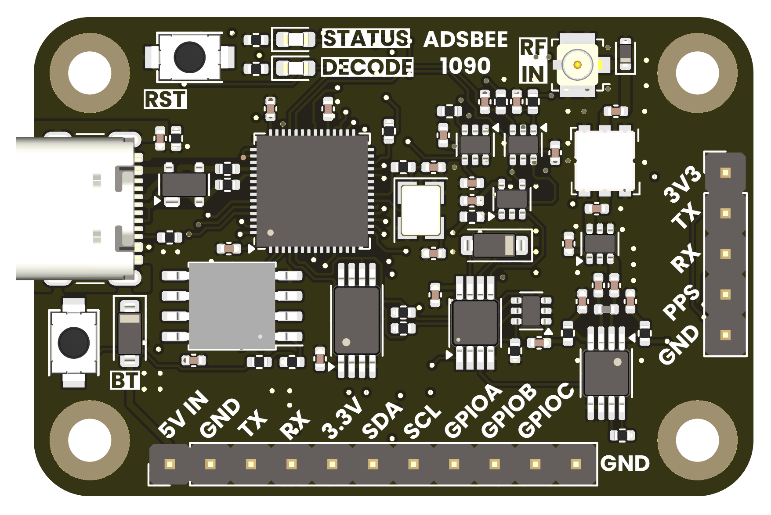
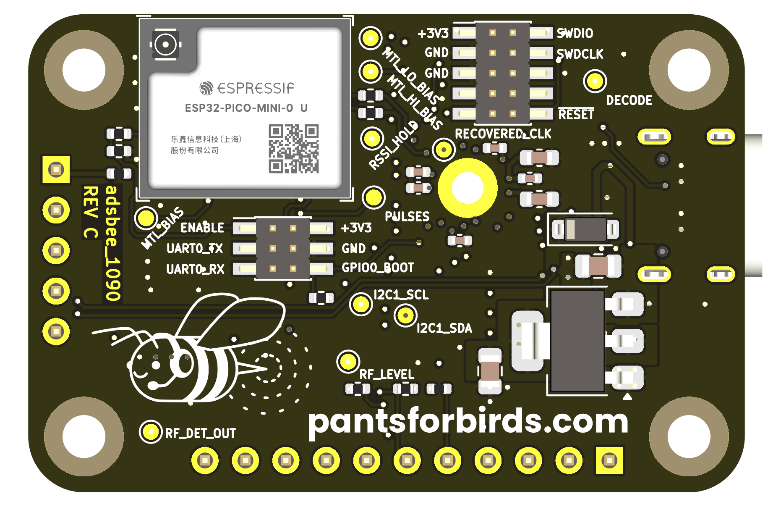
ADSBee 1090

Open Source Embedded ADS-B Receiver



Features

* 1090MHz Mode S and ADS-B packet decoding.
* Adjustable receive gain and trigger levels for customized tuning in diverse RF environments.
* Multiple output formats over UART or USB:
  + ADSBee CSV
  + MAVLink
  + GDL90 (not yet implemented)
  + More to come!
* Built-in EEPROM for storing configuration parameters in non-volatile memory.
* GNSS module input (UART + PPS) for MLAT or Remote ID applications.
* 2.4GHz 802.11 module for connecting directly to ADS-B databases via WiFi or broadcasting Remote ID beacon frames in UAS applications (not yet implemented).
* Integrated M2.5 mounting holes.
* Firmware updates over USB.



Applications

* Standalone feeder device for online ADS-B databases. No external compute required, just add power and WiFi!
* Aircraft detection for robotics and embedded projects.

Quick Specs

|  |  |
| --- | --- |
| Supply Voltage | 5V (Via USB or 5V pin) |
| Supply Current | 75mA (WiFi disabled)  ~400mA (WiFi enabled) |
| Minimum RF Input Power Level | -70dBm (not yet tested) |
| Simultaneous Aircraft Tracks Supported | ≤100 |
| Connectors | 1090MHz RF In: U.FL / MHF1  802.11 RF Out: W.FL / MHF3  Power / Data: USB C  GPIO / UART: 0.1” Pin Headers |

Open Source Hardware + Software

Github Repository: <https://github.com/coolnamesalltaken/ads-bee>

All hardware schematics and source code files required to build ADSBee 1090 are available under a GNU GPL v3 license. This means that they can be freely incorporated into other open-source projects that utilize a compatible license. The hope is that by opening the design to contributions and feedback from a community of users, the functionality of the ADSBee 1090 will be enhanced over time.

Note that the GPL v3 license applies to design and source code files, and not devices. ADSBee 1090 units purchased from Pants for Birds may be used in commercial applications without any licensing restrictions.

For commercial licensing requests of ADSBee hardware or software design files, please contact <john@pantsforbirds.com>.

Contents

[Features 1](#_Toc175911968)

[Applications **Error! Bookmark not defined.**](#_Toc175911969)

[Quick Specs 1](#_Toc175911970)

[Open Source Hardware + Software 1](#_Toc175911971)

[**1** Background 3](#_Toc175911972)

[**2** Communication Interfaces 3](#_Toc175911973)

[2.1 Console Interface 3](#_Toc175911974)

[2.2 COMMS\_UART Interface 3](#_Toc175911975)

[2.3 GNSS\_UART Interface 3](#_Toc175911976)

[**3** AT Commands 4](#_Toc175911977)

[**4** Reporting Protocols 6](#_Toc175911978)

[4.1 CSBee 7](#_Toc175911979)

[4.1.1 Aircraft Message 7](#_Toc175911980)

[4.1.2 Statistics Message 13](#_Toc175911981)

[MAVLINK 15](#_Toc175911982)

[MAVLINK ADSB\_VEHICLE (Message ID 246) Packet Definition 15](#_Toc175911983)

[MAVLINK MESSAGE\_INTERVAL (Message ID 244) Packet Definition 15](#_Toc175911984)

# Background

ADSBee 1090 was created as an attempt to build a low-cost ADS-B receiver without the use of an FPGA (found in most embedded ADS-B receivers on the market) and without the need for external compute (a requirement of most SDR-based ADS-B receivers). ADSBee 1090 accomplishes this through the use of a low-cost dual core microcontroller (RP2040) with flexible IO peripherals (PIO) that run a set of custom-written programs for preamble detection and packet decoding. By dedicating the RP2040’s PIO peripherals to the tasks required to find and decode ADS-B packets, ADSBee 1090 frees up its remaining cores to perform the logical functions required to validate checksums on decoded packets and decipher aircraft information (position, altitude, callsign, etc).

The ADSBee 1090 reports decoded aircraft information over UART and USB interfaces in a variety of protocols, and can utilize its attached ESP32 S3 module to host WiFi networks for data streaming to other devices, or connect to existing WiFi networks in order to upload data to the internet or other devices on the network.

Provisions are included for connecting an external GNSS module and UAT radio receiver to the ADSBee 1090. These features will enter development in the near future.

# Communication Interfaces

## Console Interface

The CONSOLE interface (USB-C connector) on the ADSBee 1090 can be used to supply the device with power, configure the device’s internal parameters via AT commands, and receive data from the device.

No baud rate configuration is necessary for the CONSOLE interface. Note that AT commands must be suffixed with CR+LF (“\r\n”) in order to be processed by the AT command parser.

## COMMS\_UART Interface

The COMMS\_UART interface is used for data output, and can support a number of different protocols. Data can be streamed out of the CONSOLE and COMMS\_UART interfaces simultaneously. The baud rate of the COMMS\_UART interface can be adjusted via AT commands.

**COMMS\_UART Parameters**

|  |  |
| --- | --- |
| **Parameter** | **Value** |
| Baud Rate | 115200 baud (default) |
| Data Bits | 8 |
| Stop Bits | 0 (N) |
| Parity Bits | 1 |
| Logic Level | 3.3V |

## GNSS\_UART Interface

The ADSBee 1090’s GNSS module connector includes a UART interface which can be used to receive NMEA sentences from a GNSS module. The baud rate of the GNSS\_UART interface can be adjusted via AT commands.

**GNSS\_UART Parameters**

|  |  |
| --- | --- |
| **Parameter** | **Value** |
| Baud Rate | 9600 baud (default) |
| Data Bits | 8 |
| Stop Bits | 0 (N) |
| Parity Bits | 1 |
| Logic Level | 3.3V |

# AT Commands

AT Commands are used to configure the ADSBee 1090 receiver’s internal parameters via the CONSOLE interface.

All AT command arguments are optional. Arguments will be ignored if left as blank or whitespace. For instance, to set the second parameter of AT+TL\_SET to 3000 without changing the value of the first parameter, the command “AT+TL\_SET=,3000” can be sent. Likewise, to change the first parameter to 2000 without changing the value of the second, the command “AT+TL\_SET=2000,” can be sent.

|  |  |
| --- | --- |
| Command | Parameters |
| **AT+LOG\_LEVEL** | **Log Level Command** |
| **AT+TL\_SET**  *Write with echo of values that were set.*  AT+MTLSET=<tl\_lo\_mv:uint16\_t>,<tl\_hi\_mv:uint16\_t>  +MTLSET=<tl\_lo\_mv:uint16\_t>,<tl\_hi\_mv:uint16\_t>  *Read present set value (stored setpoint, not read by ADC).*  AT+MTLSET?  +MTLSET=<tl\_lo\_mv:uint16\_t>,<tl\_hi\_mv:uint16\_t> | **RF Comparator Trigger Level (TL) Setpoint Command**  NOTE: tl\_lo\_mv should be set to a value lower than tl\_hi\_mv.  Reducing the level of tl\_lo\_mv will make the receiver more sensitive to weak RF signals, but will also increase the noise that it receives.  Increasing the difference between tl\_lo\_mv and tl\_hi\_mv will filter out signals with smaller dynamic range (difference in power level between max amplitude and min amplitude), thereby requiring a higher Signal to Noise ratio for a transponder signal to be decoded. This may reduce the likelihood that the ADSBee tries to decode a transponder signal with invalid bits that will trigger a checksum error.  tl\_lo\_mv: TL Low Threshold [milliVolts]   * 0-3300 = Low-side trigger threshold of the comparator circuit on the output of the RF detector. Refer to the AD8313 datasheet and adjustable gain stuff for a conversion from mV (RF detector output signal amplitude) to dBm (RF signal power level in).   tl\_hi\_mv: TL High Threshold [milliVolts]   * 0-3300 = High-side trigger threshold of the comparator circuit. |
|  |  |
| **AT+TL\_READ**  *Read with echo of values that were read.*  AT+MTLREAD  +MTLREAD=<tl\_lo\_mv>,<tl\_hi\_mv> | **RF Comparator Trigger Level (TL) Read Command**  Used an ADC to read the value of tl\_lo\_mv and tl\_hi\_mv. Should be roughly in line with the values of tl\_lo\_mv and tl\_hi\_mv set in the AT+TL\_SET section. |
| **AT+HELP**  AT+HELP  <command>:<command help string>  <command>:<command help string>  <…> | **Help Command**  Prints out a list of available commands and their associated help strings. |
| **AT+RX\_GAIN**  *Set gain to 100x with echo of gain value that was set.*  AT+RX\_GAIN=100  +RX\_GAIN=100  *Read gain value.*  AT+RX\_GAIN?  +RX\_GAIN=100  *Test +RX\_GAIN command.*  +RX\_GAIN=<gain:uint16\_t> | **Receiver Gain Command**  Adjust the gain of the operational amplifier located after the AD8313 in the receive signal chain. Gain is set as a positive integer value between 1-101.  gain: Receiver Gain [ratio]   * 1-101 = Gain value of operational amplifier operating on AD8313 output. |

# Reporting Protocols

ADSBee 1090 supports the following reporting protocols on CONSOLE and COMMS\_UART.

* CSBee
* GDL90 (not yet implemented)
* MAVLINK 1
* MAVLINK 2
* Mode S Beast
* Raw Packets

The CONSOLE interface reports debug messages and AT command responses in addition to the selected reporting protocol. If the CONSOLE interface is being used as a reporting interface, it is recommended to send AT+LOG\_LEVEL=SILENT to silence any debug logs that might corrupt the reported data, and to avoid sending additional AT commands while reading reported data in order to avoid the reported data being interspersed with OK and other AT command responses from the ADSBee 1090.

## CSBee

Comma Separated Bee protocol containing information about tracked aircraft as plain text.

The CSBee protocol is heavily inspired by the Aerobits Aero CSV protocol.

### Aircraft Message

This message contains information about an aircraft being tracked via ADS-B (1090MHz). Aircraft reports are provided once per second, per aircraft, until contact with the aircraft has been lost for 60 seconds.

#A:ICAO,FLAGS,CALL,SQUAWK,ECAT,LAT,LON,ALT\_BARO,ALT\_GEO,TRACK,VELH,VELV,SIGS, SIGQ,ACFPS,SFPS,SYSINFO,CRC\r\n

|  |  |  |  |
| --- | --- | --- | --- |
| **#A** | **Aircraft message start indicator** | **Format** | **Example value** |
| ICAO | ICAO number of aircraft (3 bytes). | Hex Integer | 3C65AC |
| FLAGS | Flags bitfield, see table 4.1.1.1. | Hex Integer | 12F356A8 |
| CALL | Callsign of aircraft. | String | N61ZP |
| SQUAWK | SQUAWK of aircraft. | Octal Integer | 7232 |
| ECAT | Emitter category, see table 4.1.1.2. | Integer | 14 |
| LAT | Latitude, in degrees. | Float | 57.57634 |
| LON | Longitude, in degrees. | Float | 17.59554 |
| ALT\_BARO | Barometric altitude, in feet. | Integer | 5000 |
| ALT\_GEO | Geometric altitude, in feet. | Integer | 5000 |
| TRACK | Ground track of aircraft, in degrees [0,360). | Integer | 35 |
| VELH | Horizontal velocity of aircraft, in knots. | Integer | 464 |
| VELV | Vertical velocity of aircraft, in ft/min. | Integer | -1344 |
| SIGS | Signal strength, in dBm. | Integer | -92 |
| SIGQ | Signal quality, in dB. | Integer | 2 |
| ACFPS | Number of valid Mode A and Mode C frames received from the aircraft during the last second. | Integer | 2 |
| SFPS | Number of valid Mode S frames received from the aircraft during the last second. | Integer | 5 |
| SYSINFO | Aircraft data integrity and physical dimensions, see table 4.1.1.3. | Hex Integer | 31BE89F2 |
| CRC | CRC16 (described in 4.1.1.4). | Hex Integer | 2D3E |

#### FLAGS Bitfield

Note: All bits 17-32 are momentary (cleared and updated every reporting interval).

|  |  |  |
| --- | --- | --- |
| **Bit** | **Bit Name** | **Meaning if the bit is set (1)** |
| 0 | IS\_AIRBORNE | Emitter is airborne. |
| 1 | POSITION\_VALID | Emitter has a valid position (ADSBee has received a valid pair of even and odd Compact Position Reporting packets and decoded an unambiguous location for the aircraft). |
| 2 | IS\_MILITARY | Emitter has transmitted at least one packet using a military format, such as Military Extended Squitter (DF=19). |
| 3 | IS\_CLASS\_B2\_GROUND\_VEHICLE | Emitter is actually a ground vehicle using a Class B2 transponder with a transmission power < 70W. |
| 4 | HAS\_1090\_ES\_IN | Emitter has receive capability for 1090MHz Extended Squitter transmissions. |
| 5 | HAS\_UAT\_IN | Emitter has receive capability for UAT (978MHz Universal Access Transceiver) transmissions. |
| 6 | TCAS\_OPERATIONAL | Emitter has a functional TCAS (Traffic Collision Avoidance System) onboard. |
| 7 | SINGLE\_ANTENNA | Emitter is using a single antenna, instead antennas above and below the fuselage. Transmissions may be weak or irregular during maneuvering. |
| 8 | SURFACE\_POSITION\_USES\_HEADING | Surface position messages provided by the aircraft indicate a heading and not a track angle. |
| 9 | HEADING\_USES\_MAGNETIC\_NORTH | Heading reported by the aircraft while on the surface uses magnetic north instead of true north. |
| 10 | IDENT | The aircraft has its SPI (Special Position Identification) bits set in Mode A/C or Mode S messages. This indicates that the pilot has depressed the momentary IDENT switch on their transponder, most likely at the request of air traffic control. |
| 11 | ALERT | The aircraft is issuing either a permanent or momentary alert. This could correspond to an operational mode change or something else. |
| 12 | TCAS\_RA | The aircraft has an active TCAS resolution advisory (i.e. the aircraft is warning the pilot to take action in order to avoid colliding with another aircraft). |
| 13 | RESERVED |  |
| 14 | RESERVED |  |
| 15 | RESERVED |  |
| 16 | RESERVED |  |
| 17 | UPDATED\_BARO\_ALTITUDE | Barometric altitude has been updated within the last reporting interval. |
| 18 | UPDATED\_GNSS\_ALTITUDE | GNSS altitude has been updated within the last reporting interval. |
| 19 | UPDATED\_POSITION | Position (latitude / longitude) has been updated within the last reporting interval. |
| 20 | UPDATED\_TRACK | Track has been updated within the last reporting interval. |
| 21 | UPDATED\_HORIZONTAL\_VELOCITY | Horizontal velocity has been updated within the last reporting interval. |
| 22 | UPDATED\_VERTICAL\_VELOCITY | Vertical velocity has been updated within the last reporting interval. |
| 23 |  |  |
| 24 |  |  |
| 25 |  |  |
| 26 |  |  |
| 27 |  |  |
| 28 |  |  |
| 29 |  |  |
| 30 |  |  |
| 31 |  |  |

#### ECAT Field

The ECAT field indicates the Emitter Category (i.e. airframe type) for each ADSB emitter that is being tracked. This field contains information about what kind of aircraft, ground vehicle, obstacle, or other airspace user is emitting ADS-B packets, and can be used to understand the emitter’s maneuvering capability and potential for wake vortex impact.

|  |  |
| --- | --- |
| **ECAT Value** | **Emitter Category** |
| 0 | Invalid |
| 1 | Reserved |
| 2 | No Category Information |
| 3 | Surface Emergency Vehicle |
| 4 | Surface Service Vehicle |
| 5 | Ground Obstruction |
| 6 | Glider / Sailplane |
| 7 | Parachutist / Skydiver |
| 8 | Ultralight / Hang Glider / Paraglider |
| 9 | Unmanned Aerial Vehicle |
| 10 | Space / Transatmospheric Vehicle |
| 11 | Light Aircraft (< 7,000kg) |
| 12 | Medium 1 (7,000kg – 34,000kg) |
| 13 | Medium 2 (34,000kg – 136,000kg) |
| 14 | High Vortex Aircraft |
| 15 | Heavy (> 136,000kg) |
| 16 | High Performance (> 5 G acceleration and > 400 kts speed) |
| 17 | Rotorcraft |

#### SYSINFO Bitfield

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **SYSINFO Bitfield** | | | | | | | | | | | | | | | |
| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|  |  |  | Aircraft Maximum Dimension (MDIM) | | | | | | | GNSS Antenna Offset Direction (GAOR) | GNSS Antenna Offset Distance (GAOD) | | GNSS Antenna Offset Known (GAOK) | System Design Assurance (SDA) | |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Source Integrity Level (SIL) | | Geometric Vertical Accuracy (GVA) | | Navigation Accuracy Category: Position (NACp) | | | | Navigation Accuracy Category: Velocity (NACv) | | | Navigation Integrity Category: Barometer (NICbaro) | Navigation Integrity Category (NIC) | | | |

**SYSINFO[0-3]: Navigation Integrity Category (NIC)**

The radius of containment (NIC) indicates how much trust should be placed in an aircraft’s reported location in the horizonal plane. The NIC reports a radius of containment specified by the avionics system of the emitter. The probability that the aircraft is outside of this radius of containment due to its avionics system receiving a faulty signal from one of its inputs (without displaying an error) is provided by another bitfield called the Source Integrity Level (SIL). Combined, the NIC and SIL indicate how likely it is that an aircraft is not actually contained by a bubble of a specified size, centered at the aircraft’s reported location, assuming that the avionics onboard the aircraft are functioning correctly but may be given faulty inputs.

A higher NIC value indicates more trust in an aircraft’s reported latitude / longitude position.

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **NIC­ Value** | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 |
| **Radius of Containment** | Unknown | < 20 NM | < 8 NM | < 4 NM | < 2 NM | < 1 NM | < 0.6 NM | < 0.2 NM | < 0.1 NM | < 75 m | < 25 m | < 7.5 m |

#### SYSINFO[4]: Navigation Integrity Category: Barometer (NICbaro)

The barometric altitude integrity (NICbaro) indicates how much trust should be placed in an aircraft’s reported altitude. The field is a single bit that indicates whether the aircraft uses an altimeter that has been cross-checked against other sources. Old school encoding altimeters have many parallel wires and output an altitude in a format called a Gillham Code, and have no built-in method of error checking. A single faulty wire can result in erroneous readings, so this bit lets air traffic control know whether to take altitude readings from the aircraft with a grain of salt. A 0 indicates that the transponder is outputting altitude from a Gillham coded source (with no way to cross check the value), while a 1 indicates that the transponder is outputting altitude from a Gillham coded source while using another sensor to cross-check it, or is using a more modern barometer that supports a protocol with built-in error checking.

A higher NICbaro value (i.e. 1 instead of 0) indicates more trust in the aircraft’s reported altitude.

|  |  |  |
| --- | --- | --- |
| **NICbaro Value** | 0 | 1 |
| **Barometric Altitude Integrity** | Altitude is from a Gillham-coded input, not cross checked. | Altitude is from a Gillham-coded input that is being cross-checked with another source, or from a non Gillham-coded input with built-in error checking features. |

#### SYSINFO[5-7]: Navigation Accuracy Category: Velocity (NACv)

The horizontal velocity error (NACv) indicates the expected accuracy of the reported velocity of the aircraft when systems are operating nominally. This varies depending on the accuracy capabilities of the measurement equipment onboard the aircraft, and not how often we expect said equipment to fail.

A higher NACv indicates a more accurate velocity measurement system.

|  |  |
| --- | --- |
| **NACv Value** | **Horizontal Velocity Error** |
| 0b000 | Unknown or ≥ 10 m/s |
| 0b110 | < 10 m/s |
| 0b010 | < 3 m/s |
| 0b011 | < 1 m/s |
| 0b100 | < 0.3 m/s |

#### SYSINFO[8-11]: Navigation Accuracy Category: Position (NACp)

The estimated position uncertainty (NACp) indicates the expected accuracy of the aircraft’s reported location when systems are operating nominally. This varies depending on the capabilities of the aircraft’s positioning system and not on how often we expect said equipment to fail.

A higher NACp indicates a more accurate positioning system.

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **NACp Value** | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 |
| **Estimated Position Uncertainty** | Unknown or ≥ 10NM | < 10 NM | < 4 NM | < 2 NM | < 1 NM | < 0.5 NM | < 0.3 NM | < 0.1 NM | < 0.5 NM | < 30 m | < 10 m | < 3 m |

#### SYSINFO[12-13] Geometric Vertical Accuracy (GVA)

The geometric vertical accuracy indicates the 95% confidence interval (vertical figure of merit) provided by the aircraft’s onboard GNSS system (i.e. assuming some distribution of altitudes, the aircraft’s GNSS system is confident that 95 out of 100 times, the aircraft falls within some height of the reported geometric altitude.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **GVA Value** | 0 | 1 | 2 | 3 |
| **95% Vertical Figure of Merit (VFOM)** | Unknown or ≥ 150 m | < 150 m | ≤ 45 m | < 45 m (Was previously “reserved”, the actual value of this field may change but is guaranteed to be < 45 m). |

#### SYSINFO[14-15] Source Integrity Level (SIL)

The source integrity level indicates the probability that the aircraft exceeds the bounds of its horizontal radius of containment (NIC) due to a silent fault in signals received by the aircraft (no avionics failure).

|  |  |
| --- | --- |
| **SIL Value** | **Probability of Exceeding NIC Radius of Containment Due to Silent Fault** |
| 0 | Unknown or > 1x10-3 per flight hour. |
| 1 | ≤ 1x10-3 per flight hour. |
| 2 | ≤ 1x10-5 per flight hour. |
| 3 | ≤ 1x10-7 per flight hour. |
| 4 | Unknown or > 1x10-3 per sample. |
| 5 | ≤ 1x10-3 per sample. |
| 6 | ≤ 1x10-5 per sample. |
| 7 | ≤ 1x10-7 per sample. |

**SYSINFO[16-17] System Design Assurance (SDA)**

The system design assurance indicates how robust the aircraft’s position reporting systems are to failures of various severities. For instance, SDA = 1, a low SDA value, corresponds to Software and Hardware Design Assurance Level D, which states that a minor failure could cause the aircraft to transmit misleading position information with a probability of ≤ 1x10-3 per flight hour. A more robust system with SDA = 3, corresponding to Software and Hardware Design Assurance Level B, is expected to transmit misleading position information with a probability of 1x10-7 per flight hour even under a Hazardous failure condition.

Software Design Assurance categories used in this field are classified under RTCA DO-178B, Airborne Electronic Hardware Design Assurance are classified under RTCA DO-254, and failure classification levels are defined in FAA Advisory Circular [AC-23.1309-1E](https://www.faa.gov/regulations_policies/advisory_circulars/index.cfm/go/document.information/documentID/1019681).

|  |  |  |  |
| --- | --- | --- | --- |
| **SDA Value** | **Supported Failure Condition** | **Probability of Undetected Fault causing transmission of False or Misleading Information** | **Software and Hardware Design Assurance Level** |
| 0 | Unknown / No Safety Effect | > 1x10-3 per flight hour or unknown. | N/A |
| 1 | Minor | ≤ 1x10-3 per flight hour. | D |
| 2 | Major | ≤ 1x10-5 per flight hour. | C |
| 3 | Hazardous | ≤ 1x10-7 per flight hour. | B |

**SYSINFO[18] GNSS Antenna Offset Known (GAOK)**

This field indicates whether the aircraft has reported the installation location of its GNSS antenna relative to its centerline (roll axis). A field for reporting this value is only available in ADS-B messages emitted by aircraft on the ground, and even then, the aircraft may not report a value in this field.

|  |  |
| --- | --- |
| **GANTO Value** | **Aircraft reported location of its GNSS antenna relative to roll axis?** |
| 0 | No |
| 1 | Yes |

**SYSINFO[19-20] GNSS Antenna Offset Distance (GAOD)**

This field indicates the distance that the GNSS antenna is offset from the centerline (roll axis) of the aircraft. Aircraft only report even values for their GNSS antenna offset distance, between 2-6 meters, so the reported offset distance can be calculated using the equation below.

GNSS antenna offset distance = GAOD << 1

Note that this value is only reported by some aircraft while operating on the ground. Aircraft operating in the air do not report this value. Always check the value of the GAOK bit to see if the value of GAOD is worth paying attention to.

**SYSINFO[21] GNSS Antenna Offset Direction (GAOR)**

|  |  |
| --- | --- |
| **GAOR Value** | **GNSS Antenna Offset Direction** |
| 0 | GNSS antenna is offset to the left of centerline (roll axis). |
| 1 | GNSS antenna is offset to the right of centerline (roll axis). |

**SYSINFO[22-28] Aircraft Maximum Dimension (MDIM)**

This field indicates the value of the maximum dimension (length or width) of an aircraft, and is only reported by aircraft while on the ground. This field has no special coding, and can be interpreted directly as a binary unsigned integer value.

#### CRC Field

#### CSBee messages use a 16-bit Cyclical Redundancy Checksum (CRC-16), which can be calculated using the algorithm in the C++ code snippet below. Note the “swap16” helper function which also needs to be included.

#### uint16\_t swap16(uint16\_t value) { return (value << 8) | (value >> 8); }

#### uint16\_t CalculateCRC16(const uint8\_t \*data\_p, int32\_t length) {

#### uint8\_t x;

#### uint16\_t crc = 0xFFFF;

#### while (length--) {

#### x = crc >> 8 ^ \*data\_p++;

#### x ^= x >> 4;

#### crc = (crc << 8) ^ ((uint16\_t)(x << 12)) ^ ((uint16\_t)(x << 5)) ^ ((uint16\_t)x);

#### }

#### return swap16(crc);

#### }

### Statistics Message

This message contains some useful statistics about operation of module. Format of that frame is shown below:

#S:DPS,ACFPS,SFPS,TSCAL,UPTIME,CRC\r\n

|  |  |  |
| --- | --- | --- |
| **#S** | **Statistics message start indicator** | **Example** |
| DPS | Number of attempted demodulations in the last second. | 106 |
| ACFPS | Number of MODE-A or MODE-C frames received in the last second. | 20 |
| SFPS | Number of valid Mode S frames received in the last second. | 3 |
| TSCAL | Calibration value for TS field in raw frames | 13999415 |
| UPTIME | Time from last enter to RUN mode, in seconds. | 134 |
| CRC | CRC16 (described in 4.1.2.1). | 2D3E |

#### CRC Field

See 4.1.1.4.

## MAVLINK

Tracked aircraft information is sent in MAVLINK ADSB\_VEHICLE messages, in a data burst once per second. The data burst consists of Nx ADSB\_VEHICLE messages, where N is the number of tracked aircraft, and 1x MESSAGE\_INTERVAL message as a delimiter which indicates the end of the list of tracked aircraft. Note that this is a binary protocol which is not human readable.

|  |
| --- |
| **WARNING**: [Windows has a bug](https://www.guralp.com/howtos/misindentified-serial-mouse.shtml#:~:text=The%20COM%20port%20then%20becomes,cause%20the%20PC%20to%20crash.), which causes some machines to recognize a serial port reporting MAVLINK packets as a mouse, which can result in phantom mouse movements and clicks (ask me how I know). Please use caution while running MAVLINK on a serial port while the computer is unattended. |

### MAVLINK ADSB\_VEHICLE (Message ID 246) Packet Definition

From: [https://mavlink.io/en/messages/common.html#ADSB\_VEHICLE](https://mavlink.io/en/messages/common.html%23ADSB_VEHICLE)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Field Name** | **Type** | **Units** | **Values** | **Description** |
| ICAO\_address | uint32\_t |  |  | ICAO address |
| lat | int32\_t | degE7 |  | Latitude |
| lon | int32\_t | degE7 |  | Longitude |
| altitude\_type | uint8\_t |  | [ADSB\_ALTITUDE\_TYPE](https://mavlink.io/en/messages/common.html#ADSB_ALTITUDE_TYPE) | ADSB altitude type. |
| altitude | int32\_t | mm |  | Altitude(ASL) |
| heading | uint16\_t | cdeg |  | Course over ground |
| hor\_velocity | uint16\_t | cm/s |  | The horizontal velocity |
| ver\_velocity | int16\_t | cm/s |  | The vertical velocity. Positive is up |
| callsign | char[9] |  |  | The callsign, 8+null |
| emitter\_type | uint8\_t |  | [ADSB\_EMITTER\_TYPE](https://mavlink.io/en/messages/common.html#ADSB_EMITTER_TYPE) | ADSB emitter type. |
| tslc | uint8\_t | s |  | Time since last communication in seconds |
| flags | uint16\_t |  | [ADSB\_FLAGS](https://mavlink.io/en/messages/common.html#ADSB_FLAGS) | Bitmap to indicate various statuses including valid data fields |
| squawk | uint16\_t |  |  | Squawk code |

### MAVLINK MESSAGE\_INTERVAL (Message ID 244) Packet Definition

From: [https://mavlink.io/en/messages/common.html#MESSAGE\_INTERVAL](https://mavlink.io/en/messages/common.html%23MESSAGE_INTERVAL)

|  |  |  |  |
| --- | --- | --- | --- |
| **Field Name** | **Type** | **Units** | **Description** |
| message\_id | uint16\_t |  | The ID of the requested MAVLink message. v1.0 is limited to 254 messages.  NOTE: For ADSBee 1090, message\_id is always 246, corresponding to the ADSB\_VEHICLE message. |
| interval\_us | int32\_t | us | The interval between two messages. A value of -1 indicates this stream is disabled, 0 indicates it is not available, > 0 indicates the interval at which it is sent.  NOTE: For ADSBee 1090, message\_id is always 1000 us. |