Anyleaf GNSS (CAN)

September 2023

General Description

The AnyLeaf CAN GNSS is a small device that transmits position data over CAN. It’s intended to be used with small unmanned aircraft. It periodically measures and broadcasts the following information:

- GNSS (GPS) fix and Dilution of Precision (DOP) information
- Barometric pressure
- Magnetometer data (3-axis)
- Fused position from the GNSS and IMU data
- Raw IMU readings (3-axis accelerometer and 3-axis gyroscope)
- AHRS solution (estimated attitude)

Broadcast rate, and enabling is customizable for any of these properties.

This device uses the DroneCAN protocol, and is compatible with any flightcontroller that implements the applicable DroneCAN messages.

Specifications

- **Dimensions:** 49 × 49 × 17 (height) mm. 70mm width with tabs
- **Mounting holes:** 2 × M4, spaced 60.4mm
- **Weight:** 28 grams
- **Power input:** 5V, via CAN or USB-C
- **MCU:** STM32G431. 170Mhz Cortex-M4
- **GNSS device:** Ublox SAM-M10Q
- **Inertial Measurement Unit (IMU):** TDK ICM-42688p
- **Barometer (pressure altimeter):** Infineon DPS-310
- **Magnetometer:** ST LIS3MDL
- **Update capability**: USB-C, CAN
- **Bus compatibility:** DroneCAN
- **CAN tranceiver:** NXP TJA1051TK/3
- **CAN version:** CAN-FD capable
- **CAN headers:** 2 × JST GH, 1.25mm pitch

- **Max CAN datarate:** 5Mbps
- **GNSS fix rate:** Up to 10Hz
- **Pressure and temperature update rate:** Up to 32Hz
- **Magnetometer update rate:** 155Hz
- **Position update rate with IMU and baro fusing:** Up to 1kHz
- **AHRS and IMU data update rate:** Up to 1kHz
- **Flight controller firmware compatibility:** Ardupilot, and PX4. Compatible with any firmware that supports the applicable *DroneCAN* message types.

Integrating with your aircraft

This device connects with aircraft systems using a 4-pin connection header; this powers the device, and allows two-way communication over CAN. It uses a JST-GH header, with connections labeled on the enclosure for 5V power, CAN data high, CAN data low, and ground. Because CAN is a bus, multiple peripherals can use these same wires for power and data, and routing can be set up in a way that makes sense for a given aircraft geometry. This device (and many CAN devices) includes two CAN connectors: This can be used to simplify wiring: For example, run one CAN cable from the flight controller to one of this device's connectors. Run another wire from this device's second connector to another CAN device in the same area of the aircraft.

Protocol description

This device is compatible with DroneCAN, and can exist on busses that include other DroneCAN devices. It periodically broadcasts information from its onboard sensors. The broadcast rate of this information is customization, either using PC software available on the AnyLeaf website, using the USB-C connection on the device, or via a CAN configuration message.

Most messages broadcast by this device are included in the Dronecan *List of standard data types*: https://dronecan.github.io/Specification/7._List_of_standard_data_types/. A notable exception is the configuration setup uses a custom data type.

The standard data type use allows for compatibility with any flight control firmware that supports the DroneCAN standard. (For example, Ardupilot, and PX4.)

Messages periodically broadcast:

- Gnss fix. (uavcan.equipment.gnss.Fix2)

- Static air pressure (uavcan.equipment.air_data.StaticPressure)

- Air temperature (uavcan.equipment.air_data.StaticTemperature)

- Magnetic field strength (uavcan.equipment.ahrs.MagneticFieldStrength2)

- IMU data (uavcan.equipment.ahrs.RawIMU)

- Fused position, from GNSS and IMU (uavcan.equipment.navigation.GlobalNavigationSolution, or optionally a custom format that is more compact)

- Node status (uavcan.protocol.NodeStatus)

Messages accepted:

- Node info request (uavcan.protocol.GetNodeInfo)

- Dynamic node ID allocation (uavcan.protocol.dynamic_node_id/Allocation)

- Node restart (uavcan.protocol.RestartNode)

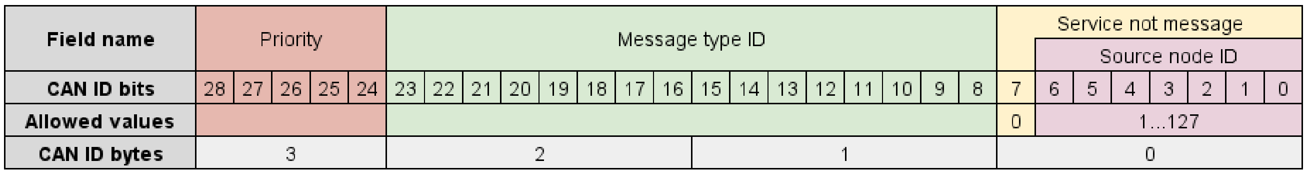
- Get/Set parameters (uavcan.protocol.param.GetSet)

DroneCAN protocol

This device uses the DroneCAN protocol. Information about its wire protocol can be found in its specification here; most notably in chapter 4:
https://dronecan.github.io/Specification/4._CAN_bus_transport_layer/

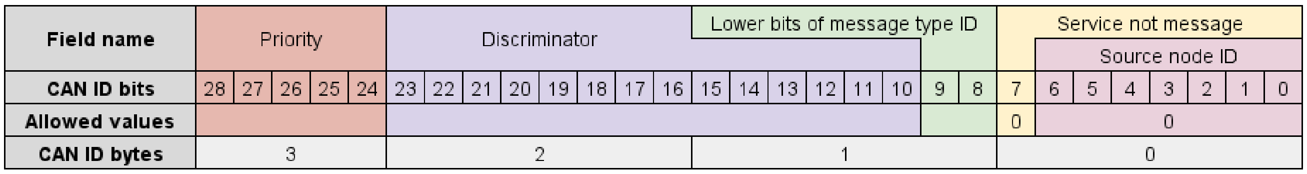
The CAN ID format is as follows. Most messages broadcast by this device, including all sensor readings, use the *Message frame* format.

Message frame



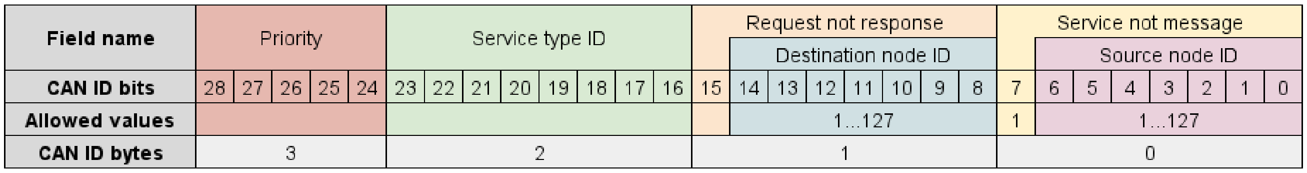
Field name	Priority					Message type ID																Service not message							
																							Source node ID						
CAN ID bits	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Allowed values																						0	1...127						
CAN ID bytes	3					2								1								0							

Anonymous message frame



Field name	Priority					Discriminator								Lower bits of message type ID								Service not message							
																							Source node ID						
CAN ID bits	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Allowed values																						0	0						
CAN ID bytes	3					2								1								0							

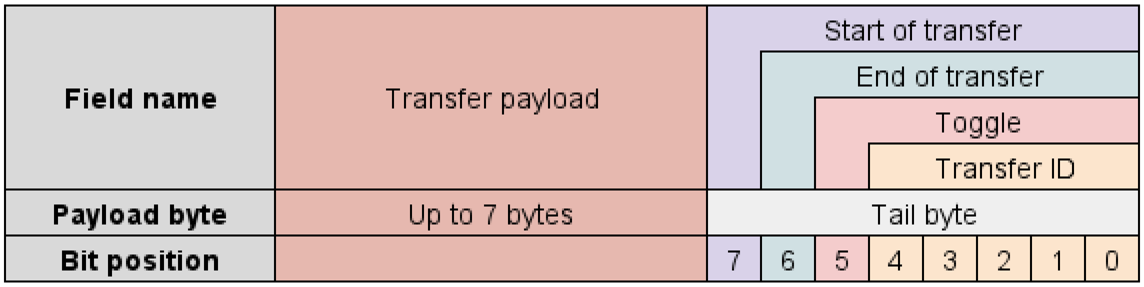
Service frame



Field name	Priority					Service type ID								Request not response								Service not message							
															Destination node ID								Source node ID						
CAN ID bits	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Allowed values															1...127							1	1...127						
CAN ID bytes	3					2								1								0							

The first two bytes of every message is the CRC; information on decoding it is found in the DroneCAN Spec linked above. The final byte in each frame called is the tail byte; this uses the following format, and contains information describing if a payload is contained in a single frame, or split across multiple ones:

CAN payload



Field name	Transfer payload	Start of transfer							
			End of transfer						
				Toggle					
					Transfer ID				
Payload byte	Up to 7 bytes	Tail byte							
Bit position		7	6	5	4	3	2	1	0

Payload contents for the message types this device broadcasts are described below.

GNSS Fix (position, velocity, time) format:

This device broadcasts GNSS fixes using the DroneCAN *Fix2* message. This uses the following bit-aligned payload format:

Bits	Description	Data type
0-63	Timestamp since node start in µs	56-bit unsigned integer
64-119	GNSS timestamp in µs since January 1, 1970	56-bit unsigned integer
120-122	GNSS time standard	Enum: 0: None; 1: TAI; 2: UTC; 3: GPS
136-143	Number of leap seconds included in the GNSS timestamp	unsigned integer
144-180	Longitude, in degrees divided by 100,000,000	signed 37-bit integer
181-217	Latitude, in degrees divided by 100,000,000	signed 37-bit integer
218-244	Height above ellipsoid, in mm; signed integer	signed 27-bit integer
245-271	Height above Mean Sea Level (MSL), in mm	signed 27-bit integer
272-303	Velocity, north component, in m/s	32-bit floating point
304-335	Velocity, east component, in m/s	32-bit floating point
336-367	Velocity, down component, in m/s	32-bit floating point
368-373	Number of satellites used	unsigned integer
374-375	Fix status	Enum: No fix; 1: Time only; 2: 2D fix; 3: 3D fix
376-379	GNSS mode (Currently only uses Single mode)	Enum: 0: Single; 1: DGPS; 2: RTK; 3: PPP
380-385	GNSS sub mode. (Currently unused)	
392-407	Position covariance matrix diagonal values; values with order: North-North, North-East, North-Down, East-East, East-Down, Down-Down	16-bit floating point [6]
408-423	Positional dilution of precision (PDOP)	16-bit floating point

Magnetic field strength format (ie magnetometer readings):

This device broadcasts magnetic field strength readings using the DroneCAN *Magnetic Field Strength 2* message. Reference the visual axis depiction on top of the device for definitions of these axes. This uses the following payload format:

Bytes	Description	Data type
0	Sensor ID	8-bit unsigned integer
1-2	Magnetic field strength along the X axis	16-bit floating point
3-4	Magnetic field strength along the Y axis	16-bit floating point
5-6	Magnetic field strength along the Z axis	16-bit floating point

Pressure format (ie barometer):

This device broadcasts pressure readings using the DroneCAN *Static Pressure* message. This uses the following payload format:

Bytes	Description	Data type
0-3	Pressure in Pascals	32-bit floating point

IMU format (accelerometer and gyroscope):

This device broadcasts IMU readings using the DroneCAN *Raw IMU* message. Reference the visual axis depiction on top of the device for definitions of these axes. This uses the following format:

Bytes	Description	Data type
0-6	Timestamp since node start in μs	56-bit unsigned integer
11-12	Gyroscope X axis reading in radians/seconds	16-bit floating point
13-14	Gyroscope Y axis reading in radians/seconds	16-bit floating point
15-16	Gyroscope Z axis reading in radians/seconds	16-bit floating point
29-30	Gyroscope X axis reading in radians/seconds	16-bit floating point
31-32	Gyroscope Y axis reading in radians/seconds	16-bit floating point
23-34	Gyroscope Z axis reading in radians/seconds	16-bit floating point

Attitude and Heading Reference System (AHRS) format:

This device broadcasts IMU readings using the DroneCAN *AHRS Solution* message. Reference the visual axis depiction on top of the device for definitions of these axes. Attitude is represented as a unit-length quaternion that uses Hamilton's conventions. This uses the following format:

Bytes	Description	Data type
0-6	Timestamp since node start in μs	56-bit unsigned integer
7-8	Attitude quaternion, X component	16-bit floating point
9-10	Attitude quaternion, Y component	16-bit floating point
11-12	Attitude quaternion, Z component	16-bit floating point
13-14	Attitude quaternion, W component	16-bit floating point
15-16	Angular velocity, X component, in radians/second	16-bit floating point
17-18	Angular velocity, Y component, in radians/second	16-bit floating point
19-20	Angular velocity, Z component, in radians/second	16-bit floating point
21-22	Linear acceleration, X component, in m/s²	16-bit floating point
23-24	Linear acceleration, Y component, in m/s²	16-bit floating point
25-26	Linear acceleration, Z component, in m/s²	16-bit floating point

Fused position data:

This device broadcasts fused position data; this is a high-update-rate position solution with fused data from the GNSS and IMU. The data depicted here is our custom format. By default, this device outputs the DroneCAN *Global Naviation Solution* format, which is more complicated. The format used is selectable when configuring over USB or CAN. Reference the visual axis depiction on top of the device for definitions of these axes.

Bytes	Description	Data type
0-7	Timestamp since node start in µs	64-bit unsigned integer
8-15	Latitude, in degrees divided by 100,000,000	64-bit signed integer
16-23	Longitude, in degrees divided by 100,000,000	64-bit signed integer
24-27	Elevation (height above ellipsoid (HAE)) in meters	32-bit floating point
28-31	Elevation (height above mean sea level (MSL)) in meters	32-bit floating point
32-35	Velocity in the north direction, in meters/second	32-bit floating point
36-39	Velocity in the east direction, in meters/second	32-bit floating point
40-43	Velocity in the down direction, in meters/second	32-bit floating point

Node status format:

This device periodically broadcasts the DroneCAN Node Status message. This message reports the following information:

Bits	Description	Data type
0-31	Timestamp since node start in µs	32-bit unsigned integer
32-34	Node health	Enum: 0: OK; 1:Warning; 2: Error; 3: Critical.
35-38	Node mode	Enum: Operational; 1: Initialization;2: Maintenance; 3: Software update; 7: Offline

This device responds to DroneCAN *GetNodeInfo* and *Restart* requests. The node info response contains the node status message above, and additional information about software version, hardware version, and the node name. The serialization format for this info is somewhat complicated, and is beyond the scope of this datasheet.

This device responds to dynamic node ID allocation requests, if the *Dynamic node ID allocation* setting, described below, is enabled. (It is enabled by default)

Note on bus saturation

It may be desirable to send certain messages, like IMU data, AHRS position, or fused position at a high data rate. This device is capable of configuring each of these at up to 1kHz. It’s possible to saturate a CAN bus when sending multiple packets (especially larger ones) at a high data rate. Using a higher-speed bus (ie 5Mbps) allows for more data. When configuring which messages to broadcast and their data rates, ensure you are not saturating the bus. The details depend on what other devices are present on the bus.

Configurable parameters

The following parameters can be customized. These settings are stored in non-volatile memory, and take effect after the device is restarted.

Node Id
Either hard-sets the node ID, or specifies the desired node ID to send to the ID allocator. See the *Dynamic node ID allocation* setting for details on this. Defaults to 70.

Dynamic node ID allocation
If set to true, node ID is determined by the DroneCAN dynamic node ID allocation process, and ID is 0 until assigned. (Broadcasts are anonymous, as defined in DroneCAN. In this case, the *Node ID* settings determines the desired ID to send to the allocator. If false, the *Node Id* settings is hard set as the ID. Defaults to true.

FD mode
Enable this to support frame-lengths up to 64 bytes. If disabled, only 8-byte frames are supported. You should only enable this if your flight controller supports and is configured to use FD mode. Defaults to disabled.

Note: If broadcasting AHRS, IMU data or fused positions at a high data rate, FD mode with a sufficiently high bit rate may be required.

CAN bit rate
Select the data bit rate to use. This has discrete settings available: 250kbps, 500kpbs, 1Mbps, 2Mbps, 4Mbps, and 5Mbps. You should only enable values higher than 1Mbps if your flight controller supports and is configured to use FD mode. Defaults to 1Mbps.

GNSS broadcast ratio
The device receives GNSS fixes at 10Hz. This ratio configures how often these packets are transmitted over CAN. A value of 1 transmits every fix. A value of 2 transmits every other fix. A value of 0 disables fix transmission. The default value of 1 is a good choice, unless there are bus saturation considerations.

Fused position broadcast rate
This controls how often the fused position solution is broadcast. It can be set between 0Hz and 1kHz. It defaults to 0 (disabled). Note that this message, especially at high data rates, uses a lot of bandwidth.

Barometer broadcast rate
This determines the rate to broadcast barometer data, in Hz. Can be set between 0 (disabled) and 255Hz. Note that the barometer only receives data at 32Hz, so setting a higher value than this is unecessary in many cases. Defaults to 32Hz.

Magnetometer broadcast rate

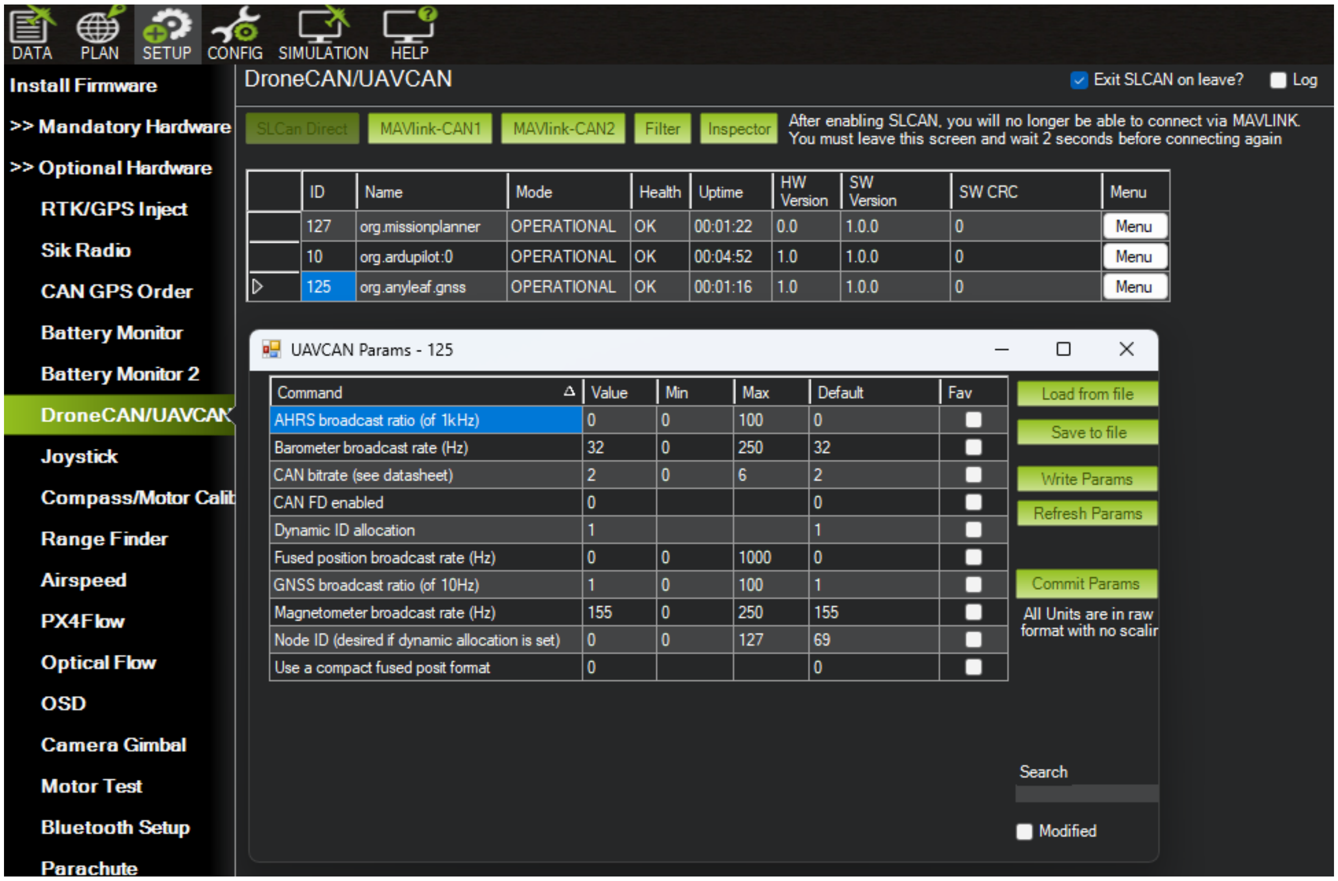
This determines the rate to broadcast magnetometer data, in Hz. Can be set between 0 (disabled) and 255Hz. Note that the barometer only receives data at 155Hz, so setting a higher value than this is unecessary in many cases. Defaults to 155Hz.

IMU data broadcast ratio
The device receives IMU data at 1kHz. This ratio configures how often these raw IMU data is transmitted over CAN. A value of 1 transmits every update A value of 2 transmits every other update A value of 0 disables transmission. It defaults to 10. (Sends updates at 100Hz) Note that if you don't plan to use IMU data, 0 is a good chioce.

Configuring and updating using Mission Planner or QGroundControl

It's possible to configure this device's settings using *Mission Planner*, *QgroundControl*, or any other software that implements DroneCAN's *Parameter Get Set* API.

This image shows the parameter set API in Mission Planner:



After enabling SLCAN, you will no longer be able to connect via MAVLINK. You must leave this screen and wait 2 seconds before connecting again

	ID	Name	Mode	Health	Uptime	HW Version	SW Version	SW CRC	Menu
	127	org.missionplanner	OPERATIONAL	OK	00:01:22	0.0	1.0.0	0	Menu
	10	org.ardupilot:0	OPERATIONAL	OK	00:04:52	1.0	1.0.0	0	Menu
▷	125	org.anyleaf.gnss	OPERATIONAL	OK	00:01:16	1.0	1.0.0	0	Menu

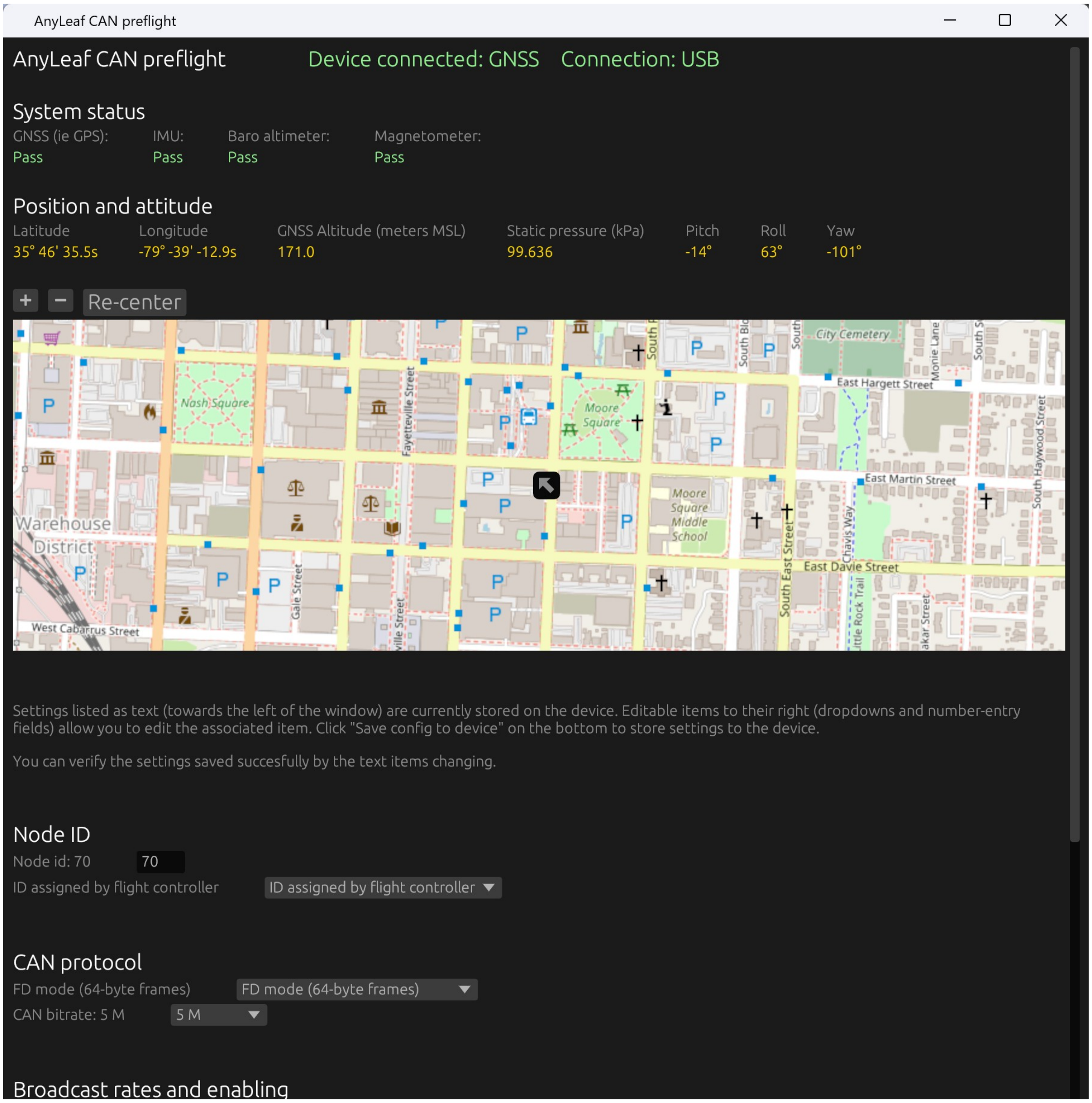
UAVCAN Params - 125

Command	Value	Min	Max	Default	Fav
AHRS broadcast ratio (of 1kHz)	0	0	100	0	☐
Barometer broadcast rate (Hz)	32	0	250	32	☐
CAN bitrate (see datasheet)	2	0	6	2	☐
CAN FD enabled	0			0	☐
Dynamic ID allocation	1			1	☐
Fused position broadcast rate (Hz)	0	0	1000	0	☐
GNSS broadcast ratio (of 10Hz)	1	0	100	1	☐
Magnetometer broadcast rate (Hz)	155	0	250	155	☐
Node ID (desired if dynamic allocation is set)	0	0	127	69	☐
Use a compact fused posit format	0			0	☐

All Units are in raw format with no scalir

Configuring and updating over USB

To configure, update, and view device status over USB, download and run the *AnyLeaf CAN Preflight* software, from the link on the Anyleaf website's page for this product. This lets you view and change settings, as well as view system status, as well as sensor readings including position and attitude.



AnyLeaf CAN preflight

AnyLeaf CAN preflight — Device connected: GNSS — Connection: USB

System status

GNSS (ie GPS):	IMU:	Baro altimeter:	Magnetometer:
Pass	Pass	Pass	Pass

Position and attitude

Latitude	Longitude	GNSS Altitude (meters MSL)	Static pressure (kPa)	Pitch	Roll	Yaw
35° 46' 35.5s	-79° -39' -12.9s	171.0	99.636	-14°	63°	-101°

Settings listed as text (towards the left of the window) are currently stored on the device. Editable items to their right (dropdowns and number-entry fields) allow you to edit the associated item. Click "Save config to device" on the bottom to store settings to the device.

You can verify the settings saved succesfully by the text items changing.

Node ID

Node id: 70 — 70

ID assigned by flight controller — ID assigned by flight controller ▼

CAN protocol

FD mode (64-byte frames) — FD mode (64-byte frames) ▼

CAN bitrate: 5 M — 5 M ▼

Broadcast rates and enabling

Configuring Ardupilot CAN settings

There are several CAN settings in ArduPilot that may need to be enabled or modified to make this device work. These are accessed by selecting the *Config* button at the top left of Mission Planner, then selected *Full Parameter List* from the menu at the left. You can then use the search window at the right to find these settings. The following settings are the most relevant:

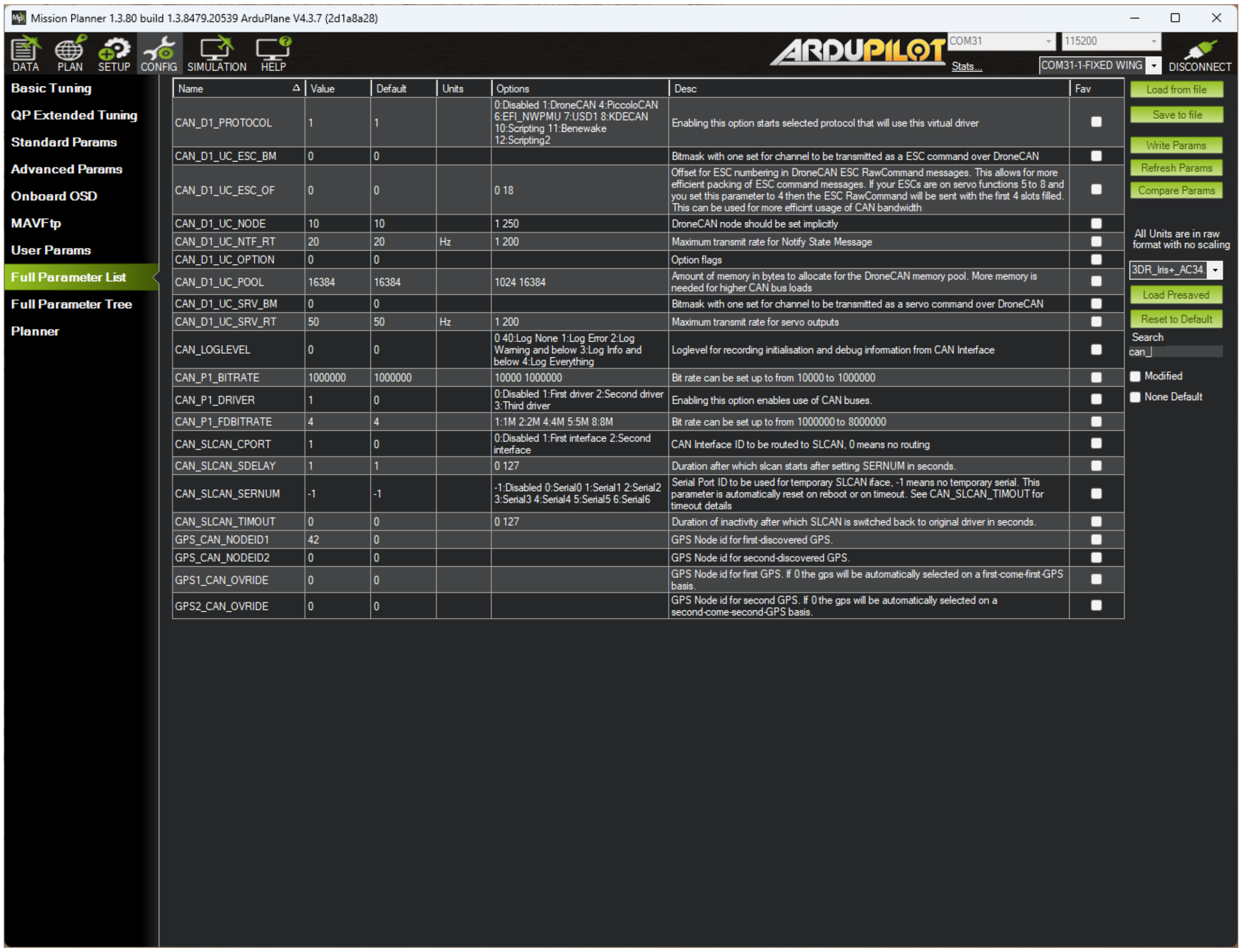
CAN_P1_DRIVER = 1 ("Enables use of CAN buses")
GPS_TYPE = 9 (DroneCAN)
GPS_AUTO_CONFIG = 2 (Auto config for Serial and DroneCAN)

If using FD mode or a different bit rate from the default, adjust these settings as required:

CAN_D1_UC_OPTION: enable FD (Value of 4, if no other flags here are set.)
CAN_P1_FDBITRATE: enum for FD bit rate. 1 for 1Mb. Defaults to 4 for 4mb. Select 5 for this device's maximum of 5Mbps.
CAN_P1_BITRATE = 1000000 Bitrate if using classic (non-FD) mode.

If troubleshooting, confirm CAN_D1_PROTOCOL = 1 (This selects DroneCAN as the CAN protocol). This should be set by default. You can set CAN_SLCAN_CPORT = 1 to enable reading CAN messages from a PC, eg using the DroneCAN GUI software.

The below image shows most of the ArduPilot CAN settings:



Name	Value	Default	Units	Options	Desc	Fav
CAN_D1_PROTOCOL	1	1		0:Disabled 1:DroneCAN 4:PiccoloCAN 6:EFI_NWPMU 7:USD1 8:KDECAN 10:Scripting 11:Benewake 12:Scripting2	Enabling this option starts selected protocol that will use this virtual driver	☐
CAN_D1_UC_ESC_BM	0	0			Bitmask with one set for channel to be transmitted as a ESC command over DroneCAN	☐
CAN_D1_UC_ESC_OF	0	0		0 18	Offset for ESC numbering in DroneCAN ESC RawCommand messages. This allows for more efficient packing of ESC command messages. If your ESCs are on servo functions 5 to 8 and you set this parameter to 4 then the ESC RawCommand will be sent with the first 4 slots filled. This can be used for more efficint usage of CAN bandwidth	☐
CAN_D1_UC_NODE	10	10		1 250	DroneCAN node should be set implicitly	☐
CAN_D1_UC_NTF_RT	20	20	Hz	1 200	Maximum transmit rate for Notify State Message	☐
CAN_D1_UC_OPTION	0	0			Option flags	☐
CAN_D1_UC_POOL	16384	16384		1024 16384	Amount of memory in bytes to allocate for the DroneCAN memory pool. More memory is needed for higher CAN bus loads	☐
CAN_D1_UC_SRV_BM	0	0			Bitmask with one set for channel to be transmitted as a servo command over DroneCAN	☐
CAN_D1_UC_SRV_RT	50	50	Hz	1 200	Maximum transmit rate for servo outputs	☐
CAN_LOGLEVEL	0	0		0 40:Log None 1:Log Error 2:Log Warning and below 3:Log Info and below 4:Log Everything	Loglevel for recording initialisation and debug information from CAN Interface	☐
CAN_P1_BITRATE	1000000	1000000		10000 1000000	Bit rate can be set up to from 10000 to 1000000	☐
CAN_P1_DRIVER	1	0		0:Disabled 1:First driver 2:Second driver 3:Third driver	Enabling this option enables use of CAN buses.	☐
CAN_P1_FDBITRATE	4	4		1:1M 2:2M 4:4M 5:5M 8:8M	Bit rate can be set up to from 1000000 to 8000000	☐
CAN_SLCAN_CPORT	1	0		0:Disabled 1:First interface 2:Second interface	CAN Interface ID to be routed to SLCAN, 0 means no routing	☐
CAN_SLCAN_SDELAY	1	1		0 127	Duration after which slcan starts after setting SERNUM in seconds.	☐
CAN_SLCAN_SERNUM	-1	-1		-1:Disabled 0:Serial0 1:Serial1 2:Serial2 3:Serial3 4:Serial4 5:Serial5 6:Serial6	Serial Port ID to be used for temporary SLCAN iface, -1 means no temporary serial. This parameter is automatically reset on reboot or on timeout. See CAN_SLCAN_TIMOUT for timeout details	☐
CAN_SLCAN_TIMOUT	0	0		0 127	Duration of inactivity after which SLCAN is switched back to original driver in seconds.	☐
GPS_CAN_NODEID1	42	0			GPS Node id for first-discovered GPS.	☐
GPS_CAN_NODEID2	0	0			GPS Node id for second-discovered GPS.	☐
GPS1_CAN_OVRIDE	0	0			GPS Node id for first GPS. If 0 the gps will be automatically selected on a first-come-first-GPS basis.	☐
GPS2_CAN_OVRIDE	0	0			GPS Node id for second GPS. If 0 the gps will be automatically selected on a second-come-second-GPS basis.	☐

For more information, reference the ArduPilot CAN setup documentation:
https://ardupilot.org/copter/docs/common-canbus-setup-advanced.html

FD CAN and Classic CAN selection

ArduPilot, PX4, and this device all default to using classic CAN; this is a good choice for compatibility. If any nodes on a given bus do not support FD mode, classic is the only viable option for that bus. If all devices support FD mode, selecting it, with the maximum bitrate supported by all nodes on the bus is the best option. Enabling FD mode with a high bitrate may be required if there are many devices on the bus, or a device is sending high-bandwidth data. This device, if configured to send AHRS or fused position data at a high rate, may saturate a low-bitrate bus.

This device supports FD mode, with a datarate up to 5Mbps. If all other devices on the bus support this, this is the recommended setting. Important: This device and any devices on the bus it communicates with (notably the flight controller) must be configured the same way in regards to FD mode vice classic mode, and datarate.

Updating firmware

This device’s firmware can be updated over USB. To do so, download firmware from the AnyLeaf website. Open the device’s lid.

Mounting on smaller frames

If you wish to mount this device on frames that do not accommodate the plastic enclosure, you can remove the bare circuit board and mount it directly. To remove the circuit board, press gently on the front and back of the lid, while pulling it away from the base. Once removed, the PCB can be removed from the enclosure by removing the 4 Phillips screws securing it to the base.

The base circuit board can be mounted using its 28mm-spaced M3 mounting holes, or by applying an adhesive or securing mechanism to its bottom surface.

Support

If you have any questions, or support requests, contact us by email: anyleaf@anyleaf.org.