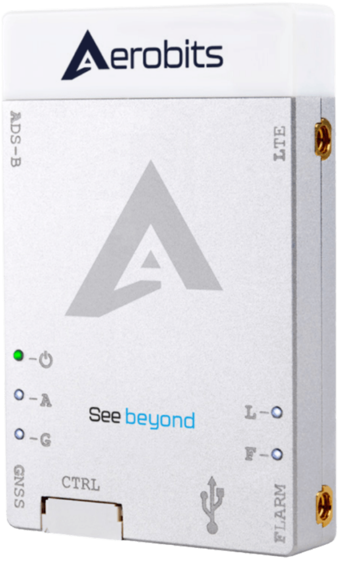
Aerobits

Subsystems for the
UAS intergration into
the airspace

The Hook-On-Device

Data sheet & User manual



Introduction

The Hook-On-Device (HOD) for UAS and other aircraft (such as helicopters) to transmit their own position data. Thanks to its low weight, the **HOD** can be attached to any aircraft.

It contains an **LTE** modem and a SIM card. The device transmits its current GNSS position via LTE to the UAS Traffic Management (UTM) system. The device is capable of receiving **FLARM** and **ADS-B** from surrounding air traffic and sending this data additionally to its own position to the UTM system. The UAS operator receives the UAS own position and the position data of other relevant air traffic in the vicinity via the web based UTM tracker. At the same time, the HOD transmits its position via FLARM (flight alarm). In this way, the aircraft also becomes visible to other airspace users in the vicinity who use FLARM.

For more information please contact: support@aerobits.pl.

Main features

- Connectivity: 4G LTE Cat1, GNSS, ADS-B, FLARM - wide band/multi-constellation/1090 MHZ/868MHz
- LTE modem to track aircraft via LTE
- Internal antennas: Built-in LTE antenna can be activated optionally
- External antennas: LTE, GNSS, FLARM and ADS-B antennas
- Compatible with all FLARM systems in aircraft and UAS
- Licensed FLARM transceiver (0.025 Watt output power) - broadcasting its own position
- Barometric sensor on board
- Programming via AT commands
- Simple plug&play integration

Contents

1 Technical parameters

1.1 Basic technical information

Parameter	Description	Typ.	Unit
FLARM frequencies	transmitter and receiver	868	MHz
FLARM range	highly dependent on antenna installation (also at other airspace users)		
ADS-B frequency	receiver	1090	MHz
ADS-B range	can be reduced by the user to limit data usage	250	km
Antenna orientation	GNSS oriented upwards (writing ”Pulse” facing up) LTE / FLARM / ADS-B - oriented downwards		
Temperature range	Operating temperature	-30 to +85	°C
Storage temperature	Optimal storage temperature	-5 to +40	°C
Dimension		58 x 38 x 9.5	mm
Weight	without antennas	35	grams
	with antennas	84	
	with USB and antennas	89	
	with all antennas and power bank	149	

Table 1: General technical parameters.

1.2 Electrical specification

1.2.1 Basic electrical parameters

Parameter	Value
External (power) connector	6 pin JST (power and data) or mirco USB (power and data), MMCX (antennas)
Power supply	5 V (external power supply, such as a power bank or aircraft’s power supply)
Power consumption	< 400 mA

Table 2: General electrical parameters.

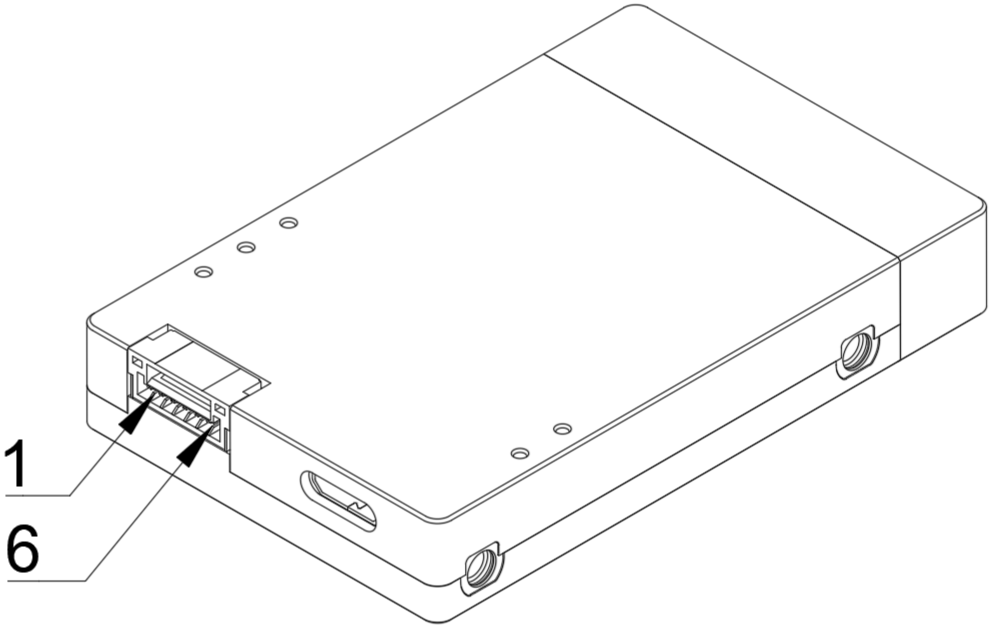


Figure 1: Appendant drawing of The Hook-On-Device .

1.2.2 PIN definition

PIN	Color	Name	Function
1	-	+5 V	Power supply
2	-	RX	MAVLink, AERO RXD
3	-	TX	MAVLink, AERO TXD
4	-	NC	Not connected
5	-	NC	Not connected
6	-	GND	Ground

Table 3: Pin definition.

1.2.3 LED indicators

LED	Color	Function
A (ADS-B)	White	Flashing – reception of 1090 MHz avionics frame (ADS-B)
F (FLARM)	White	Flashing – reception of valid FLARM frame (868 MHz)
L (LTE)	White	Flashing – LTE communication initialized Constant light – LTE communication in progress Off – No mobil network, wait or change position for better network coverage
G (GNSS)	White	Flashing – GNSS fixed Off – No GNSS fix, wait or change position for better satellite coverage
P (Power)	Green	Constant light - Power supply presence Off – No power, connect or recharge power source

Table 4: LED indicators.

1.3 Mechanical specification

1.3.1 Mechanical parameters

Parameter	Value
Dimensions	58 mm x 38 mm x 9.5 mm
Weight	35 g

Table 5: Mechanical parameters of The Hook-On-Device

1.3.2 Dimensions

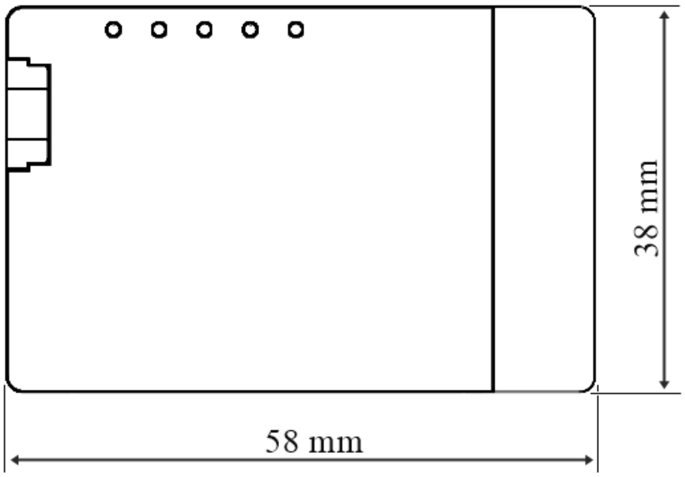


Figure 2: Dimensions of The Hook-On-Device

1.3.3 Connectors

Connector	Type	Example
Main	Installed on board	MOLEX, 47346-1001
	Mating connector	Common micro USB B connector
Antenna	Installed on board	MOLEX, 73415-0961
	Mating connector	ASMK025X174S11

Table 6: Connectors

2 Principle of operation

During work module goes through multiple states. In each state operation of the module is different. Each state and each transition is described in paragraphs below.

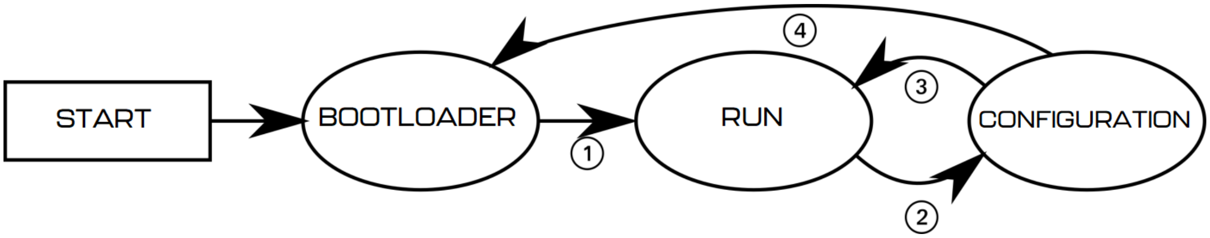


Figure 3: State machine of The Hook-On-Device

2.1 States of operation

2.1.1 BOOTLOADER state

This is an initial state of The Hook-On-Device after restart. Firmware update is possible here. Typically module transits automatically to RUN state. It is possible to lock module in this state (prevent transition to RUN state) using one of BOOTLOADER triggers. UART baud is constant and is set to 115200bps. After powering up module, it stays in this state for up to 3 seconds. If no BOOTLOADER trigger is present, module will transit to RUN state. Firmware upgrade is possible using Micro ADS-B App software. For automated firmware upgrading scenarios, aerobits_updater software is available. To acquire this program please contact: support@aerobits.pl.

2.1.2 RUN state

In this state module is working and receiving the data from aircrafts. It uses selected protocol to transmit received and decoded data to the host system. In this state of operation module settings are loaded from non-volatile internal memory, including main UART interface's baud.

2.1.3 CONFIGURATION state

In this mode change of stored settings is possible. Operation of the module is stopped and baud is set to fixed 115200bps. Change of settings is done by using AT-commands. Changes to settings are stored in non-volatile memory on exiting this state. Additional set of commands is also available in this state, allowing to e.g. reboot module into BOOTLOADER state, check serial number and firmware version. It is possible to lock module in this state (similarly to BOOTLOADER) using suitable command.

2.2 Transitions between states

For each of state transitions, different conditions must be met, which are described below. Generally, the only stable state is RUN. Module always tends to transit into this state. Moving to other states requires host to take some action.

2.2.1 BOOTLOADER to RUN transition

BOOTLOADER state is semi-stable: the module requires additional action to stay in BOOTLOADER state. The transition to RUN state will occur automatically after short period of time if no action will be taken. To prevent transition from BOOTLOADER state, one of following actions must be processed:

- Send `AT+LOCK=1` command while device is in BOOTLOADER state (always after power on for up to 3s)
- Send `AT+REBOOT_BOOTLOADER` command in CONFIGURATION state. This will move to BOOTLOADER state and will lock module in this state.

If none of above conditions are met, the module will try to transit into RUN state. Firstly it will check firmware integrity. When firmware integrity is confirmed, module will transit into RUN state, if not, it will stay in BOOTLOADER state.

To transit into RUN state:

- If module is locked, send `AT+LOCK=0` command

When module enters RUN mode it will send `AT+RUN_START` command.

2.2.2 RUN to CONFIGURATION transition

To transit from RUN into CONFIGURATION state, host should do one of the following:

- Send `AT+CONFIG=1` (using current baud).

When module leaves RUN state it sends `AT+RUN_END` message, then `AT+CONFIG_START` message on entering CONFIGURATION state. The former is sent using baud from settings, the latter always uses 115200bps baud.

2.2.3 CONFIGURATION to RUN transition

To transit from CONFIGURATION into RUN state, host should do one of the following:

- Send `AT+CONFIG=0` command.

When module leaves CONFIGURATION state it sends `AT+CONFIG_END` message, then `AT+RUN_START` message on entering RUN state. The former is always sent using 115200bps baud, the latter uses baud from settings.

2.2.4 CONFIGURATION to BOOTLOADER transition

To transit from CONFIGURATION into BOOTLOADER state, host should do one of the following:

- Send `AT+REBOOT_BOOTLOADER` command.
- Send `AT+REBOOT` and when module enters BOOTLOADER state, prevent transition to RUN state.

When entering the bootloader state, the module sends `AT+BOOTLOADER_START` .

3 UART configuration

Communication between module and host device is done using UART interface.

In CONFIGURATION and BOOTLOADER state transmission baud is fixed at 115200bps.

The UART interface uses settings as described in table 7.

UART Settings				
Parameter	Min.	Typ.	Max	Unit
Baud	-	115200	-	bps
Stop Bits Number	-	1	-	-
Flow Control	-	None	-	-
Parity Bit	-	None	-	-

Table 7: UART settings.

4 Settings

In RUN state, operation of the module is determined based on stored settings. Settings can be changed in CONFIGURATION state using AT-commands. Settings can be written and read.

NOTE: New values of settings are saved in non-volatile memory when transitioning from CONFIGURATION to RUN state.

Settings are restored from non-volatile memory during transition from BOOT do RUN state. If settings become corrupted due to memory fault, power loss during save, or any other kind of failure, the settings restoration will fail, loading default values and displaying the `AT+ERROR (Settings missing, loaded default)` message as a result. This behavior will occur for each device boot until new settings are written by the user.

4.1 Write settings

After writing a new valid value to a setting, an `AT+OK` response is always sent.

`AT+SETTING=VALUE`
For example `AT+PROTOCOL=1`
Response: `AT+OK`

4.2 Read settings

`AT+SETTING?`
For example: `AT+PROTOCOL?`
Response: `AT+PROTOCOL=1`

4.3 Settings description

`AT+SETTING=?`
For example: `AT+PROTOCOL=?`
Response:

```
Setting: PROTOCOL
       Description: Selected protocol (0: NONE, 2: CSV, 3: MAVLINK)
       Type: Integer decimal
       Range (min.): 0
       Range (max.): 5
       Is preserved: 1
       Is restart needed: 0
```

4.4 Errors

Errors are reported using following structure:
`AT+ERROR (DESCRIPTION)`
DESCRIPTION is optional and contains information about error.

4.5 Command endings

Every command must be ended with one of the following character sequences: “\n”, “\r” or “\r\n”. Commands without suitable ending will be ignored.

4.6 Uppercase and lowercase

All characters (except preceding `AT+`) used in command can be both uppercase and lowercase, so following commands are equal:

```
AT+PROTOCOL?
AT+pRoToCoL?
```

NOTE: This statement is true in configuration state, not in bootloader state. in bootloader state all letters must be uppercase.

4.7 Available settings

Setting	Min	Max	Def	Comment
`GNSS_LOG`	0	2	0	GNSS NMEA forwarding 0 - No forwarding 1 - RMC Messages only 2 - All
`FLARM_LOG`	0	2	0	FLARM debugging (1 - basic, 2 - detailed)
`FLARM_TX`	0	1	1	Flarm TX enabled
`FLARM_LOG`	0	2	0	Flarm debugging (1 - basic, 2 - detailed)
`FLARM_AIRCRAFT_TYPE`	0	15	0	Flarm aircraft type (0 - UNKNOWN, 1 - GLIDER, 2 - TOW-PLANE, 3 - HELICOPTER, 4 - PARACHUTE, 5 - DROP-PLANE, 6 - FIXED_HG, 7 - SOFT_HG, 8 - ENGINE, 9 - JET, 10 - UFO, 11 - BALLOON, 12 - AIRSHIP, 13 - UAV, 15 - STATIC)
`TT_LOG`	0	1	0	Show TT log
`LTE_LOG`	0	1	0	Show LTE log
`LTE_EXTERNAL_ANTENNA`	0	1	0	Enable external LTE ANTENA
`APN_NAME`	-	-	-	LTE APN name
`APN_USER`	-	-	-	LTE APN user name
`APN_PASSWORD`	-	-	-	LTE APN user password
`MQTT_LOG`	0	1	0	Show MQTT log
`MQTT_TLS`	0	1	0	Enables TLS in MQTT
`MQTT_BROKER_ADDRESS`	-	-	-	MQTT broker address
`MQTT_BROKER_PORT`	0	65535	0	MQTT broker port
`MQTT_USER`	-	-	-	MQTT user name
`MQTT_PASS`	-	-	-	MQTT broker password
`MQTT_CLIENT_ID`	-	-	-	MQTT client ID, left blank to use IMSI
`MQTT_KEEPALIVE`	60	3600	60	MQTT keepalive interval
`MQTT_WEBSOCKET`	0	1	0	Use websocket with MQTT stream (0 - disable, 1 - enable)
`FILTER_DISTANCE`	0	700000	0	Range of distance filter (Meters) (0 - Disabled), parameter works only when HOD has GNSS signal
`AUX_DATA_TOPIC`	-	-	-	Topic for transmitting data received by aux port
`AERO_JSON_BITMASK`	0	3F	3B	Determine, what data will be sent over AeroJson protocol 0x01 - ADSB 0x02 - FLARM 0x04 - RAW 0x08 - STATUS 0x10 - GNSS 0x20 - SENSOR
`AERO_JSON_TOPIC`	-	-	-	AeroJson MQTT topic
`UAS_PAYLOAD`	-	-	-	UAS Payload nature/type
`UAS_PRIORITY`	-	-	-	Urgency of UAS operation (low, medium, high)
`UAS_RECOVERY_CAPS`	-	-	-	UAS Recovery capabilities
`UAS_ID`	-	-	-	UAS identification code
`PRESSURE_LOG`	0	1	0	Show barometer log

Table 8: Settings

Setting	Min	Max	Def	Comment
`OTA_URL`	-	-	-	URL to server with JSON configuration file. Text "sn" inside URL will be replace with device Serial Number. Empty URL mean that OTA feature is disable.
`OTA_INTERVAL`	0	1000	0	Time interval in hours, in which device will check new firmware on server.Value "O" means that OTA will be performed only during boot.
`OTA_USERNAME`	-	-	-	User name required for authorization on server.
`OTA_PASSWORD`	-	-	-	User password required for authorization on server.

Table 9: OTA Settings. For further information, contact: support@aerobits.pl

4.8 Example

As an example, to switch The Hook-On-Device module to CSV protocol, one should send following commands. "<<" indicates command sent to module, ">>" is a response.

```
<< AT+CONFIG=1\r\n
>> AT+OK\r\n
<< AT+PROTOCOL=2\r\n
>> AT+OK\r\n
>> AT+OK\r\n
<< AT+CONFIG=0\r\n
```

5 Commands

Apart from settings, module supports set of additional commands. Format of this commands are similar to those used for settings, but they do not affect operation of module in RUN state.

5.1 Commands in BOOTLOADER and CONFIGURATION state

5.1.1 AT+LOCK

`AT+LOCK=1` - Set lock to enforce staying in BOOTLOADER or CONFIGURATION state
`AT+LOCK=0` - Remove lock
`AT+LOCK?` - Check if lock is set

5.1.2 AT+BOOT

`AT+BOOT?` - Check if module is in BOOTLOADER state

Response:

`AT+BOOT=0` - module in CONFIGURATION state
`AT+BOOT=1` - module in BOOTLOADER state

5.2 Commands in CONFIGURATION state

5.2.1 AT+CONFIG

`AT+CONFIG=0` - Transition to RUN state.
`AT+CONFIG?` - Check if module is in CONFIGURATION state.

Response:

`AT+CONFIG=0` - module in RUN state
`AT+CONFIG=1` - module in CONFIGURATION state

5.2.2 AT+SETTINGS?

`AT+SETTINGS?` - List all settings. Example output:

```
AT+PROTOCOL=2
AT+SUBPROTOCOL=0
AT+BAUDRATE=0
```

5.2.3 AT+HELP

`AT+HELP` - Show all settings and commands with descriptions. Example output:

```
SETTINGS:
AT+PROTOCOL=2          [Selected protocol (0: NONE, 2: CSV, 3: MAVLINK)]
AT+SUBPROTOCOL=0       [Subprotocol of selected protocol]
COMMANDS:
AT+HELP                [Show this help]
AT+TEST                [Responds "AT+OK"]
AT+SETTINGS_DEFAULT    [Load default settings]
AT+REBOOT              [Reboot system]
```

5.2.4 AT+SETTINGS_DEFAULT

`AT+SETTINGS_DEFAULT` - Set all settings to their default value.

5.2.5 AT+SERIAL_NUMBER

`AT+SERIAL_NUMBER?` - Read serial number of module.

Response:

```
AT+SERIAL_NUMBER=07-0001337
```

5.2.6 AT+FIRMWARE_VERSION

`AT+FIRMWARE_VERSION?` - Read firmware version of module.

Response:

```
AT+FIRMWARE_VERSION=10101017(May 11 2018)
```

5.2.7 AT+REBOOT

`AT+REBOOT` - Restart module.

5.2.8 AT+REBOOT_BOOTLOADER

`AT+REBOOT_BOOTLOADER` - Restart module to BOOTLOADER state.

NOTE: This command also sets lock.

5.3 Commands in RUN state

`AT+CONFIG=1` - transition to CONFIGURATION state.

NOTE: This command also sets lock.

6 Protocols

6.1 Network communication modes

HOD communicates through the network using MQTT 3.1 protocol. Connection can be configured to use username and password authentication, as well as TLS encryption. All data is transmitted into single MQTT topic.

6.2 JSON Protocol

HOD uses JSON compliant with RFC 8259 to transmit its messages. Each message is encoded as separate JSON object, without any excess whitespace, consisting of fields described in table 10. If multiple objects are sent in same message, they are concatenated without any delimiter.

Name	Description	Value type
`{`		
`"src": "23-0000001",`	HOD's serial number.	String
`"ts": 69061337,`	Timestamp in milliseconds, relative to last UTC midnight. Value `69061337` encodes 19:11:01.337. Omitted if unknown.	Unsigned integer
`"ver": 1,`	JSON protocol version. See details below.	Unsigned integer
`"gnss": {...}`	One or more of the data fields, described in subchapters below.	Object or array
`}`		

Table 10: Description of main JSON fields.

NOTE: The order of JSON object fields in any part of message may vary between firmware revisions and messages.

Some JSON objects have fields, of which values may sometimes be unknown. In this case, they are skipped in JSON output. In following chapters, each of those fields are explicitly marked as ommitable.

NOTE: In case of JSON objects consisting of only ommitable fields, if none of them are set, the whole object may be omitted.

The `"ver"` field indicates JSON protocol version. Future ICD versions may introduce additional fields without changing the version number. If a breaking change occurs in HOD JSON specification, the version number is guaranteed to be incremented.

NOTE: The version number of JSON protocol described in this document is 1.

6.2.1 Status section

The `"status"` section contains status information related to HOD itself. The example JSON message with this section fields described, is shown in table 11.

JSON field	Description	Value type
`{`		
`"src": "23-0000001",`	see table 10.	
`"ts": 69061337,`		
`"ver": 1,`		
`"status": {`		
`"fw": "30903679(Jan 15 2021)",`	Firmware version, with same syntax as `AT+FIRMWARE_VERSION` command. Value `30903679` is version 3.9.3.679.	String
`"simcom_fw": "LE20B03SIM7600M21-A",`	SIMCOM LTE module firmware versionn.	String
`"imsi": "240074231736782",`	SIM card IMSI.	String
`"network": {`	Mobile network connection information.	
`"type": "4G",`	Network type. Omitted if unknown.	String
`"protocol": "LTE",`	Mobile network protocol. Omitted if unknown.	String
`"rssnr": 11,`	Reference Signal Signal to Noise Ratio, in dB. Available only in 4G. Omitted if unknown.	Unsigned integer
`"rsrp": -90.0,`	Reference Signal Received Power, in dBm. Available only in 4G. Omitted if unknown.	Floating point
`"rsrq": -12.4,`	Reference Signal Received Quality, in dB. Available only in 4G. Omitted if unknown.	Floating point
`"cid": 4327240`	Cell ID. Omitted if unknown	Unsigned integer
`"ecio": -8`	Downlink Carrier to Interference ratio. Available only in 3G. Omitted if unknown.	Floating point
`"rscp": -85`	Received Signal Code Power. Available only in 3G. Omitted if unknown.	Signed integer
`"rssi": -75`	Received Signal Strength Indicator, in dBm. Available only in 4G. Omitted if unknown.	Floating point
`"rxlev": 15`	Reception Level. Available only in 2G and 3G. Omitted if unknown.	Signed integer
`},`		
`"uas": {`	ES-282 UAS status/identification fields.	
`"id": "MFR1C123456789ABC",`	UA Identification field. Omitted if set to empty value.	String

`"gnd": 0,`	Set to 1 if vehicle is on ground, otherwise 0.	Unsigned integer
`"payload": "meds",`	Payload nature/type. Omitted if set to empty value.	String
`"priority": "low",`	Urgency of UA operation. Omitted if value is unknown. Allowed values: `low`, `medium`, `high`.	String
`"recoveryCap": "parachute"`	UA Emergency recovery capability. Omitted if set to empty value.	String
`},`		
`"hw": {`	ES-282 hardware/subsystem status fields.	
`"ua": "nom",`	General UA status. Omitted if value is unknown. Possible values: - `nom`: nominal, - `deg`: degraded, - `emerg`: emergency	String
`"rps": "nom",`	RPS status. Omitted if value is unknown. Possible values: - `nom`: nominal, - `deg`: degraded, - `emerg`: emergency	String
`"c2": "nom",`	C2 link status. Omitted if value is unknown. Possible values: - `nom`: nominal, - `deg`: degraded, - `lost`: total loss of capability	String
`"fcs": "nom",`	FCS status. Omitted if value is unknown. Possible values: - `nom`: nominal, - `deg`: degraded, - `lost`: total loss of capability	String
`"engine": "nom",`	Engine status. Omitted if value is unknown. Possible values: - `nom`: nominal, - `deg`: degraded, - `lost`: engine failure detected	String
`"power": "nom",`	Power status. Omitted if value is unknown. Possible values: - `nom`: nominal, - `deg`: degraded, - `low`: low power	String
`"daaCoop": "nom",`	Cooperative DAA status. Omitted if value is unknown. Possible values: - `nom`: nominal, - `deg`: degraded, - `lost`: total loss of capability	String

`"daaNonCoop": "nom",`	Non-Cooperative DAA status. Omitted if value is unknown. Possible values: - `nom`: nominal, - `deg`: degraded, - `lost`: total loss of capability	String
`"trajectory": "nom",`	Trajectory status. Omitted if value is unknown. Possible values: - `nom`: nominal, - `deconf`: De-confliction, - `avoid`: Collision avoidance - `emerg`: Following trajectory from emergency procedure, - `ooc`: Out of control	String
`}`		
`}` `}`		

Table 11: Descriptions of JSON sensor section fields.

6.2.2 GNSS section

The `"gnss"` section contains basic GNSS information. This message is sent once per second. The example JSON message with `"gnss"` section fields described, is shown in table 12.

JSON field	Description	Value type
`{`		
`"src": "23-0000001",`	see table 10.	
`"ts": 69061337,`		
`"ver": 1,`		
`"gnss": {`		
`"fix": 1,`	Set to 1 if onboard GNSS currently has fix, otherwise 0.	Unsigned integer
`"lat": 53.42854,`	Last known latitude. Omitted if there was no GNSS fix since device boot.	Floating point
`"lon": 14.55281,`	Last known longitude. Omitted if there was no GNSS fix since device boot.	Floating point
`"altWgs84": 499.6,`	Last known WGS-84 Altitude, in meters. Omitted if there was no GNSS fix since device boot.	Floating point
`"altMsl": 508.6,`	Last known MSL Altitude, in meters. Omitted if there was no GNSS fix since device boot.	Floating point
`"track": 127.3,`	Track angle, 0°..360°, relative to true north. Omitted if unknown.	Floating point
`"hVelo": 10.5,`	Horizontal velocity, in knots. Omitted if unknown.	Floating point
`"vVelo": 25.00,`	Vertical velocity, in m/s. Positive value is upwards. Omitted if unknown.	Floating point
`"gndSpeed": [`		
`5.2, 2.1`	Ground speed in east-west and north-south axes respectively, in knots. Positive value is East and North. Derived from `track` / `hVelo` values. Omitted if unknown.	Floating point
`],`		
`"acc": {`		
`"lat": 5.2,`	Accuracy of latitude, in meters. Omitted if unknown.	Floating point
`"lon": 2.1,`	Accuracy of longitude, in meters. Omitted if unknown.	Floating point
`"alt": 3.6`	Accuracy of altitude, in meters. Omitted if unknown.	Floating point
`},`		
`"nacp": 12`	Navigational Accuracy Category for Position value, as defined in ED-282. Omitted if unknown.	Unsigned integer
`"nacv": 2`	Navigational Accuracy Category for Velocity value, as defined in ED-282. Omitted if unknown.	Unsigned integer
`"nic": 12`	Navigation Integrity Category as defined in ED-282. Omitted if unknown.	Unsigned integer
`}` `}`		

Table 12: Descriptions of JSON GNSS section fields.

NOTE: The `nacp`, `nacv` and `nic` values are not available in regular HOD hardware.

6.2.3 Sensor section

The `"sensor"` section contains values acquired from miscellaneous sensors present in HOD hardware and consists of fields shown in table 13. This message is sent once per second. All fields are optional - they are sent only if appropriate sensor is enabled.

JSON field	Description	Value type
`{`		
`"src": "23-0000001",`	see table 10.	
`"ts": 69061337,`		
`"ver": 1,`		
`"sensor": {`		
`"pressure": 1013.25,`	Current pressure sensor value, in hPa.	Floating point
`"altQfe": 500.0,`	Barometric altitude (QFE). The reference value is sampled on issuing the takeoff command. Omitted if unknown or on ground.	Floating point
`"temp": 20.5,`	Current temperature sensor value, in degrees Celsius.	Floating point
`}` `}`		

Table 13: Descriptions of JSON sensor section fields.

6.2.4 Raw ADS-B section

The `"raw"` section contains raw, unprocessed and unfiltered ADS-B frames gathered by HOD, which can be used e.g. for multilateration and other low-level analysis. Raw messages are encoded as JSON array with at least one entry. Each array entry is a separate array containing values as described in table 14

JSON field	Description	Value type
`{`		
`"src": "23-0000001",`	see table 10.	
`"ts": 69061337,`		
`"ver": 1,`		
`"raw": [`		
`[`		
`"18A9725A4C842D",`	Raw frame bytes, formatted as uppercase hexadecimal. Short Mode-S frames encode 7 bytes, long frames contain 14 bytes.	String
`696,`	Signal strength, in mV.	Unsigned integer
`68,`	Signal quality, in mV.	Unsigned integer
`"295CAB573A77"`	UTC-calibrated time of reception, formatted as uppercase hexadecimal, in nanoseconds. Example translates to 12:37:57.988350583	String
`]`		

`]`
`}`		

Table 14: Descriptions of JSON raw ADS-B section fields.

NOTE: Due to constrained throughput of HOD communication, transmission of some raw frames may be skipped in heavy aircraft traffic situations.

6.2.5 Processed ADS-B reports

The `"adsb"` section contains aircraft information determined by HOD's internal ADS-B processing engine. The messages are encoded as JSON array with at least one entry. Each entry is an object consisting of fields denoted in table 15. Reports for each ADS-B aircraft are updated once every second.

JSON field	Description	Value type
`{`		
`"src": "23-0000001",`	see table 10.	
`"ts": 69061337,`		
`"ver": 1,`		
`"adsb": [`		
`{`		
`"icao": "DABABE",`	ICAO address, 24-bit value encoded in uppercase hexadecimal, with leading zeros.	String
`"sigStr": 696,`	Signal strength.	Unsigned integer
`"sigQ": 68,`	Signal quality.	Unsigned integer
`"fps": 5,`	Number of raw Mode-S frames received from aircraft during last second.	Unsigned integer
`"lat": 53.42854,`	Latitude. Omitted if position is unknown.	Floating point
`"lon": 14.55281,`	Longitude. Omitted if position is unknown.	Floating point
`"baroAlt": 1725,`	Barometric altitude, in feet. Omitted if unknown.	Signed integer
`"geoAlt": 1712,`	Geometric altitude, in feet. Omitted if unknown.	Signed integer
`"track": 72.18,`	Track angle, -180°..180°. Omitted if unknown.	Floating point
`"hVelo": 10.5,`	Horizontal velocity, in knots. Omitted if unknown.	Floating point
`"vVelo": 50,`	Vertical velocity, in ft/min, positive value is upwards. Omitted if unknown.	Signed integer
`"ident": "TEST8",`	Callsign, up to 8 chars. Omitted if unknown.	String
`"squawk": "7232",`	Squawk, 8 octal digits. Omitted if unknown.	String
`"ecat": 13,`	Emitter category code, see table 16. Omitted if unknown.	Unsigned integer
`"nacp": 3,`	NAC_P value, as described in ED-102A. Omitted if value is 0 (unknown).	Unsigned integer
`"nacv": 1,`	NAC_V value, as described in ED-102A. Omitted if value is 0 (unknown).	Unsigned integer
`"nicBaro": 1,`	NIC_BARO value, as described in ED-102A. Omitted if value is 0.	Unsigned integer
`"nic": 2,`	NIC value, as described in ED-102A. Omitted if value is 0 (unknown).	Unsigned integer
`"surf": 1`	Set to 1 if plane is on earth surface. Omitted if plane is in air or unknown.	Unsigned integer
`}`		

`]`
`}`		

Table 15: Descriptions of JSON ADS-B section fields.

The emitter category values returned in `"ecat"` field is shown in table 16.

`"ecat"` value	Description
0	Unknown.
1	Light (below 15500 lbs.).
2	Small (15500 - 75000 lbs.).
3	Large (75000 - 300000 lbs.).
4	High-Vortex Large (aircraft such as B-757).
5	Heavy (above 300000 lbs.).
6	High performance (above 5g acceleration and above 400 knots).
7	Rotorcraft.
8	Reserved.
9	Glider, Sailplane.
10	Lighter-Than-Air.
11	Parachutist, Skydiver.
12	Ultralight, hang-glider, paraglider.
13	Reserved.
14	Unmanned Aerial Vehicle.
15	Space, Trans-atmospheric Vehicle.
16	Reserved.
17	Surface Vehicle - Emergency Vehicle.
18	Surface Vehicle - Service Vehicle.
19	Point Obstacle (includes Tethered Balloons).
20	Cluster obstacle.
21	Line obstacle.

Table 16: ADS-B emitter category values in JSON protocol.

6.2.6 Processed FLARM reports

The `"flarm"` section contains aircraft information determined by HOD's internal FLARM processing engine. The messages are encoded as JSON array with at least one entry. Each entry is an object consisting of fields denoted in table 17. Reports for each FLARM aircraft are updated once every second.

JSON field	Description	Value type
`{`		
`"src": "23-0000001",`	see table 10.	
`"ts": 69061337,`		
`"ver": 1,`		
`"flarm": [`		
`{`		
`"idType": 1,`	Aircraft id type. 0: randomized, 1: ICAO, 2: FLARM.	Unsigned integer
`"id": "DABABE",`	Aircraft id, 32-bit value encoded in uppercase hexadecimal, with leading zeros.	string
`"type": 13,`	Aircraft type, see table 18.	Unsigned integer
`"danger": 1,`	Danger level (1-3). Omitted if 0 (no danger)	Unsigned integer
`"lat": 53.42854,`	Latitude.	Floating point
`"lon": 14.55281,`	Longitude.	Floating point
`"alt": 525,`	Barometric altitude, in meters.	Signed integer
`"track": 72`	Track angle, in degrees.	Unsigned integer
`"hVelo": 50,`	Horizontal velocity, in m/s.	Unsigned integer
`"vVelo": 200,`	Vertical velocity, in cm/s.	Unsigned integer
`"movMode": 5,`	Movement mode. 1: stationary, 4: circling right, 5: flying, 7: circling left.	Unsigned integer
`"stealth": 1,`	Set to 1 if target has Stealth flag set, otherwise omitted.	Unsigned integer
`"notrack": 1`	Set to 1 if target has Notrack flag set, otherwise omitted.	Unsigned integer
`}`		

`]`
`}`		

Table 17: Descriptions of JSON FLARM section fields.

The list of possible FLARM “Aircraft type” values returned in `"type"` field is shown in table 18.

`"type"` value	Description
0	Reserved.
1	Glider, Motor glider.
2	Tow plane, tug plane.
3	Helicopter, gyrocopter, rotocraft.
4	Skydiver, parachute.
5	Drop plane for skydivers.
6	Hang glider (hard).
7	Hang glider (soft).
8	Aircraft with reciprocating engine.
9	Aircraft with jet / turboprop engine.
10	Reserved.
11	Balloon (hot, gas, weather, static).
12	Airship, blimp, zeppelin.
13	Unmanned Aerial Vehicle (UAV).
14	Reserved.
15	Static obstacle.

Table 18: FLARM Aircraft type values in JSON protocol.

6.3 MAVLink protocol

The Hook-On-Device can be switched to use MAVLink protocol. This can be achieved by altering PROTOCOL setting. When MAVLink protocol is used, module is sending list of aircrafts every second. MAVLink messages have standarized format, which is well described on official protocol webpage (mavlink.io/en/messages).

6.3.1 ADS-B Aircraft message

Aircrafts are encoded using ADSB_VEHICLE message (mavlink.io/en/messages/common.html#ADSB_VEHICLE). MAVLink message contains several data fields which are described below.

Field Name	Type	Description
`ICAO_address`	`uint32_t`	ICAO address
`lat`	`int32_t`	Latitude, expressed as degrees * 1E7
`lon`	`int32_t`	Longitude, expressed as degrees * 1E7
`altitude`	`int32_t`	Barometric/Geometric Altitude (ASL), in millimeters
`heading`	`uint16_t`	Course over ground in centidegrees
`hor_velocity`	`uint16_t`	The horizontal velocity in centimeters/second
`ver_velocity`	`uint16_t`	The vertical velocity in centimeters/second, positive is up
`flags`	`uint16_t`	Flags to indicate various statuses including valid data fields
`squawk`	`uint16_t`	Squawk code
`altitude_type`	`uint8_t`	Type from ADSB_ALTITUDE_TYPE enum
`callsign`	`char[9]`	The callsign, 8 chars + `NULL`
`emitter_type`	`uint8_t`	Type from ADSB_EMITTER_TYPE enum
`tslc`	`uint8_t`	Time since last communication in seconds

Table 19: MAVLink ADSB_VEHICLE message description

The ADS-B vehicle may transmit barometric, as well as geometric altitude. The SUBPROTOCOL setting allows for toggling altitude transmit priority:

- When set to 0, altitude field will be filled with geometric altitude first. If not available, barometric altitude will be used.
- When set to 1, barometric altitude wil be preferred.

6.3.2 FLARM Aircraft message

Aircrafts reported by FLARM use ADSB_VEHICLE message in same format as described in ADS-B Aircraft message section, with following restrictions:

- The FLARM “Aircraft Type” field is translated to MAVLink “Emitter Category” field as shown in table 20.
- ICAO field contains FLARM id value.

Aircraft Type index	Aircraft Type description	Emitter Category index	Emitter Category description
0	Reserved	0	No information
1	Glider, Motor glider	9	Glider
2	Tow plane, tug plane	1	Light
3	Helicopter, gyrocopter, rotocraft	7	Rotorcraft
4	Skydiver, parachute	11	Parachute
5	Drop plane for skydivers	1	Light
6	Hang glider (hard)	12	Ultra light
7	Hang glider (soft)	12	Ultra light
8	Aircraft with reciprocating engine	1	Light
9	Aircraft with jet / turboprop engine	3	Large
10	Reserved	0	No information
11	Balloon (hot, gas, weather, static)	10	Lighter than air
12	Airship, blimp, zeppelin.	10	Lighter than air
13	Unmanned Aerial Vehicle (UAV)	14	UAV
14	Reserved	0	No information
15	Static obstacle	0	No information

Table 20: FLARM Aircraft Type to Emitter Category translation table

6.3.3 FLARM Collision message

Apart from ADS-B messages, FLARM subsystem emits COLLISION messages (Mavlink documentation). Detailed information about given aircraft can be obtained from ADSB_VEHICLE message directly preceding given COLLISION message.

NOTE: The Mavlink protocol is sent via the JST connector.

7 Quick start

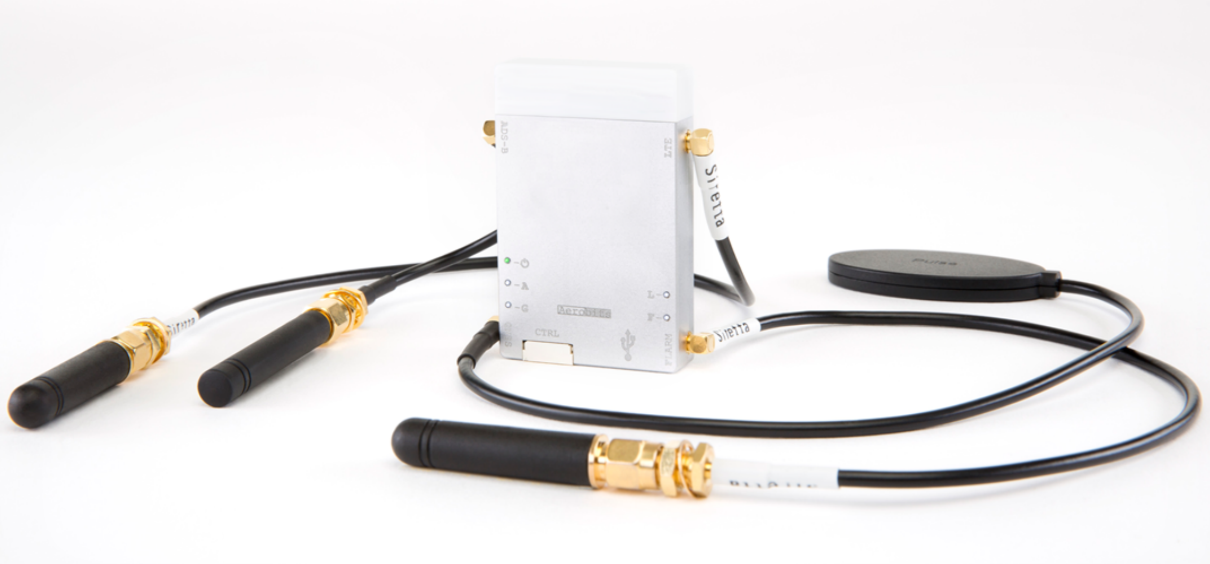


Figure 4: Combination overview

7.1 Specification of used anntena

Connector	Part number	Connector type
ADS-B	DELTA1A/X/SMAM/S/S/11	GSM/GPRS, 3G and Ism Stubby Antenna SMA Male
GNSS	PulseLarsen Antennas GNSSCGMSMA	GPS Antenna SMA Male
FLARM	TG.22.0111	GSM/GPRS Antenna SMA Male
LTE	TG.22.0111	GSM/GPRS Antenna SMA Male

Table 21: Description of commonly used anntena

7.2 Scope of delivery

1. Hook-On-Device
2. Power bank 3600mAh
3. Mounting bracket for HOD and power bank on drone
4. ADS-B antenna (GSM/GPRS, 3G and Ism Stubby Antenna SMA Male)
5. FLARM antenna (GSM/GPRS Antenna SMA Male)
6. LTE antenna (GSM/GPRS Antenna SMA Male)
7. GNSS antenna (GPS/LTE Antenna MMCX)
8. 2x MMCX -> SMA cable (250mm), 1 x MMCX -> SMA cable (100mm)
9. USB cable
10. 2 x Hook&loop, 2 x mounting tapes, 2 x M1.6 screws and allen key

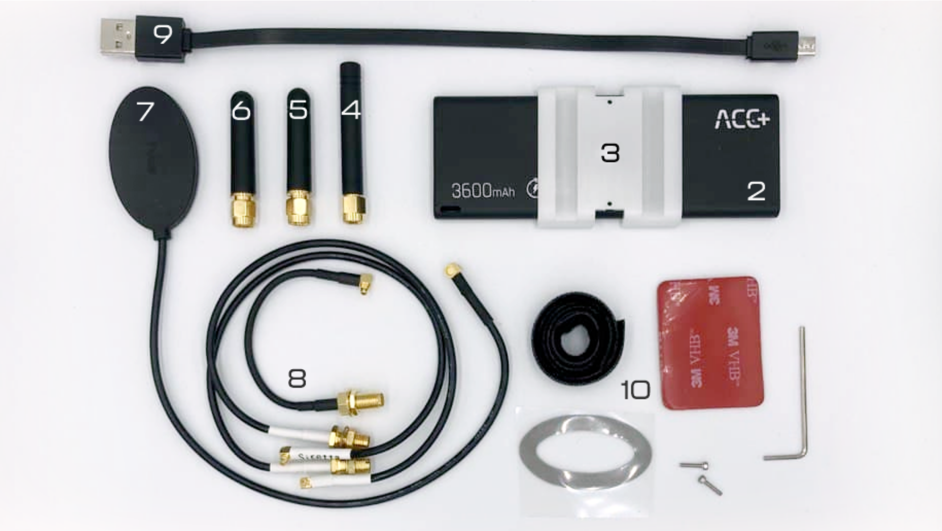


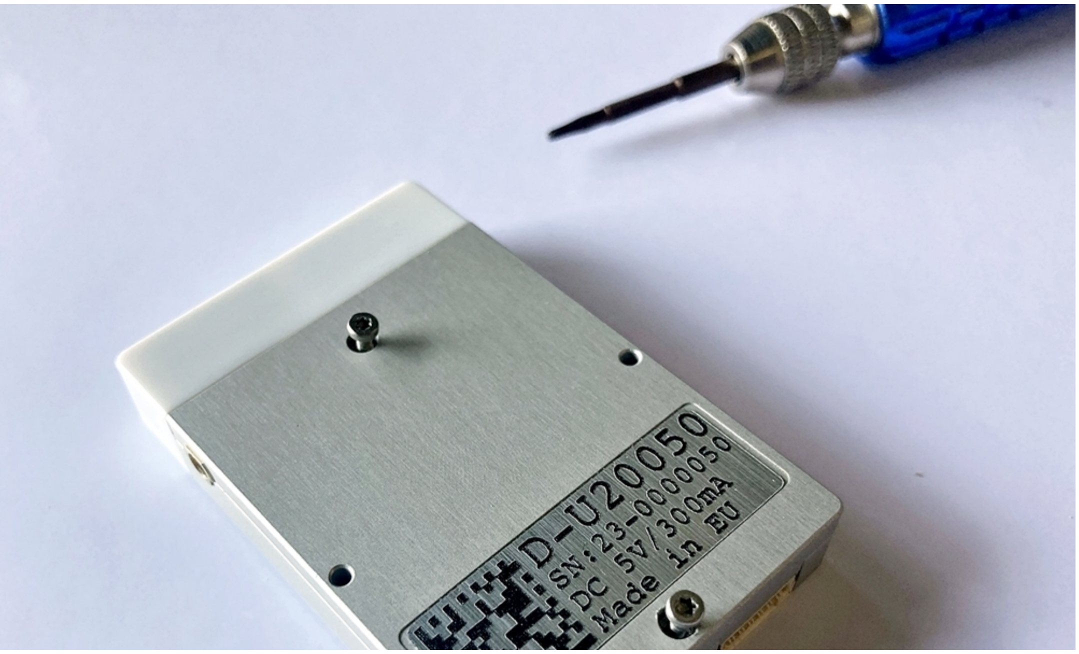
Figure 5: HOD equipment kit

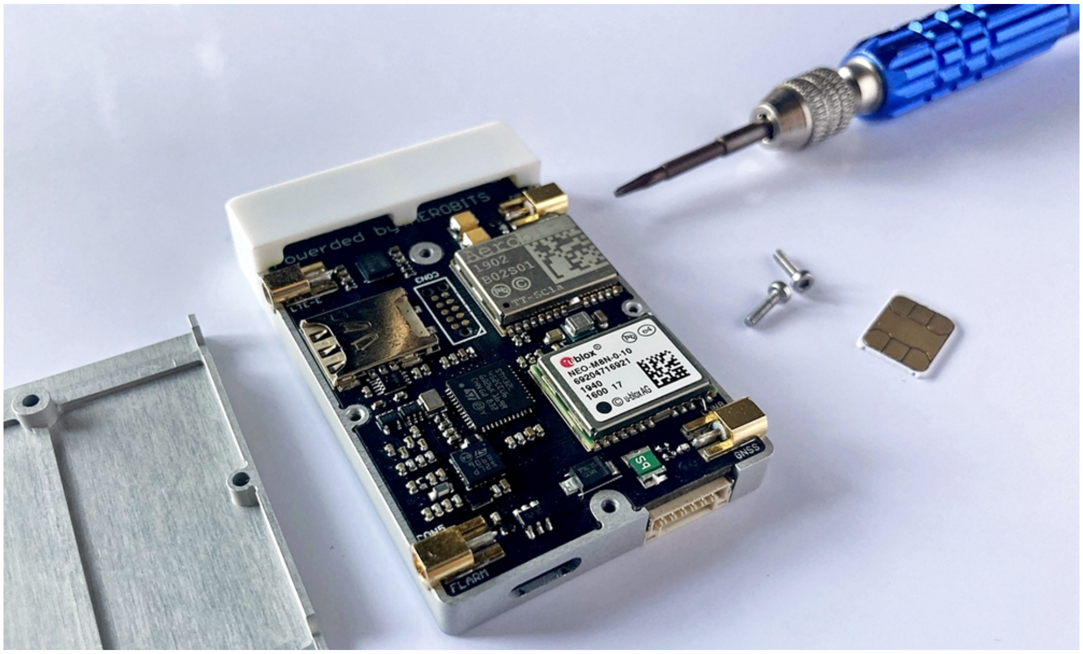
NOTE: LTE and FLARM antennas are identical and can be used interchangeably.

7.3 Inserting a SIM/chip card

To work properly the HOD V2 needs a valid SIM card. Recommended way to change/insert SIM card is described below.

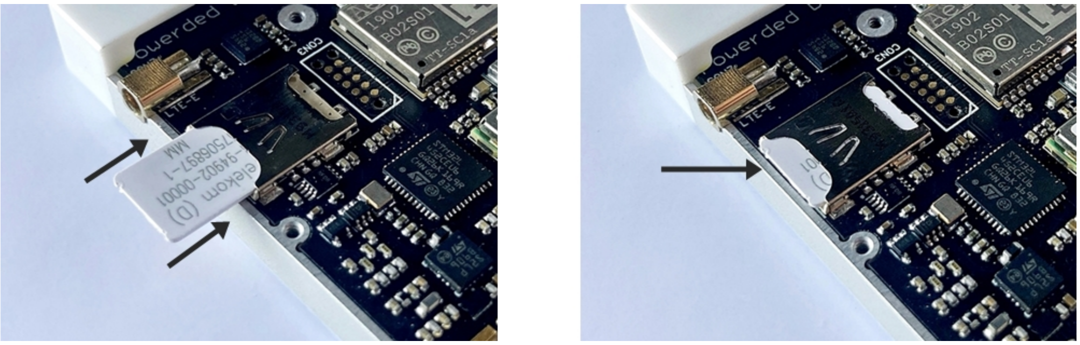
1. Disconnect module and unmount cover.





Ensure that you have HOD disconnected and take off module cover. To unmount cover you will need a asterisk screwdriver.

2. Find SIM card slot and insert SIM card.



Insert SIM card into slot visible on the picture above. A simple way to do it is to grab SIM with tweezers and push it carefully directly into slot.

7.4 Installation

7.4.1 Recharging of batteries

The HOD V2 uses an external power source (such as the supplied power bank), which can be recharged with the enclosed USB cable.

7.4.2 Usage of the Hook-On-Device Version 2

- Can be used for UAVs
- Can be used for all other air traffic such as helicopters, airplanes, gliders, etc.

7.4.3 Installation process

1. Click the antennas into their respective slots on the device
2. Connect the device to a power supply (via USB or JST connector)
3. Secure the device and the antennas to the UAV or the aircraft and consider the important notices for the installation given below. Securing all parts can be done with e.g. adhesive tape, table ties, Velcro tape, specially designed mounts etc. and is unique to every vehicle. Always ensure a secure mount of all parts before taking off.
4. The UAV or aircraft is ready to fly

NOTE: Once powered up, the device communicates its status via the LED descriptions mentioned in section above.

7.5 Antennas mount

For better performance antenna must be mounted on a ground-plane (carbon plate, PCB plate etc.) to radiate efficiently. The antenna should be mounted at the edge of the ground-plane of the mainboard of the device. Also no metal should be used near the antenna, with at least 20mm of clearance required, the more clearance the better. The best way to properly mount antennas is read manufacturer documentation. Example of proper mounting vertical antenna **DELTA1A/X/SMAM/S/S/11** below:

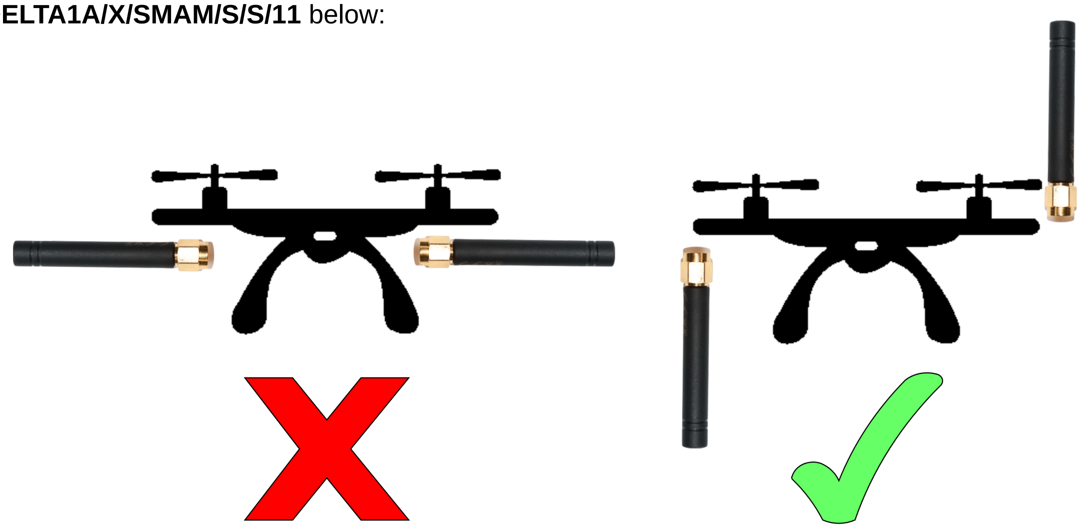


Figure 6: Proper antennas mount

7.5.1 Important notices during installation

- Keep separation between antennas and do not cross over antenna cables.
- GNSS antenna should be directed to the sky for optimal satellite signal reception.
- The external LTE antenna should be placed underneath the vehicle ideally in a vertical orientation facing towards the ground.

  NOTE: The device has an additional internal LTE antenna, which can be used instead of the external one, if the signal strength in the vicinity is sufficient. To switch from external to internal or vice versa the respective setting has to be changed in the software parameters via the downloadable Droniq HOD V1 software. Settings are: External LTE antenna = 1 for external use or External LTE antenna = 0 for internal.
- FLARM antenna should be placed in such a way to potentially “see” all other air traffic in the vicinity of the vehicle (e.g. mounted facing in the main direction of flight of the vehicle).

- ADS-B antenna like FLARM antenna (

  NOTE: Since the typical ADS-B signal strength is far greater than the one of FLARM the antenna mounting position is less crucial in comparison to FLARM and can be compromised in FLARMs favor).

- It is necessary to protect the device against vibrations of the vehicle as good as possible.
- The device is not secured against water, hence using it during rain or in wet conditions is not permitted.

8 General information

8.1 Important safety notes

NOTE: When using the Hook-On-Device Version 2 the following safety notes must be followed to reduce injuries, accidents and short circuit to a minimum.

1. The manual must be read carefully.
2. All instructions must be followed.
3. Medical equipment (such as hearing aids, pace makers, etc.) which are not shielded properly, can be affected in their functions. Please inform your doctor if this could be the case.
4. The device is operating with a voltage 5V.
5. Do not open the device, there are no parts that can be serviced by the user. Opening the device leads to the loss of warranty and infringes the general terms and conditions.
6. The user is solely responsible for the risk while using this device. Droniq and the manufacturer are not responsible for any injuries or accidents that are a result of improper use.

Please read carefully