Anyleaf Power module (CAN)

January 2024

General Description

The AnyLeaf power module accepts a battery input of up to 36V (For example, an 8S LiPo), and outputs power at 5V, and either 7.4V or 12V, selected by a jumper. Each of these two output lines can supply up to 3A of current. The maximum voltage of the backup battery is 26V(4S).

This device is intended to be used with small unmanned vehicles, but can be used more broadly. It powers electronics and servos, and periodically measures and broadcasts the following information over CAN:

- Main battery voltage
- Backup battery voltage and current
- Cell voltage for up to 6 battery cells
- Voltage and current on the 5V output
- Voltage and current on the 7.4V or 12V output
- Estimated battery life remaining, in time, and portion used
- Battery in use

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

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

Specifications

- **Dimensions:** 69 × 49 × 17 (height) mm. 92mm width with tabs
- **Mounting holes:** 2 × M4, spaced 80mm
- **Weight:** 48 grams
- **Power input:** 5 – 36V, via XT-60, XT-30, or USB.
- **Maximum input battery voltage:** 36V(8S)
- **Maximum input backup battery voltage:** 26V(6S)
- **Maximum individual cell voltage, if connected:** 4.5V
- **Power output 1:** 5V, via 2xJST-GH CAN connectors, or MR-30. 1 amp each.
- **Power output 2:** 7.4V or 12V, via an MR-30 connector. 3 amps.
- **Power output 2 selection:** 2-pin jumper, with 2.54mm spacing.
- **MCU:** STM32G431. 170Mhz Cortex-M4
- **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
- **Flight controller firmware compatibility:** Ardupilot, and PX4. Compatible with any firmware that supports the applicable *DroneCAN* message types.

Power output description

This device outputs power on two lines: A 5V line, and a 7.4V or 12V line. The 5V line is intended to power electronics via its CAN connection on one of the JST-GH ports. This can include flight controllers, GNSS devices, AHRS systems, and other sensors. A maximum of 1A is supported on each GH port. This line is shared with a MR30 connector, and the combined current draw on this line must remain below 3A. The MR30 is intended to power servos.

The 7.4V and 12V line is intended to power servos, and video transmission systems. An included 2.54mm, 2-pin jumper selects the voltage on this line. Leave unconnected for 7.4V, or connect it for 12V. Current on this line must not exceed 3A.

If the 5V line voltage measures bettween 4.8V and 5.2V, a *5V healthy* LED will illuminate green. If the second power line voltage measures between 7.2V and 7.6V, or between 11.7V and 12.3V, a *7V healthy* LED will illuminate green. Note that if the module is powered only by USB, it is likely that neither of these will light, due to power on the 5V line being potentially between 4.5V and 4.8V.

Backup battery description

An optional backup battery can be connected using the XT-30 connector. The maximum voltage supported on this battery is 26V (6S). If the primary battery voltage drops below 7V for whatever reason, this battery immediately takes over, with no power loss. The intent behind this behavior is to connect a lightweight secondary battery capable of powering electronics and servos. If the main battery is inadvertently disconnected, runs out of power, or otherwise has a fault, this can allow safe recovery of fixed-wing aircraft, and easier recovery by keeping systems other than propulsion online. If main battery power is restored, normal operation continues. An amber LED towards the top-left of the device indicates that the backup battery is in use. When not in use, it experiences minimal current draw.

Important: The backup battery must not exceed primary battery voltage. For example, if using a 6S main battery, use a 4S backup or lower. Otherwise, the backup battery may drain itself prematurely.

Cell voltage measurement

This device measures up to 6 individual battery cells. (It can't measure voltage on cells 7 and 8 due to space requirements). It connects to cells using a balance connector, with 2.54mm spacing. It is non-latching, to accommodate balance connectors of varying cell count. It assumes battery cells are connected in series. This connector is purely for measurement: It doesn't draw current from these pins. Make sure to connect the balance connector to this according to the markings next to it! Doing this incorrectly (Backwards, or shifted) can cause permanent damage to this device's CPU by overvolting its ADC pins!

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:

- Circuit status (uavcan.equipment.power.CircuitStatus)
- Power Supply Stats (uavcan.equipment.power.PrimaryPowerSupplyStatus)
- AnyLeaf power stats (Custom)
- 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	Destination node 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															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.

AnyLeaf power stats format. All voltages are in mV. All currents are in mA.

Bytes	Description	Data type
0-1	Primary battery voltage	16-bit unsigned integer
2-3	Backup battery voltage	16-bit unsigned integer
4-5	Voltage on the 5V line	16-bit unsigned integer
6-7	Voltage on the 7.4V or 12V line	16-bit unsigned integer
8-9	Battery cell 1 voltage	16-bit unsigned integer
10-11	Battery cell 2 voltage	16-bit unsigned integer
12-13	Battery cell 3 voltage	16-bit unsigned integer
14-15	Battery cell 4 voltage	16-bit unsigned integer
16-17	Battery cell 5 voltage	16-bit unsigned integer
18-19	Battery cell 6 voltage	16-bit unsigned integer
20-21	(unused)	16-bit unsigned integer
22-23	(unused)	16-bit unsigned integer
24-25	(unused)	16-bit unsigned integer
26-27	Backup battery current	16-bit unsigned integer
28-29	Current on the 5V line	16-bit unsigned integer
30-31	Current on the 7.4V or 12V line	16-bit unsigned integer
32	Estimated portion through of the primary battery, on a scale of 0 - 255	8-bit unsigned integer
33-34	Estimated time remaining on the main battery, in seconds	16-bit unsigned integer
35	Battery in use: 0 for primary, 1 for backup, 2 for none	8-bit unsigned integer

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)

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.

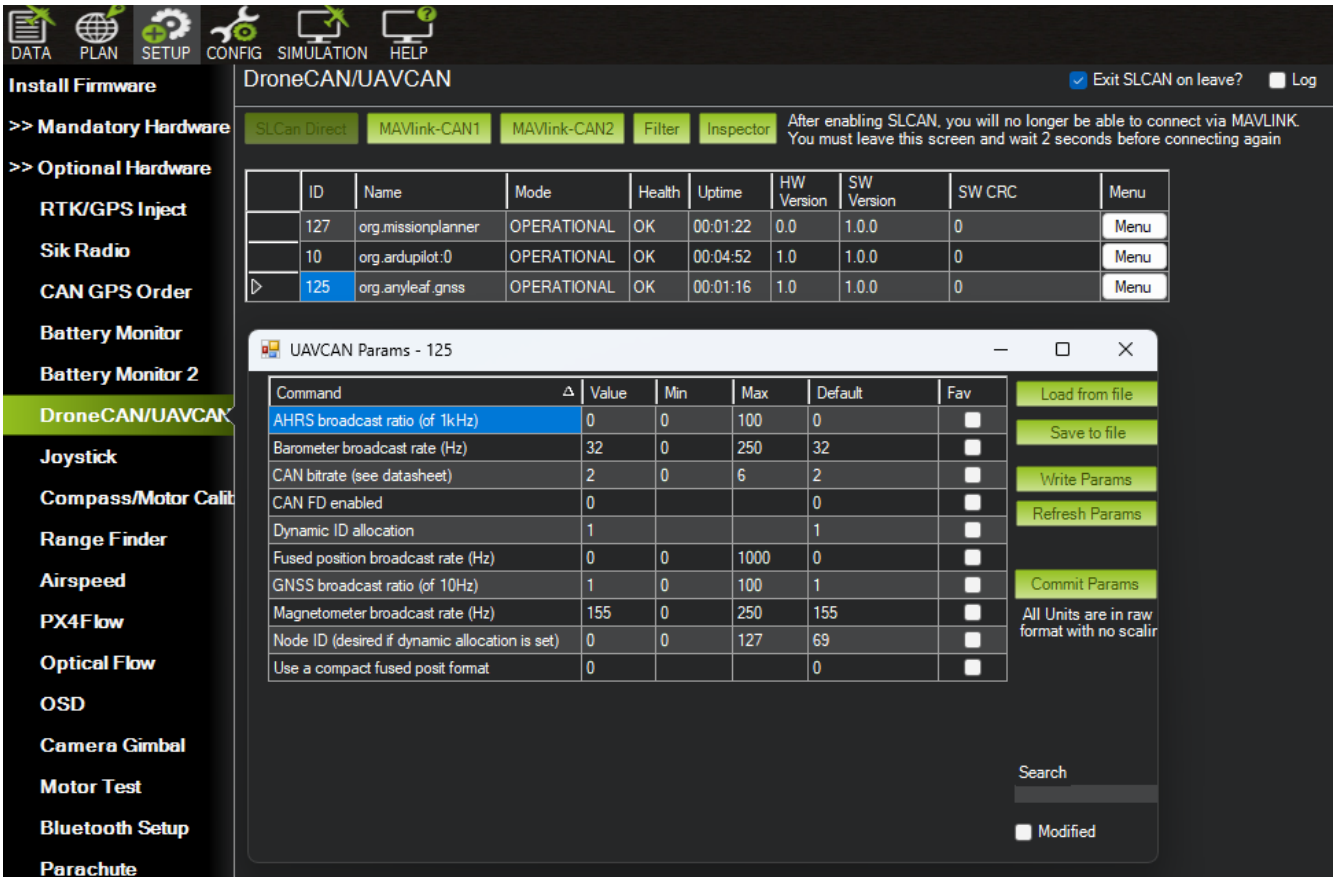
Note: If broadcasting AHRS, IMU data or fused positions at a high data rate, FD mode with a sufficiently high bit rate may be required.

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.

Configuring and updating using Mission Planner or QGroundControl

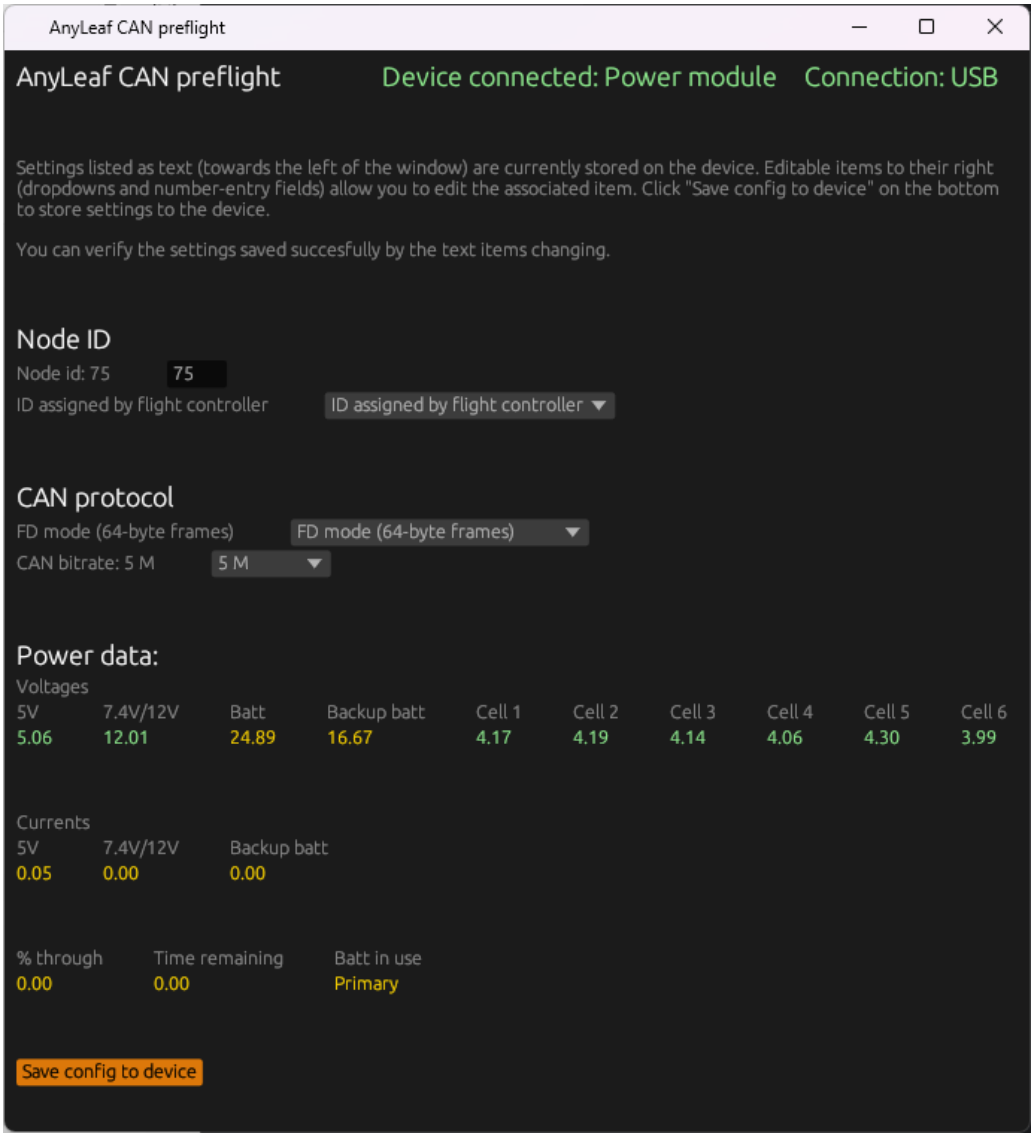
It's possible to configure this device's settings using *Mission Planner*, *QgroundControl*, or any other software that implements DroneCAN's *Parameter Get Set* API.

This image shows the parameter set API in Mission Planner:



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.



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:

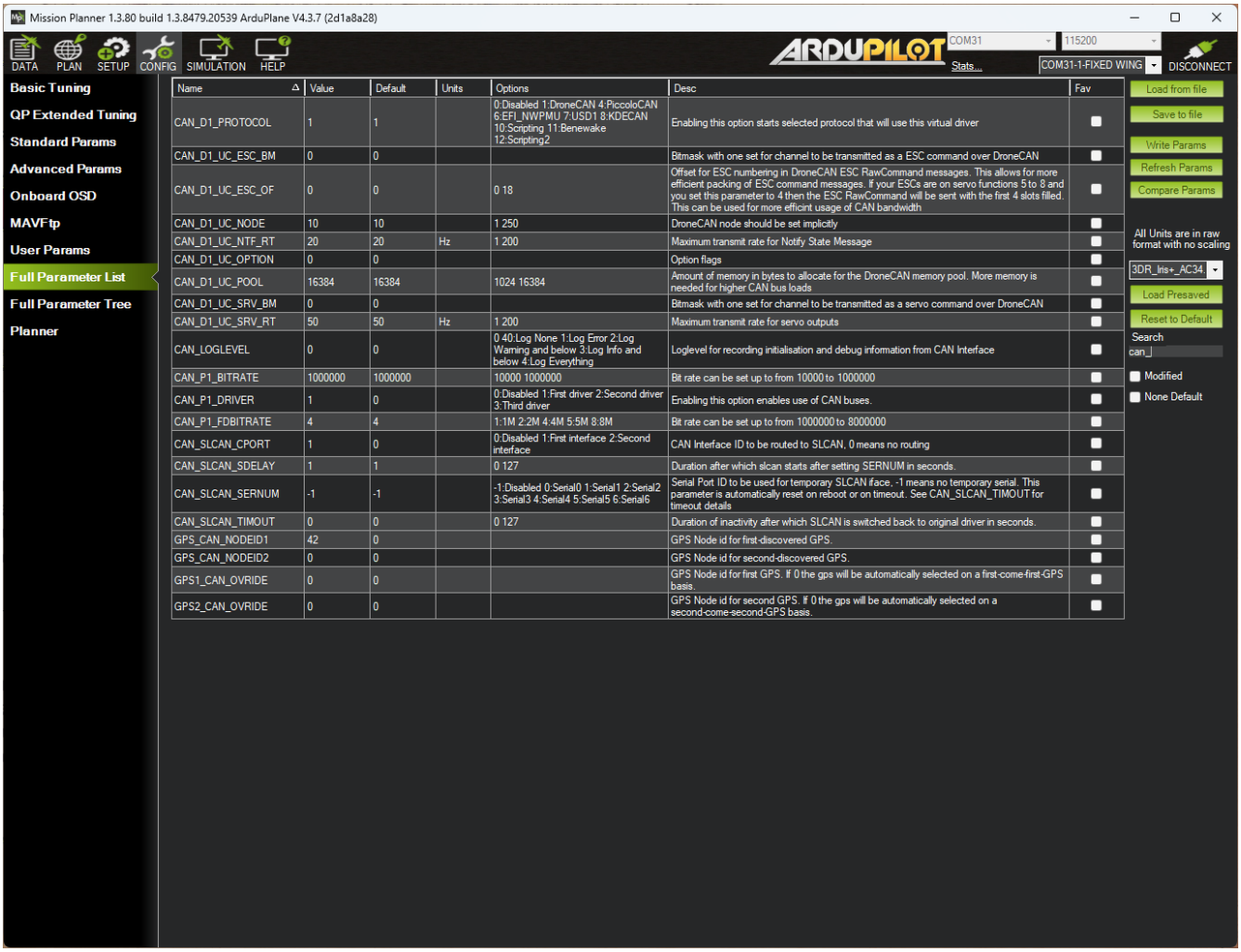
CAN_P1_DRIVER = 1 ("Enables use of CAN buses")
BAT_MONITORx = 9 (CAN; x is the battery monitor you wish to configure.)

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.