EVALUATION KIT AVALIABLE

TT-MC1 series

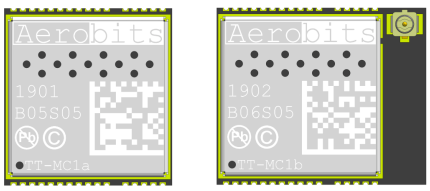
(TT-MC1a, TT-MC1b)

General Description

TT-MC1 is a high-performance OEM receiver series operating at 1090MHz. It is based on the proven FPGA-In-The-LoopTM technology, which is a unique combination of a multi-core processor and FPGA. The patented solution allows high-speed RF data processing with significantly reduced number of electronic components. Simultaneous miniaturization of the module and its OEM nature open a wide range of possible applications. The basic version of TT-MC1 offers the possibility of receiving and decoding ADS-B and Mode-A/C/S in different modes. The analysis of the power/quality of the RF signal and the use of time stamps facilitates the implementation of multilaterations, and the fast UART interface allows for the simple integration of the module with the user's system. In addition, large reserves of computing power open the way to customize the firmware and extend the module with non-standard functions. There are several communication interfaces, protocols and special functionalities available on request, such as ADS-B/Out (digital data). For more information please contact: support@aerobits.pl.

Applications

- SAA / DAA (Sense and Avoid / Detect and Avoid)
- UAS ground stations and high-density traffic surveillance
- UTM / U-Space construction (traffic surveillance network)
- Traffic-flow analysis and statistics
- Monitoring of 1090MHz band (signal integrity check)
- ADS-B/In/Out devices that meet the NextGen/SESAR philosophy
- Etc.



Features

- Fastest ADS-B implementation on a surface of <4cm2
- Receiving of ADS-B, Mode-A/C/S with RF signal strength/quality analysis
- Time stamp (raw data only) for multilateration
- High-resolution ADC with real-time signal processing; best-in-class aircraft tracking
- High sensitive front-end, jamming and ESD protection (only TT-MC1b) with ranges over 300 km (open space, 1dBi antenna)
- Simple module integration via UART interface and AT commands
- Multiple supported protocols, i.a. MAVLink, GDL90
- Scalable OEM solution with enormous customization potential (additional functions or interfaces on request)
- Firmware update capability (uC and FPGA)
- Power consumption 5V/60mA (RF part), 3.3V/200mA (digital part)
- Small outline: 18.0 x 19.0 x 2.65mm, ver. (a); 22.5 x19.0 x 2.65mm, ver. (b), weight < 2g
- Designed to meet MOPS defined in TSO-C199

Ordering Information

Type	Symbol	Description
OEM Module	TT-MC1a	Without antenna connector
OEM Module	TT-MC1b	With antenna connector (U.FL)
Evaluation kit	EVAL-TT-MC1	Based on TT-MC1a

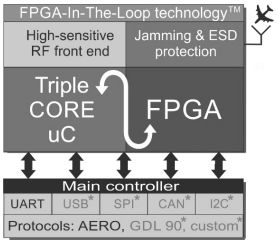


Figure 1: Structure, interfaces and protocols of TT-MC1 series.
*on request only (custom solutions)

Contents

1 Electrical characteristics

Absolute Maximum Ratings			
Parameter	Min.	Max.	Unit
Storage temperature	-40	+85	°C
Supply voltage (VCC1)	2.7	3.6	DCV
Supply voltage (VCC2)	4.0	5.5	DCV
Other pin voltage	VSS-0.4	VCC + 0.4	DCV
RF input power	-	+15	dBm
Recommended Operating Conditions			
Parameter	Min.	Max.	Unit
Temperature	-40	+70	°C
Supply voltage (VCC1)	3.0	3.6	DCV
Supply voltage (VCC2)	4.5	5.5	DCV

Table 1: Absolute maximum ratings and recommended operation conditions.

General Electrical Specification					
Parameter	Description	Min.	Typ.	Max.	Unit
Carrier frequency		-	1090	-	MHz
RX sensitivity	For operation at 50Ω U.fl connector (TT-MC1b)	-	-85	-	dBm
Input Low Voltage	RESET, UARTs, CAN, USB, SPI, I2C	-0.3	-	0.8	DCV
Input High Voltage	RESET, UARTs, CAN, USB, SPI, I2C, GPIO	0.7 VCC1	-	VCC1 + 0.3	DCV
Output Low Voltage	UARTs, CAN, USB, I2C, SPI, GPIO	-	-	0.4	DCV
Output High Voltage	UARTs, CAN, USB, I2C, SPI, GPIO	VCC1 - 0.4	-	-	DCV
Current consumption (digital part)		-	200	-	mA
Current consumption (RF front end)		-	60	-	mA

Table 2: General electrical specification.

2 Pin definition

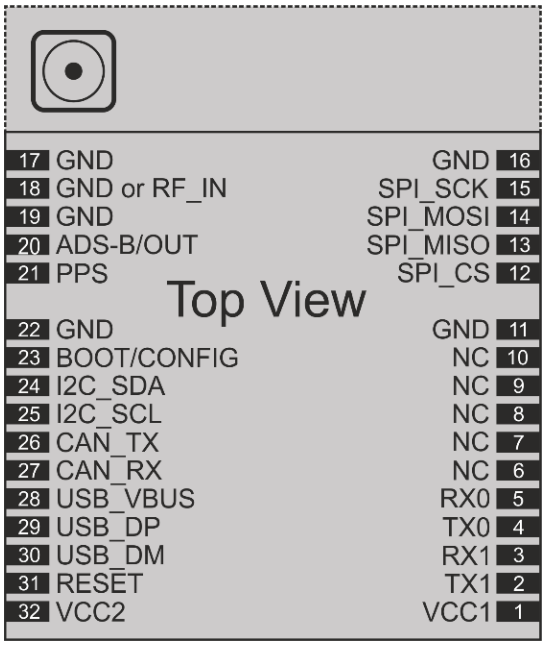


Figure 2: Pin assignment for TT-MC1a and TT-MC1b.

Pin No.		Pin Name	Pin Type	Description
1		VCC1	Power	3.3V (digital supply)
2		TX1	CMOS Output	UART1 data output
3		RX1	CMOS Input	UART1 data input
4		TX0	CMOS Output	UART0 data output
5		TX1	CMOS Input	UART0 data input
6		NC	Reserved	Keep floating
7		NC	Reserved	Keep floating
8		NC	Reserved	Keep floating
9		NC	Reserved	Keep floating
10		NC	Reserved	Keep floating
11		GND	GND	Common ground
12		SPI_CS	CMOS Output	Serial Peripheral Interface / chip select
13		SPI_MISO	CMOS Input	Serial Peripheral Interface / data input
14		SPI_MOSI	CMOS Output	Serial Peripheral Interface / data output
15		SPI_SCK	CMOS Output	Serial Peripheral Interface / clock
16		GND	GND	Common ground
17		GND	GND	Common ground
18	TT-MC1a	RF_IN	RF Input	RF Input (antenna)
	TT-MC1b	GND	GND	Common ground
19		GND	GND	Common Ground
20		ADSB_OUT	CMOS Output	ADS-B Output (digital)
21		PPS	CMOS Input	1PPS GNSS signal
22		GND	GND	Common ground
23		BOOT/CONFIG	CMOS Input	Bootloader / Configuration mode
24		I2C_SDA	Bi-directional	I2C Data line
25		I2C_SCL	Bi-directional	I2C Clock line

Pin No.	Pin Name	Pin Type	Description
26	CAN_TX	CMOS Output	CAN transmit
27	CAN_RX	CMOS Input	CAN receive
28	USB_VBUS	Power	USB Power line
29	USB_DP	Bi-directional	USB+
30	USB_DM	Bi-directional	USB-
31	RESET	CMOS Input	Reset input / active low
32	VCC2	Power	5V (RF front end supply)
	Interfaces and I/Os not used in standard design (for custom implementation only)		

Table 3: Pinout.

3 General 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.

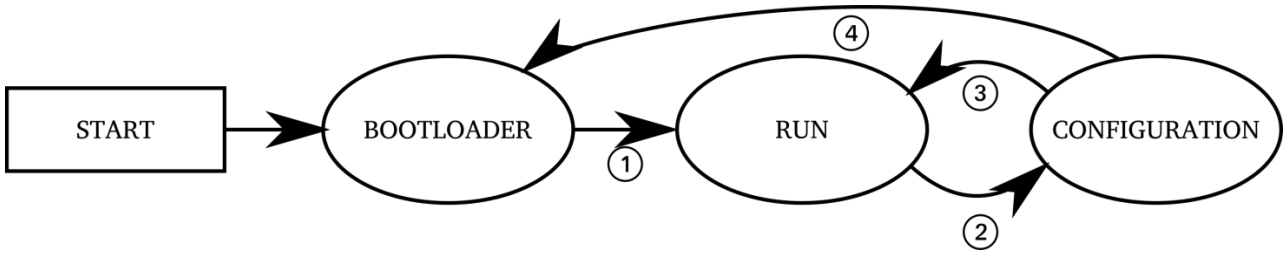


Figure 3: State machine of TT-MC1module.

3.1 States of operation

3.1.1 BOOTLOADER state

This is an initial state of TT-MC1module 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 baudrate is constant and is set to 115200bps. After powering up module, it stays in this state for 1s. If any BOOTLOADER trigger is not present, module will transit to RUN state. Firmware upgrade is possible with use of AEROBITS_updater.exe program. To acquire this program please contact: support@aerobits.pl.

3.1.2 RUN state

In this state module is working and receiving the data from aircraft. 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 baudrate.

3.1.3 CONFIGURATION state

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

3.2 Transitions between states

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

3.2.1 BOOTLOADER to RUN transition

BOOTLOADER state is semi-stable, that means the module requires additional action to stay in BOOTLOADER state. This transition will occur automatically if any of the required actions will not be taken. To stay in BOOTLOADER state (prevent this transition) one of following action must be processed:

- Pull BOOT/CONFIG pin low during start of module
- Send `AT+LOCK=1` command while device is in BOOTLOADER state (always after power on for approximately 1s)

- Send AT+REBOOT_BOOTLOADER command in CONFIGURATION state. This will move to BOOTLOADER state and will lock module in this state.

If none of preceding conditions are meet, 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:

- Release or pull high BOOT/CONFIG pin
- If module is locked, send AT+LOCK=0 command

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

3.2.2 RUN to CONFIGURATION transition

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

- Pull BOOT/CONFIG pin low
- Send AT+CONFIG=1 (using current baudrate). This method is not recommended, because module will support multiple protocols in future and Aerobits Sp. z o.o. cannot ensure that this command will be present in all protocols.

When moduleleaves RUN state it will send AT+RUN_END command and then AT+CONFIG_START command, on entering CONFIGURATION state. The former will be send using baudrate from settings, the latter will be send using 115200bps baudrate.

3.2.3 CONFIGURATION to RUN transition

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

- Release or pull high BOOT/CONFIG pin
- Send AT+CONFIG=0 command.

When module leaves CONFIGURATION state it will send AT+CONFIG_END command and then AT+RUN_START command, on entering RUN state. The former will be send using 115200bps baudrate, the latter will be send using baudrate from settings.

3.2.4 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 .

4 UART configuration

Communication between module and host device is done using UART interface. This interface has following settings. In CONFIGURATION and BOOTLOADER state baudrate is fixed at 115200bps.

UART Settings				
Parameter	Min.	Typ.	Max	Unit
Baudrate	115200	115200	3000000	bps
Stop Bits Number	-	1	-	-
Flow Control	-	None	-	-
Parity Bit	-	None	-	-

Table 4: UART settings.

5 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. Values are stored in non-volatile memory when leaving CONFIGURATION state, so they will be preserved during restart.

5.1 Write settings

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

5.2 Read settings

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

5.3 Settings description

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

```
Setting: PROTOCOL
       Description: Selected protocol: 0 NONE, 1 - RAW HEX, 2- CSV, 3 - MAVLINK,
4 - ASTERIX, 5 - GLD90
       Type: Integer decimal
       Range (min.): 0
       Range (max.): 5
       Is preserved: 1
       Is restart needed: 0
```

5.4 Errors

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

5.5 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.

5.6 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.

5.7 Available settings

Setting	Min value	Max value	Default value	Comment
BAUDRATE	0	2	0	Baudrate in RUN state 0 - 115200bps 1 - 921600bps 2 - 3000000bps
PROTOCOL	0	5	2	Selected protocol 0 - None 1 - RAW HEX 2 - CSV (AERO) 3 - MAVLink 4 - ASTERIX 5 - GDL90
SUBPROTOCOL	0	0	0	Reserved for future use

Table 5: Settings

5.8 Example

As an example, to switch MC1 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
```

6 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.

6.1 Commands in BOOTLOADER and CONFIGURATION state

6.1.1 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

6.1.2 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

6.2 Commands in CONFIGURATION state

6.2.1 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

6.2.2 AT+SETTINGS?

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

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

6.2.3 AT+HELP

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

```
SETTINGS:
AT+PROTOCOL=2         [Selected protocol: 0 - NONE, 1 - RAW HEX, 2- CSV, 3 - MAVLINK,
4 - ASTERIX, 5 - GLD90]
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]
```

6.2.4 AT+SETTINGS_DEFAULT

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

6.2.5 AT+SERIAL_NUMBER

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

Response:

```
AT+SERIAL_NUMBER=0202041E43
```

6.2.6 AT+FIRMWARE_VERSION

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

Response:

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

6.2.7 AT+REBOOT

`AT+REBOOT` - Restart module.

6.2.8 AT+REBOOT_BOOTLOADER

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

NOTE: This command also sets lock.

6.3 Commands in RUN state

AT+CONFIG=1 - transition to CONFIGURATION state.

NOTE: This command also sets lock.

NOTE: This command is depreciated and could be not supported in all future protocols. It is better to use BOOT/CONFIG pin to make module transit to CONFIGURATION state.

7 CSV protocol (AERO)

CSV protocol is simple text protocol, that allows fast integration and analysis of tracked aircrafts. CSV messages start with "#" character and ends with "\r\n" characters. There are 2 types of messages. AIRCRAFT message and STATISTICS message.

7.1 CRC

Frames include CRC value for consistency check. CRC value is calculated using standard CRC16 algorithm and is based on every character in frame starting from '#' to last comma ',' (excluding last comma). After calculation, value is appended to frame using hexadecimal coding. Function that calculates CRC is shown below.

```
uint16_t crc16(const uint8_t* data_p, uint32_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);
}
```

7.2 Aircraft message

This message describes state vector of aircraft and is send once per second.

```
#A:ICAO,FLAGS,CALL,SQ,LAT,LON,TRACK,HEA,VELH,VELV,SIGS,SIGQ,FPS,RES,CRC\r\n
```

#A	Aircraft message start indicator	Example value
ICAO	ICAO number of aircraft (3 bytes)	3C65AC
FLAGS	Flags (see below)	1
CALL	Callsign of aircraft	N61ZP
SQ	SQUAWK of aircraft	7232
LAT	Latitude in degrees	57.57634
LON	Longitude in degrees	17.59554
ALT	Pressure altitude in feets	5000
TRACK	Track of aircraft in degrees [0,360)	35
VELH	Horizontal velocity of aircraft in knots	464
VELV	Vertical velocity of aircraft in ft/min	-1344
SIGS	Signal strength in mV	840
SIGQ	Signal quality in mV	72
FPS	Number of raw MODE-S frames received from aircraft during last second	5
RES	Reserved for future use	
CRC	CRC16 (described in CRC section)	2D3E

Table 6: Detailed message description.

Value	FLAG type	Description
0x0001	PLANE_ON_THE_GROUND	The aircraft is on the ground
0x0002	PLANE_IS_MILITARY	The aircraft is military object
0x0004	PLANE_ALTITUDE_BASED_ON_GNSS	The aircraft has no pressure altitude, but altitude based on GNSS
0x0100	PLANE_UPDATE_ALTITUDE	During last second altitude of this aircraft was updated
0x0200	PLANE_UPDATE_POSITION	During last second position (LAT & LON) of this aircraft was updated
0x0400	PLANE_UPDATE_TRACK	During last second track of this aircraft was updated
0x0800	PLANE_UPDATE_VELO_H	During last second horizontal velocity of this aircraft was updated
0x1000	PLANE_UPDATE_VELO_V	During last second vertical velocity of this aircraft was updated

Table 7: Flags description.

If data of any field of frame is not available, then this field is empty. For example:

```
#A:4CA948,300,,2122,52.99750,13.76526,37000,169,442,0,814,72,3,,6F1C\r\n
#A:424313,,,2362,52.43431,14.84535,37000,65,456,0,806,61,0,,6843\r\n
```

NOTE: SIGS and SIGQ fields are updated based on raw MODE-S frames. They are calculated from frames received in last second. If there were no receiver frames (FPS=0), those fields will not be updated.

NOTE: SIGS is measured based on analog RF signal. This signal has DC offset of about 700mV.

7.3 Statistics message

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

```
#S:CPU,RES,RES,FPSS,FPSAC,TSCAL,CRC
```

#S	Statistics message start indicator	Example
CPU	CPU load in %	12.1
RES	Reserved for future use	-
RES	Reserved for future use	-
FPSS	Number of MODE-S frames received in last second	3
FPSAC	Number of MODE-A or MODE-C frames received in last second	35
TSCAL	Calibration value for TS field in raw frames	13999415
CRC	CRC16 (described in CRC section)	2D3E

Table 8: Statistics message description.

NOTE: TSCAL field is available when precise PPS signal from GNSS source is applied to module to 1PPS pin.

7.4 Calibration of raw frames

To get precise time, TS field from raw frames must be calibrated using TSCAL field from statistics message. This allow obtaining precise time which have passed between most recent PPS pulse and reception of that particular frame.

$$TS_{CALIB}[ns] = \frac{TS}{TSCAL}\frac{1s}{10^9}$$

8 RAW protocol

This protocol is dedicated for raw Mode-A/C/S frames acquisition. In this special mode of operation, output frames are not processed, nor validated in any way. All processing, checksum validation, etc. must be done on user's side. All raw frames, regardless of type, start with '*' and end with ';' ASCII characters, whereas their content is encoded in hexadecimal format, MSB first. At the end, extended fields are appended to frame.

```
*RAW_FRAME;(SIGS,SIGQ,TS)\r\n
```

Var.	Description	Example
SIGS	Signal strength in mV	840
SIGQ	Signal quality in mV	72
TS	Timestamp for multilateration. Time from last PPS pulse in hex format.	20CB3

Table 9: Extended messages description.

NOTE: To use multilateration TS value must be calibrated using calibration value from statistics message.

NOTE: TS field is available when precise PPS signal from GNSS source is applied to module to 1PPS pin.

8.1 Mode-S raw frames

Short and long frames consist accordingly of 7 or 14 data bytes. Examples of raw MODE-S frames:

- Short frame: `*5D4B18FFFC710B;(899,58,20CB3)\r\n`
- Long frame: `*8D4CA7E858B9838206BA422BBD7B;(995,164,20CB3)\r\n`

8.2 Mode-AC raw frames

Each frame consists of 2 data bytes. Bits of first 3 nibbles represent data pulses of MODE-A/C frames in the following order:

```
C1, A1, C2, A2, C4, A4, B1, D1, B2, D2, B4, D4
```

The fourth nibble has least significant bit set, which indicates presence of SPI pulse:

- 0 - SPI pulse not present in frame
- 1 - SPI pulse present in frame

Examples of raw MODE-AC frames:

- `*6610;(979,151,20CB3)\r\n`
- `*E3E0;(995,167,20CB3)\r\n`

NOTE: It is impossible to reliably distinguish between MODE-A and MODE-C frames based only on received signal on 1090MHz.

9 MAVLink protocol

TT-MC1 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. Aircrafts are encoded using ADSB_VEHICLE message. MAVLink messages has standard format, which is well described on this protocol webpage (ardupilot.org/dev/docs/mavlink-commands.html). 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_type	uint8_t	Type from ADSB_ALTITUDE_TYPE enum
altitude	int32_t	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
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
flags	uint16_t	Flags to indicate various statuses including valid data fields
squawk	uint16_t	Squawk code

Table 10: MAVLink ADSB_VEHICLE message description

10 ASTERIX protocol

TT-MC1 can be switched to use ASTERIX binary protocol. This can be achieved by altering PROTOCOL setting. When ASTERIX protocol is used, module is sending list of aircrafts every second. Aircrafts are encoded using I021 ver. 0.23 message. Also, once per second device sends I023 Ground Station Status frame as a hearbeat.

For further reference of reading ASTERIX frames see relevant documentation:

- I021 messages: CAT021 - EUROCONTROL Specification for Surveillance Data Exchange Part 12: Category 21
- I023 messages: CAT023 - EUROCONTROL Specification for Surveillance Data Exchange Part 16: Category 23

11 GDL90 protocol

TT-MC1 can be switched to use GDL90 binary protocol. This can be achieved by altering PROTOCOL setting. When GDL90 protocol is used, module is sending list of aircrafts every second. Aircrafts are encoded using Traffic Report (#20) message. Also, once per second device sends Heartbeat (#0), Ownship Report (#10) and Ownship Geometric Altitude (#11) messages.

For further reference of parsing GDL90 frames see relevant documentation: GDL90 Data Interface Specification

12 Minimum integration recommendation

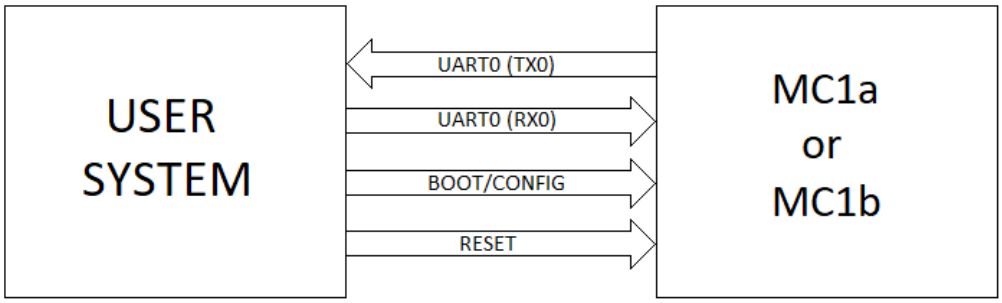


Figure 4: Simplest module integration with user MCU.

13 Mechanical drawing

All dimensions are in mm.

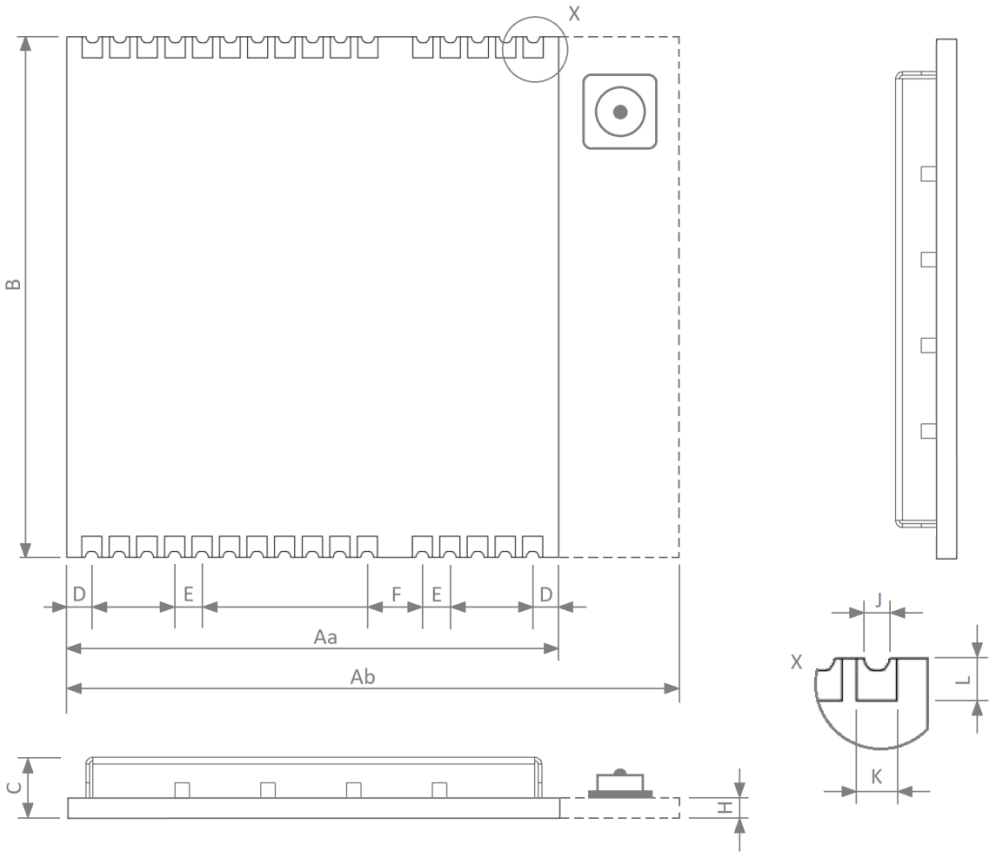


Figure 5: Dimensions.

Symbol	Min. (mm)	Typ. (mm)	Max. (mm)
Aa (TT-MC1a)	17.9	18.0	18.1
Ab (TT-MC1b)	22.4	22.5	22.6
B	18.9	19.0	19.1
C	2.55	2.65	2.75
D	0.9	1.0	1.1
E	0.9	1.0	1.1
F	1.9	2.0	2.1
H	0.6	0.7	0.8
J	0.4	0.5	0.6
K	0.6	0.7	0.8
L	0.7	0.8	0.9

Table 11: Dimensions and tolerances.

14 Recommended layout

All dimensions are in mm.

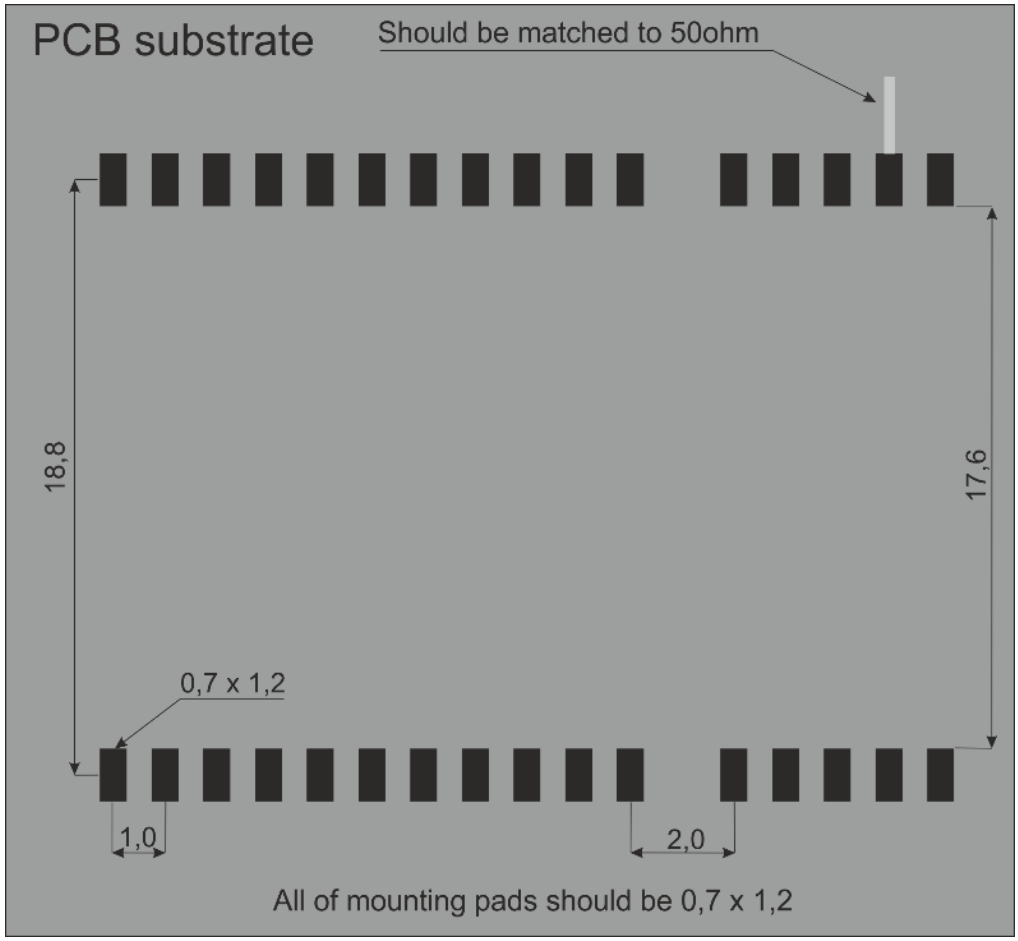


Figure 6: Recommended footprint.

NOTE: In case of TT-MC1a the RF input (pad 18) should be matched to 50ohm.

15 Revision history

Date	Revision	Changes
10-May-2018	1	Initial release.
24-May-2018	2	Corrected inconsistencies.
05-Nov-2019	3	Described ASTERIX and GDL protocols, changed `AT+PROTOCOL` description, described `AT+SETTING?=` syntax and `AT+HELP` command

Table 12: Document revision history.

Please read carefully

Information contained in this document is provided solely in connection with Aerobits products. Aerobits reserves the right to make changes, corrections, modifications or improvements to this document, and to products and services described herein at any time, without notice. All Aerobits products are sold pursuant to our own terms and conditions of sale. Buyers are solely responsible for the choice, selection and use of the Aerobits products and services described herein, and Aerobits assumes no liability whatsoever, related to the choice, selection or use of Aerobits products and services described herein. No license, express or implied, by estoppel or otherwise, to any intellectual property rights is granted under this document. If any part of this document refers to any third party products or services, it shall not be deemed a license granted by Aerobits for use of such third party products or services, or any intellectual property contained therein or considered as a warranty covering use, in any manner whatsoever, of such third party products or services or any intellectual property contained therein.

UNLESS OTHERWISE SET FORTH IN AEROBITS TERMS AND CONDITIONS OF SALE, AEROBITS DISCLAIMS ANY EXPRESS OR IMPLIED WARRANTY WITH RESPECT TO USE AND/OR SALE OF AEROBITS PRODUCTS INCLUDING, WITHOUT LIMITATION, IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE (AND THEIR EQUIVALENTS UNDER THE LAWS OF ANY JURISDICTION), OR INFRINGEMENT OF ANY PATENT, COPYRIGHT OR OTHER INTELLECTUAL PROPERTY RIGHT. UNLESS EXPRESSLY APPROVED IN WRITING BY AN AUTHORIZED AEROBITS REPRESENTATIVE, AEROBITS PRODUCTS ARE NOT RECOMMENDED, AUTHORIZED OR WARRANTED FOR USE IN LIFE SAVING, OR LIFE SUSTAINING APPLICATIONS, NOR IN PRODUCTS OR SYSTEMS WHERE FAILURE OR MALFUNCTION MAY RESULT IN PERSONAL INJURY, DEATH, OR SEVERE PROPERTY OR ENVIRONMENTAL DAMAGE.

Information in this document supersedes and replaces all previously supplied information.