navigation Jerk Limit (WP_JERK)

Description

`WP_JERK` defines the "Smoothness" of your drone's speed changes during autonomous missions.

ArduPilot uses **S-curve** path planning to move between waypoints. Unlike older "step" changes in speed, S-curves ramp acceleration up and down gradually. This parameter sets the limit for that ramp (the derivative of acceleration).

- **Higher Value:** Faster, snappier acceleration and braking.
- **Lower Value:** More "graceful," cinematographic movements.
- **0:** (Default). The autopilot sets the jerk limit to match the maximum acceleration (WP_ACCEL).

Tuning & Behavior

- **Recommendation:** Leave at **0** for most applications.
- **Usage:** For heavy lifting or filming drones, you can set this to a lower value (e.g. 1.0 or 2.0) to prevent sudden jerks that could cause gimbal vibration or payload swinging.

WP_MAX_RADIUS (Waypoint Maximum Radius)

DEFAULT: 0	RANGE: 0 1000	UNITS: m

Waypoint Maximum Radius (WP_MAX_RADIUS)

Description

Safety limit to prevent the aircraft from indefinitely circling a waypoint it cannot reach.

Tuning & Behavior

- **Default Value:** 0 m (Disabled)
- **Range:** 0 1000 m

WP_NAVALT_MIN (Minimum navigation altitude)

DEFAULT: 0	RANGE: 0 5	UNITS: m

Minimum navigation altitude (WP_NAVALT_MIN)

Description

Minimum altitude for navigation to begin.

Tuning & Behavior

- **Default Value:** 0
- **Range:** 0 5

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





WP_PIVOT_ANGLE (Pivot Turn Threshold Angle)

DEFAULT: 60	RANGE: 0 360	UNITS: deg

Pivot Turn Threshold Angle (WP_PIVOT_ANGLE)

Description

`WP_PIVOT_ANGLE` determines when a Rover or Boat should stop and spin in place rather than trying to make a wide arc turn.

This is only used for **Skid-Steer** (tank-like) vehicles. If the turn required to reach the next waypoint is sharper than this angle, the vehicle will come to a complete halt, spin until its nose points at the target, and then resume driving forward.

- **0:** Disabled. The vehicle always attempts a wide steering turn.
- **60 (Default):** Standard for most rovers.

Tuning & Behavior

- **Recommendation:** Set between **45 and 90** degrees.
- **Significance:** Prevents the vehicle from making massive, uncontrolled "S-turns" when trying to reach a waypoint that is behind it or at a sharp angle.

WP_PIVOT_DELAY (Pivot Turn Pause Delay)

DEFAULT: 0	RANGE: 0 60	UNITS: s

Pivot Turn Pause Delay (WP_PIVOT_DELAY)

Description

`WP_PIVOT_DELAY` introduces a brief "Rest" after a Rover or Boat spins on the spot.

Pivot turns can be aggressive and cause the vehicle to shake or the GPS heading to become slightly noisy. This delay allows the mechanical momentum to dissipate and the navigation filters to settle before the vehicle applies forward power.

Tuning & Behavior

- **Default:** 0 seconds (No delay).
- **Recommendation:** Use **0.5 or 1.0 seconds** if your rover tends to "wobble" or depart from the track immediately after finishing a pivot turn.

WP_PIVOT_RATE (Pivot Turn Max Rate)

DEFAULT: 60	RANGE: 1 360	UNITS: deg/s

Pivot Turn Max Rate (WP_PIVOT_RATE)

Description

`WP_PIVOT_RATE` sets the "Spin Speed" for on-the-spot turns.

This is the target angular velocity the rover attempts to maintain while it is performing a pivot turn (stopped in one place).

Tuning & Behavior

- **Default:** 60 deg/s.
- **Recommendation:** If your rover spins too violently and slips its treads/wheels, decrease this to **30 or 40**. If the turn feels sluggish, increase it.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





WP_RADIUS (Waypoint Radius)

DEFAULT: 90	RANGE: 1 12700	UNITS: m

Waypoint Radius (WP_RADIUS)

Description

Defines the distance from a waypoint at which the aircraft considers the waypoint reached and transitions to the next leg of the mission.

Tuning & Behavior

- **Default Value:** 90 m (Plane), 200 cm (Copter - internal WPNAV_RADIUS)
- **Range:** 1 to 12700 m
- **A larger WP_RADIUS usually leads to smoother turns and better tracking in windy conditions.**

WP_SPEED (Waypoint Speed (Default))

DEFAULT: 2.0	RANGE: 0 100	UNITS: m/s

Waypoint Speed (WP_SPEED)

Description

`WP_SPEED` sets the "Cruise Control" for your vehicle during an autonomous mission.

It defines how fast the drone or rover attempts to travel between waypoints. Individual waypoints can have their own speed settings in the mission file, but `WP_SPEED` is the value used if no specific speed is provided.

- **Standard Setting:** 2.0 m/s to 5.0 m/s for Rovers. 5.0 m/s to 15.0 m/s for Multicopters.

Tuning & Behavior

- **Default:** 2.0 m/s.
- **Recommendation:** Set this to a speed that allows your vehicle to maneuver comfortably. If the vehicle "overshoots" waypoints or oscillates at high speed, reduce this value or increase WP_RADIUS.
- **Dynamic Adjustment:** You can change this speed in real-time during a mission using a MAVLink "Change Speed" command from the GCS.

WP_YAW_BEHAVIOR (Waypoint Yaw Behavior)

DEFAULT: 1	RANGE: 0 3

Waypoint Yaw Behavior (WP_YAW_BEHAVIOR)

Description

`WP_YAW_BEHAVIOR` tells the drone where to "Look" while it's flying a mission.

- **0: Never Change.** The drone maintains whatever heading it had when the mission started.
- **1: Face Next Waypoint (Default).** The drone always points its nose toward the destination waypoint. (Most natural for FPV and forward-flight performance).
- **2: Face Next Waypoint except RTL.** Same as 1, but when returning home, the drone maintains its current heading.
- **3: Face Along GPS Track.** The drone points in the direction it is physically moving.

Tuning & Behavior

- **Default:** 1.
- **Recommendation:** Use **1 (Face Next Waypoint)** for most missions. If you are performing a mapping mission and want the camera to stay oriented in one direction regardless of the flight path, use **0 (Never Change)**.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





WAYPOINT NAVIGATION CONFIGURATION (WPNAV)

Overview

The **WPNAV** parameter group provides the low-level **Path Planning** and **Speed** settings for **ArduCopter** autonomous flight (Auto, Guided, RTL).

While the `PSC` group handles PID control, the `WPNAV` group defines the "Target Velocity" and "Acceleration" profiles that the drone follows between points.

Key Concepts

1. Speed Control (`WPNAV_SPEED` / `DN` / `UP`)

- `WPNAV_SPEED`: Maximum horizontal cruising speed (cm/s).
- `WPNAV_SPEED_UP`: Maximum climb rate (cm/s).
- `WPNAV_SPEED_DN`: Maximum descent rate (cm/s).

2. Cornering (`WPNAV_RADIUS`)

Similar to `WP_RADIUS`, this defines the acceptance circle for waypoints.

- **Spline Turns:** In newer versions, the drone uses "Spline" paths to fly through waypoints smoothly without stopping.

3. Acceleration (`WPNAV_ACCEL` / `Z`)

Defines how fast the drone can speed up or slow down.

- **Smoothness:** Lower values make for a more graceful, cinematic flight.

Integration Guide

- **Efficiency:** For long-range mapping, increase `WPNAV_SPEED` to the airframe's most efficient cruise speed.
- **Safety:** Ensure `WPNAV_SPEED_DN` is not set higher than the drone's stable descent capability (to prevent "Wobble of Death").

Developer Notes

- **Library:** `libraries/AC_WPNav`.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





WPNAV_ACCEL (Waypoint Horizontal Acceleration)

DEFAULT: 100	RANGE: 50 500	UNITS: cm/s/s

Waypoint Horizontal Acceleration (WPNAV_ACCEL)

Description

`WPNAV_ACCEL` is the "Gas Pedal" limit for your drone's autonomous brain. It determines how hard the drone should lean forward to reach its cruising speed (`WPNAV_SPEED`) and how hard it should lean back to stop at a waypoint.

- **Low Value (e.g. 50):** The drone accelerates and brakes very gently. Flight feels very smooth and "lazy." Ideal for cinematography.
- **High Value (e.g. 200):** The drone snaps to its target speed instantly. Flight feels very robotic and aggressive.
- **Default (100):** 1.0 m/s². A standard, safe value that is comfortable for most payloads.

Physical Context: 100 cm/s² is roughly **0.1G**. Setting this to 500 would be 0.5G, which requires a steep lean angle and high battery current.

The Mathematics

This parameter defines the acceleration limit (a_max) for the 2D path shaper. When starting a mission segment:

$$V(t) = V_initial + \text{constrain}(a_calc, -a_max, a_max) \cdot dt$$

Where:

- a_max is `WPNAV_ACCEL`.

The **Total 2D Acceleration** is limited, meaning the combination of North and East acceleration will never exceed this value, resulting in smooth, circular turn profiles rather than "square" corners.

The Engineer's View

This parameter is the `_wp_accel_cmss` member in `AC_WPNav`.

It is a core input to the **S-Curve Kinematic path generator**. One subtle detail is that **WPNAV_ACCEL also determines the Braking Distance**.

$$\text{Stop_Dist} = \frac{V^2}{2 \cdot a_max}$$

If you set `WPNAV_SPEED` very high but `WPNAV_ACCEL` very low, the drone will have to start braking 50 meters before the waypoint, which can make the mission feel very slow.

Tuning & Behavior

- **Default Value:** 100 cm/s² (1.0 m/s²)
- **Range:** 50 - 500 cm/s²
- **Effect of Increasing:** Faster stop-and-go maneuvers. Drone reaches waypoint speed sooner. Better for high-wind handling in missions.
- **Effect of Decreasing:** Smoother cinematic motion. Less "jerking" of the gimbal. Safer for unstable payloads.

Use Case Recommendations

- **Cinematography: Decrease to 50.** Provides extremely smooth transitions between waypoints.
- **Rapid Delivery / Rescue: Increase to 150 - 200.** Every second counts; you want the drone at its cruise speed as fast as possible.
- **Mapping: Keep Default.** Standard values are sufficient for consistent overlap.

Troubleshooting

- **Scenario:** Drone "lunges" forward violently at the start of a mission segment.
  - **Diagnosis:** `WPNAV_ACCEL` is too high for the vehicle's power-to-weight ratio.
  - **Fix:** Reduce to 80.
- **Scenario:** Drone consistently overshoots the waypoint during a mission.
  - **Diagnosis:** `WPNAV_ACCEL` is too low, and the drone doesn't have enough "braking power" to stop from its current speed.
  - **Fix:** Increase `WPNAV_ACCEL` or reduce `WPNAV_SPEED`.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY

WPNAV_ACCEL_C (Waypoint Cornering Acceleration)

DEFAULT: 0	RANGE: 0 500	UNITS: cm/s/s

Waypoint Cornering Acceleration (WPNAV_ACCEL_C)

Description

`WPNAV_ACCEL_C` controls how "hard" the drone turns at waypoints during an autonomous mission.

It defines the maximum centripetal acceleration allowed during a cornering maneuver. This parameter determines whether the drone will slow down significantly to hit a sharp point, or maintain speed by flying a wider, more graceful arc.

- **0 (Default):** Uses **2x** the value of `WPNAV_ACCEL`. This is usually the best setting for a smooth mission.
- **High Value:** The drone will maintain high speed through corners, pulling high Gs. This looks aggressive and "jerky" but completes missions faster.
- **Low Value:** The drone will slow down significantly as it approaches waypoints to avoid high lateral forces.

The Mathematics

The maximum cornering speed v_max is calculated based on the radius of the turn r and this acceleration limit a_c:

$$v_max = \sqrt{a_c \cdot r}$$

Where a_c is `WPNAV_ACCEL_C`.

If the drone is flying at `WPNAV_SPEED` and encounters a turn where the radius is too tight for the current a_c, it must decelerate before entering the turn.

The Engineer's View

Used in `AC_WPNav::advance_wp_target_along_track()` and passed to the S-Curve trajectory generator.
The S-Curve logic uses this to calculate the "cornering arc" that connects two mission legs. If `WPNAV_ACCEL_C` is very high, the drone will start the turn much closer to the physical waypoint.

Tuning & Behavior

- **Default Value:** 0 (Defaults to $2 \times \text{WPNAV_ACCEL}$)
- **Recommendation:** Leave at **0** for general use.
- **Racing/Speed Missions:** Increase to **400 or 500** to maintain maximum momentum. Ensure your `ANGLE_MAX` and motor thrust can handle the resulting lean angles.
- **Cinematography:** Decrease to **50 or 100** (or keep default 0) to ensure the camera gimbal isn't subjected to high lateral G-forces during turns.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





WPNAV_ACCEL_Z (Waypoint Vertical Acceleration)

DEFAULT: 100	RANGE: 50 500	UNITS: cm/s/s

Waypoint Vertical Acceleration (WPNAV_ACCEL_Z)

Description

`WPNAV_ACCEL_Z` is the "Vertical Gas Pedal" limit for autonomous flight. It determines how aggressively the drone initiates a climb or starts braking for a descent during a mission.

- **Low Value (e.g. 50):** The drone starts and stops its climbs very gently. Ideal for delicate payloads or cinematography.
- **High Value (e.g. 200):** The drone "punches" its way into climbs. Flight feels more robotic and rapid.
- **Default (100):** 1.0 m/s². A safe, balanced setting for most multirotors.

Constraint: This value should generally be set **equal to or lower than** `PSC_ACCZ_P` authority. If you set `WPNAV_ACCEL_Z` higher than what the drone can physically achieve, you will get "Altitude Slop" where the drone falls behind its mission path.

The Mathematics

This parameter sets the acceleration limit (a_z_max) for the vertical path shaper:

$$V_z(t) = V_z_initial + \text{constrain}(a_z_calc, -a_z_max, a_z_max) \cdot dt$$

Where:

- a_z_max is `WPNAV_ACCEL_Z`.

The vertical shaper uses this to ensure that the transition between different vertical speeds follows a smooth ramp, preventing the "instant torque" shocks that can damage motor mounts.

The Engineer's View

This parameter is the `_wp_accel_z_cmss` member in `AC_WPNav`.

One important engineering detail: **This limit is combined with the Jerk limit (`WPNAV_JERK`)**. The shaper will prioritize the jerk limit during the start of a move, and only reach full `WPNAV_ACCEL_Z` if the move is long enough.

```
// AC_WPNav.cpp
// Vertical shaper logic uses _wp_accel_z_cmss to calculate the velocity profile.
```

Tuning & Behavior

- **Default Value:** 100 cm/s² (1.0 m/s²)
- **Range:** 50 - 500 cm/s²
- **Effect of Increasing:** Snappier altitude corrections. Drone handles vertical "staircase" missions more efficiently.
- **Effect of Decreasing:** Smoother, more flowing altitude changes. Greatly reduces the "bobbing" often seen in camera gimbal footage.

Use Case Recommendations

- **Cinematography: Decrease to 50 - 80.** Prevents the "jerk" seen on the horizon when the drone finishes a climb.
- **Rapid Mapping (Terrain Following): Increase to 150.** When flying close to hills, the drone needs to accelerate vertically fast enough to follow the ground without crashing.
- **Heavy Lift Cargo: Decrease to 50.** Minimizes the vertical G-load on the airframe during transitions.

Troubleshooting

- **Scenario:** Drone makes a loud "thump" sound from the motors at the very top of a climb.
  - **Diagnosis:** `WPNAV_ACCEL_Z` is too high, causing a sudden motor current drop that triggers mechanical ringing.
  - **Fix:** Reduce `WPNAV_ACCEL_Z` to 80.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





WPNAV_JERK (Waypoint Jerk Limit)

DEFAULT: 1.0	RANGE: 1 20	UNITS: m/s^3

Waypoint Jerk Limit (WPNAV_JERK)

Description

`WPNAV_JERK` is the "refinement" setting for mission flight. While `WPNAV_ACCEL` tells the drone how hard to accelerate, `WPNAV_JERK` tells it how **smoothly** to apply that acceleration.

In physics, **Jerk** is the rate at which acceleration changes. High jerk means the drone snaps into a lean instantly; low jerk means it rolls into the lean gracefully. By setting this limit, ArduPilot ensures that autonomous missions (Auto, RTL) look professional and don't place unnecessary mechanical stress on the frame.

- **Low Value (e.g. 1.0):** Flight is extremely fluid. The drone takes a noticeable moment to "roll into" its forward flight. Best for high-end cinematography.
- **High Value (e.g. 10.0):** The drone snaps into its acceleration ramps instantly. Flight looks more robotic and "aggressive."
- **Default (1.0):** Optimized for standard, smooth multirotor flight.

The Mathematics

Jerk (j) is the third derivative of position (p):

$$j(t) = \frac{da}{dt} = \frac{d^3p}{dt^3}$$

In the ArduPilot shaper, this parameter limits the curvature of the velocity graph. A mission segment's acceleration (a) must satisfy:

$$|\Delta a| \leq \text{WPNAV_JERK} \cdot dt$$

This creates the characteristic **S-Curve velocity profile**, where the transition from hover to cruising speed is rounded off at both the beginning and the end.

The Engineer's View

This parameter is the `_wp_jerk` member in `AC_WPNav`.

It defines the "snappiness" of the **SCurve path shaper**. One interesting technical detail is that `WPNAV_JERK` is expressed in **m/s³**, whereas `WPNAV_ACCEL` is in **cm/s²**.

- A value of 1.0 m/s³ means the drone can increase its acceleration by 100 cm/s² every second.
- With the default `WPNAV_ACCEL` of 100, it would take exactly 1.0 second to reach full acceleration.

Tuning & Behavior

- **Default Value:** 1.0 m/s³
- **Range:** 1.0 - 20.0 m/s³
- **Effect of Increasing:** The drone reacts faster to the mission path. Corners are tighter (less drift).
- **Effect of Decreasing:** Maximum smoothness. Essential for capturing professional aerial video without any "horizon jitter."

Use Case Recommendations

- **Cinematography: Keep at 1.0.** This is the professional standard.
- **High-Speed Grid Mapping: Increase to 2.0 - 5.0.** When flying a tight survey grid at 15 m/s, a higher jerk value is needed to ensure the drone can stop and turn fast enough at the end of each line.
- **Small Rigid Racers (Auto-Mission): Set to 10.0.** Smaller frames have very little inertia and can handle high jerk rates with ease.

Troubleshooting

- **Scenario:** Drone "bobs" or twitches when starting or ending a mission leg.
  - **Diagnosis:** `WPNAV_JERK` is too high, exciting a frame resonance.
  - **Fix:** Reduce to 1.0 or 0.5.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





WPNAV_RADIUS (Waypoint Hit Radius)

DEFAULT: 200	RANGE: 10 1000	UNITS: cm

Waypoint Hit Radius (WPNAV_RADIUS)

Description

`WPNAV_RADIUS` defines the "bullseye" size for your autonomous missions. It tells the drone exactly how close it needs to get to a waypoint before it can move on to the next one.

- **Small Radius (e.g. 50cm):** The drone will be extremely precise. It will slow down almost to a hover to ensure it crosses the exact coordinate. This makes the mission very accurate but "jerky" as the drone stops at every point.
- **Large Radius (e.g. 500cm):** The drone will "fly past" the waypoints in a smooth, flowing curve. It doesn't need to reach the exact center, so it can maintain its speed. This is known as **Corner Cutting**.

Recommendation: For standard mapping, a radius of **200 (2 meters)** is ideal. For precision delivery or drop-off, reduce to **50 - 100**.

The Mathematics

The path planner constantly monitors the 2D distance (d) between the current EKF position and the waypoint coordinate:

$$d = \sqrt{(X_target - X_actual)^2 + (Y_target - Y_actual)^2}$$

A waypoint transition is triggered when:

$$d < \text{WPNAV_RADIUS}$$

Spline Interaction: If you are using "Spline" waypoints, this parameter is less critical because the drone follows a pre-calculated curve. However, for standard "Waypoints," this parameter determines the smoothness of the turns.

The Engineer's View

This parameter is the `_wp_radius_cm` member in `AC_WPNav`.

It is used in `AC_WPNav::check_wp_hit()`. A critical engineering detail is that **higher speeds require larger radii**. If the drone is flying at 15 m/s, it may physically be unable to turn tight enough to stay within a 2-meter radius. In this case, the drone will circle the waypoint (Toilet Bowling) trying to get inside the radius.

```
// AC_WPNav.cpp
// Logic for waypoint transition based on _wp_radius_cm.
```

Tuning & Behavior

- **Default Value:** 200 cm (2 meters)
- **Range:** 10 - 1000 cm
- **Effect of Increasing:** Smoother mission flight. Higher average speed. Drone "rounds off" the corners of the path.
- **Effect of Decreasing:** More precise path tracking. Drone "stops and turns" at each waypoint.

Use Case Recommendations

- **Aerial Mapping (Photogrammetry): Set to 200 - 300.** You want smooth flight to avoid motion blur in the photos, and a 2-3 meter error is usually acceptable for the photo center.
- **Precision Spraying (Agriculture): Decrease to 100.** You need to ensure the chemicals are applied exactly to the edge of the field.
- **Search and Rescue: Set to 500.** Speed and coverage area are more important than hitting the exact center of a search grid.

Troubleshooting

- **Scenario:** Drone reaches a waypoint but then starts doing wide circles around it instead of going to the next one.
  - **Diagnosis:** `WPNAV_RADIUS` is too small for the current `WPNAV_SPEED`. The drone's turn radius is larger than the hit radius.
  - **Fix:** Increase `WPNAV_RADIUS` or reduce `WPNAV_SPEED`.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





WPNAV_RFND_USE (Waypoint Mission Rangefinder Usage)

DEFAULT: 1	RANGE: 0 1

Waypoint Mission Rangefinder Usage (WPNAV_RFND_USE)

Description

`WPNAV_RFND_USE` is the master switch for **Terrain Following** in autonomous missions. It determines whether the drone's height is measured from where it started (Barometer) or from the surface directly beneath it (Rangefinder).

- **Enabled (1):** The drone uses its rangefinder (Lidar/Sonar) to maintain its height above the ground. If the drone flies over a hill, it will climb to maintain the requested altitude relative to the slope.
- **Disabled (0):** The drone uses the Barometer or GPS to maintain a fixed altitude relative to its takeoff point ("Home"). If it flies over a hill, it will stay at the same absolute height, potentially crashing into the terrain.

Pre-requisite: You must have a functioning rangefinder configured (`RNGFND1_TYPE` > 0) for this parameter to have any effect.

The Mathematics

This parameter toggles the **Altitude Reference** in the vertical control loop:

$$Z_target = \begin{cases} \text{Ground_Level} + \text{Alt_WP}, & \text{if RFND_USE} = 1 \\ \text{Home_Level} + \text{Alt_WP}, & \text{if RFND_USE} = 0 \end{cases}$$

Where `Alt_WP` is the altitude specified in the mission file (MAVLink `MISSION_ITEM`).

The Engineer's View

This parameter is the `_rangefinder_use` member in `AC_WPNav` .

It is checked in `AC_WPNav::update_z_controller()` . When enabled, the code continuously queries the rangefinder for the current distance to ground. One critical safety feature is that **ArduPilot will ignore this parameter if the rangefinder health is low**. If the Lidar loses its signal (e.g., flying over water), the system will automatically fall back to Barometric altitude to prevent a crash.

Tuning & Behavior

- **Default Value:** 1 (Enabled)
- **Range:** 0 (Off) or 1 (On)
- **Effect of Enabling:** Drone follows the contours of the ground. Essential for low-altitude mapping and agriculture.
- **Effect of Disabling:** Drone stays at a constant barometric height. Safer for high-altitude flight where ground texture may be unreliable.

Use Case Recommendations

- **Agricultural Spraying: Always Enable (1).** You must maintain a precise distance from the crops regardless of the field's slope.
- **Power Line Inspection: Always Enable (1).** Ensures the drone tracks the vertical profile of the terrain accurately.
- **High-Altitude Photography (>50m): Disable (0).** Most hobbyist rangefinders are unreliable above 30-40 meters. Using the Barometer is safer at these heights.

Troubleshooting

- **Scenario:** Drone "bobs" up and down violently during a mission over tall grass.
  - **Diagnosis:** The rangefinder is getting inconsistent readings from the grass.
  - **Fix:** Disable `WPNAV_RFND_USE` or increase `EK3_HGT_DELAY` .

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY

WPNAV_SPEED (Waypoint Horizontal Speed)

DEFAULT: 500	RANGE: 0 2000	UNITS: cm/s

Waypoint Horizontal Speed (WPNAV_SPEED)

Description

`WPNAV_SPEED` sets the "Cruise Speed" for your autonomous missions.

- **500 (Default):** 5 meters per second (11 mph). Good for stable surveying.
- **1000:** 10 m/s (22 mph). Faster transit.

Tuning & Behavior

- **Default Value:** 500 cm/s.
- **Recommendation:** Set based on your drone's power and the requirements of your mission.

WPNAV_SPEED_DN (Waypoint Vertical Speed Down)

DEFAULT: 150	RANGE: 0 500	UNITS: cm/s

Waypoint Vertical Speed Down (WPNAV_SPEED_DN)

Description

`WPNAV_SPEED_DN` controls how fast the drone drops when moving between waypoints.

- **150 (Default):** 1.5 meters per second.

Tuning & Behavior

- **Default Value:** 150 cm/s.
- **Recommendation:** Keep lower than `WPNAV_SPEED_UP` to prevent the drone from falling into its own propwash (vortex ring state).

WPNAV_SPEED_UP (Waypoint Vertical Speed Up)

DEFAULT: 250	RANGE: 0 1000	UNITS: cm/s

Waypoint Vertical Speed Up (WPNAV_SPEED_UP)

Description

`WPNAV_SPEED_UP` controls how fast the drone climbs when moving between waypoints of different altitudes.

- **250 (Default):** 2.5 meters per second.

Tuning & Behavior

- **Default Value:** 250 cm/s.
- **Recommendation:** Keep at or below **250** to avoid overwhelming the vertical controller.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY

WPNAV_TER_MARGIN (Terrain Following Margin)

DEFAULT: 10.0	RANGE: 1 100	UNITS: m

Terrain Following Margin (WPNAV_TER_MARGIN)

Description

`WPNAV_TER_MARGIN` is the "safety cushion" for drones using a digital map to stay away from the ground.

ArduPilot can maintain its altitude by downloading a **Terrain Database** (SRTM data) which contains the elevation of the entire earth. However, this data is often low-resolution (typically 30-90 meters per pixel) and does not include objects like tall trees, power lines, or new buildings.

`WPNAV_TER_MARGIN` adds a fixed vertical offset to your mission altitude to ensure you clear these unknown obstacles.

- **Low Value (e.g. 2m):** Extremely risky. The drone will follow the ground map precisely, but even a small tree or a mistake in the map data will cause a crash.
- **High Value (e.g. 20m):** Very safe. The drone stays well above any potential unmapped obstacles.
- **Default (10m):** A robust setting for most cross-country autonomous missions.

The Mathematics

The target altitude (Z_target) is calculated by summing the Terrain Elevation (E), the Mission Altitude (A), and the Margin (M):

$$Z_target = E_database + A_mission + \text{WPNAV_TER_MARGIN}$$

Example: If you set a mission altitude of 50 meters, and the map says the hill is 100 meters high, the drone will fly at an absolute height of 160 meters ($100 + 50 + 10$).

The Engineer's View

This parameter is the `_terrain_margin` member in `AC_WPNav`.

It is applied in the altitude calculation logic. One critical technical detail is that **Terrain Margin is only used when the Rangefinder is NOT available or is out of range**. If you have a Lidar sensor and it sees the ground, ArduPilot will prioritize the real sensor data and ignore the `WPNAV_TER_MARGIN`.

Tuning & Behavior

- **Default Value:** 10.0 m
- **Range:** 1.0 - 100.0 m
- **Effect of Increasing:** Safer autonomous missions. Drone stays higher above potential obstacles.
- **Effect of Decreasing:** Drone flies closer to the terrain. Better for low-level tactical or inspection flight.

Use Case Recommendations

- **Long-Distance BVLOS: Increase to 20m.** You cannot see the drone, so you must assume the map data might be old or inaccurate.
- **Copter Mapping (Flat Area): Keep Default (10m).**
- **Precision Low-Level Flight: Decrease to 5m.** Only do this if you have visually confirmed there are no unmapped obstacles in the path.

Troubleshooting

- **Scenario:** Drone "climbs" unexpectedly during a mission over a flat desert.
  - **Diagnosis:** The terrain database thinks the ground is higher than it actually is.
  - **Fix:** Reduce `WPNAV_TER_MARGIN` or switch to a high-accuracy custom terrain map.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





WHEEL RATE CONTROL CONFIGURATION (WRC)

Overview

The **WRC** parameter group configures the **Closed-Loop Wheel Speed** controller for **ArduRover**.

Without WRC, a rover uses "Open-Loop" control, sending a fixed throttle percentage. With WRC, the autopilot uses feedback from **Wheel Encoders** to ensure the wheels are spinning at exactly the commanded speed, regardless of surface friction, slope, or battery voltage.

Key Concepts

1. Velocity Feedback

The system calculates the error between the "Commanded Wheel Speed" and the "Actual Wheel Speed" (measured by encoders).

- `WRC_ENABLE`: Master switch.

2. PID Tuning

(If present in group) Standard gains to control the responsiveness of the wheel motor response.

Integration Guide

1. **Hardware:** Install and calibrate wheel encoders (`WENC` group).
2. **Enable:** Set `WRC_ENABLE = 1`.
3. **Tuning:** Adjust gains to ensure the rover can maintain a crawl up a hill without stalling or jerking.

Developer Notes

- **Library:** `libraries/AP_WheelEncoder/AP_WheelRateControl.cpp`.
- **Dependency:** This group is functionally useless without the `WENC` group being configured first.

WRC_ENABLE (Wheel Rate Control Enable)

DEFAULT: 0	RANGE: 0 1

Wheel Rate Control Enable (WRC_ENABLE)

Description

`WRC_ENABLE` activates the closed-loop speed controller for a Rover's wheels.

Standard Rovers often use "Open-Loop" control – if you command 50% throttle, the autopilot sends 1500 PWM to the ESC. With **Wheel Rate Control**, the autopilot uses sensors (Wheel Encoders) to measure exactly how fast the wheels are turning and adjusts the throttle in real-time to maintain the requested speed, even when driving over uneven terrain or up hills.

Tuning & Behavior

- **Default:** 0 (Disabled).
- **Requirement:** Requires functional wheel encoders (WENC_TYPE).
- **Benefit:** Dramatically improves speed consistency and navigation accuracy for ground vehicles.

WRC_RATE_MAX (Wheel Rate Maximum)

DEFAULT: 0	RANGE: 0 100	UNITS: rad/s

Wheel Rate Maximum (WRC_RATE_MAX)

Description

`WRC_RATE_MAX` sets the physical speed limit for your wheels in the rate controller.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





WEATHER VANE CONFIGURATION (WVANE)

Overview

The **WVANE** parameter group configures the **Weather Vane** (Wind Vane) subsystem for **multicopters** and **QuadPlanes**. This allows the vehicle to autonomously rotate its nose (Yaw) into the wind while hovering or during VTOL transition.

Key Concepts

1. Into-The-Wind Navigation

Multicopters are most stable and efficient when pointing into the wind (reduced drag, better cooling, less motor imbalance).

- `WVANE_ENABLE`: Master switch.
- `WVANE_GAIN`: How aggressively the drone rotates into the wind.

2. Transition Assist

For QuadPlanes, pointing into the wind during a vertical-to-forward transition is critical to prevent the wind from getting under the wings and flipping the aircraft (stalling).

Parameter Breakdown

- `WVANE_ENABLE`:
  - **0:** Disabled.
  - **1:** Always rotate into the wind.
  - **2:** Only in specific modes (e.g., Loiter).
- `WVANE_ANG_MIN`: The minimum wind angle required to start a rotation.

Developer Notes

- **Library:** `libraries/AC_WeatherVane`.
- **Estimation:** Can use a physical wind vane or estimate wind direction using the multicopter's lean angle (drag model).

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





WVANE_ANG_MIN (Weathervaning min angle)

DEFAULT: 1	RANGE: 0 10	UNITS: deg

Weathervaning min angle (WVANE_ANG_MIN)

Description

This parameter sets a "Deadzone" for the weather vane controller. It defines the minimum amount of roll (or pitch) lean the aircraft must experience before the controller begins to apply any yaw correction.

This prevents the aircraft from constantly "hunting" or jittering in response to small turbulence or minor trim offsets. It ensures the controller only reacts to significant wind forces.

The Mathematics

The deadzone is subtracted from the absolute error angle.

$$Error_effective = \max(|Angle_measured| - ANG_MIN, 0)$$

If the measured angle is less than `ANG_MIN`, the error is zero, and no yaw output is generated.

The Engineer's View

In `AC_WeatherVane::get_yaw_out`, the parameter is converted to centi-degrees (`deadzone_cdeg`) and used as a threshold.
It is applied to both Roll and Pitch (if `PITCH_ENABLE` is set) axes.

Tuning & Behavior

- **Default Value:** 1.0 degree
- **Range:** 0 to 10 degrees
- **Effect of Increasing:** Increases the deadzone. The aircraft will tolerate more lean before correcting. Useful if the aircraft is poorly trimmed or if you want to allow some side-slip.
- **Effect of Decreasing:** Makes the controller more sensitive. Setting it to 0 means it will react to even the slightest roll angle, which may cause continuous small yaw corrections.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





WVANE_ENABLE (Weathervaning Enable)

DEFAULT: 0	RANGE: -1 4

Weathervaning Enable (WVANE_ENABLE)

Description

`WVANE_ENABLE` activates a feature that allows the drone to act like a weather vane. When enabled, the drone will automatically rotate (yaw) so that it is always facing into the wind while hovering in a position-controlled mode (like Loiter or QLoiter).

This is extremely beneficial for **QuadPlanes** and **Long-Endurance Multirotors**. By pointing the nose into the wind, the drone minimizes its surface area and drag, which reduces the motor power needed to maintain position, leading to significantly longer flight times and better stability in gusty conditions.

- **0: Disabled.** No automatic rotation.
- **1: Nose into wind.** The drone points its front toward the wind.
- **2: Nose or tail into wind.**
- **3: Side into wind.**
- **4: Tail into wind.**

The Mathematics

The controller monitors the drone's current "Leaning Angle" (Roll/Pitch). Since the drone must lean against the wind to stay stationary, the roll angle ϕ is used as an error signal:

$$\text{YawRate}_{cmd} = \text{WVANE_GAIN} \cdot (\phi - \text{WVANE_ANG_MIN})$$

The drone yaws until the roll angle required to maintain position is minimized (i.e., it is facing directly into the wind).

The Engineer's View

Used in `AC_WeatherVane::get_yaw_out()`.
The logic includes a 2-second stability buffer to prevent the drone from spinning wildly during momentary gusts. Pilot yaw stick input always overrides the weathervaning action.

Tuning & Behavior

- **Default Value:** 0 (Disabled)
- **Recommendation:**
  - **QuadPlanes: Set to 1 (Nose into wind)**. This prevents the large wings from acting as a sail and destabilizing the hover.
  - **Multicopters:** Set to **1** if you frequently fly in windy conditions to improve battery efficiency.
- **Dependencies:** Requires a healthy position estimate (GPS) and `WVANE_GAIN` set correctly (usually 1.0).

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





WVANE_GAIN (Weathervaning gain)

DEFAULT: 0	RANGE: 0.5 4	UNITS: 1/s

Weathervaning gain (WVANE_GAIN)

Description

This parameter controls the "strength" of the weather vaning response. It acts as a multiplier that converts the aircraft's lean angle (caused by wind) into a command to rotate (yaw) into the wind.

A higher gain means the aircraft will turn faster into the wind for a given amount of lean. A lower gain results in a gentler, slower turn.

The Mathematics

The output is calculated by applying a Low Pass Filter (LPF) to the product of the error and the gain.

$$Rate_yaw = LPF(Error_deg \cdot Gain)$$

Where:

- $Error_deg$ is the calculated misalignment (mostly Roll angle, or Pitch angle if options enabled).
- $Gain$ is `WVANE_GAIN`.
- LPF is a smoothing filter: $y_n = 0.98y_n - 1 + 0.02x_n$

For example, if the aircraft is rolled 10 degrees due to wind and the Gain is 2.0, the target yaw rate will settle at 20 degrees per second.

The Engineer's View

The parameter is used in `AC_WeatherVane::get_yaw_out`.
The heavy low-pass filter ($\alpha = 0.02$) is hardcoded to prevent oscillation and ensure smooth yaw changes. This means the weathervaning response is intentionally damped and will lag slightly behind the instantaneous roll angle.

Tuning & Behavior

- **Default Value:** 1 (Plane/Copter), 0 (Disabled/Other)
- **Range:** 0.5 to 4.0
- **Effect of Increasing:** The aircraft will yaw more aggressively into the wind. If set too high, it may overshoot or oscillate (wag its tail).
- **Effect of Decreasing:** The aircraft will take longer to align with the wind.
- **Dependencies:** `WVANE_ENABLE` must be active. `WVANE_OPTIONS` affects how the error is calculated.

WVANE_HGT_MIN (Weathervaning Minimum Height)

DEFAULT: 0	RANGE: 0 100	UNITS: m

Weathervaning Minimum Height (WVANE_HGT_MIN)

Description

`WVANE_HGT_MIN` prevents the drone from rotating while it is still very close to the ground. This ensures that the drone is safely clear of landing pads, grass, or personnel before it begins its automatic alignment with the wind.

WVANE_LAND (Weathervaning Land Enable)

DEFAULT: 0	RANGE: 0 1

Weathervaning Land Enable (WVANE_LAND)

Description

`WVANE_LAND` determines if the drone should face into the wind while it is touching down. This is highly recommended for QuadPlanes, as landing with a strong tailwind or crosswind can cause the drone to tip over or lose stability just as the motors spool down.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





WVANE_OPTIONS (Weathervaning options)

DEFAULT: 0	RANGE: null	UNITS: Bitmask

Weathervaning options (WVANE_OPTIONS)

Description

This parameter controls advanced behavior modifiers for the Weather Vane controller. The Weather Vane controller automatically yaws the aircraft into the wind during VTOL flight to reduce drag and improve stability.

By default, the controller primarily uses the **Roll** angle to detect if the aircraft is being blown sideways and applies a Yaw correction. Enabling specific options here allows the controller to also use **Pitch** information, providing a faster and more robust response when the wind is coming from the front or back.

The Mathematics

When **Bit 0 (PITCH_ENABLE)** is active, the controller adds the pitch error to the total error term, increasing the aggressiveness of the yaw correction.

Standard Logic (Bit 0 = 0):

$$Error = \max(|Roll_deg| - Deadzone, 0)$$

Enhanced Logic (Bit 0 = 1):
For **Nose In** weather vaning:

$$Error = |Roll_deg| + \max(Pitch_deg - Deadzone, 0)$$

(Only applied if Pitch > Deadzone)

This effectively means that if the aircraft is pitching up to fight the wind (while nose-in), that pitch effort is treated as "misalignment" energy that needs to be converted into a yaw correction to face the wind more directly.

The Engineer's View

This parameter is read in `AC_WeatherVane::get_yaw_out` inside `libraries/AC_AttitudeControl/AC_WeatherVane.cpp`.

It is cast to the `AC_WeatherVane::Options` enum.

- **Bit 0 (`PITCH_ENABLE`)**:
  - Defined as `(1<<0)`.
  - When set, it modifies the error calculation for `Direction::NOSE_IN` and `Direction::TAIL_IN`.
  - For `NOSE_IN`: It checks if `pitch_cdeg` is positive (nose up) and exceeds the deadzone.
  - For `TAIL_IN`: It checks if `pitch_cdeg` is negative (nose down) and exceeds the deadzone (in the negative direction).

This is particularly useful for QuadPlanes or Tailsitters where wind resistance manifests as both a roll moment (side force) and a pitch moment (drag).

Tuning & Behavior

- **Bit 0 (Value 1): Enable Pitch Input.**
  - **Recommended For:** Aircraft that tend to "hang" in the wind with a high pitch angle but fail to yaw into it aggressively enough.
  - **Effect:** Increases weathervaning gain effectively when the aircraft is pitching to hold position against the wind.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





WVANE_SPD_MAX (Weathervaning max ground speed)

DEFAULT: 0	RANGE: 0 50	UNITS: m/s

Weathervaning max ground speed (WVANE_SPD_MAX)

Description

This parameter defines a maximum ground speed threshold for active weathervaning. If the aircraft is moving across the ground faster than this speed, the weather vane controller is automatically disabled.

This is typically used to ensure that weathervaning only occurs during hover or slow maneuvering, and not during high-speed forward flight where the aerodynamics of the airframe (like a vertical tail) or other navigation controllers should take precedence.

The Mathematics

The controller checks the magnitude of the horizontal velocity vector (V_xy).

$$If \quad ||V_xy|| > SPD_MAX \quad \rightarrow \quad Disable \quad Weathervaning$$

The logic requires a valid velocity estimate from the AHRS/EKF. If speed is unknown, weathervaning is disabled for safety.

The Engineer's View

In `AC_WeatherVane::get_yaw_out`, this check uses `vel_ned.xy().length_squared()` against `_max_vel_xy` squared to avoid square root operations.
If `SPD_MAX` is set to 0, this check is skipped (weathervaning allowed at any speed).

Tuning & Behavior

- **Default Value:** 0 (Disabled/Unlimited) for Plane, typically higher for Copter.
- **Range:** 0 to 50 m/s.
- **Effect of Increasing:** Allows weathervaning at higher ground speeds.
- **Effect of Decreasing:** Restricts weathervaning to lower speeds.
- **Note:** If set to 0, the speed check is ignored.

WVANE_TAKEOFF (Weathervaning Takeoff Enable)

DEFAULT: 0	RANGE: 0 1

Weathervaning Takeoff Enable (WVANE_TAKEOFF)

Description

`WVANE_TAKEOFF` allows the drone to automatically rotate to face the wind as soon as it leaves the ground. This ensures that the climb-out phase of a mission is as efficient and stable as possible.

WVANE_VELZ_MAX (Weathervaning Max Vertical Speed)

DEFAULT: 0	RANGE: 0 5	UNITS: m/s

Weathervaning Max Vertical Speed (WVANE_VELZ_MAX)

Description

`WVANE_VELZ_MAX` sets a "Safety Cap" on weathervaning during climbs and descents.

If the drone is climbing or descending very fast, the aerodynamic forces on the airframe change significantly. This parameter ensures that the autopilot stops trying to rotate the drone into the wind if the vertical speed is too high, which could cause instability.

- **0 (Default):** Always allowed (Disabled vertical speed limit).
- **Recommendation:** Leave at **0** unless you notice heading instability specifically during fast climbs or descents.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **XTRACK** parameter group defines the limits for **Crosstrack Error** management. "Crosstrack" is the perpendicular distance from the vehicle to the planned mission line.

Key Concepts

1. Error Limits (`XTRACK_MAX`)

Defines the maximum allowed distance (meters) the vehicle can deviate from the path before the autopilot considers the mission "Invalid" or takes emergency action (e.g., stopping or circling).

2. Correction Angle (`XTRACK_ANG_LIM`)

(Sub/Plane) Limits the angle at which the vehicle is allowed to turn *towards* the path to correct an error.

- **Purpose:** Prevents the vehicle from turning 90 degrees to the path, which could cause a stall or unstable behavior.

Developer Notes

- **Library:** `libraries/AP_L1_Control`.

XTRACK_ANG_LIM (Crosstrack correction angle limit)

DEFAULT: 45	RANGE: 10 90	UNITS: deg

Crosstrack correction angle limit (XTRACK_ANG_LIM)

Description

This parameter sets the maximum angle the vehicle (Submarine) is allowed to turn towards the track line to correct for crosstrack error. A higher value allows for more aggressive course corrections to return to the path quickly, while a lower value results in smoother, shallower intercept trajectories. This is crucial for towed systems or vehicles with high inertia to prevent oscillation around the track.

The Mathematics

The navigation controller limits the heading error correction demand:

$$\theta_{error} = \text{Target Bearing} - \text{Current Heading}$$

$$|\theta_{correction}| \leq \text{XTRACK_ANG_LIM}$$

The Engineer's View

Defined in `ArduSub/Parameters.cpp`. This value is used by the WPNav library (specifically the crosstrack error controller) to clamp the output angle. It prevents the vehicle from turning 90 degrees to the track, which can be undesirable for ROVs with tethers.

Tuning & Behavior

- **Default Value:** 45 deg
- **Range:** 10 - 90
- **Effect of Increasing:** Faster return to track, potential for "S-turning" or oscillation.
- **Effect of Decreasing:** Smoother, slower return to track. Better for stability.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





XTRACK_MAX (Sailing vehicle max cross track error)

DEFAULT: 10	RANGE: 5 25	UNITS: m

Sailing vehicle max cross track error (XTRACK_MAX)

Description

This parameter defines the width of the "Sailing Corridor" for autonomous sailboats. When sailing upwind (into the wind), the boat must zigzag (tack) to reach its destination.

`XTRACK_MAX` sets the maximum allowable distance the boat can drift from the direct line to the waypoint before it is forced to tack (turn 90 degrees) to get back on course.

The Mathematics

This parameter effectively creates a virtual hallway of width $2 \times XTRACK_MAX$ centered on the line between the previous and next waypoint.

$$\text{Trigger Tack IF: } |CrossTrackError| \geq XTRACK_MAX$$

The logic also checks the wind direction to ensure the tack will actually reduce the error (i.e., turning *towards* the center line, not away).

The Engineer's View

In `Sailboat::calc_heading()` (`Rover/sailboat.cpp`):

1. The function computes `cross_track_error` using the navigation library.
2. It checks `if ((fabsf(cross_track_error) ≥ xtrack_max) && ...`.
3. If triggered, `should_tack` becomes `true`.
4. If the boat is on Starboard tack and error is positive (right side of corridor), it tacks to Port.
5. If on Port tack and error is negative (left side of corridor), it tacks to Starboard.

Tuning & Behavior

- **Default Value:** 10 meters
- **Range:** 5 to 25 meters
- **Effect of Increasing:** The boat will tack less frequently, making longer legs. This is generally more efficient as tacking slows the boat down.
- **Effect of Decreasing:** The boat will stay very close to the direct path, tacking frequently. This is useful in narrow channels but inefficient in open water.
- **0:** Disables auto-tacking based on crosstrack error.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY

Overview

The **XY** parameter group (specifically `XY_CTRL_MUL`) configures the gain scaling for horizontal movement.

Key Concepts

1. Gain Multiplier (`XY_CTRL_MUL`)

(ArduPlane/SysID) This is a safety parameter used during automated tuning or experimental modes. It scales the horizontal position and velocity controller gains.

- **Purpose:** Allows for "detuning" the position hold to prevent it from interfering with a test signal (e.g., during System Identification).

Developer Notes

- **Library:** `ArduPlane/systemid.cpp`.

XY_CTRL_MUL (System identification XY control multiplier)

DEFAULT: 0.1	RANGE: 0.05 1.0

XY_CTRL_MUL: System identification XY control multiplier

Description

A multiplier for the XY velocity and position controller when using systemID in VTOL modes that do horizontal position and velocity control.

Values

- **Range:** 0.05 to 1.0
- **Default:** 0.1

Description

This parameter scales down the position and velocity controller gains during a System Identification flight test on a QuadPlane.

- **Function:** When System ID is active in a position-controlled mode (like QLoiter), the normal PID gains might be too aggressive combined with the injected chirp signal. `XY_CTRL_MUL` reduces the effectiveness of the outer loops (Position/Velocity) to allow the inner loops (Attitude/Rate) to be excited cleanly without the drone fighting the test inputs too hard.
- **Usage:** Only relevant when `SID_AXIS` is set to a position or velocity axis.
- **Safety:** Prevents the drone from over-reacting to the test signal and flying away.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **YAW** parameter group handles configuration for the **Heading (Yaw) Axis** across various vehicle types.

Key Concepts

1. Heading Maintenance

Yaw is critical for ensuring the vehicle's nose is pointed correctly for navigation, sensors (cameras/Lidar), or aerodynamics.

- `YAW_BEHAVE`: Defines how the nose behaves during mission flight (Face next WP, Face Home, etc.).

2. PID Tuning (`YAW_RATE_P/I/D`)

(If present in group) Standard gains to control the responsiveness of the yaw rotation.

Developer Notes

- **Library:** `libraries/AC_AttitudeControl`.
- **See Also:** ATC for more details on rate PID loops.

YAW2SRV_DAMP (Yaw Damping)

DEFAULT: 0	RANGE: 0 2.0

Yaw Damping (YAW2SRV_DAMP)

Description

`YAW2SRV_DAMP` is the "Anti-Wag" gain for planes.

When a plane hits a gust of wind, the tail often "Fishtails" back and forth. This gain uses the gyro to detect that horizontal motion and applies counter-rudder to stop it instantly. This is the most important parameter for a stable, professional-feeling flight on fixed-wing aircraft.

- **Higher Value:** Stiffer, more damped tail.
- **Too High:** The rudder will chatter or vibrate.

YAW2SRV_IMAX (Yaw Integrator Limit)

DEFAULT: 1500	RANGE: 0 4500	UNITS: cdeg

Yaw Integrator Limit (YAW2SRV_IMAX)

Description

`YAW2SRV_IMAX` prevents the automatic rudder trim from taking over the entire plane.

It limits how many degrees of deflection the YAW2SRV_INT term can command. If your plane is badly out of alignment, the integrator could theoretically "Walk" the rudder all the way to its mechanical limit. This safety cap ensures you always have visual control remaining.

- **Default:** 1500 (1/3rd of the total rudder travel).

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY

YAW2SRV_INT (Sideslip Control Integrator)

DEFAULT: 0	RANGE: 0 2.0

Sideslip Control Integrator (YAW2SRV_INT)

Description

`YAW2SRV_INT` acts like a "Self-Trimming Rudder."

If your plane naturally pulls to one side because of motor torque or a warped tail, the I-term learns the required offset to keep the flight perfectly coordinated without any pilot input.

YAW2SRV_RLL (Yaw Coordination Gain)

DEFAULT: 1.0	RANGE: 0.8 1.2

Yaw Coordination Gain (YAW2SRV_RLL)

Description

`YAW2SRV_RLL` is the "Turn Finisher."

Once a turn is established, this gain fine-tunes the rudder to ensure the plane's tail is perfectly tracking its nose. If the plane tends to skid (tail out) or slip (tail in) during a steady circle, this value is adjusted to fix it.

- **1.0 (Default):** Physics-based calculation.
- **Values > 1.0:** Increase yaw into the turn.
- **Values < 1.0:** Decrease yaw into the turn.

YAW2SRV_SLIP (Sideslip Control Gain)

DEFAULT: 0	RANGE: 0 4.0

Sideslip Control Gain (YAW2SRV_SLIP)

Description

`YAW2SRV_SLIP` helps a plane fly "Straight" through the air during turns.

For aircraft with a significant vertical tail or fuselage area, a sideslip causes lateral (sideways) acceleration. This parameter uses the accelerometer to detect that sideways force and automatically applies rudder to cancel it out, ensuring the plane's nose is always pointing exactly where it is going.

- **Higher Value:** More aggressive rudder correction to keep the ball centered.
- **0:** Disabled.

Tuning & Behavior

- **Default Value:** 0.
- **Recommendation:** Only tune this after YAW2SRV_DAMP is set. Do not use on flying wings or gliders without a large vertical tail.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





YAW_BEHAVE (Follow Yaw Behavior)

DEFAULT: 1	RANGE: 0 3

Follow Yaw Behavior (YAW_BEHAVE)

Description

`YAW_BEHAVE` defines what the drone points its "Face" (camera) at while it is chasing a target.

- **0: None.** The drone keeps its current heading and does not automatically rotate.
- **1: Face Target (Default).** The drone always turns to point its nose directly at the person or object it is following. Good for filming yourself.
- **2: Face Next Waypoint.**
- **3: Face Direction of Travel.** The drone points its nose where it is going, like a real airplane.

Tuning & Behavior

- **Default Value:** 1.
- **Use Case:** If you are using "Follow Me" to film a mountain biker from the side, you want **Mode 1** so the camera stays locked on the subject.

YAW_CORR (Yaw Correction Gain)

DEFAULT: 0.1	RANGE: 0 1.0

Yaw Correction Gain (YAW_CORR)

Description

`YAW_CORR` defines how much the autopilot trusts the compass or GPS heading compared to its internal motion sensors (IMU).

Over time, the IMU "Drifts" and loses track of true North. The autopilot uses the compass (or GPS ground track) to "Pull" the IMU back into alignment.

- **Higher Value:** Faster alignment to the compass, but more susceptible to compass glitches (e.g. flying near large metal structures).
- **Lower Value:** Smoother heading, but more prone to long-term drift.

YAW_MAX (Mount Yaw angle maximum)

DEFAULT: 180	RANGE: -180 180	UNITS: deg

Mount Yaw angle maximum

Note: This parameter configures instance MAX. It functions identically to MNT1_YAW_MAX.

YAW_MIN (Mount Yaw angle minimum)

DEFAULT: -180	RANGE: -180 180	UNITS: deg

Mount Yaw angle minimum

Note: This parameter configures instance MIN. It functions identically to MNT1_YAW_MIN.

YAW_M_NSE (Visual odometry yaw measurement noise)

DEFAULT: 0.2f	RANGE: 0.05 1.0	UNITS: rad

Visual odometry yaw measurement noise

Note: This parameter configures instance None. It functions identically to VISO_YAW_M_NSE.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





YAW_RANGE (Antenna Tracker Yaw Range)

DEFAULT: 360	RANGE: 0 360	UNITS: deg

Antenna Tracker Yaw Range (YAW_RANGE)

Description

`YAW_RANGE` defines the mechanical limits of the tracker's pan axis.

Antenna trackers often have slip-rings (allowing continuous 360 rotation) or cable limits. This parameter tells the software how far it can turn before it needs to stop or "unwind."

- **360 (Default):** Full rotation allowed (assuming slip ring).
- **180:** Limited rotation (e.g. +/- 90 degrees).

Tuning & Behavior

- **Safety:** If you do not have a slip ring, setting this correctly is vital to prevent tearing your antenna cables.

YAW_RATE_D (Yaw Rate D Gain)

DEFAULT: 0	RANGE: 0 0.05

Yaw Rate D Gain (YAW_RATE_D)

Description

`YAW_RATE_D` acts as a brake for horizontal rotation.

It is rarely needed for multirotors, as the drag of the propellers usually provides enough natural damping. However, on large drones with heavy batteries mounted far from the center, a small amount of D-gain can help stop the rotation more cleanly.

YAW_RATE_D_FF (FF FeedForward Gain)

DEFAULT: 0	RANGE: null	UNITS: null

FF FeedForward Gain

Note: This parameter configures instance D_FF. It functions identically to RLL_RATE_D_FF.

YAW_RATE_ENABLE (Yaw Rate Controller Enable)

DEFAULT: 1	RANGE: 0 1

Yaw Rate Controller Enable (YAW_RATE_ENABLE)

Description

`YAW_RATE_ENABLE` activates the yaw stabilization loop.

- **1 (Default):** The autopilot actively uses the motors (and rudder/tail) to hold the drone's heading.
- **0:** Yaw stabilization is disabled. The pilot has 100% manual control over yaw torque. (Highly unstable for multirotors!).

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY

YAW_RATE_FF (Yaw Rate Feed-Forward)

DEFAULT: 0	RANGE: 0 0.5

Yaw Rate Feed-Forward (YAW_RATE_FF)

Description

`YAW_RATE_FF` gives the drone a "Head Start" on turning.

It injects a portion of your stick command directly into the motors, bypassing the PID calculation. This makes the drone feel more connected and less "mushy."

Tuning & Behavior

- **Default Value:** 0.
- **Recommendation:** Tune until the drone starts rotating exactly when you move the stick, without waiting for the error to build up.

YAW_RATE_FLTD (PID D term filter frequency in Hz)

DEFAULT: AC_PID_DFILT_HZ_DEFAULT	RANGE: null	UNITS: Hz

PID D term filter frequency in Hz

Note: This parameter configures instance FLTD. It functions identically to RLL_RATE_FLTD.

YAW_RATE_FLTE (PID Error filter frequency in Hz)

DEFAULT: AC_PID_EFILT_HZ_DEFAULT	RANGE: null	UNITS: Hz

PID Error filter frequency in Hz

Note: This parameter configures instance FLTE. It functions identically to RLL_RATE_FLTE.

YAW_RATE_FLTT (PID Target filter frequency in Hz)

DEFAULT: AC_PID_TFILT_HZ_DEFAULT	RANGE: null	UNITS: Hz

PID Target filter frequency in Hz

Note: This parameter configures instance FLTT. It functions identically to RLL_RATE_FLTT.

YAW_RATE_I (Yaw Rate I Gain)

DEFAULT: 0.02	RANGE: 0 0.1

Yaw Rate I Gain (YAW_RATE_I)

Description

`YAW_RATE_I` ensures the drone stays pointed in the correct direction over long periods.

If the drone is constantly being pushed by a crosswind or has an imbalanced motor, the I-term builds up pressure to hold the nose exactly where it is supposed to be.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





YAW_RATE_IMAX (Yaw Rate I Max)

DEFAULT: 0.222	RANGE: 0 1.0

Yaw Rate I Max (YAW_RATE_IMAX)

Description

`YAW_RATE_IMAX` limits how much power the "Memory" (I-Term) can use to fix a yaw error.

- **0.222 (Default):** The integrator can use up to 22.2% of the motor range.
- **1.0:** The integrator can use 100% of the motor range. (Dangerous!).

Tuning & Behavior

- **Safety:** This prevents "Wind-Up." If you block the drone's rotation with your hand, the integrator would normally keep building up power to try and move it. `IMAX` caps that power so the drone doesn't explode when you let go.

YAW_RATE_P (Yaw Rate P Gain)

DEFAULT: 0.25	RANGE: 0.1 0.5

Yaw Rate P Gain (YAW_RATE_P)

Description

`YAW_RATE_P` controls how aggressively the drone stops its horizontal rotation.

When you let go of the yaw stick, the drone should stop spinning and hold its new heading. If the drone is slow to stop or "Drifts" around, you need more P.

- **Too High:** The tail will "Wag" or vibrate horizontally.
- **Too Low:** The drone will be slow to respond to yaw commands and will not hold its heading well in the wind.

YAW_RATE_PDMX (PD sum maximum)

DEFAULT: 0	RANGE: null	UNITS: null

PD sum maximum

Note: This parameter configures instance PDMX. It functions identically to RLL_RATE_PDMX.

YAW_RATE_SMAX (Slew rate limit)

DEFAULT: 0	RANGE: 0 200	UNITS: null

Slew rate limit

Note: This parameter configures instance SMAX. It functions identically to RLL_RATE_SMAX.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





YAW_SLEW_TIME (Yaw Slew Time)

DEFAULT: 2	RANGE: 0 20	UNITS: s

Yaw Slew Time (YAW_SLEW_TIME)

Description

`YAW_SLEW_TIME` acts as a speed governor for the Antenna Tracker's pan axis.

It forces the tracker to move smoothly between targets rather than snapping instantly. This reduces mechanical stress on the gears and prevents the heavy high-gain antenna from wobbling when it stops.

- **0:** Unlimited speed (Servo max speed).
- **2 (Default):** Takes at least 2 seconds to do a full rotation.

Tuning & Behavior

- **Recommendation:** Increase this value if you have a heavy antenna array. Smoother movement is generally better for video link quality than jerky tracking.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **YAW2SRV** parameter group configures the **Yaw Damping** and **Ground Steering** PID gains for **ArduPlane**.

Key Concepts

1. Yaw Damping (`YAW2SRV_DAMP`)

Prevents the tail of the plane from "wagging" (Dutch Roll) in turbulence.

- **Action:** Automatically deflects the rudder to oppose sudden uncommanded yaw movements.

2. Side-Slip Correction (`YAW2SRV_SLIP`)

Automatically applies rudder during a turn to ensure the ball is centered (coordinated turn), preventing the tail from slipping or skidding.

Developer Notes

- **Library:** `libraries/APM_Control/AP_YawController.cpp`.
- **Context:** Only relevant for **ArduPlane**.

YAW2SRV_DAMP (Yaw Damping Gain (Fixed-Wing))

DEFAULT: 0	RANGE: 0 2.0

Yaw Damping Gain (YAW2SRV_DAMP)

Description

`YAW2SRV_DAMP` is the primary tool for stopping a plane's tail from "wagging."

In a fixed-wing aircraft, aerodynamic forces or sudden wind gusts can cause the tail to swing left and right (Yaw Oscillation). This parameter applies a counter-force via the rudder based on the yaw rate measured by the gyro. It "resists" fast rotation on the yaw axis, leading to a much smoother and more stable flight path.

Tuning & Behavior

- **Default:** 0 (Disabled).
- **Recommendation:** Start with **0.1** and increase in small increments.
- **Optimal Setting:** The plane should fly straight and true, even in turbulent air, without any noticeable oscillating or "hunting" behavior from the rudder.
- **Dependency:** This should be tuned after setting the basic rudder mix (KFF_RDDRMIX).

YAW2SRV_IMAX (Yaw Integrator Limit)

DEFAULT: 1500	RANGE: 0 4500	UNITS: cdeg

Yaw Integrator Limit (YAW2SRV_IMAX)

Description

`YAW2SRV_IMAX` prevents the "Integrator Windup" problem.

The integrator (YAW2SRV_INT) slowly adds rudder trim to correct for persistent errors. However, if this value can grow too large, the rudder might get "stuck" over to one side for a long time even after the error is gone. This parameter sets a hard limit on how much the integrator can move the servo.

- **1500 (Default):** Allows up to 15 degrees of rudder trim (approx 1/3 of total travel).

Tuning & Behavior

- **Recommendation:** Leave at **1500**. If your plane needs more than 15 degrees of constant rudder to fly straight, you have a major mechanical build issue.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY

YAW2SRV_INT (Sideslip Control Integrator)

DEFAULT: 0	RANGE: 0 2.0

Sideslip Control Integrator (YAW2SRV_INT)

Description

`YAW2SRV_INT` (Integral) eliminates long-term steady-state sideslip.

While `YAW2SRV_DAMP` reacts to sudden movements, and `YAW2SRV_SLIP` reacts to lateral forces, the Integrator term looks at persistent errors. If the plane is consistently flying with its tail slightly to one side (e.g. due to a slightly crooked vertical fin or asymmetric motor thrust), the Integrator will slowly "build up" and apply a constant rudder trim to straighten the aircraft.

Tuning & Behavior

- **Default:** 0.
- **Recommendation:** Start with a very small value (e.g. **0.05** or **0.1**).
- **Effect:** Too much integrator will cause the tail to "Wander" or hunt slowly left and right. Too little will leave the plane flying "crabbed" through the air.

YAW2SRV_RLL (Yaw Coordination Gain (Bank Angle))

DEFAULT: 1.0	RANGE: 0.8 1.2

Yaw Coordination Gain (Bank Angle) (YAW2SRV_RLL)

Description

`YAW2SRV_RLL` fine-tunes the turn geometry.

For a fixed-wing aircraft to turn efficiently, the nose must rotate (yaw) at a specific rate that matches the bank angle and airspeed. If the yaw is too slow, the plane slips into the turn. If too fast, it skids out. This parameter adjusts the calculation of that "perfect" yaw rate.

Tuning & Behavior

- **Default:** 1.0 (Ideal physics).
- **Values > 1.0:** Demands *more* yaw into the turn.
- **Values < 1.0:** Demands *less* yaw into the turn.
- **Significance:** If you have accurate airspeed calibration, leave this at **1.0**. If you see persistent sideslip during steady turns, adjust YAW2SRV_SLIP first.

YAW2SRV_SLIP (Sideslip Coordination Gain)

DEFAULT: 0	RANGE: 0 4.0

Sideslip Coordination Gain (YAW2SRV_SLIP)

Description

`YAW2SRV_SLIP` is used to achieve perfectly "Coordinated" turns (where the ball stays centered in a physical cockpit).

When an aircraft banks into a turn, it often experiences "Sideslip" – moving through the air slightly sideways. This parameter uses the accelerometer to sense that sideways force and automatically applies the correct amount of rudder to keep the nose pointed exactly where the plane is flying.

- **0: Disabled.** Standard for flying wings and gliders.
- **Non-Zero:** Active coordination for planes with a vertical tail.

Tuning & Behavior

- **Default:** 0.
- **Usage:** Only use this for traditional aircraft with a large vertical fin/rudder and significant fuselage side area.
- **Tuning Order:** Tune this **Last**, after YAW2SRV_DAMP and YAW2SRV_INT are finalized.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





Overview

The **ZIGZ** parameter group configures the **ZigZag** flight mode. This is a specialized semi-autonomous mode for **Agricultural Sprayers** and **Surveying**.

It allows the pilot to define two points (A and B) and then simply push the stick forward to have the drone fly a perfect straight line to the other side, "step" to the next row, and wait for the next command.

Key Concepts

1. Line Sequencing

The mode automates the repetitive "Back and Forth" work of crop spraying.

- `ZIGZ_SIDE_DIST`: Defines the "Step" or width (meters) of each row.
- `ZIGZ_DIRECTION`: Selects if the step should be to the left or right.

2. Auto-Spraying (`ZIGZ_SPRAYER`)

Can be linked to the `SPRAY_` group to automatically turn the pump on during the straight segments and off during the turns.

Developer Notes

- **Library:** `ArduCopter/mode_zigzag.cpp`.

ZIGZ_AUTO_ENABLE (ZigZag Auto Enable)

DEFAULT: 0	RANGE: 0 1

ZIGZ_AUTO_ENABLE: ZigZag Auto Enable

Description

Allows you to enable (1) or disable (0) the ZigZag Auto feature.

Values

Value	Meaning
0	Disabled
1	Enabled

- **Default:** 0

Description

This parameter enables the advanced automation features of **ZigZag Mode** (Copter/Rover).

- **ZigZag Mode:** A manual/semi-auto flight mode designed for crop spraying. The pilot flies to one side of the field and flips a switch. The drone records that point as "Point A." The pilot flies to the other side and flips the switch again to record "Point B."
- **Auto Enable:** When this parameter is set to **1**, the drone will automatically fly back and forth between the two defined lines (advancing forward by the `ZIGZ_SIDE_DIST` each time), essentially creating an instant grid mission without a GCS.

Take Your Professional Drone Operations
to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





ZIGZ_DIRECTION (ZigZag Mode Step Direction)

DEFAULT: 0	RANGE: 0 1

ZigZag Mode Step Direction (ZIGZ_DIRECTION)

Description

`ZIGZ_DIRECTION` determines which way the drone "marches" across the field.

- **0: Right.**
- **1: Left.**

Tuning & Behavior

- **Default:** 0.
- **Usage:** Set this based on your starting position and the shape of your field. If you start on the left edge of a field, you will want the drone to step **Right (0)** to cover the remaining area.

ZIGZ_LINE_NUM (ZigZag Mode Total Lines)

DEFAULT: 0	RANGE: -1 32767	UNITS: Lines

ZigZag Mode Total Lines (ZIGZ_LINE_NUM)

Description

`ZIGZ_LINE_NUM` defines the "Mission Length" for ZigZag mode.

- **-1: Infinity.** The drone will continue flying back and forth between the A and B points and stepping sideways until the pilot cancels the mode or the battery runs low.
- **0:** Single Step. The drone will only perform the lateral movement when commanded, then wait.
- **1-N:** The drone will automatically complete this many parallel legs across the field and then come to a halt.

Tuning & Behavior

- **Usage:** Set this to the number of rows in your field to automate the entire spraying or scanning job.

ZIGZ_SIDE_DIST (ZigZag Mode Step Distance)

DEFAULT: 4.0	RANGE: 0.1 100.0	UNITS: m

ZigZag Mode Step Distance (ZIGZ_SIDE_DIST)

Description

`ZIGZ_SIDE_DIST` defines the "Width" of your scanning pattern.

In **ZigZag** mode (commonly used for agricultural crop spraying), the pilot defines two points (A and B). The drone flies back and forth between these points. When it reaches one end, the pilot can command a "Step" to the side. This parameter sets how many meters the drone moves sideways for each of those steps.

Tuning & Behavior

- **Default:** 4.0 meters.
- **Recommendation:** Set this to the effective width of your spraying boom or the "swath" of your sensor to ensure 100% coverage with no gaps or excessive overlaps.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY





ZIGZ_SPRAYER (ZigZag Mode Auto Sprayer)

DEFAULT: 0	RANGE: 0 1

ZigZag Mode Auto Sprayer (ZIGZ_SPRAYER)

Description

`ZIGZ_SPRAYER` automates the crop-spraying process.

When enabled, the autopilot will automatically turn on the sprayer (via a relay or PWM) as the drone begins a "Zig" or "Zag" leg, and turn it off when it reaches the end point or while it is performing a lateral step. This ensures that chemicals are only applied over the intended area and not wasted during turns.

ZIGZ_WP_DELAY (ZigZag Waypoint Delay)

DEFAULT: 0	RANGE: 0 127	UNITS: s

ZigZag Waypoint Delay (ZIGZ_WP_DELAY)

Description

`ZIGZ_WP_DELAY` introduces a pause at the turn-around points.

When the drone reaches point A or point B in ZigZag mode, it will wait for this many seconds before stepping sideways and continuing to the next leg. This is useful for allowing mechanical equipment (like a sprayer pump) to fully cycle or for the drone's attitude to stabilize before the next movement.

Tuning & Behavior

- **Default:** 0 seconds.
- **Recommendation:** Use **1 or 2 seconds** if you notice the drone's "Zig" and "Zag" legs aren't perfectly parallel due to aggressive turn-around maneuvers.

Take Your Professional Drone Operations to the next level with MAVLink HUD
GET IT ON GOOGLE PLAY



