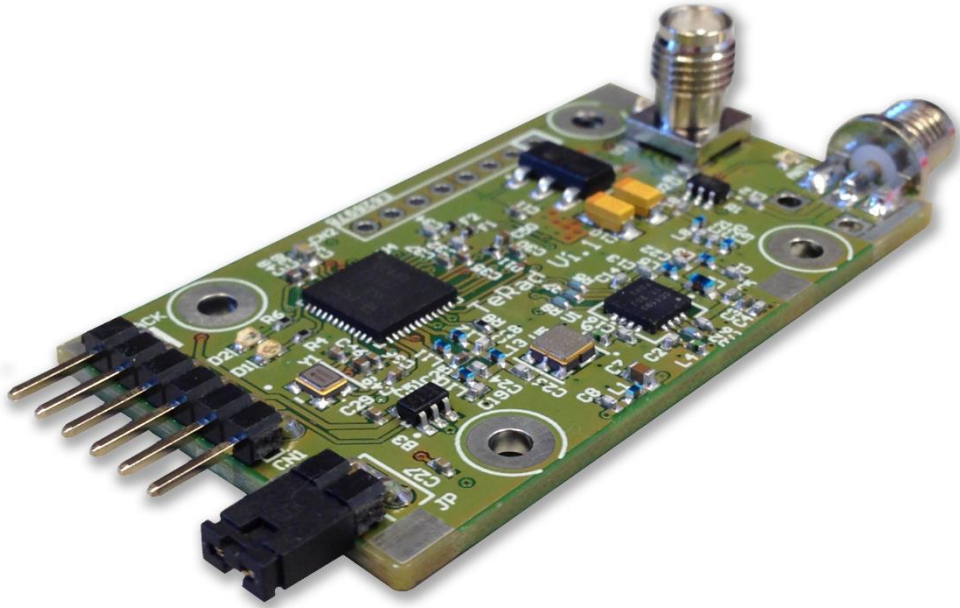


AMORES TeRad v1.1

Quick Start Guide



AMORESROBOTICS
AUTONOMOUS MOBILE REMOTE SENSING

Contents

1	Overview	3
2	Specification	3
3	Connecting power and I/O signals.....	3
4	Connecting to PC.....	4
5	Host connection	4
5.1	Flow control operation	5
5.2	Throughput and latency considerations	5
6	Using the Software Updater.....	5
6.1	Obtaining the latest build of the Software Updater	5
6.2	Running the Software Updater	6
6.2.1	Select build	6
6.2.2	Setup options	7
6.2.3	Flash device	7
7	Radio setup options/settings	9
7.1	Overview	9
7.2	Operating mode	12
7.3	Radio setup	13
7.4	Radio channel	15
7.5	Link address	15
7.6	Error correction	15
7.7	RF booster	16
7.8	Pre-boost TX power	16
7.9	Antenna	16
7.10	Datagram framing	17
7.11	UART baudrate	17
7.12	UART parity	17
7.13	UART stopbits.....	17
7.14	UART flow control	17
8	Recommended antennas	17
8.1	$\lambda/2$ dipole	17
8.2	$\lambda/4$	18
9	Range	18
10	Mechanical details	18

1 Overview

The aim of this document is to contain all relevant technical information about the AMORES Telemetry Radio. AMORES TeRad is a state-of-the-art, high speed telemetry radio intended for professional UAV applications. It enables 2-way communication between a central computer based station and the autopilot. It uses the license free 868 MHz ISM band with a maximum RF output power of 26 dBm.

2 Specification

Supply Voltage	5VDC
Maximal Current Consumption	365 mA
Maximal Output Power	26dBm
Modulation	MSK, GFSK
RF data transmission bit rate	17-185 kbit/sec
Nominal Frequency	869 MHz
Antenna connector	Standard 50 Ohm SMA connector
Estimated Range	10 km
Operating Temperature	from -40°C to +85°C
Size	54×3 mm
Weight	12 g

3 Connecting power and I/O signals

On the TeRad board, the CN1 connector provides the UART connection and power supply as per the following table. Signals named from radio's perspective, eg. RXD means the radio receives data on this line, which must be driven by the host.

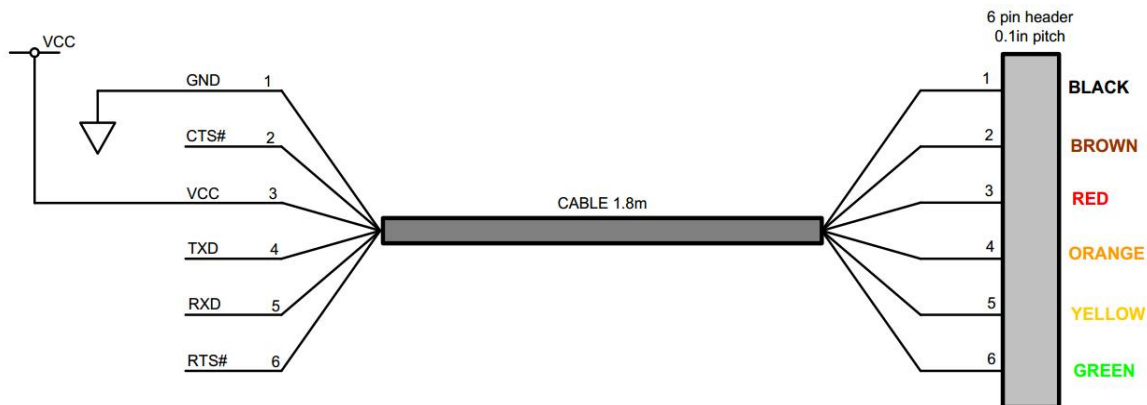


Pin	Name	Type	Signal level
1	Ground	GND	Ground
2	CTS	Output	3,3V/5V
3	VCC	Power	5V
4	RXD	Input	3,3V/5V
5	TXD	Output	5V
6	RTS	Input	3,3V/5V

4 Connecting to PC

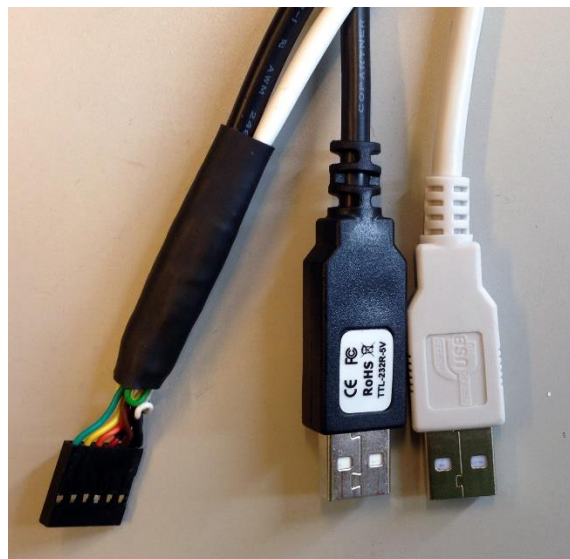
For the connection with PC, using an USB FTDI cable is recommended.

The following picture shows standard color codes as used on FTDI cables. Signals are named from host computer's perspective, eg. the connected system must drive the RXD signal, which is used by the host to receive data.



Because, of the current limit of the FTDI chip, another cable should be used for supplying the device.

Recommended solution:



If it is needed, FTDI driver can be downloaded from <http://www.ftdichip.com/FTDrivers.htm>

5 Host connection

The host is connected to the telemetry radio via UART (through the CN1 connector). The UART connection implemented in tmradio has the following properties:

- Full duplex operation: independent receiving and sending of frames;
- Configurable speed: 9600, 19200, 38400, 57600, 115200, 230400 Baud;
- Configurable parity (None, Even, Odd);
- Configurable number of stopbits (1 or 2);
- Optional RTS/CTS flow control (strongly encouraged).

5.1 Flow control operation

When enabled, RTS/CTS flow control is used by tmradio. It must also be supported by the host. The flow control consists of two active-low signals RTS/ and CTS/:

RTS/ must be driven by the host and must be low when the host is prepared to receive further data frames (this should be the default state of the RTS/ line). It must be pulled high (inactive) when the host needs to temporarily inhibit tmradio from sending data. tmradio will then buffer data and resume sending frames over UART when RTS/ is pulled low by the host. It is guaranteed that tmradio will send at most one extra UART frame after the host has pulled RTS/ high.

CTS/ is driven by tmradio and is low when it is capable of receiving further data. It will be pulled high when tmradio's inbound buffer has less than 64 bytes of free space. The host must suspend sending of data within this amount to prevent data loss over the UART. tmradio will re-activate (pull low) CTS/ when its inbound buffer free space exceeds 256 bytes.

5.2 Throughput and latency considerations

When a pair of radios is communicating in normal mode, they arbitrate the usage of the radio channel (transmission direction) between themselves. Even when there is no user data to be sent over the air, the two radios keep in touch with each other, regularly exchanging the right to immediately start sending data.

The rate at which this exchange is done (5 per sec) determines an initial worst-case latency on the start of transmission: in case the other end has just acquired the right to send when user data starts arriving over the serial link, the arriving data will need to be buffered for 0.2 seconds. The UART receive buffer is 1024 bytes, which at higher baudrates will have easily filled by this time, and some data sent by the host will eventually be dropped. Thus it is important to implement flow control between the radio and host.

Note: this initial latency is not equivalent with the general transmit latency of the data transmission, which - once a logical packet is scheduled for transmission - is much, much lower. As long as at least one of the endpoints keeps sending payload, logical packets will be scheduled continuously and the aforementioned arbitration exchange mechanism will not come into play.

Also, as long as only one of the endpoints has payload to send, the other device will have its right to transmit waived, so the sending endpoint will not experience the aforementioned startup latency even in case of long gaps between blocks of payload to send.

6 Using the Software Updater

The PC-based software application **atrswu** is used for flashing the TeRad device with tmradio software. It handles factory (first) programming as well as in-field updating of already programmed units.

The device must be connected to the host running atrswu via its regularly used serial interface (see the CN1 connector). Note: when the Entry method pulse is used (see below; this generally only happens at factory programming), additional two connections are required to the pins RST and TEST.

6.1 Obtaining the latest build of the Software Updater

Before using it, please make sure that you have the latest release of atrswu. Since each release of atrswu cumulatively contains all versions of tmradio published up to the date of the particular atrswu release, older versions should not be kept around.

Currently atrswu releases are uploaded to the AMORES autobuild repository, accessible via <https://nuku.icss.hu> (user authentication required). After logging in, browse to AMORES -> Repository -> autobuild and select one of the below editions:

- atrswu-win32.exe: Windows (32 and 64 bit; Windows XP or later)
- atrswu-Linux-i686: Linux (32 bit)
- atrswu-Linux-x86_64: Linux (64 bit)

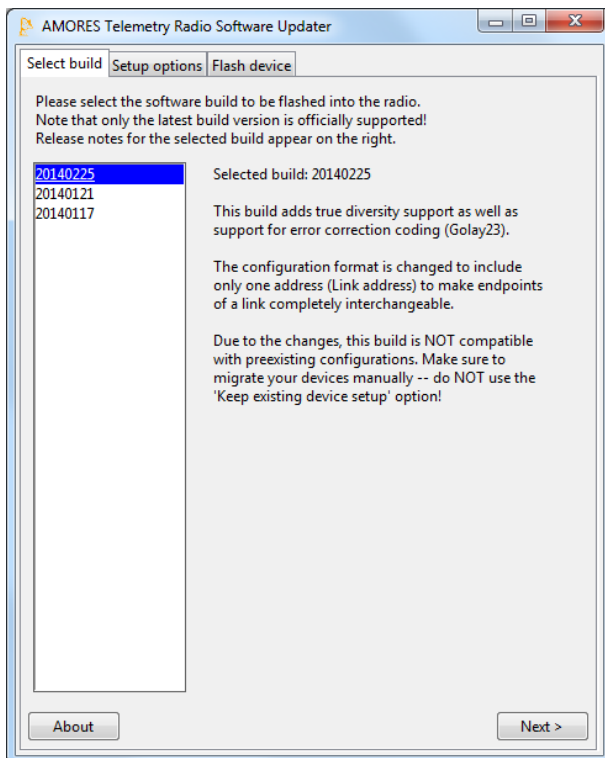
Download and run the executable suitable for your environment. Each of the above editions is a standalone self-extracting executable. Apart from decompressing to the system's temporary folder (automatically deleted on application exit), they do not install anything to the file system or operating system registry.

6.2 Running the Software Updater

The software updater application user interface is structured as a simple step-by-step wizard offering three steps.

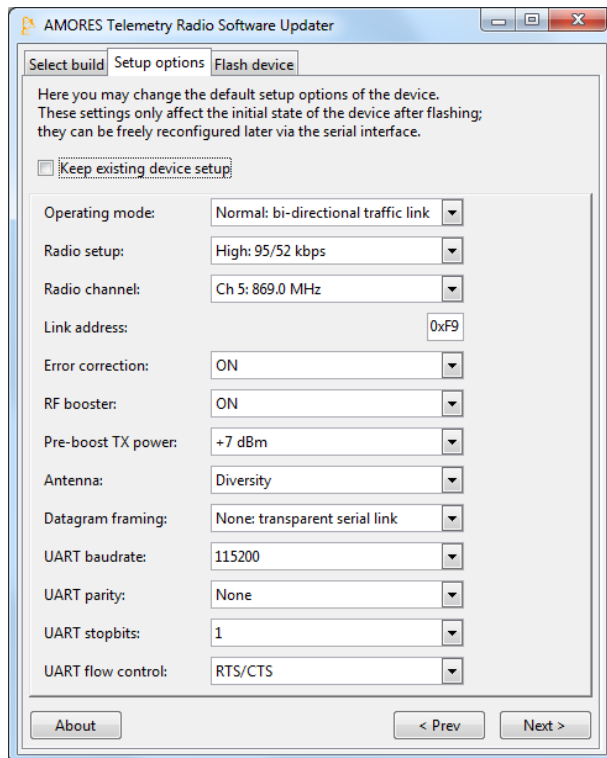
The steps may be sequenced through in order via the *Next >* and *< Prev* buttons, or may be visited in any order via clicking on their labels at the top.

6.2.1 Select build



As the first step the TeRad build to be downloaded into the device must be selected. Builds are presented in an ordered list (latest at the top). The first list item, which corresponds to the latest build, is selected as a default. When selecting a build, its changelog appears to the right of the list selector. This is a description of changes that went into the software compared to the previous version. User visible changes to and incompatibilities with earlier versions will be explicitly noted here.

6.2.2 Setup options



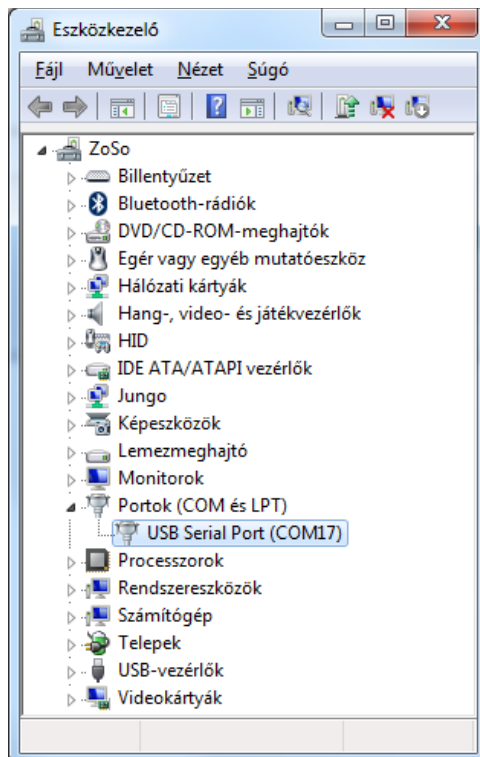
You may change the default setup of the device you are about to flash.

You may also opt to keep the existing setup of your device, in which case the device area where the setup is stored won't be overwritten. You may only do this if the device already contains a valid setup (that is, it is already programmed and functioning well). A further requirement of this option is that the newer version must not introduce any incompatible changes to the device setup, as compared to the existing version. You are informed about any such incompatible changes via each build's changelog at the first step.

Note: due to the fact that the software updater does not know the build version already in the device (if any), you must check this condition manually yourself.

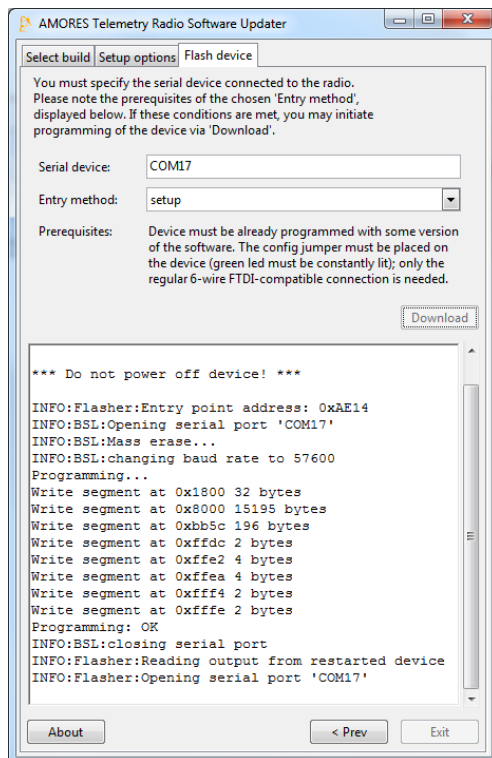
6.2.3 Flash device

To be able to flash the device, you must first connect it to an available serial port of your computer. To enumerate serial ports on Windows, open the Device manager:



As you can see, in this case we are using a USB-serial converter that provides us with a virtual serial port named COM17. Please make sure that the serial port is not opened by any other program (eg. terminal emulator). Now we may proceed to the actual flashing of the device.

Important: Always place a Setup jumper on the device before powering it with the intention of a software update. After powering the device, the Green Led must be continuously lit, signalling that the jumper was detected and the device entered Setup mode.



There are two so-called "entry methods" which allow the software updater application to interact with the device. They are useful in different situations and have different prerequisites.

Entry method

Prerequisites

setup

Device must be already programmed with some version of the software. Only the regular 6-wire FTDI-compatible connection is needed.

pulse

Device may be blank (or dysfunctional), but the serial interface must have RTS and DTR output pins (FTDI green and grey); connected to board pins TEST (CN2.6) and RST (CN2.7), respectively. The device setup area cannot be preserved in this mode.

Entry method setup is the default. It should be used by the vast majority of users; in fact it is not expected that regular users will ever need the pulse method. However, support for factory programming and recovering from the rare but possible case of a failed download (eg. caused by a power outage) necessitate its inclusion.

The pulse method does not require anything on the part of the device, but requires connections to board pins TEST and RST via a serial adapter that supports not only an RTS, but also a DTR line. This method should be used only if (for example due to an earlier failed programming attempt) the device is in a state where tmradio does not correctly start on power up. It is also the method used for factory programming of a blank device.

Please remember that after a download, each device **MUST** be power-cycled (powered off before being powered on again) before any further use. However, the software updater application itself can

be used to program several devices in a row without restarting it. A common use case is the flashing of a pair of devices to form a link:

- configure and flash the first device;
- disconnect the device from the 6-pin connector, and connect the second device;
- press the Download button again.

The download process itself takes about 10 seconds. Upon completion of a download, the device's setup mode is initiated again and its output is read back into the text area. You should at least skim through this output and verify that the device is now indeed running the selected software build with the configured setup options.

7 Radio setup options/settings

7.1 Overview

On powering up the device with the Setup jumper on, Setup mode is entered. The Green Led is continuously lit while in this mode.

When in Setup mode, the device communicates via the UART connection with the following fixed parameters (regardless of the values of UART configuration items):

- 9600 baud;
- no parity;
- 1 stop bit;
- no flow control.

Therefore, any computer with a serial port and ordinary terminal emulator software can be used to access Setup mode; no special client application needs to be installed.

Immediately after device power up, a banner is printed containing the current build version. A menu listing follows showing setup options with their currently active values. Altering these options is possible by pressing the designated keys in the left column.

```
*****
**      AMORES Telemetry Radio -- Setup Mode      **
**      Firmware v0.99   build Feb 28 2014        **
**      http://amores-robotics.hu                 **
*****

Press key to change      [current setting]

0 - Operating mode      [Normal: bi-directional traffic link]
1 - Radio setup         [High: 95/52 kbps]
2 - Radio channel       [Ch 5: 869.0 MHz]
3 - Link address        [F9]
4 - Error correction    [OFF]
5 - RF booster          [ON]
6 - Pre-boost TX power  [+7 dBm]
7 - Antenna             [ANT 1]
8 - Datagram framing    [None: transparent serial link]
9 - UART baudrate       [115200]
A - UART parity         [None]
B - UART stopbits       [1]
C - UART flow control   [RTS/CTS]

W - Write configuration to FLASH and exit
X - Exit without committing changes

>
```

For example, to change the **Radio channel** setting, press 2. A single keystroke is all that is needed; don't press Enter. A submenu is presented showing the setup item, its current value, and appropriate choices for the new value:

```
> 2
==[ Radio channel          [Ch 5: 869.0 MHz]
0 - Ch 0: 868.0 MHz
1 - Ch 1: 868.2 MHz
2 - Ch 2: 868.4 MHz
3 - Ch 3: 868.6 MHz
4 - Ch 4: 868.8 MHz
5 - Ch 5: 869.0 MHz
6 - Ch 6: 869.2 MHz
7 - Ch 7: 869.4 MHz
8 - Ch 8: 869.6 MHz
9 - Ch 9: 869.8 MHz
```

Again, pressing any of the offered numbers on the left will choose the indicated setting for Radio channel (the setup item we're now in). Let's set the channel to **Ch 3** by pressing 3:

```
> 3
--> Radio channel          [Ch 3: 868.6 MHz]

Press key to change        [current setting]

0 - Operating mode         [Normal: bi-directional traffic link]
1 - Radio setup             [High: 95/52 kbps]
2 - Radio channel          [Ch 3: 868.6 MHz]
3 - Link address           [F9]
4 - Error correction        [OFF]
5 - RF booster             [ON]
6 - Pre-boost TX power     [+7 dBm]
7 - Antenna                [ANT 1]
8 - Datagram framing       [None: transparent serial link]
9 - UART baudrate          [115200]
A - UART parity            [None]
B - UART stopbits          [1]
C - UART flow control      [RTS/CTS]

W - Write configuration to FLASH and exit
X - Exit without committing changes

>
```

Subsequent output starts with acknowledging our change, followed by a full reprint of the main menu where the change to the current setting is now also reflected. Now we can change another item, or go on to save our changes to Flash (or abandon the changes we made).

When changing the `Link` address setting, the input method is different, but only slightly so. Let's change the address to `a5`. This begins by pressing `3`, since we need to select the setup item `Link` address first. Then just key in `a5` (press `a`, then press `5`) without hitting Enter:

```
> 3
==[ Link address          [F9]
Enter new value as a 2-digit hex number:
> a5
--> Link address          [A5]
```

You could have entered `A5` just as well - note that in the acknowledgment following your input, the value is already parsed and printed in upper case. Note that in case of a single-digit address, you must type a leading `0` so the input will be two digits.

Please note that individual changes are not automatically saved – you must explicitly press `w` or `W` in the main menu to overwrite the setup in permanent Flash storage, so it will be effective from the next power up. But only do this if you are satisfied with all settings – after writing the setup to Flash, the software will exit and you will need to power cycle the device to get into a functional Setup mode again. The red LED will light up for the duration (a brief moment) of writing the setup to Flash.

As you have already guessed, the menu is not case sensitive – for example, you can type either `a` or `A` to access the UART baudrate setting.

Non-parseable (garbage) input is handled gracefully; characters not allowed at the given point are rejected with messages

Selection out of range (at menu choices);

Invalid character input (at address entries);

Eh? (in the main menu).

Now that we've covered general usage of the Setup mode, let's dig deeper into each item.

7.2 Operating mode

```
0 - Normal: bi-directional
    traffic link
1 - Diagnostic: link quality
    assessment
2 - RF test: continuous
    transmission
```

The radio software may be operating in one of the above modes. To use the radio for payload transmission (normal usage) choose normal mode.

The second setting may be used to run the radio in a special diagnostic mode. In this mode, the two endpoints generate dummy traffic between each other. The UART interface is used to report statistics about several aspects of operation:

- received/transmitted bytes;
- number of errors corrected by ECC;
- fragment transmission retries
- dropped (lost) datagrams so far;
- values of some internal radio control variables;
- antenna selection counters;
- RSSI statistics;
- link distance (estimated from propagation delay);
- device temperature.

Some of these values are reported for both endpoints (values of the remote device are transmitted and displayed as well as the local values). Other values are displayed for the local endpoint only. Please see section 8 for full reference.

This mode may be used for link quality testing (measurement of signal levels, payload capacities etc.) and general testing and debugging of device operation. Naturally, both endpoints must be in this mode.

RF test mode puts the radio into an RF measurement mode in which no data exchange takes place at all. The radio will be transmitting an infinite sequence of random bytes, effectively driving the RF circuits with a stationary signal. This is used for RF measurements only, when the antenna outputs are connected to RF equipment with proper impedance termination. Do not connect the radio to antennas (or leave the antenna ports unconnected) when running in this mode! The latter may actually cause permanent damage to the device. If you are not really sure what you are doing, do not use RF test mode at all!

7.3 Radio setup

0 - Extreme: 185/104 kbps
1 - High: 95/52 kbps
2 - Medium: 46/25 kbps
3 - Low: 32/17 kbps

This option controls the actual radio modem: air speed, modulation format, and lots of accompanying internal settings. The most obvious implications of this setting are payload capacity and usable range. As a quick reference, approximate payload capacities (with Error correction OFF/ON respectively) are displayed as part of the option.

Available radio modes are summarized by the following table. For each setting, the table gives the approximate sustainable net payload capacity and the nominal UART baudrate matching that capacity with and without error correction

Designation	Modulation	Air speed [kBaud]	Error correction	Payload capacity [kbit/s]	Matching UART speed [kBaud]
Extreme	MSK	274.5	OFF	185	230.4
			ON	104	115.2
High	2-GFSK	136	OFF	95	115.2
			ON	52	57.6
Medium	2-GFSK	67.6	OFF	46	57.6
			ON	25	38.4
Low	2-GFSK	48	OFF	32	38.4
			ON	17	19.2

NB.: When matching payload capacity and UART speed setting, keep in mind that the actual UART capacity is at most 80% of its line speed (assuming no parity and 1 stop bit as usual) and may be as low as 67% (in case of 2 stopbits with parity enabled). Taking this into account, it is easy to see that the available payload capacity over the radio closely matches the corresponding UART speed (or, in case ECC is used, even exceeds it). The only exception is Medium with ECC on - this is a bit too slow for a 38.4 kBaud UART (which can sustain 30 kbps of payload data) but too fast for a 19.2 kBaud UART.

All the above boils down to a very simple bottom line recommendation: you can easily select the radio mode recommended for you based on the UART speed you demand (per the above table). Note that in this case the UART speed that counts is that which your payload actually requires (nearly or completely saturates) to be transmitted.

Of course, you are not constrained to setting the UART speed as per the above; it is merely an indication of the setting that will provide an even payload capacity throughout your data path with no bottlenecks. You may have legitimate reasons to deviate from these settings; in particular, when interfacing the radio with other hardware (eg. an autopilot or other on-board equipment) you may be forced to use the UART setting of this other device. This sets an upper bound of data that the device is able to communicate to/from the radio; the actual effectively used rate may be much lower.

Please note, however, that in case the UART speed you select is lower than the one matching the radio setting, data may be correctly transmitted over the radio link only to be dropped at the receiving end. This is a problem especially if there is no RTS/CTS flow control over the serial line.

Naturally, higher air speed brings with it not only greater payload capacity, but also reduced receiver sensitivity and an increased susceptibility to noise and interference. The ultimate result is a shorter usable range for the same transmit power and antennas. As a rule of thumb, reverse the radio setting designation to get an indication of its robustness and range: go for a Low setting to obtain Extreme range / robustness, and so on.

As a corollary, it really pays to research the actual payload capacity requirement of your application before deciding upon the radio mode to use. You should always select the lowest setting that is able to accommodate the capacity requirement of your application, since this will give you the best available robustness of transmission and longest range.

In case extra link robustness is required, consider enabling error correction. Please see section 7.6 for further info.

7.4 Radio channel

0	-	Ch 0:	868.0 MHz
1	-	Ch 1:	868.2 MHz
2	-	Ch 2:	868.4 MHz
3	-	Ch 3:	868.6 MHz
4	-	Ch 4:	868.8 MHz
5	-	Ch 5:	869.0 MHz
6	-	Ch 6:	869.2 MHz
7	-	Ch 7:	869.4 MHz
8	-	Ch 8:	869.6 MHz
9	-	Ch 9:	869.8 MHz

7.5 Link address

The radio software supports a point-to-point link protocol. The link address is an identifier number assigned to the link, and is included in the header of each radio packet exchanged by the endpoints. Each radio discriminates incoming packets based on this address. Only packets with a link address matching the device's own setting will be considered; all others will be dropped.

A pair of radios must be configured so both of them are set to the same link address.

Each address is a one-byte value. It is not recommended to use 0x00 and 0xFF (255) due to their conventional use as broadcast addresses. (Technically these values can be used, but it is better to stay away from them).

7.6 Error correction

The software supports a half-rate error correction coding (ECC) scheme to add extra robustness against random bit errors. The ECC used is the well known binary Golay code (famously used by the Voyager spacecrafts).

If used, all data of transmitted radio packets are protected by ECC (the headers as well as the payload fragments) resulting in a significant robustness improvement against random noise. Since there is less need for fragment retransmissions, the realised payload capacity tends to be more constant (sporadic retries do not eat into it).

The Golay encoder emits a 24 bit codeword for each 12 bits of input, thus halving the available payload capacity for each radio setup mode. Actually, it is a bit better than that (see values in section 7.3) for the reason that radio packet preambles and inter-packet guard times need not be duplicated.

The upside is that when decoding, up to three bit errors in each 24 bit codeword are corrected. This amounts to tolerance against a whopping 12.5% BER (bit error rate) value – provided, of course, that the bit errors are evenly distributed across the received data.

Regardless of whether ECC is used or not, the received data is still error checked by means of 16 bit CRC sums for each 64 bytes of payload. Any fragment with a CRC error is discarded by the receiver, and very likely re-transmitted by the sender. Thus, the chance of outputting corrupt data to the host is negligible.

7.7 RF booster

- | |
|---------|
| 0 - OFF |
| 1 - ON |

This setting controls whether the CC1190 booster chip on the board is used. More precisely, this setting controls the HGM (High Gain Mode) input of the CC1190. If HGM is off, the booster outputs with the same power setting as received from the CC430.

7.8 Pre-boost TX power

- | |
|-------------|
| 0 - +10 dBm |
| 1 - +7 dBm |
| 2 - +5 dBm |
| 3 - 0 dBm |
| 4 - -10 dBm |
| 5 - -15 dBm |
| 6 - -20 dBm |
| 7 - -30 dBm |

This setting controls the transmit power of the CC430 which drives the CC1190 booster chip. In case the RF booster is off, this is approximately equal to the transmit power on the antenna output.

In case the booster is on, the optimal setting for driving it with full transmit power is +7 dBm. This will result in full transmit power (+26 dBm / 400 mW) on the antenna output without overdriving the booster input.

Important: all settings up to this point **MUST** be the same for both endpoints!

7.9 Antenna

- | |
|---------------|
| 0 - ANT 1 |
| 1 - ANT 2 |
| 2 - Diversity |

This setting controls which antenna port to use. You may use either one of the antenna ports if you have only one antenna. However, it is strongly recommended to use diversity mode with two antennas connected. This way the radio will always be able to select the antenna with the better momentary signal strength for receiving each radio packet. This is especially important when the endpoints are moving against each other and the terrain, as is the case with UAVs.

The diversity scheme employs received signal strength (RSSI) measurements on both antennas during the preamble of each incoming radio packet. Thus, each packet payload is received on the antenna with the stronger signal. Since radio packets are short, this provides a frequent re-evaluation of the selected antenna – effectively mitigating the deep signal dropouts that result from multipath fading.

7.10 Datagram framing

0 - None: transparent serial link
1 - AirGuardian
2 - MAVLink 0.9
3 - MAVLink 1.0

7.11 UART baudrate

0 - 9600
1 - 19200
2 - 38400
3 - 57600
4 - 115200
5 - 230400

7.12 UART parity

0 - None
1 - Even
2 - Odd

7.13 UART stopbits

0 - 1
1 - 2

7.14 UART flow control

0 - None
1 - RTS/CTS

8 Recommended antennas


It is recommended to use $\lambda/2$ dipole or $\lambda/4$ antennas on the mobile device, and higher gain YAGI antenna on base station

8.1 $\lambda/2$ dipole



Length	145 mm
Width	8 mm
Reflection Loss (869 MHz)	-21 dBm
Gain	2,15 dBi
Weight	3,7 g

8.2 $\lambda/4$

	Length	27,5 mm
	Diameter	9,5 mm
	Reflection Loss (869 MHz)	-3 dB
	Gain	1,9 dBi
	Weight	5 g

9 Range

10 Mechanical details

