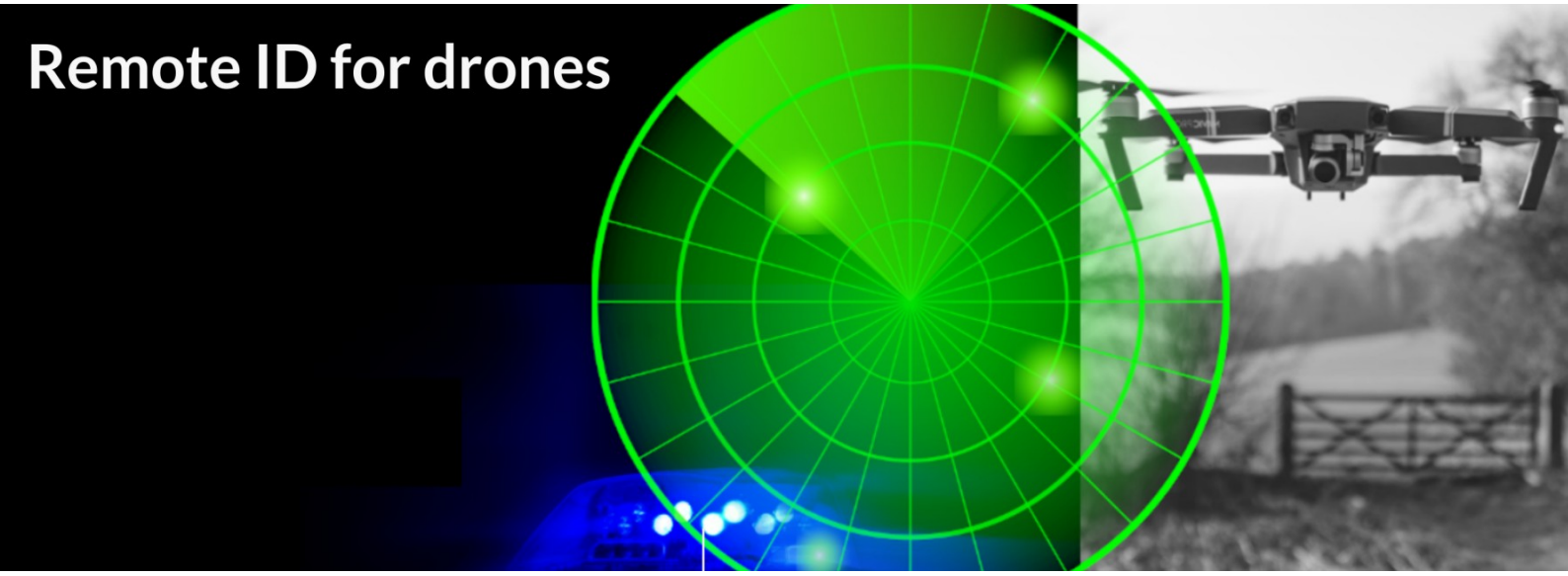
DroneScout - Receiver Manual 230 and 240-series

June 2025 - version 2.3



The latest version of this manual is located here:
https://download.bluemark.io/ds230.pdf

Intended audience: system integrators

Disclaimer: we are not responsible or liable for errors or incomplete information in this document.

Version history

version	date	description
1.0	May 2022	● Initial release
1.1	November 2022	● Added FCC information ● How to install a MQTT broker on the ds230
1.2	December 2022	● Added transmit_mode and aggregate_data options ● Improved update and reboot function
1.3	January 2023	● Describe how to solve DHCP issues in chroot mode ● Add typical RSSI values at 1 meter distance
1.4	March 2023	● Added transmit_mode 2 description
1.5	June 2023	● Added potential risks about transmit_mode 1 ● Describe how to configure a static IP address in section 1.6
1.6	September 2023	● Added the ds240 received to the manual
1.7	January 2024	● Better description how to set a static IP address ● MQTT trouble shooting
1.8	January 2024	● Added section about antenna installation for the ds240
1.9	June 2024	● Update manual to with latest firmware and added more antenna information.
2.0	August 2024	● updated a link ● improved description to set the IP address to static/fixed. (Directly copy/paste of the commands in one block did not work.)
2.1	September 2024	● Description of the new MQTT messages location and network ● Added section about the new ADS-B 1090 MHz and UAT 978 MHz receiver options ● Added a section about the benefits of bandpass filter options in areas with nearby LTE base stations
2.2	February 2025	● Added section about manufacturer/model identification with latest firmware ● Added section how system time can be set using the LTE-add-on (GPS) to allow offline operation. ● Added information about practical detection range
2.3	June 2025	● Minor changes to reflect the current firmware. ● Describe access using the serial console

Contents

1 INTRODUCTION

Thank you for purchasing and using DroneScout products!

The latest version of this user manual may be downloaded at the following link, where the most up-to-date version will be found:
https://download.bluemark.io/ds230.pdf

(Direct/Broadcast) Remote Identification (Remote ID) adds "beacon" capability to drones to broadcast basic information of airborne drones, such as the operator's registration number, drone serial number and current position. The EU and USA are planning new rules that make Remote ID mandatory for drones over 250 grams weight. The beacon information can be used by general public, law enforcement and drones to give better situation awareness of the airspace around them.

BlueMark Innovations BV offers Remote ID transponders and receivers. DroneBeacon is an add-on (transponder) for drones which broadcasts Remote ID beacon signals. DroneScout is a receiver that detects Remote ID signals of nearby drones up to several km distance (in open space). See https://dronescout.co for more information about our products. We also sell a smaller RemoteID receiver called DroneScout Bridge https://dronescout.co/bridge/

1.1 Audience

This document is intended for system integrators that want to use the *DroneScout 230 or DroneScout 240* receiver in their own product. This product is not intended for end users!

1.2 Receiver

The DroneScout receiver family consists of three models:

- *ds230* - Cost-effective basic Remote ID receiver
- *ds240* - Long Range Remote ID receiver, 3x more detection range compared to the ds230.
- *ds240 barebone* - Long Range Remote ID receiver (receiver only). No antennas, antenna cables and surge protectors are provided.

ds230

The DroneScout ds230 receiver consists of an embedded system and several radio-interfaces to collect remote ID signals.

Key specifications:
- Quad-Core Cortex-A53 ARM CPU 1.8 GHz
- 2 GByte RAM
- 8 GByte eMMC flash storage
- 10/100M/1000M Ethernet interface
- Compliant with international standards
  - EU ASD-STAN DIN EN 4709-002
  - USA ASTM Remote ID Standard ASTM F3411-22a-RID-B

- 1x Bluetooth (LE and BLE-Long Range) radio
  - Sensitivity:
    - BLE -97 dBm
    - BLE Long Range -105 dBm
- 2x triple-band WiFi radio: 2.4, 5.2 and 5.8 GHz
  - Sensitivity:
    - WiFi Beacon + WiFi NaN: -85 dBm
- PoE (Power over Ethernet): 802.3af/at
  - connectivity and power
- Power consumption: < 5 W
- Outdoor enclosure IP67
  - 1x Bluetooth antenna connector (N-type)
  - 2x WiFi antenna connector (N-type)
  - omni-directional antennas with 5 dBi gain
- Practical detection area: 80 km^2 / 5 km radius, see also section 1.9. The actual detection area can be bigger or smaller as it depends on the installation location, such as height and obstruction of large buildings.
- Size: 27.2 x 27.6 x 9.6 cm (without antennas).
- Operating temperature: -20°C to +50°C
- Weight: around 1.4 kg (with mast mount 1.9 kg

ds240

The DroneScout ds240 receiver consists of an embedded system and several radio-interfaces to collect remote ID signals.

Key specifications:
- Quad-Core Cortex-A53 ARM CPU 1.8 GHz
- 2 GByte RAM
- 8 GByte eMMC flash storage
- 10/100M/1000M Ethernet interface
- Compliant with international standards
  - EU ASD-STAN DIN EN 4709-002
  - USA ASTM Remote ID Standard ASTM F3411-22a-RID-B
- 1x Bluetooth (LE and BLE-Long Range) radio
  - Sensitivity:
    - BLE -97 dBm
    - BLE Long Range -105 dBm
- 2x 2.4 GHz WiFi radio
- 1x 5 GHz band WiFi radio
  - Sensitivity:
    - WiFi Beacon + WiFi NaN: -85 dBm
- PoE (Power over Ethernet): 802.3af/at
  - connectivity and power
- Power consumption: < 5 W
- Outdoor enclosure IP67
  - 1x Bluetooth antenna connector (N-type)
  - 3x WiFi antenna connector (N-type)
- Antennas (the ds240 barebone receiver does not include these items):
  - 4x surge protectors 0 - 6 GHz
  - 3x 15 dBi omni-directional 2.4 GHz antenna: dimensions 1485*68 mm / 910 gram
  - 1x 15 dBi omni-directional 5 GHz antenna: dimensions 740*68 mm / 440 gram
  - 4x 100 cm N-connector antenna cables
- Practical detection area: 700 km^2 / 15 km radius, see also section 1.9
- Size: 27.2 x 27.6 x 9.6 cm (without antennas).

- Operating temperature: -20°C to +50°C
- Weight: around 1.4 kg (with mast mount 1.9 kg (excluding antennas) Antennas, RF cables, surge protectors are roughly 5 kg in total.

ds240 antenna patterns

2.4 GHz (Bluetooth & 2.4 GHz WiFi)

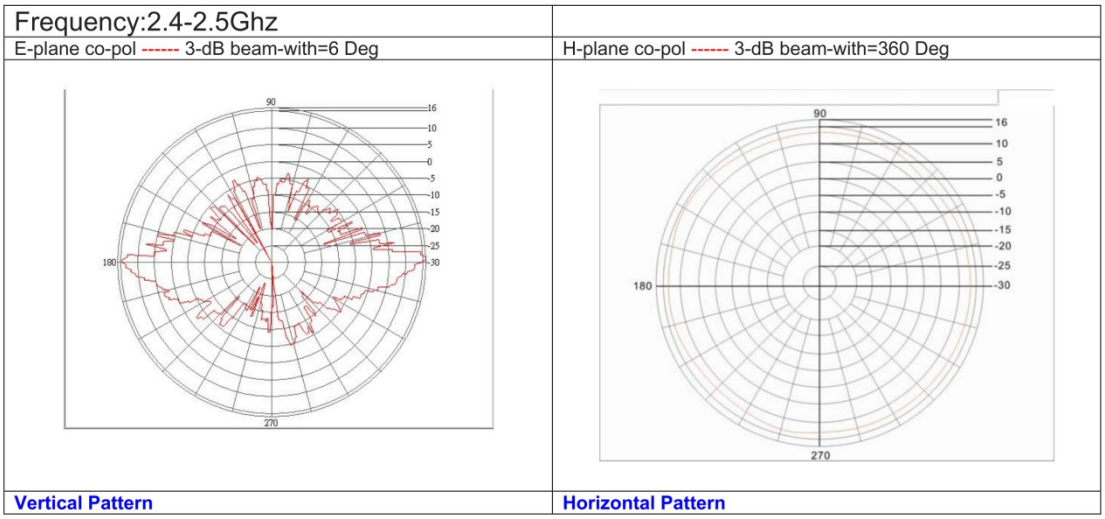


Figure 1 - DroneScout 240 antenna pattern 2.4 GHz

5 GHz

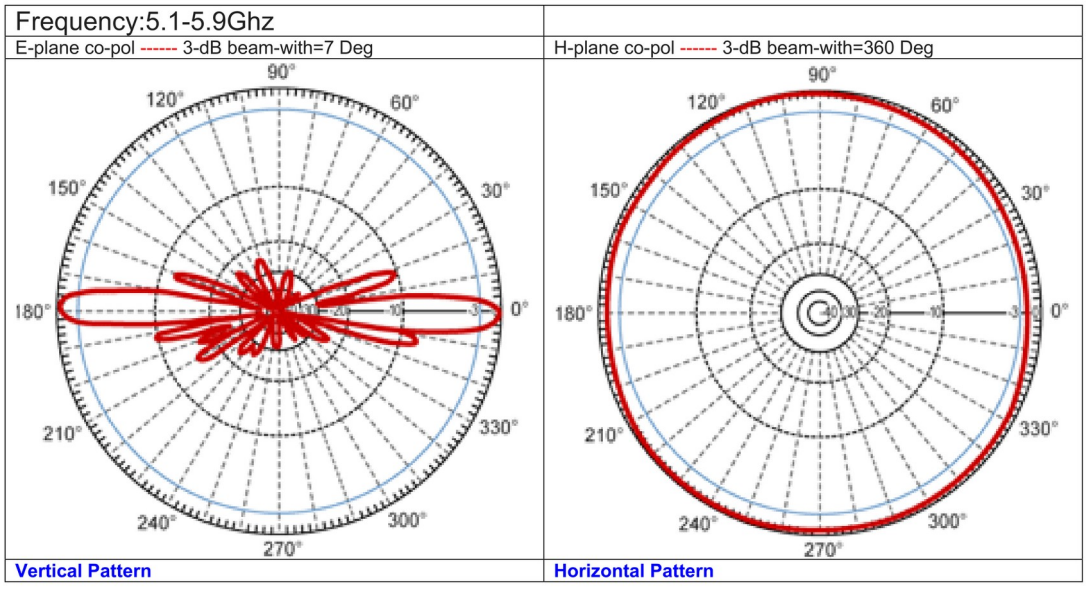


Figure 2 - DroneScout 240 antenna pattern 5 GHz

FCC/CE compliance

The ds230/ds240 receivers have been evaluated and tested by the Dutch lab EMCMCC for compliance with FCC and CE regulation. The receivers can be used in Europe and the USA. Contact us for more information.

Bluetooth scanning

The Bluetooth radio scans continuously for Bluetooth LE and Bluetooth LE Long-Range packets with Remote ID payload.

WiFi scanning

Remote ID signals can be broadcast on several frequencies. This means that the WiFi radio interface will hop every second to a new channel. If no Remote ID signals are detected, both radio interfaces will keep sensing for Remote ID signals on all WiFi channels. If there is a Remote ID signal found, radio 1 will keep hopping and scanning for new Remote ID signals. Radio 2 on the other hand will permanently tune to the channel where a Remote ID signal has been found.

In case of multiple Remote ID signals and multiple WiFi channels, radio 2 will hop every second to another channel where Remote ID signals have been found. If no Remote ID signals are found for 60 seconds, the channel will be removed from the list.

ds240 receiver
The ds240 receiver has a dedicated radio for 5 GHz signals. This means that both 2.4 GHz radio work as above. In case a Remote ID signal is found in the 5 GHz, the 5 GHz antenna will dedicate 50% of the time to tracking the 5 GHz signal and the other 50% scanning for new Remote ID signals. Compared to the ds230, the ds240 has faster detection of Remote ID signals and has a 3x larger detection range.

Remote ID can be broadcast in two WiFi formats: WiFi Beacon and WiFi NaN. WiFi NaN is also called Wi-Fi Aware and those signals can only be found on WiFi channel 6 (2.4 GHz), 44 (5 GHz) and 149 (5 GHz). For this reason, these channels are more frequently scanned for Remote ID signals. WiFi Beacon on the other hand is a format that is similar to the signals that a regular Access Point transmits. Those Remote ID signals can be found on all WiFi channels.

There is a configuration file where you can specify which WiFi channels will be scanned. See the next chapter for more details.

Antenna connectors

ds230

The ds230 uses 3 antennas. Below the N-connector there is a marking: ANT1/2/3/4

ANT 2 - WLAN 1
ANT 3 - WLAN 2
ANT 4 - Bluetooth

Newer ds230 receivers use the following scheme:

ANT 1 - WLAN 1
ANT 2 - WLAN 2
ANT 3 - Bluetooth

ds240
The ds240 uses 4 antennas. Below the N-connector there is a marking: ANT1/2/3/4

ANT 1 - WLAN 1 - 2.4 GHz
ANT 2 - WLAN 2 - 2.4 GHz

ANT 3 - WLAN 3 - 5 GHz
ANT 4 - Bluetooth

Figure 3 - DroneScout 230 receiver

Figure 4 - DroneScout 240 receiver

1.3 DroneScout add-ons

DroneScout receivers provide several optional add-ons. See also https://dronescout.co/

- *DroneScout SDK* - If you want to use our DroneScout technology in your own sensor, use DroneScout SDK. A radio-board and a Docker container that runs our firmware application (x86_64 / ARM64).
- *DroneScout Dashboard* - DroneScout Dashboard is a dashboard installed on a DroneScout receiver to visualize, store and export captured RemoteID signals. It designed for basic single sensor deployments.
- *LTE modem* - DroneScout receivers can equipped with an LTE modem. The LTE modem can be used for connectivity, but also the built-in GNSS (GPS) receiver allows to retrieve the position of the sensor
- *Band pass filter pack* - In urban environments, the spectrum is heavily utilized. Strong signals for instance from nearby LTE base stations limit the sensitivity. Adding these external filters, will remove those nearby strong signals, improving the sensitivity. See also section 1.4.
- *ADS-B receiver* - RemoteID is used for drones, but there are other wireless number plate technologies that identify aircraft. One of the is ADS-B 1090 MHz that is used by commercial airplanes. This option adds an ADS-B receiver to receive those signals as well.
- *UAT receiver* - Besides ADS-B, in the USA (and some other countries) smaller aircraft (at lower altitude) can also use UAT at 978 MHz. This option adds an UAT receiver to receive those signals as well.
- *ADS-L receiver* - In Europe a ADS-B light technology has been introduced called ADS-L. Smaller aircraft and drones can use this wireless number plate technology at 868 MHz. This option is under development. Expected to be released in H1 2025.

ADS-B receiver

The ADS-B receiver add-on is an add-on that can be installed when purchasing a DroneScout receiver or installed afterwards. It consists of:

- SDR USB dongle to decode ADS-B signals
- USB hub for installation purposes
- SMA to N cable to have an antenna N-connector at the outside of the enclosure
- 100 cm N-cable
- Band pass filter with N-male and N-female connectors
- 6 dBi antenna (61.5 cm 2 cm diameter, N-connector) + mounting kit

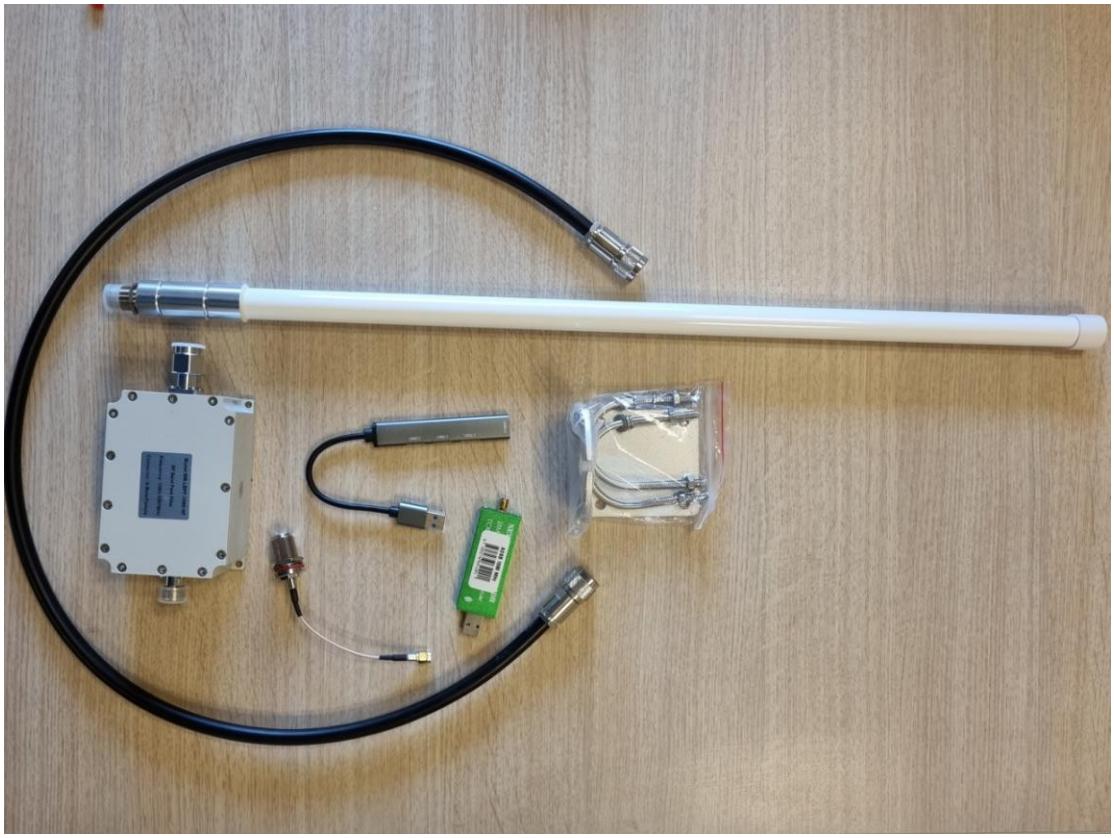


Figure 5 - ADS-B receiver add-on

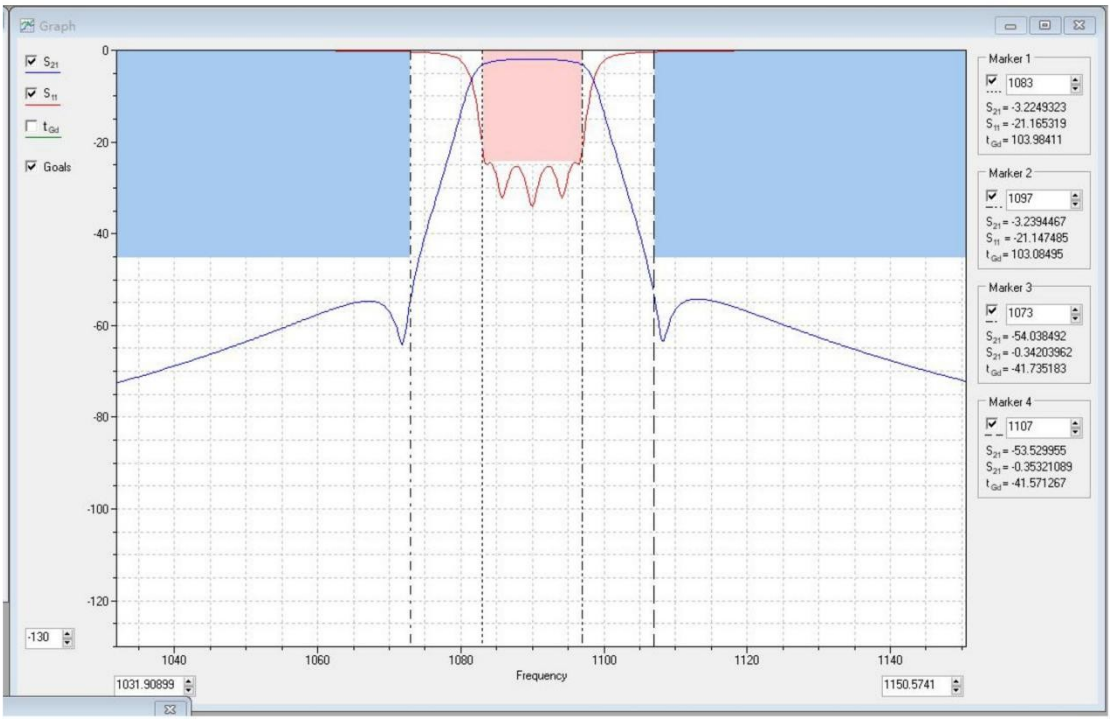


Figure 6 - ADS-B band pass filter

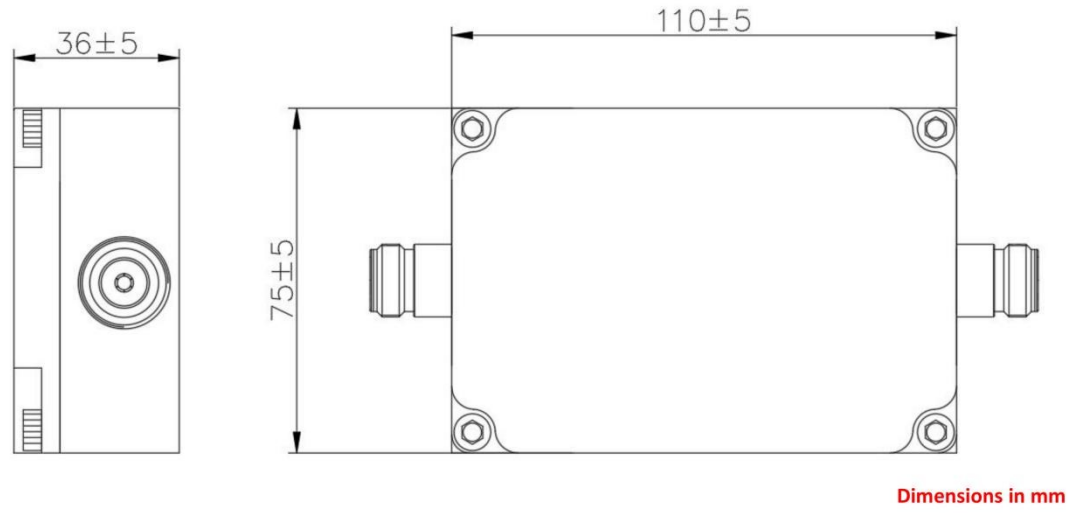


Figure 7 - ADS-B band pass filter dimensions

UAT receiver

The UAT receiver add-on is an add-on that can be installed when purchasing a DroneScout receiver or installed afterwards. It consists of:

- SDR USB dongle to decode UAT signals
- USB hub for installation purposes
- SMA to N cable to have an antenna N-connector at the outside of the enclosure
- 100 cm N-cable
- Band pass filter with N-male and N-female connectors
- 6 dBi antenna (61.5 cm 2 cm diameter, N-connector) + mounting kit

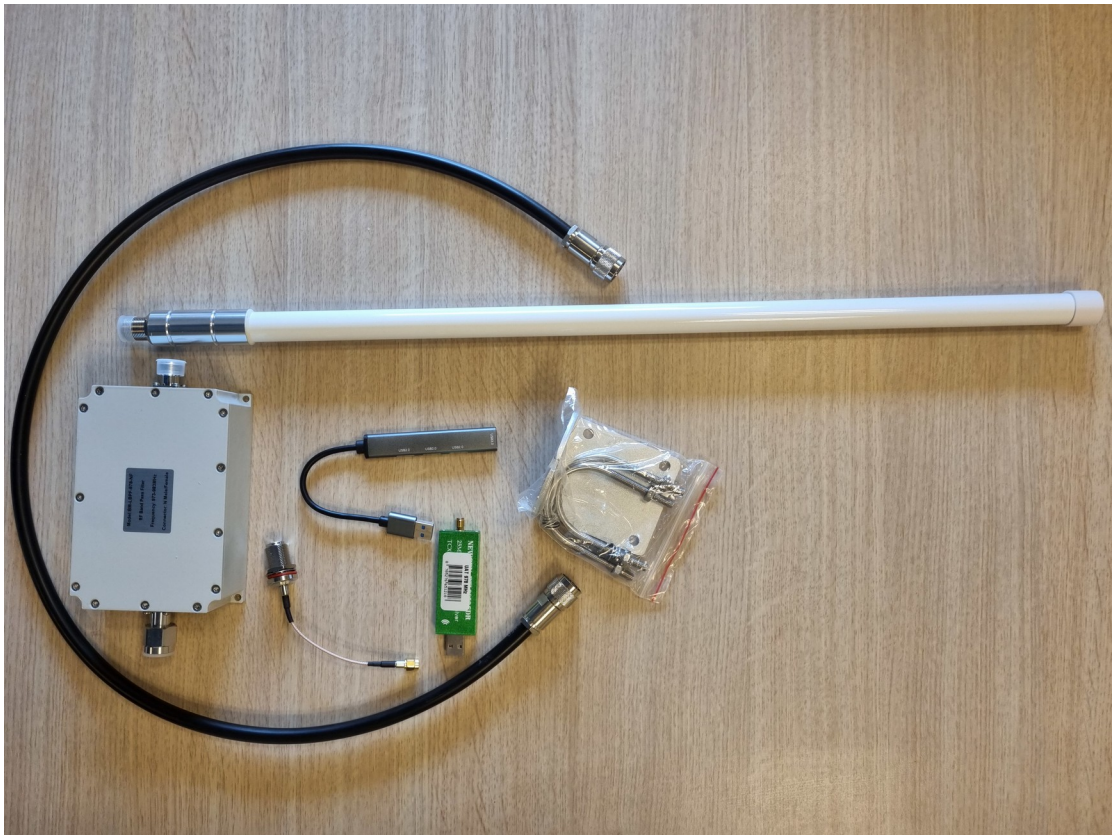


Figure 8 - UAT receiver add-on

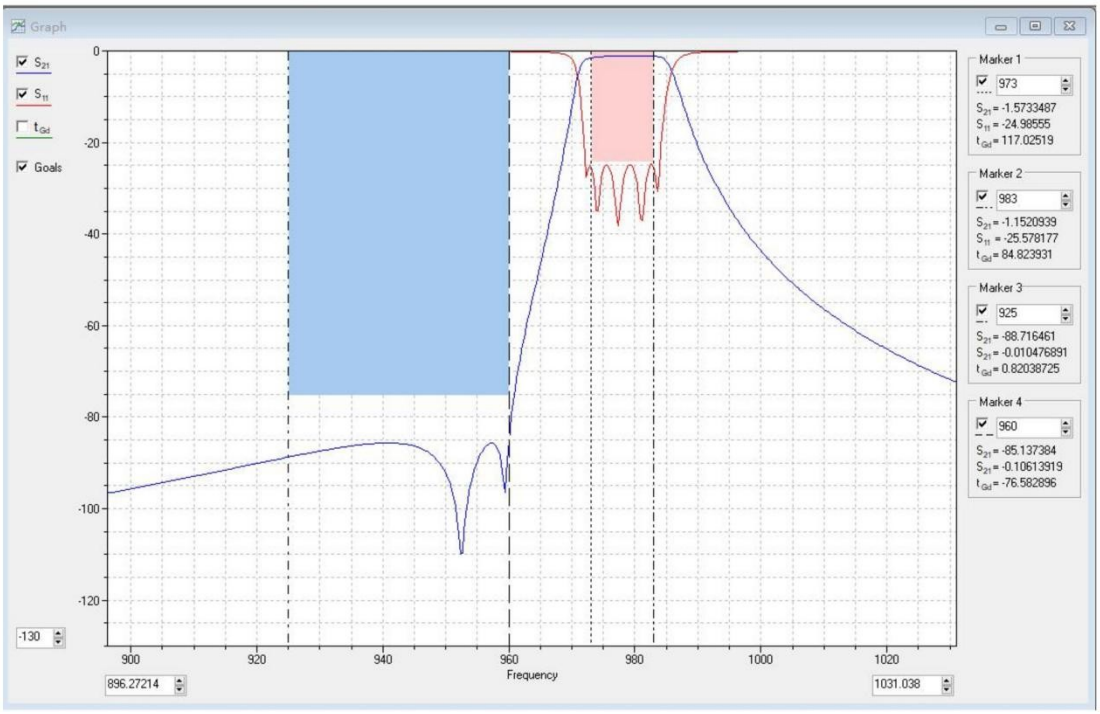


Figure 9 - UAT band pass filter

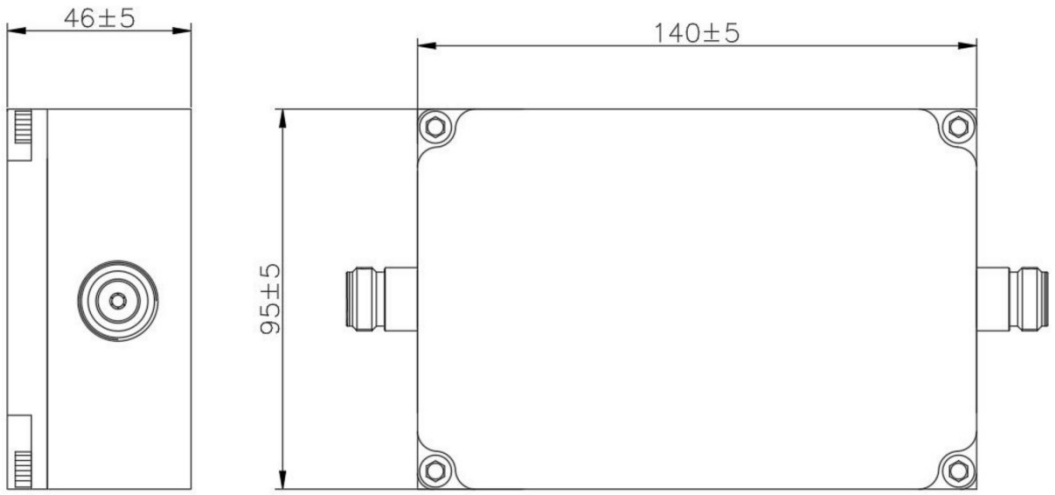


Figure 10 - UAT band pass filter dimensions

1.4 Installation

Summary

- *Install the receiver to a wall or pole mast using the included mast mount.*
- *ds230: connect/screw the 3 antennas to the antenna connectors on the enclosure.*
- *ds240: see ds240 mast mounting section*
  - *Powering up the receiver **without antennas** may **damage** the Bluetooth and WiFi radios.*
  - ***Removing or attaching antennas** when the **receiver is powered up**, will **damage** the Bluetooth and WiFi radios.*
  - *The small antenna is 5 GHz, connect it to ANT3 connector!*
- *Connect the receiver to Ethernet and for outdoor installations make sure this connection is waterproof (by using the included waterproof accessories). This Ethernet cable needs to have power (PoE 802.3af/at) and also connectivity. The receiver acts as a DHCP client in the network.*

Location - The receiver detects remote ID signals from all directions, it is *omnidirectional*. For installation, it is therefore important not to install receivers near the border of the detection area, but instead in the center. It also depends a bit on the situation. If you have nearby WiFi networks, you don't want the receiver nearby it, as (busy) WiFi networks will reduce the detection range. Basically, install the receiver away from areas where there are signals in the 2.4/5 GHz band or large nearby objects (house) that can block detection of signals/drones from that direction.

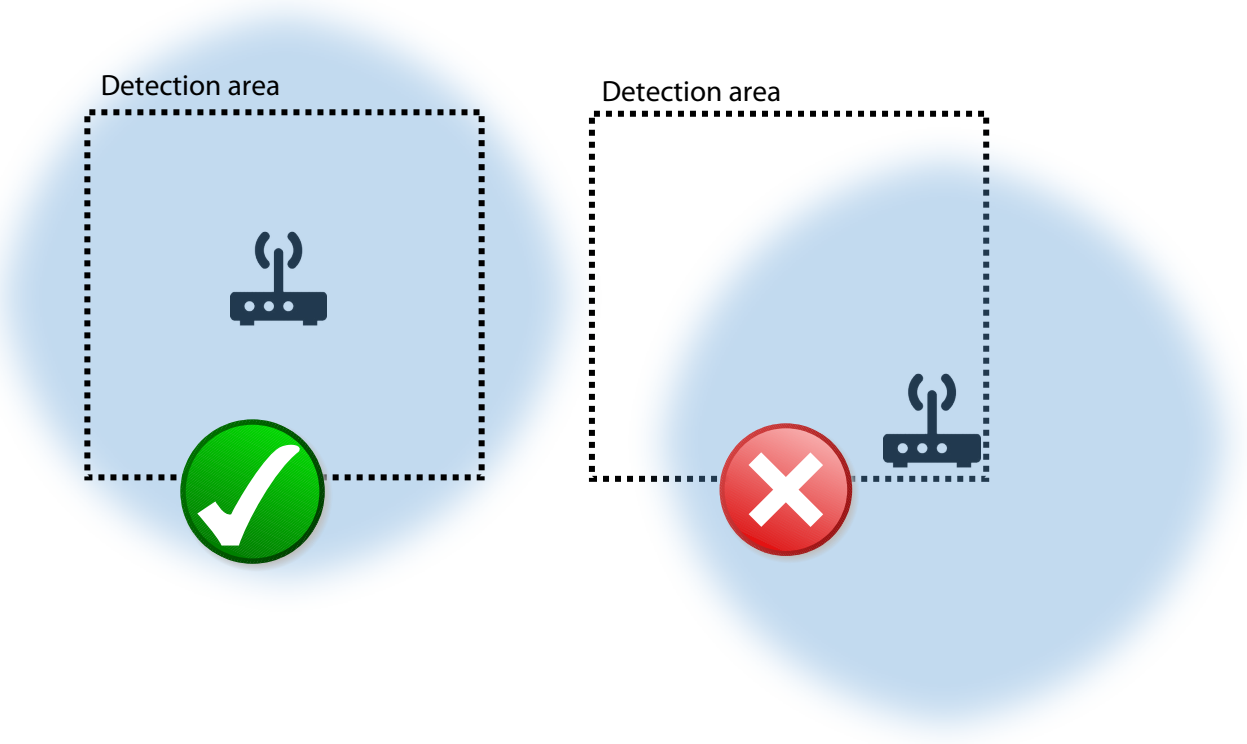


Figure 11 - install the receiver in the center of the detection area.

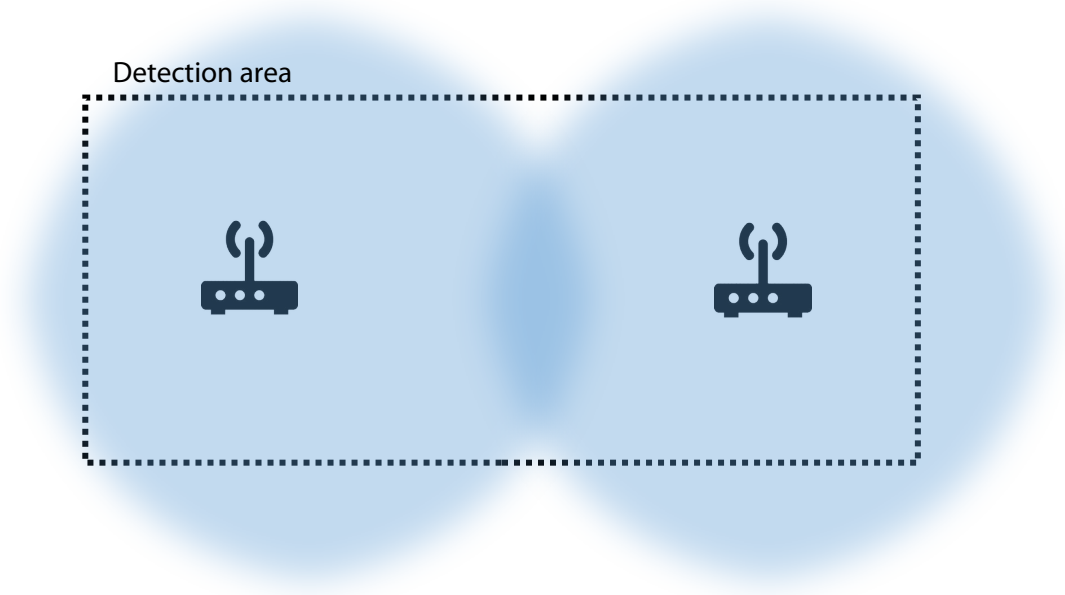


Figure 12 - use multiple receivers to cover the detection area in case the detection area is not square, or circle shape.

Construction materials - Construction materials (wood, concrete) attenuate wireless signals. This means that the detection area is reduced, if a receiver is installed behind or in such an object. This is especially true for the 5 GHz band. Also, it may introduce *blind spots* in the detection area where a drone is not detected. Installing multiple receivers is a solution to avoid blind spots.

For optimal performance install the receiver in open space, not surrounded by nearby objects. As a reference please find below a table describing the RF attenuation by various construction materials.

For instance, if a drone is detected in the 5 GHz band with a 114 mm wooden fence between receiver and drone, the signal strength is 13 dB less compared to no fence. Also, moist air will limit range, because the 2.4 GHz is the frequency where water molecules resonate. Hence, this is why it is a free ISM band. See also the section Urban versus Rural detection range.

Material and thickness	2.4 GHz	5 GHz
Red brick (hollow), 89 mm	5	15
Window glass (uncoated), 6 mm	1	1
Plasterboard, 13 mm	1	0
Wood dry, 114 mm	7	13
Plywood dry, 13 mm	1	0
Bricks (concrete, hollow), 203 mm	11	15
Concrete (C8 mix), 203 mm	35	56
Reinforcing steel mesh (19 mm Ø, 70-mm-grid)	10	3
Reinforced concrete (C8, 19 mm Ø, 70-mm-grid)	37	58

Table 1: RF attenuation by various construction materials. (source c't 9/2021, page 139 using data from William C. Stone, Electromagnetic Signal Attenuation in Construction Materials, 1997)

Height - Preferred installation height is 2 to 40 meters. Installation lower, near the ground, will reduce the detection area as objects in the detection area will block wireless signals more. Installing the receiver higher on the other hand will increase the detection area, but may prevent detecting remote ID signals very nearby.

Angle - The receiver has omnidirectional antennas. Install the receiver with zero angle (vertical plane). This means that the receiver should looks straight ahead. Not down or up under an angle.

Power - The receiver needs power and is powered via Power over Ethernet (PoE), 802.11af. Connect the Ethernet port of the receiver to an PoE capable switch/router to have both power and connectivity.

Connectivity **-** Connect the Ethernet port of the receiver to your router. The receiver needs Ethernet to upload data to the MQTT broker. It can also be used for management purposes. The network name is the serial number that is printed on back of the receiver (dsxxxxxxxxxxx) e.g. ds220300000101.

Urban versus Rural detection range

The detection range of the ds230 and ds240 receivers are limited by noise. Normally, there is thermal noise. Electronics, air have a temperature that introduces noise in the system. This is the so called Johnson–Nyquist noise https://en.wikipedia.org/wiki/Johnson%E2%80%93Nyquist_noise This means also that in cooler areas, the detection range is slightly improved due to lower noise levels. RemoteID signals needs a minimum Signal-to-Noise Ratio (SNR). Signals above the SNR are received. In urban environments, there are a lot of nearby Wi-Fi networks in the 2.4 GHz and other electronic equipment installed that emit low levels of noise. This man-made noise is in urban areas much higher than the thermal noise. This will be the limiting factor for the detection range. It could be that in dense urban areas, the detection range is severely limited to 2 km or less. This is a known fact when radio-communications are deployed. Also, moist air will limit range, because the 2.4 GHz is the frequency where water molecules resonate. Hence, this is why it is a free ISM band.

Real-life field deployment experiences confirm the same. The range of the ds230/ds240 can be severely limited in dense urban areas to less than 1.5 km detection range when the 2.4 GHz is full of WLAN networks. In quiet rural areas, the detection range is up to 18 km for the ds240 (real measurements). Sporadic RemoteID detections can be up to 30 km. Placing the sensor outside

urban areas and directional antennas could be a solution to reduce the number of sensors. Also big metal objects (bridges, towers) can influence the detection range, because metal reflects radio signals and on the other hand block signals behind such object.

Selectivity and the use of external band pass filters

The detection range of the ds230/ds240 is not only limited by noise, but can also be limited due to strong nearby signals at a different frequency. Such strong signal will negatively influence the dynamic range/sensitivity of the inside radios. Bluetooth and WLAN receiver chips are designed for consumer applications where this is not an issue. However, the same radio chips can run into problems when they are installed on high-rise buildings with nearby mobile network base stations (4G/5G). Even if those base stations are installed 200 meter away. A solution to this problem is to mount external band pass filters to the antennas in order to remove those strong signals. To evaluate if a band-pass filter is useful, it is advisable to perform a spectrum survey using a spectrum analyzer (or cost-effective SDR solution) at the DroneScout receiver location to investigate if there are nearby strong signals (1.5 GHz - 3 GHz). See also Section 1.5 for the specifications.

Mast mount

For each receiver a mast mount is provided. It can be used to install the receiver to a mast or directly to a wall. See details below.

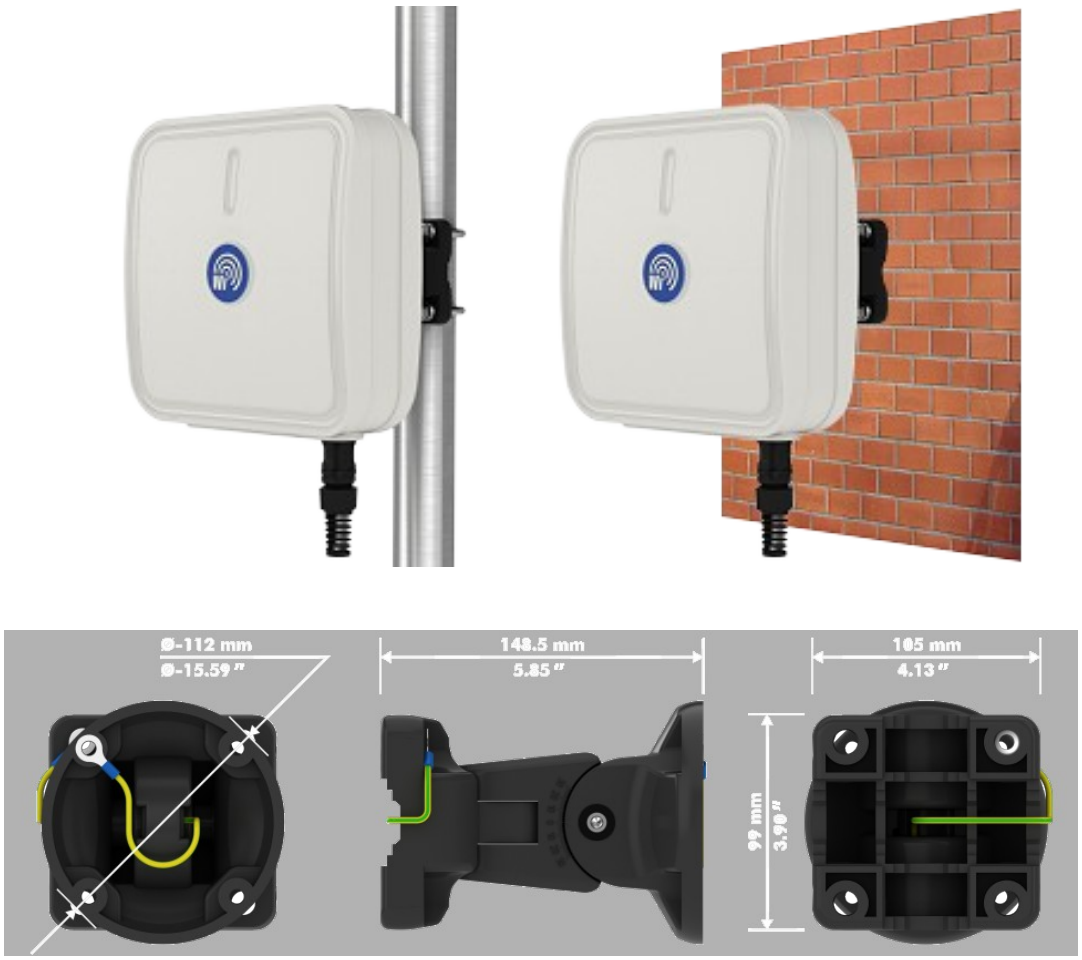


Figure 13 - mast mount to install the receiver to a mast or directly to a wall.

DS240 antenna mounting

For optimal ds240 antenna installation use the guidelines in this link:
https://southwestantennas.com/articles/antenna-spacing-considerations-for-multi-antenna-systems

Summary for optimal antenna installations:

- In case of a metal pole mast, install the antennas as high as possible, so the metal pole mast will not affect the antenna pattern. (The reason is that the metal pole mast can act as reflector in such a case, that will affect the omni-directional characteristics of the antenna)
- To avoid antenna coupling, install the antennas at least one wavelength (12.25 cm) apart. Preferably, three wavelengths (36.75 cm). Antenna coupling means that due to close proximity the antennas will influence each other e.g. blocking radio signals from a particular direction.
- The lightning surge protectors are used to prevent damage when a lightning hits one of the antennas (or very nearby). In some cases, there is already other lightning protection available on-site, in such cases these surge protectors are not needed.
- The short antenna is 5 GHz, connect it to ANT3.

If the antennas are placed in closer proximity, the receiver will still work, but the omni-directional antenna patterns will be (slightly) converted into a directional antenna pattern. Use a drone with RemoteID broadcast to verify the antenna pattern/detection range from various antenna directions.

Opening angle

The ds240 antennas have a very small opening angle (6 degrees). High-gain antennas have always a small opening angle in order to have a high gain. This means that there will be nearby dead zones. Also large objects in the line of sight of the antenna will block reception in that direction. Best location of the ds240 antennas is on a roof, higher than surrounding trees and buildings.

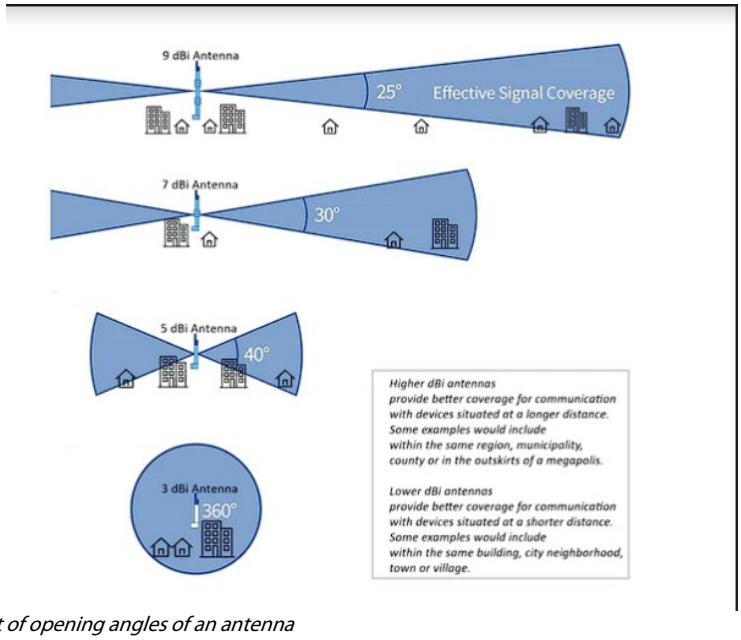


Figure 14 - the effect of opening angles of an antenna

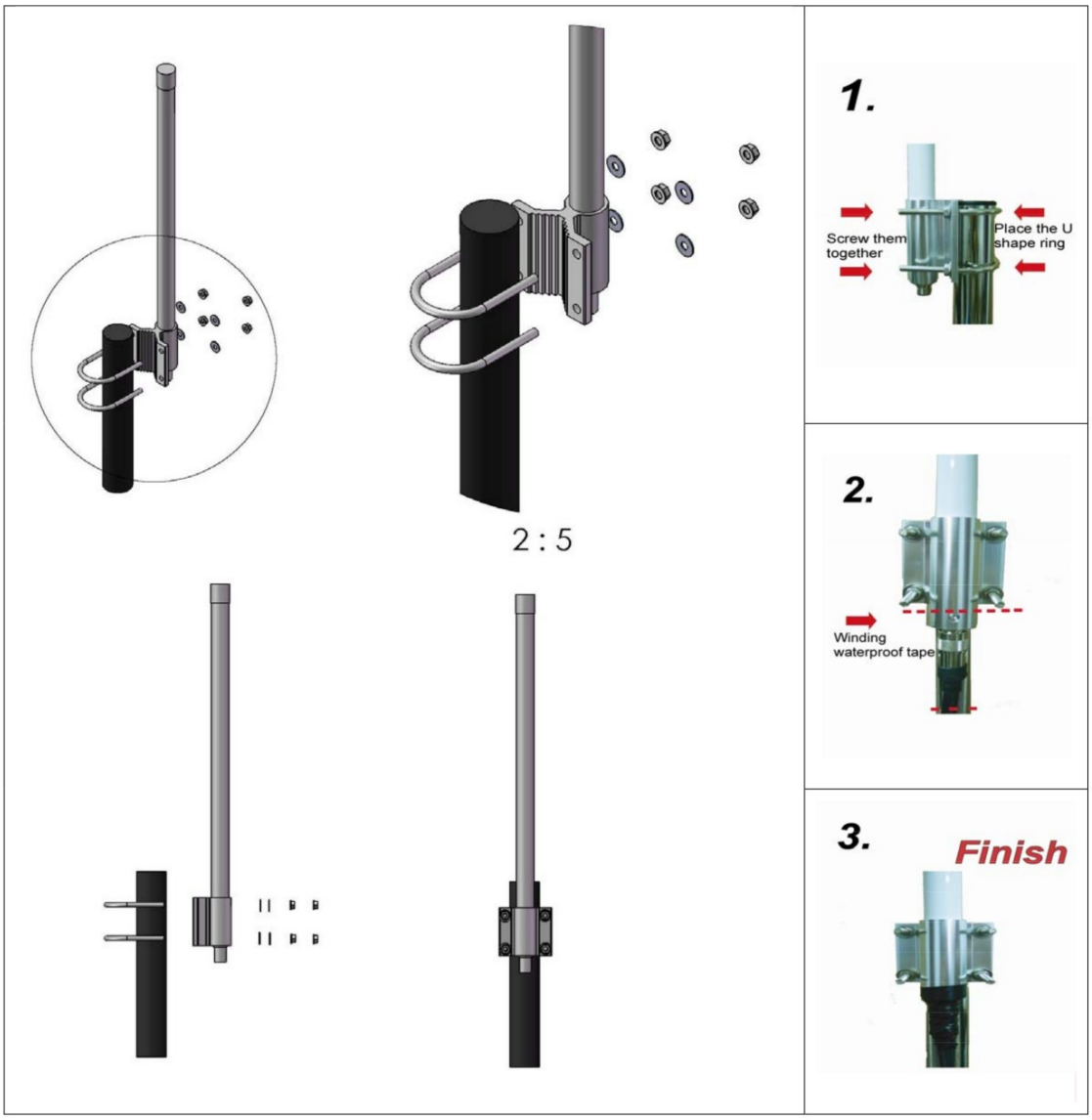


Figure 15 - mounting of the ds240 antennas

DS240 lightning surge protectors

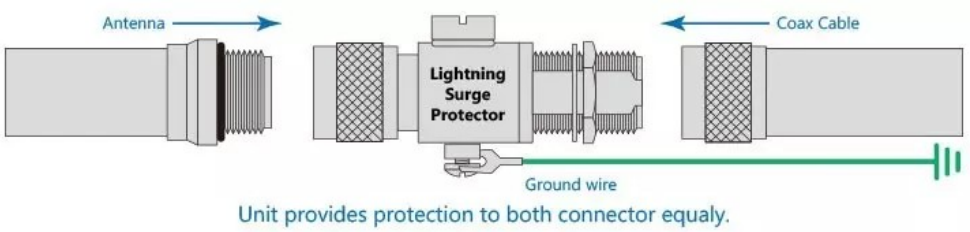


Figure 16 - mounting of lightning surge protectors

1.5 Band pass filter specifications

The RemoteID band pass filter pack is designed for the ds240 receiver. It consists of 3x 2.4 GHz band pass filters and 1x 5 GHz

2.4 GHz band pass

Filter characteristics, removes strong signals such as nearby LTE base (2. 6 GHz)

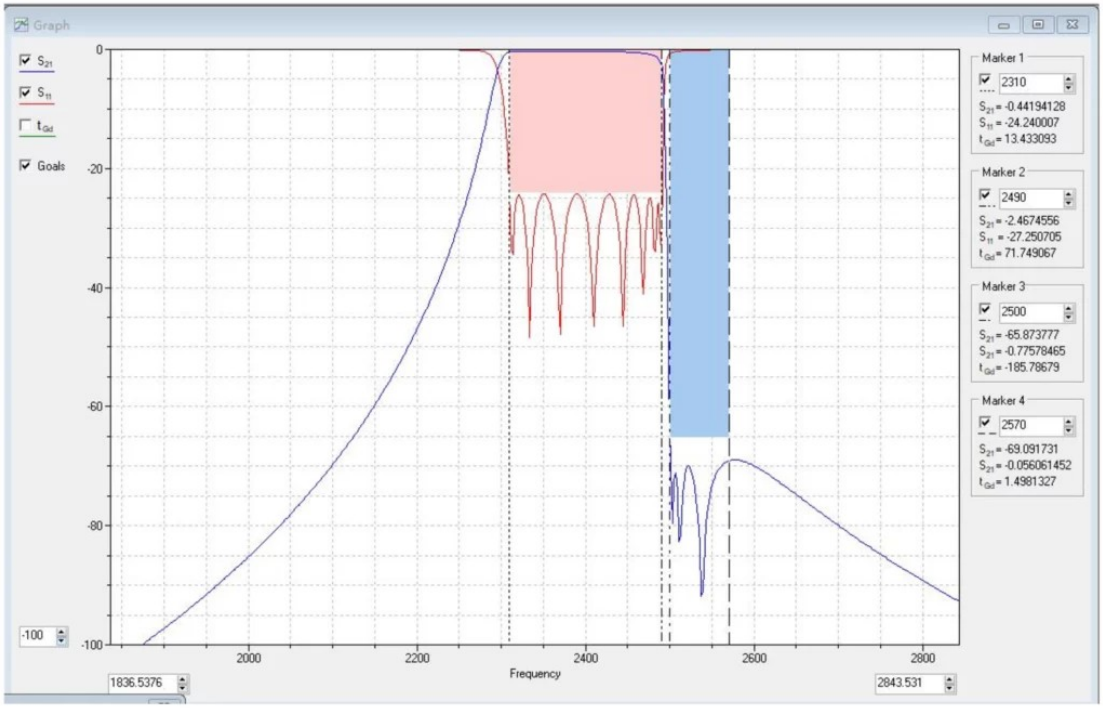


Figure 17 - 2.4 GHz band pass filter

Dimensions

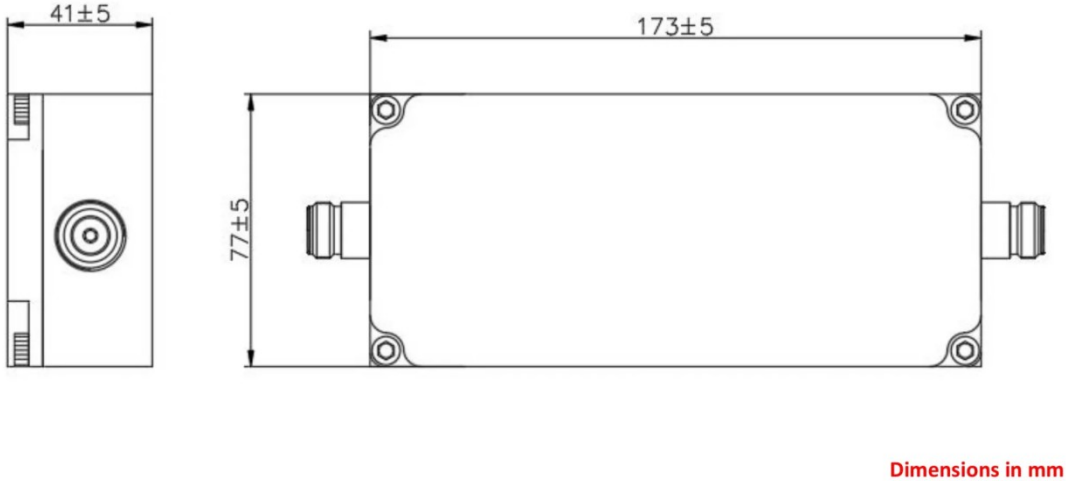


Figure 18 - 2.4 GHz band pass filter dimensions

5 GHz band pass

Filter characteristics

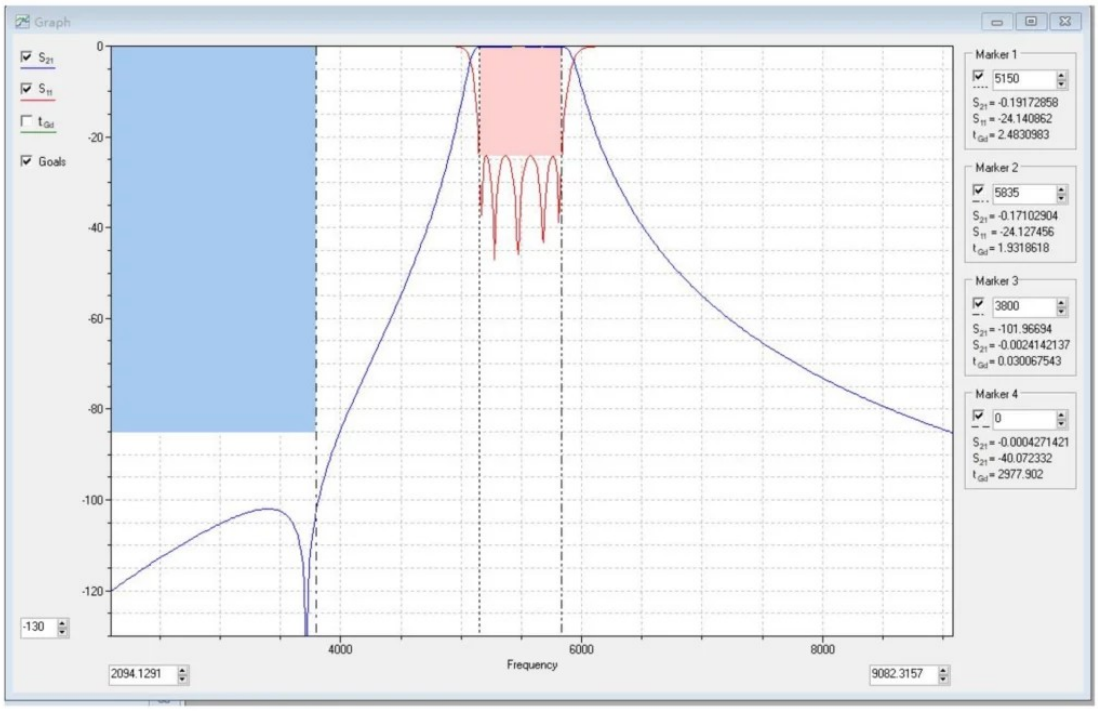


Figure 19 - 5 GHz band pass filter

Dimensions

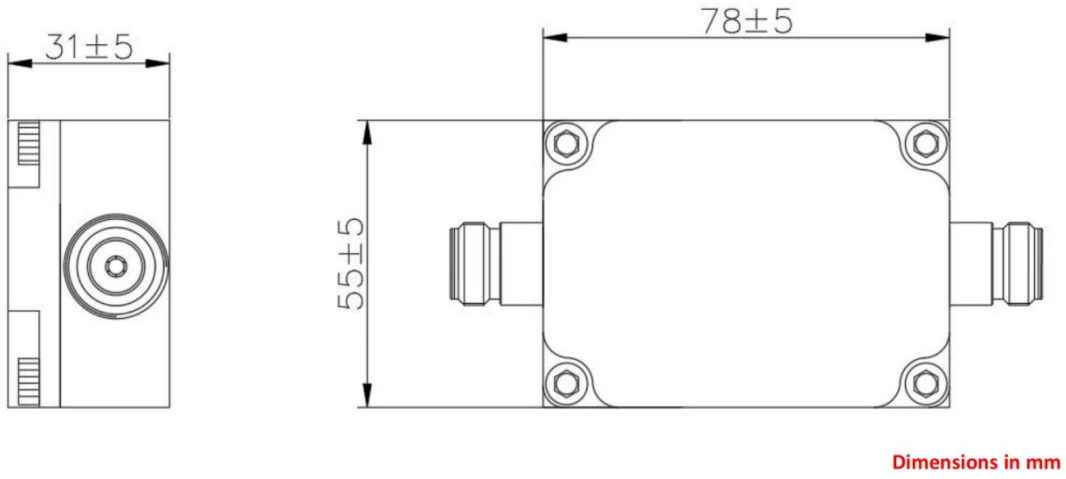


Figure 20 - 5 GHz band pass filter

1.6 Login

The receiver runs on Armbian Linux distribution (ARM64); https://www.armbian.com/. It can be accessed via SSH on the local network.

Login details

User name: root
Password: bluemark

Service: SSH
Port: 22
IP address: DHCP client in local network

Note: please change the password in production deployments!

Serial console

It is also possible to get access to the receiver using the serial console pins on the processing board (ARM64). To get access to this serial console, you need to open the enclosure and locate the 3-pin header near the HDMI-connector. This is shown in the photo below. The pins are GND, RX, TX (bottom to top).

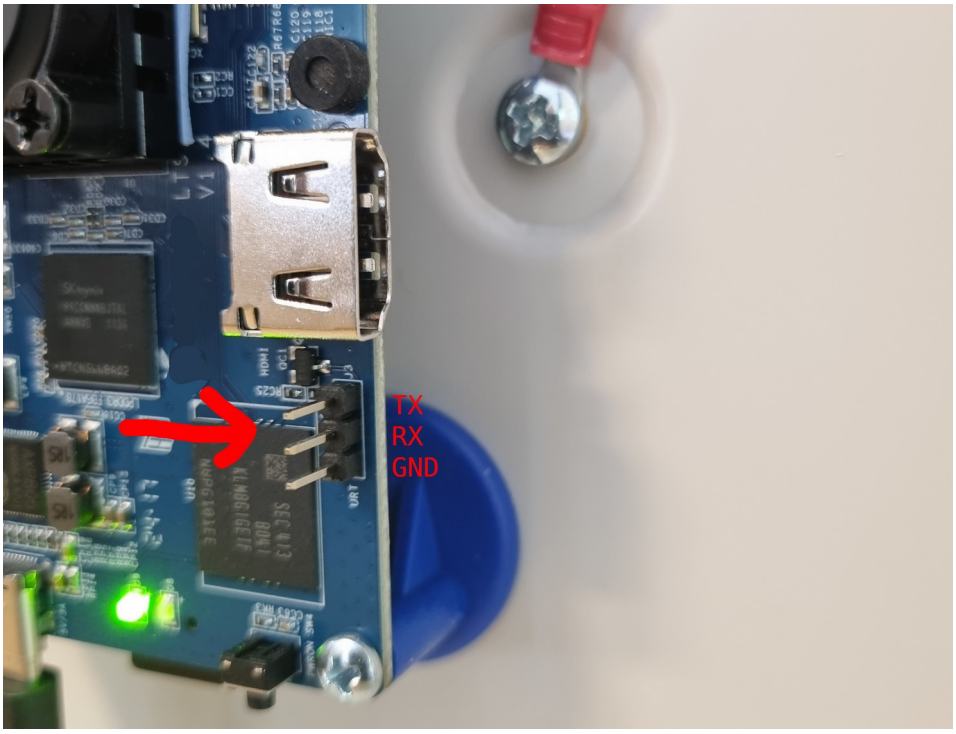


Figure 21 – Serial console pins

Connect a USB to Serial adapter to the pins. The baud rate is 115200.

The following command (on your computer) using the Linux utility screen will give you access:

```
screen /dev/ttyUSB0 115200
```

Of course, you can use other utilities like Putty.

1.7 Read-only file system

The receiver uses a so-called *overlayroot* file system. The file system is mounted read-only and there is a read-write file system in memory on top of it. This means that the system can be used normally, but after a reboot all changes to the file system are lost. Using a read-only file system prevents for instance corrupt file systems after an unexpected power loss.

To make changes permanent:

- Enter in the SSH console: `overlayroot-chroot`

- After this command you can make changes to the filesystem. Enter `exit` to exit this mode.
- If you need full access with internet, using apt etc, **execute prior to overlayroot-chroot, the following commands:**

```
mount -t sysfs none /media/root-ro/sys;
mount -t proc none /media/root-ro/proc;
mount --bind /dev/ /media/root-ro/dev;
mount --bind /tmp/ /media/root-ro/tmp;
mount --bind /dev/pts /media/root-ro/dev/pts;
mount -o bind /etc/resolv.conf /media/root-ro/etc/resolv.conf;
```

- Or mount the read-only partition as read-write instead. I.e enter `mount -o remount,rw /media/root-ro`
  - Make the changes in the folder /media/root-ro
  - Remount as read-only: `mount -o remount,rw /media/root-ro`

Reboot afterwards.

1.8 Change IP address of the sensor

To change the IP address of the DroneScout receiver from DHCP client to a static IP address, use the following commands (modify where needed). **Do not execute the commands in section 1.5.**

Step 1: Check which Ethernet interface is used for connectivity

```
nmcli dev status #available network devices
DEVICE TYPE STATE CONNECTION
enx00e04c680aaa ethernet connected Wired connection 1
eth0 ethernet unavailable --
lo loopback unmanaged -
```

In this case Ethernet adapter *enx00e04c680aaa* is used that is labeled as connection: *Wired connection 1*

Step 2: Change the wired connection to a static IP address. In this example, we will change *Wired connection 1* to static IP address: 192.168.100.144 with a gateway of 192.168.100.1 and a DNS server of 1.1.1.1. Change these example values to fit your use case accordingly.

```
#modify to static IP setting: IP 192.168.100.144;
nmcli con mod "Wired connection 1" ipv4.addresses 192.168.100.144/24;
nmcli con mod "Wired connection 1" ipv4.gateway 192.168.100.1;
nmcli con mod "Wired connection 1" ipv4.dns 1.1.1.1;
nmcli con mod "Wired connection 1" ipv4.method manual;
nmcli con up "Wired connection 1" #apply new settings and save
```

Step 3: Verify that you still can access the sensor with these new IP settings. If needed, connect to the sensor again using the new configured IP address. **Only after you have confirmed that these new IP settings work as intended, make the change permanent.** Test for instance that you can

ping an public IP address such as 8.8.8.8 or URL such as google.com. If you cannot access the sensor anymore, or you need other settings, just reboot the sensor and it will revert to the old settings.

If you wired connection is different than Wired connection 1, change the cp command accordingly.

```
#copy the config to the permanent partition
mount -o remount,rw /media/root-ro;

cp /etc/NetworkManager/system-connections/Wired\ connection\ 1.nmconnection
 \ /media/root-ro/etc/NetworkManager/system-connections/;

sync ;
mount -o remount,ro /media/root-ro;
reboot -f #
```

Delete IP settings/revert to original settings
Log in to the sensor via SSH. Remove the custom IP configuration file.

```
mount -o remount,rw /media/root-ro;
cd /media/root-ro/etc/NetworkManager/system-connections;

rm Wired\ connection\ 1.nmconnection

sync;
mount -o remount,ro /media/root-ro;
reboot -f # or power cycle and the sensor will be a DHCP client
```

1.9 Open Drone ID

DroneScout uses the Open Drone ID framework to encode Remote ID signals. The framework can be found on this page:

https://www.opendroneid.org/

1.10 Global architecture

The receiver has a binary, dronescout, in the root-folder (/root) to sense for Remote ID signals. There is also a configuration file *dronescout.conf* to configure MQTT and other settings. The file *wlan_channels.conf* is used to specify the WiFi channels that will be scanned. Finally, there are some auxiliary files that are discussed in the next chapter.

1.11 Maximum detection range

The maximum detection range of the ds230/ds240 receiver depends on several factors:

- **Effective radiated power** (ERP) of the transponder (transmit power, antenna design)
- **Antenna height** of the receiver and transponder. Detection range increases with higher height as it converges to free-space propagation.

- **Antenna gain and directivity** of the receiver. The maximum ERP power of the transponder is limited by the WiFi/Bluetooth technology standard.
- **Line of sight versus non-line of sight** to the transponder due to buildings, trees, hills etc.
- **Weather**: rain, fog or other "wet" weather will limit the range.
- **Moving drone**: if the drones moves, it will impact negatively on the range. The average RSSI will vary more (signals may be lost more easy). Also if the drones move fast, typically the drone will tilt. In such a case, the drone may block partly the signals towards the receiver.
- **Probability threshold** of detection.
- **Urban vs Rural areas**, see "Urban versus Rural detection range" in section 1.4. In dense urban areas the detection range can be reduced to 1.5 km!

No guarantees can be made about the detection range, due to the complex nature of wireless propagation, as shortly described above. Typically, the detection range is at least multiple kilometers.

The ds240 receiver has 3x larger detection range compared to the ds230 receiver due to the use of high gain antennas that result in a 10 dB extra increase in the sensitivity.

The detection range below is calculated based on internal measurements, that have been extrapolated using the "best-case" Free space propagation model[1].

In general, free-space is considered the most ideal situation. In real-life situation there is more attenuation between the receiver and beacon. Typically, how higher the receiver is installed, how more the propagation mimics free-space. Hence, the detection range increases. This is supported for instance by publications like: *"Characterization of Radio Path Loss in Seaport Environment for WiMAX Applications" - Ming-Tuo Zhou , Joe Jurianto , Jaya Shankar , M. Fujise*[2]

The free-space path loss model is:

$PL = 20 \log_{10} (d / d_0) + c$

Where d is the distance, d_0, a reference distance and c a constant value that among other depends on the frequency. If the distance d doubles i.e. $d = 2d$, the path loss will increase by 6 dB and if d would be 10 times larger, the path loss increases by 20 dB.
Measurement setup (ds230 receiver):

- receiver antenna height 2.5 m
- drone/transponder height 10 m
- transponder ERP power 20 dBm.
- RSSI measurements based on a *slow* moving drone equipped with the DroneBeacon transponder.
- flat agricultural land
  - nearby trees > 15 meter (behind antenna), nearby building > 25 meter (behind antenna)
  - nearby building can act as reflector, so results could be (slightly) too optimistic.
- sunny weather

BLE legacy

- Average RSSI at 500 m: -78 dBm
- Sensitivity radio -97 dBm

So 19 dB (-78 - -97) above sensitivity level. Assuming free space propagation (6 dB loss per doubling of the distance), the maximum detection range is: 500*10^(19/20) = **4.4 km (ds230 receiver)**. This results in an detection area of **61 km²**.

[1] https://en.wikipedia.org/wiki/Free-space_path_loss
[2] http://ap-s.ei.tuat.ac.jp/isapx/2006/pdf/3B1b-4.pdf (Table 1).

The ds240 receiver has 10 dB more sensitivity, the maximum detection range is **13.2 km (ds240 receiver)**. This results in an detection area of **547 km²**.

BLE Long Range

- Average RSSI at 500 m: -78 dBm
- Sensitivity radio -105 dBm

So 27 dB (-78 - -105) above sensitivity level. Assuming free space propagation (6 dB loss per doubling of the distance), the maximum detection range is: 500*10^(27/20) = **11.2 km (ds230 receiver)**. This results in an detection area of **394 km².**

The ds240 receiver has 10 dB more sensitivity, the maximum detection range is **33.6 km (ds240 receiver)**. This results in an detection area of **3545 km².**

WiFi NaN 2.4 GHz

- Average RSSI at 500 m: -61 dBm
- Sensitivity radio -85 dBm

So 19 dB (-61 - -85) above sensitivity level. Assuming free space propagation (6 dB loss per doubling of the distance), the maximum detection range is: 500*10^(19/20) = **7.9 km (ds230 receiver)**. This results in an detection area of **196 km².**

The ds240 receiver has 10 dB more sensitivity, the maximum detection range is **23.7 km (ds240 receiver)**. This results in an detection area of **1764 km².**

WiFi Beacon 2.4 GHz

- Average RSSI at 500 m: -60 dBm
- Sensitivity radio -85 dBm

So 25 dB (-60 - -85) above sensitivity level. Assuming free space propagation (6 dB loss per doubling of the distance.) So maximum detection range is: 500*10^(25/20) = **8.9 km (ds230 receiver)**. This results in an detection area of **249 km².**

The ds240 receiver has 10 dB more sensitivity, the maximum detection range is **26.7 km (ds240 receiver)**. This results in an detection area of **2238 km².**

WiFi Beacon 5.2 GHz

- Average RSSI at 500 m: -68 dBm
- Sensitivity radio -85 dBm

Typically, the 5.2 and 5.8 GHz frequency band have lower detection range due to the higher frequency.

So 17 dB (-68 - -85) above sensitivity level. Assuming free space propagation (6 dB loss per doubling of the distance), the maximum detection range is: 500*10^(17/20) = **3.5 km (ds230 receiver)**. This results in an detection area of **39 km².**

The ds240 receiver has 10 dB more sensitivity, the maximum detection range is **10.5 km (ds240 receiver)**. This results in an detection area of **346 km².**

Extending the detection range

The detection range can be extended in the following ways:

- Install the receiver at a higher place. In this case the received signal will be stronger as propagation is more similar to free-space propagation.
- Replace the antennas with a higher antenna gain. Rule of thumb is that every 6 dB increase, will result in doubling of the detection range. Hence, a 15 dBi antenna would triple (3x) the maximum detection distance. High-gain antennas and/or high receiver location may prevent detection of nearby drones. Experiments with the 5 dBi antennas and flying 50 m above the receiver, did not give any problem detecting the transponder. (The signal was strong received.) The same holds for a 50 m distance experiment, where the drone was moved from 3m to 50 meter height. In all cases the drones was received with a strong signal.
- The range can not be extended in all situations. For instance, if the sensor is used in urban areas, see "Urban versus Rural detection range" in section 1.4.

1.12 RSSI values at 1 meter distance

The following RSSI values were measured at 1 meter distance using a db120 transponder in an office environment and a ds230 receiver. If you measure much weaker RSSI signals as the ones stated below with the db120 transponder (larger as 10 dB), typically the ds230 receiver is broken or the antennas are not mounted properly.

Using a transponder of a different brand can result in different (lower RSSI values), for instance if the transmit power of that transponder is lower or a different antenna is used. In general the RSSI values also depend on the environment and position of the transponder. This can result in RSSI values that are 3 dB higher or lower.

WLAN transmission protocols: on average -20 dBm +/- 3 dB.
Bluetooth transmission protocols: ~ -37 dBm +/- 3 dB.

1.13 System time

The DroneScout ds230/ds240 receiver don't have an onboard RTC clock. It means that when the system is powered off, the current time is not saved. If the DroneScout receiver is booted, it will use chrony, a Network Time Protocol (NTP) client (https://chrony-project.org/) to set the correct system time.

Do you need the correct system time?

DroneScout receivers will detect RemoteID signals regardless of the system time. If the time is incorrect all (local) timestamps in the MQTT messages are wrong. If your own processing of the RemoteID data does not care about the correct timestamp, there is no need to have the correct time.

In case of DroneScout Dashboard you *need* to have the correct system time. In this solution we compare the reported UTC time by the client browser, with the time of detection. If it is too old, it is not shown/displayed. Also, we use this to determine if a sensor is still up and running.

In firmware *20250228-1115* there is also an extra option *set_system_time*. You only need to set this option, if you are in an offline environment (and don't have other options for setting the system

time). If enabled, the receiver will set the system time based on the reported time by the GNSS receiver of the LTE add-on. Note that this method of setting the system time is less accurate, but the time accuracy should still be less than second.

In file /var/log/syslog you will see messages like this to confirm that the system time has been set:

```
2025-02-28T11:23:56.136865+00:00 ds221000000230 root: BlueMark application:
set system time 1740741836.13
```

The GNSS receiver of the LTE add-on can take up to 15 minutes to get a GPS fix. The application will set the system time when there is a GPS fix and will update every few hours this system time.

1.14 RemoteID manufacturer and model identification

In firmware *20250228-1115* DroneScout receivers contain an embedded database that will use the detected serial number to identify:

- present, set to 1 if the detected RemoteID signal contains a Serial Number
- basic checks if the serial number is valid (does it follow the ANSI/CTA-2063-A requirements)
- brand like DJI
- model like Mavic 3
- type like multirotor
- application, free text like consumer or industrial
- weight, the Maximum Take-off Weight (mtow) in kg
- dimensions, free text, the size is reported in mm.

The database is based Open Source Intelligence (OSINT) sources. In the first release, the database contains more than 50 brands and more than 300 popular drone models with RemoteID. Additional fields like type, application, weight, dimensions are free text fields that can be incomplete, or contain errors.

2 CONFIGURATION

The firmware is protected by a license/device key. In case the receiver does not work anymore due to license errors, please contact support.

The ds230/ds240 needs a MQTT broker for uploading data. In the default configuration none is configured. For test purposes you can also install a MQTT broker on the sensor it self. This is described in section 2.5.

root password

For production environments it is **strongly** advised to change the default password.

```
overlayroot-chroot
passwd #interactive tool to change the password
exit
reboot # new password will work after a reboot
```

dronescout.conf

The file /root/dronescout.conf is the main configuration file.

The default contents are shown below.

```
#
# Configuration file
# (c) Bluemark Innovations BV 2022 - 2025

[global]
sensorID = ds220500000100 ; receiver ID up to 256 characters
status_interval_s = 60 ; how often status messages are generated in [s].
temp_dummy_load_enabled = 1 ; if 1 it will generate a dummy load

[mqtt]
host = localhost ; MQTT host
port = 1883 ; MQTT port
topic = ; leave empty to use default topic based on receiverID
QoSlevel = 1 ; QoS level 0, 1 or 2
username = ; leave empty if not used
password = ; leave empty if not used
keepalive = 60 ; keep alive period in seconds.
clientID =  ; set clientID, leave empty for default setting
ssl = 0 ; 0 disable SSL in MQTT connection 1, enable SSL (recommended)
ssl_verify = 0 ; disable/enable SSL verification
CAfile = /root/certs/ca.crt ; location to CA file for SSL connection
CRTfile = /root/certs/client.crt ; location to CRT file for SSL connection
KEYfile = /root/certs/client.key ; locaton to KEY file for SSL connection
compression = lzma ; none or lzma. In case of lzma, payload is compressed.
retain = 0 ; set to 1 in order to retain messages on mqtt broker
```

```
transmit_mode = 2; 0 send MQTT message for each new message, 1 every second
#                  (truncate), 2 combine all and send at transmit mode 2
#                  interval ms period
transmit_mode_2_interval_ms = 250; set the transmission interval (ms) in
#                  transmit mode 2 (valid range: 50 to 60000)
raw_data = 0 ; 0 do not include raw data (of the air interface),
#                   1 include raw data
aggregate_data = 1; 0 send MQTT message with latest received message only, 1
#                  send all information about this RemoteID device

[interface]
WLAN_USB_1 = wlan1 ; interface for WLAN 1 adapter
WLAN_USB_2 = wlan2 ; interface for WLAN 2 adapter
# WLAN_USB_3 = wlan3 ; [ds240 only] interface for WLAN 3 adapter,
BT_UART_1 = /dev/ttyACM0 ; interface for USB UART adapter

[threshold]
WLAN_USB = -200 ; signals weaker as the threshold won't be processed.
BT_UART = -200 ; signals weakers as the threshold won't be processed.

# new configuration parameters/keys for 20240927-1205 firmware upwards
[rid]
enabled = 1 ; enable RemoteID detections, default 1 (enabled)
wlan = 1 ; enable RemoteID WLAN detections, default 1 (enabled)
ble = 1 ; enable RemoteID BLE (Bluetooth) detections, default 1 (enabled)
extra = 1; enable extra RemoteID info like make, model default 1 (enabled)

# GPS section only relevant if LTE add-on is installed
[gps]
enabled = 1 ; enable GPS function of LTE modem
port = /dev/ttyUSB2 ; port for sending AT commands
location = 1 ; enable locations messages
network = 1 ; enable mobile network messages
set_system_time = 0 ; if set to 1, the sensor will set the system time
#                  using the GNSS receiver of the LTE-add on

# ADS-B section only relevant if ADS-B receiver add-on is installed
[ads-b]
enabled = 0 ; enable ADS-B receiver
gain = 48 ; gain of SDR radio. 0 - 50 dB. Set to -1 for auto gain
aggregate_data = 1 ; if 1, use older data from aircraft to fill missing
#                  data fields

# UAT section only relevant if UAT receiver add-on is installed
[uat]
enabled = 0 ; enable UAT receiver
gain = 48 ; gain of SDR radio. 0 - 50 dB. Set to -1 for auto gain
aggregate_data = 1 ; if 1, use older data from aircraft to fill missing
#                  data fields
```

Use the following commands to edit file */root/dronescout.conf*:

```
overlayroot-chroot
nano /root/dronescout.conf  # use nano (or vi) as editor
exit
reboot # to apply changes
```

sensorID

The sensorID is a string up to 256 characters. It is used to identify the DroneScout receiver and is also used in the MQTT payload.

status_interval_s

In firmware *20240528-1739* and higher: how often should a status message be generated in seconds. Default is 60 seconds. See Chapter 3 for more information about status messages.

temp_dummy_load_enabled

Introduced in firmware *20240108-1539* and later. It will create a dummy CPU load to increase the temperature in cold environments. This should benefit the reliability if the outside temperature is -20 degree Celsius. This dummy load is only triggered/generated if the outside temperature is around 0 degree Celsius or lower.

MQTT

The receiver uses internally the MQTT mosquitto library (https://mosquitto.org/). Settings in this MQTT section relate to this library.

For non-encrypted MQTT brokers, only set the *host* and *port*. Make sure that *ssl* is set to 0. If no *topic* is specified, the receiver will use the topic: */sensor/<sensorID>/upload*

If *compression* is set to *none*, the receiver will publish JSON payload in plain text. If compression is set to *lzma* the entire JSON payload will be compressed with LZMA. Typically, LZMA achieves over 80% compression ratio. In plain text mode, the payload is typically around 1400 bytes, with LZMA compression, it is around 260 bytes.

For production deployments, it is **strongly** advised to enable SSL encrypted communication! In this case set *ssl* to 1. Also, set the related file locations: *CAfile*, *CRTfile* and *KEYfile* to the files needed for SSL-encrypted communication to the MQTT broker. If case of self-generated SSL keys, set *ssl_verify* to 0.

For more security also a *username*, *password* can be configured, if the MQTT broker requires this.

In firmware *20221208-1250* and higher, two new options are available: transmit_mode and aggregate_data. The default value for both variables is 1. If transmit_mode is 1, roughly 2 times per second (2 Hz) the sensor will generate a MQTT message for each detected Remote ID device (if new signals are detected for that device). If transmit_mode is 0, a MQTT message is generated every time a Remote ID signal is detected. (No throttling.) In case transmit_mode is 0, the MQTT broker may be overloaded, if a lot of signals are detected and a lot of sensors are uploading data to the broker. In such cases new MQTT messages will get stalled at the sensor. In most smaller setups, it is safe to use transmit_mode is 0.

In firmware *20230329-1042* and higher transmit mode 2 has been introduced. In this mode all received MQTT messages are combined into one payload. At the interval *transmit_mode_2_interval_ms* the combined payload (in milliseconds) is transmitted via MQTT. The valid range is between 250 ms and 60000 (1 minute). Use this transmit mode if you want to publish all received Remote ID signals, but at the same time want to limit the message rate to prevent overloading of the MQTT broker/publishing capacity. Note for transmit mode 2 you also need the latest MQTT subscriber application (Chapter 4).

Potential risks of transmit mode 1 Although transmit mode 1 prevents overloading of a MQTT broker, there is a small probability that an attacker could use this throttling, to broadcast a similar malicious Remote ID signal where some values have been changed (like location data). If the attacker uses the correct timing, it can prevent the ds230 to receive the original Remote ID signal. For that reason newer ds230 receiver will use transmit mode 2 by default. Also for existing deployments we recommend to switch from transmit mode 1 to 2.

The other option is *aggregate_data*. In transmission mode BT4 legacy, only one part of the Remote ID signal is broadcast per time. E.g. one message for the Basic ID, another one for the Location data etc. For other transmission modes, the messages will typically contain all Remote ID information. If aggregate_data = 1, the sensor will save in memory the latest contents of the Remote ID device. So if a BT4 Basic ID message is received, that information element is updated. Other elements are unchanged. The sensor will output an MQTT signal with all available information elements. In case of aggregate_data = 0, all information elements are set to zero and only the latest received information will be saved to memory and the sensor will output an MQTT signal accordingly. In firmware *20240717-1353* and higher, the option raw_data was introduced. If set to 1, the MQTT message will have a message field *raw*, which contains the base64 encoded raw data of the air interface (payload).

interfaces

This section configures the location of the Bluetooth and WiFi radios. Leave to default settings.

threshold

Advanced setting that you typically don't need to change. All radios sense with maximum sensitivity. In case you want to reduce the detection range, you can specify here a threshold. Signals lower as the threshold won't be processed.

RID

Introduced in firmware *20240927-1205* and higher. Advanced setting, this section allows you to enable or disable RemoteID detections. (If this section does not exist, DroneScout receivers will assume you enable RemoteID detection.) This section has three options: *enabled*, if set to 1 it will enable RemoteID detections. If set to 0, it will disable it. The two other options *wlan* and *ble* allows more fine grained configuration. The option *wlan* allows to you to enable (1) or disable (0) RemoteID detections based on WLAN. The option *ble* allows to you to enable (1) or disable (0) RemoteID detections based on Bluetooth BLE.

In firmware *20250228-1115* the setting *extra* is introduced. If set to 1 (default), the MQTT JSON output will contain an extra section where the DroneScout receiver uses an embedded database to identify manufacturer, model and other information a based on the detected Serial Number. See also Chapter 3 and Section 1.14.

GPS

Introduced in firmware *20240927-1205* and higher. The optional LTE add-on/modem can also be used as GNSS (GPS) receiver. It can send the location and mobile network details as MQTT message. This section has 4 options.

The option *enabled* determines if the GPS function of the LTE modem should be activated. If it is set to disabled (0), other values of this section are not used. Also if it is disabled, the LTE modem will still provide connectivity. The option *port* sets the port to which AT commands are sent to the modem. Default */dev/ttyUSB2*. The option *location* and *network* are used to generate MQTT messages. If the option *location* is enabled, the MQTT location message is generated each minute. See Chapter 3. If the option *network* is enabled, the MQTT mobile network message is generated. In firmware *20250228-1115* there is also an extra option *set_system_time*. In normal setups, DroneScout

receivers have access to NTP server to synchronize the local clock with the current time. In offline environments this is not possible. If enabled, the firmware will set the system clock using the GNSS receiver of the LTE add-on. See also Section 1.13 System time.

ADS-B

Introduced in firmware *20240927-1205* and higher. It is only relevant if the ADS-B receiver add-on is installed. This section has 3 options. The option *enabled* determines if the ADS-B receiver add-on should be activated. The option gain determines the used gain in the SDR receiver. Lower gain means lower detection range. Leave it to 48 or set to -1 for automatic gain mode. In ADS-B data is sent using multiple different packet types. If the option *aggregate_data* is enabled (1, default value), it will use data from the last detected ADS-B packet for this aircraft. Data from older packets will be used to fill missing data fields.

UAT

Introduced in firmware *20240927-1205* and higher. It is only relevant if the UAT receiver add-on is installed. This section has 3 options. The option *enabled* determines if the UAT receiver add-on should be activated. The option gain determines the used gain in the SDR receiver. Lower gain means lower detection range. Leave it to 48 or set to -1 for automatic gain mode. In UAT data is sent using multiple different packet types. If the option *aggregate_data* is enabled (1, default value), it will use data from the last detected UAT packet for this aircraft. Data from older packets will be used to fill missing data fields.

remote SSH login

When receivers are placed behind a router, they can't be accessed remotely. Reverse SSH is a method to be able to login remotely without changing router or firewall settings. See https://www.howtogeek.com/428413/what-is-reverse-ssh-tunneling-and-how-to-use-it/ for background information.

Default remote SSH login is disabled.

If remote SSH is enabled (by setting *enabled to* "1" in */root/remote.conf*), the receiver will try to connect to the configured SSH server (using *server*, *port* and *user* in *remote.conf*). It will open a port (*remote_port* in *remote.conf*) that can be used to remotely login. Each receiver needs an unique port, otherwise those receivers will compete for the same port..

In addition, passwordless login by using SSH keys is assumed. This means that the receiver SSH key (*/root/.ssh/id_rsa*) needs to be accepted by the remote SSH server. More information can be found in the mentioned background article.

Use the following commands to edit file */root/remote.conf*:

```
overlayroot-chroot
nano /root/remote.conf  # use nano (or vi) as editor
exit
reboot # to apply changes
```

Login from the SSH server

The receiver can be accessed from the SSH server by entering the following command: `ssh root@localhost -p10230`

The number 10230 is the *remote_port* in *remote.conf*.

wlan_channels.conf

The /root/wlan_channels.conf file specifies which WiFi channels are scanned by the ds230/ds240 receiver. The ds240 will automatically filter this list and direct the 5 GHz channels to the 5 GHz radio interface and the 2.4 GHz to both 2.4 GHz radios. The default contents is show below:

```
0,1
1,2
2,3
3,4
4,5
5,6
6,44
7,149
8,7
9,8
10,9
11,10
12,11
13,13
14,6
15,149
16,44
17,36
18,40
19,48
20,52
21,56
22,60
23,149
24,6
25,44
26,13
27,11
28,10
29,9
30,8
31,7
32,149
33,44
34,6
35,5
36,4
37,30
38,2
39,1
40,64
41,161
42,165
43,153
```

```
44,157
```

Use the following commands to edit file /root/remote.conf:

```
overlayroot-chroot
nano /root/wlan_channels.conf  # use nano (or vi) as editor
exit
reboot # to apply changes
```

The first number on the row is the sequence number. The second number is the channel number. Up to 256 channels can be configured. If the WiFi radio reaches the end of the sequence it will start with the first element. The second radio will scan the same channels, but with half period delay. Typically leave this file to default settings.

WiFi NaN (also called Wi-Fi Aware) signal can only be found on WiFi channel 6 (2.4 GHz), 44 (5 GHz) and 149 (5 GHz). WiFi Beacon signals can be found on all WiFi channels.

The following channels can be configured (capabilities of WiFi radio. The number in brackets [ ] is the channel number:

- 2412 MHz [1]
- 2417 MHz [2]
- 2422 MHz [3]
- 2427 MHz [4]
- 2432 MHz [5]
- 2437 MHz [6]
- 2442 MHz [7]
- 2447 MHz [8]
- 2452 MHz [9]
- 2457 MHz [10]
- 2462 MHz [11]
- 2467 MHz [12]
- 2472 MHz [13]
- 2484 MHz [14]
- 5075 MHz [15]
- 5080 MHz [16]
- 5085 MHz [17]
- 5090 MHz [18]
- 5100 MHz [20]
- 5120 MHz [24]
- 5140 MHz [28]
- 5160 MHz [32]
- 5180 MHz [36]
- 5200 MHz [40]
- 5220 MHz [44]
- 5240 MHz [48]
- 5260 MHz [52]
- 5280 MHz [56]
- 5300 MHz [60]
- 5320 MHz [64]
- 5340 MHz [68]
- 5360 MHz [72]
- 5380 MHz [76]
- 5400 MHz [80]
- 5420 MHz [84]
- 5440 MHz [88]
- 5460 MHz [92]
- 5480 MHz [96]
- 5500 MHz [100]
- 5520 MHz [104]
- 5540 MHz [108]
- 5560 MHz [112]
- 5580 MHz [116]
- 5600 MHz [120]
- 5620 MHz [124]
- 5640 MHz [128]
- 5660 MHz [132]

- 5680 MHz [136]
- 5700 MHz [140]
- 5720 MHz [144]
- 5745 MHz [149]
- 5765 MHz [153]
- 5785 MHz [157]
- 5805 MHz [161]
- 5825 MHz [165]
- 5845 MHz [169]
- 5865 MHz [173]
- 5885 MHz [177]

MQTT broker on the DroneScout receiver

For test purposes you can also install a MQTT broker (mosquitto) on the ds230/ds240 sensor. (In the default ds230/ds240configuration none is configured.) Installing a MQTT broker on the sensor may result in network vulnerabilities. For that reason we don't advise this for production environments.

There are two options for installing the MQTT broker: a) permanent b) temporarily (lost after a reboot). For option a) permanent installation, see Section 1.5 to make permanent changes to the file system.

- First install the mosquitto MQTT broker:

```
apt update; apt install -y mosquitto
```

- Add these lines to the mosquitto configuration (`/etc/mosquitto/mosquitto.conf`)

```
echo "listener 1883" >> /etc/mosquitto/mosquitto.conf
echo "allow_anonymous true" >> /etc/mosquitto/mosquitto.conf
```

- Restart mosquitto MQTT broker

```
service mosquitto restart
```

- The default configuration of the dronescout receiver (file /root/dronescout.conf) publishes messages to the local MQTT broker. Make sure that host, port and ssl are set correctly as shown below.

```
#
# Configuration file
# (c) Bluemark Innovations BV 2022 - 2025

[global]
sensorID = ds220500000100 ; sensor ID
status_interval_s = 60 ; how often status messages are generated in [s].
temp_dummy_load_enabled = 1 ; if 1 it will generate a dummy load

[mqtt]
host = localhost ; MQTT host
port = 1883 ; MQTT port
topic = ; leave empty to use default topic based on sensorID
QoSlevel = 1 ; QoS level 0, 1 or 2
username = ; leave empty if not used
password = ; leave empty if not used
```

```
keepalive = 60 ; keep alive period in seconds.
clientID =  ; set clientID or to random to generate random ID,
#                   leave empty for default setting
ssl = 0 ; 0 disable SSL in MQTT connection 1, enable SSL
ssl_verify = 0 ; disable SSL verification of SSL key of MQTT broker
#                   (useful for self-generated keys)
CAfile = /root/certs/ca.crt ; location to CA file for SSL connection
CRTfile = /root/certs/client.crt ; location to CRT file for SSL connection
KEYfile = /root/certs/client.key ; locaton to KEY file for SSL connection
compression = lzma ; none or lzma. In case of lzma, payload is compressed.
retain = 0 ; set to 1 in order to retain messages on mqtt broker
transmit_mode = 2; 0 send MQTT message for each new message, 1 every second
transmit_mode_2_interval_ms = 250; set the transmission interval (ms)
#                   in transmit mode 2 (valid range: 50 to 60000)
raw_data = 0 ; 0 do not include raw data (of the air interface),
#                    1 include raw data
aggregate_data = 1; 0 send MQTT message with latest received message only,
#                   1 send all information about this RemoteID device

[interface]
WLAN_USB_1 = wlan1 ; interface for WLAN 1 adapter
WLAN_USB_2 = wlan2 ; interface for WLAN 2 adapter
# WLAN_USB_3 = wlan3 ; interface for WLAN 3 adapter (ds240 only)
BT_UART_1 = /dev/ttyACM0 ; interface for Bluetooth adapter

[threshold]
WLAN_USB = -200 ; signals weaker as the threshold won't be processed.
BT_UART = -200 ; signals weakers as the threshold won't be processed.

# new configuration parameters/keys for 20240927-1205 firmware upwards
[rid]
enabled = 1 ; enable RemoteID detections, default 1 (enabled)
wlan = 1 ; enable RemoteID WLAN detections, default 1 (enabled)
ble = 1 ; enable RemoteID BLE (Bluetooth) detections, default 1 (enabled)
extra = 1; enable extra RemoteID info like make, model default 1 (enabled)

# GPS section only relevant if LTE add-on is installed
[gps]
enabled = 0 ; enable GPS function of LTE modem
port = /dev/ttyUSB2 ; port for sending AT commands
location = 1 ; enable locations messages
network = 1 ; enable mobile network messages
set_system_time = 0 ; if set to 1, the sensor will set the system time
#                   using the GNSS receiver of the LTE-add on

# ADS-B section only relevant if ADS-B receiver add-on is installed
[ads-b]
enabled = 0 ; enable ADS-B receiver
gain = 48 ; gain of SDR radio. 0 - 50 dB. Set to -1 for auto gain
aggregate_data = 1 ; if 1, use older data from aircraft to fill missing
#                   data fields

# UAT section only relevant if UAT receiver add-on is installed
[uat]
enabled = 0 ; enable UAT receiver
gain = 48 ; gain of SDR radio. 0 - 50 dB. Set to -1 for auto gain
aggregate_data = 1 ; if 1, use older data from aircraft to fill missing
#                   data fields
```

- Restart the dronescout application
  `pkill dronescout`

- (Optional) point the mqtt subscriber application to the ds230/ds240 sensor:
  https://github.com/BluemarkInnovations/RemoteID-MQTT-subscriber/blob/main/mqtt_sub.py
  - Change `broker` to the IP address of the sensor
  - Change `port` to 1883
  - Comment (#) the line starting with client_pem:
    `#client_pem = "./certs/client.pem"`

The python script now looks like (line 19 to 31):

```
...
broker = 'IP_address_of_sensor'
port = 1883
topic = "#"

# generate client ID with pub prefix randomly
client_id = f'mqtt-subscriber-{random.randint(0, 100)}'

#optional user/password for connecting to the MQTT broker, uncomment
if used.
#username = 'myusername'
#password = 'mypassword'

#file containing full SSL chain, uncomment when MQTT broker uses
encrypted messages [preferred]
#client_pem = "./certs/client.pem"
...
```

You should now be able to receive ds230/ds240 MQTT messages.

For option a) permanent installation, enter the command `exit` and reboot the sensor to apply the changes: `reboot -f`.

3 MQTT MESSAGES

The receiver generates two types of MQTT messages (JSON format):

- *Status messages* - every minute, the receiver will send a status message
- *Data messages* - data messages with Remote ID data.
- *Location messages* - if the LTE add-on is installed it can send the location of the sensor
- *Mobile Network messages* - if the LTE add-on is installed it can send details of the mobile network
- Aircraft messages - if an ADS-B, UAT, or ADS-L (under development) , it will send detected aircraft using this message.

Status messages

The receiver will publish every minute a status message (period can be configured). An example message is shown below.


```
{
   "protocol":1.0,
   "status":{
      "sensor ID":"900",
      "timestamp":1652093100000,
      "firmware version":"20220509-1035",
      "model":"ds230",
      "status":"normal"
   }
}
```


Figure 22 - Example receiver status message.

This message contains several sections:

- *protocol* - indicates the protocol version. Currently only protocol 1.0 exists.
- *status* - a status message contains a section status.
- *sensor ID* - the sensor ID set in dronescout.conf
- *timestamp* - the epoch time stamp (in milliseconds)
- *firmware version* - the current firmware version of the receiver
- *model* - the model of the receiver: can be ds230 or ds240.
- *status* - the status, can be “normal” or “invalid license”. If the license is invalid, no data will be published.

Data messages

The receiver will publish a data message when a RemoteID message is detected. An example message is shown below.


```
⊟{
   "protocol":1.0,
   "data":⊟{
      "sensor ID":"900",
      "RSSI":-39,
      "channel":6,
      "timestamp":1740766152022,
      "MAC address":"52:87:F5:E0:E3:59",
      "type":"WiFi beacon",
      "UASdata":"AQAAAAEAAAAxNTgxRjZNSzEyMzQ1Njc4OTAxMgAAAAAAAAAAAA
      AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAgLRDAAB/QwAAfEI
      AAAAAAAAAAAAAAAAAAAAAAAB6xAAAesQAAAAAAAAB6xAAAAAAAAAAAAAAAAA
      AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
      AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
      AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
      AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
      AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
      AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
      AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
      AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
      AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
      AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
      AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
      AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
      AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
      AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
      AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
      AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
      EAAAAAAHrEAAB6xAAAAAAAAAAAAAB6xAAAAAAAAAAAAAAAAAAAAAAAAAAAA
      AAAAAAAAAAAAAAAAAAAAAEAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
      ",
      "extra":⊟{
         "SN present":1,
         "SN valid":1,
         "manufacturer":"DJI",
         "model":"Mavic 3 Pro Cine",
         "type":"multirotor",
         "application":"consumer/camera",
         "weight":"1",
         "dimensions":"348 x 291 x 108"
      }
   }
}
```


Figure 23 - Example remote ID data message.

This message contains several sections:

- *protocol* - indicates the protocol version. Currently only protocol 1.0 exists.
- *data* - a data message contains a section data.
- *sensor ID* - the sensor ID set in dronescout.conf
- *RSSI* - the RSSI of the received Remote ID packet.
- *channel* - the channel on which the Remote ID packet is received. It is zero for BLE Remote ID signals, otherwise it is the WiFi channel number.
- *timestamp* - the epoch time stamp of the message in milliseconds.
- *MAC address* - the MAC address that broadcast the Remote ID packet
- *type* - the remote ID type. It can be BLE legacy, BLE long range, WiFi NaN or WiFi beacon.
- *raw* - if the setting raw_data is enabled, the *raw* field contains the base64-encoded of the air interface (payload)
- *UASdata* - *this contains the Remote ID data. It is base64-encoded. The binary data itself is the Open Drone ID structure defined on* ***line 401*** *in file opendroneid.h:* https://github.com/opendroneid/opendroneid-core-c/blob/master/libopendroneid/opendroneid.h

It is shown below:

```
typedef struct ODID_UAS_Data {
  ODID_BasicID_data BasicID[ODID_BASIC_ID_MAX_MESSAGES];
  ODID_Location_data Location;
  ODID_Auth_data Auth[ODID_AUTH_MAX_PAGES];
  ODID_SelfID_data SelfID;
  ODID_System_data System;
  ODID_OperatorID_data OperatorID;

  uint8_t BasicIDValid[ODID_BASIC_ID_MAX_MESSAGES];
  uint8_t LocationValid;
  uint8_t AuthValid[ODID_AUTH_MAX_PAGES];
  uint8_t SelfIDValid;
  uint8_t SystemValid;
  uint8_t OperatorIDValid;
} ODID_UAS_Data;
```

The data structures used in ODID_UAS_Data are also defined in opendroneid.h file. It means that *ODID_BASIC_ID_MAX_MESSAGES* is 2 and *ODID_AUTH_MAX_PAGES* is 16. See Chapter 5 for the reference Python code to decode this structure.

The current firmware version uses git commit *4785de4570e2ecd418543d130d16147108181d0e* of the Open Drone ID project.

In the next chapter, reference Python source code is described to subscribe to the broker and parse these MQTT messages.

In firmware *20250228-1115* the JSON data can contain an extra section that uses the Serial Number (SN) to identify the manufacturer and model of the drone. See also Section 1.14. An example message is shown in Figure 23.

This extra section contains several fields:

- SN present - indicates if the detected RemoteID signal has a BasicID with Serial Number (1), otherwise it is 0.
- SN valid - check if the Serial Number is compliant with the ANSI/CTA-2063-A standard that is used for Serial Numbers. Valid (1), otherwise 0.

- manufacturer - manufacturer like for instance DJI. Set to unknown, if it is not available.
- model - the model like Mavic 3 Set to unknown, if it is not available.
- type - type of the drone like multirotor or fixed wing
- application - the application where the drone is used for like industrial or consumer (free text)
- weight - the Maximum Take-off Weight in kg. (free text)
- dimensions - the dimensions of the drone in mm. (free text)

Location messages

The receiver will publish every minute a location message (period can be configured), if the LTE add-on has been installed. Introduced in firmware *20240927-1205* and higher. An example message is shown below.


```
{
   "protocol":1.0,
   "location":{
      "sensor ID":"901",
      "timestamp":1727346540000,
      "latitude":52.23636166,
      "longitude":6.84949334,
      "altitude MSL":28.0
   }
}
```


Figure 24 - Example location message.

This message contains several sections:

- *protocol* - indicates the protocol version. Currently only protocol 1.0 exists.
- *location* - a location message contains a section location.
- *sensor ID* - the sensor ID set in dronescout.conf
- *timestamp* - the epoch time stamp of the message in milliseconds.
- *latitude* - the GPS latitude in Decimal Degrees
- *longitude* - the GPS longitude in Decimal Degrees
- *altitude MSL* - the altitude reference; Mean Sea Level (MSL)

Mobile network messages

The receiver will publish every minute a mobile network message (period can be configured) if the LTE-add has been installed. Introduced in firmware *20240927-1205* and higher. An example message is shown below.

```
{
   "protocol":1.0,
   "mobile network":{
      "sensor ID":"901",
      "timestamp":1727346540000,
      "PLMN":"20408",
      "operator":"GSM 900",
      "band":"GSM 900",
      "access technology":"GSM",
      "RSSI":22,
      "BER":99
   }
}
```


Figure 25- Example mobile network message.

The values of this messages are based on the output of the following AT commands: *AT+QNWINFO*, *AT+CSQ* and *AT+COPS?* for the Quectel EC25 modems.

AT Commands Manual: https://cloud.bluemark.io/s/3wxp9DjLzRSdyKF

This message contains several sections:

- *protocol* - indicates the protocol version. Currently only protocol 1.0 exists.
- *mobile network* - a mobile network message contains a section mobile network.
- *sensor ID* - the sensor ID set in dronescout.conf
- *timestamp* - the epoch time stamp of the message in milliseconds.
- *PLMN* - the PLMN (Public Land Mobile Network) is numeric code that describes the mobile network. It consists of a country code (MCC) and mobile network code (MNC).
- *operator* - the full name of the operator like Vodafone, T-mobile
- *band* - the frequency band to which the modem is connected, such as GSM 900.
- *access technology* - the used access technology like GSM, FDD LTE.
- *RSSI* - the reported signal strength RSSI
- *BER* - the reported bit error rate BER

Aircraft messages

The receiver will publish a aircraft message when a aircraft message is detected using the ADS-B/UAT or ADS-L (under development) receiver add-on. Introduced in firmware *20240927-1205* and higher. An example message is shown below.

```
{
   "protocol":1.0,
   "aircraft":{
      "sensor ID":"901",
      "timestamp":1727346775193,
      "type":"ADSB",
      "frequency MHz":1090,
      "raw":"oAAXGMZQADCkAABEfXYA",
      "address":"440050",
      "address type":0,
      "flight":"AUA4MY ",
      "emergency status":0,
      "flight state":1,
      "category":5,
      "altitude":10972,
      "altitude type":1,
      "speed":196,
      "track":294,
      "vertical rate":0,
      "latitude":52.2829114,
      "longitude":6.07999093,
      "message count":1397,
      "SBS":"MSG,4,,,440050,,,,,,,,382,294,,,0,,0,0,0,0"
   }
}
```


Figure 26- Example aircraft message

This message contains several sections:

- *protocol* - indicates the protocol version. Currently only protocol 1.0 exists.
- *network* - a aircraft message contains a section aircraft.
- *sensor ID* - the sensor ID set in dronescout.conf
- *timestamp* - the epoch time stamp of the message in milliseconds.
- *type* - the receiver type: ADSB, UAT or ADSL (under development)
- *frequency MHz* - the receiver frequency: 1090 MHz (ADSB), 978 MHz (UAT), 868 (ADSL)
- *raw* - the *base64*-encoded raw received data
- *address* - the address of the aircraft
- *address type* - the address type. Type 0 is ICAO address ADS-B, 1 is reserved, 2 is ICAO address using TIS-B, 3 is TIS-B track file address, 4 is vehicle address, 5 is fixed ADS-B Beacon address and type 6 and 7 are reserved.
- *flight* - the flight ID
- *emergency status* - emergency state of the aircraft. status 1 means emergency, 0 means normal operation.
- *flight state* - the state of the aircraft. 0 means ground, 1 means air

- category - defines the vortex category of the aircraft. Category 0 means no information, 1 means light <= 7000 kg, 2 is medium wake 7000 - 34000 kg, 3 is medium wake 34000 - 136000 kg, 4 is medium wake high vortex 34000 - 136000 kg, 5 is heavy >= 136000 kg, 6 is highly maneuverable, 7 is rotorcraft, 9 is glider/sailplane, 10 is lighter than air, 11 is parachutist / sky diver, 12 is ultra light / hang glider / paraglider, 14 is UAV, 15 is space / transatmospheric, 17 is emergency vehicle, 18 is service vehicle, 19 is point obstacle, 20 is cluster obstacle, 21 is line obstacle. Other values are reserved.
- *altitude* - he altitude in meters
- *altitude type* - the altitude type (reference): 1 means barometric (pressure) altitude, 2 is GNSS altitude
- *speed* - the aircraft speed in m/s
- *track* - the track (course) of the aircraft speed in degrees
- *vertical rate* - the vertical speed in m/s
- *latitude* - the GPS latitude in Decimal Degrees
- *longitude* - the GPS longitude in Decimal Degrees
- *message count* - number of messages received for this aircraft
- *SBS* - the SBS output string (useful for export to ADS-B visualization software) This can be multiple lines.

4 MQTT SUBSCRIBER (REFERENCE CODE)

You need to write your own MQTT subscriber application to process the MQTT messages published by the DroneScout receivers.

Reference Python3 source code is available to process MQTT messages of the ds230 or ds240 receiver. It can be found here: https://github.com/BluemarkInnovations/RemoteID-MQTT-subscriber

5 FIRMWARE UPDATE

The firmware can be updated to the latest version by executing the following update-script.

```
sh /root/update.sh
```

Note, this update script does not remove files, nor does it overwrite the file remote.conf.

Firmware history
The firmware history can be found at:
https://download.bluemark.io/dronescout/firmware/history.txt

6 TROUBLE SHOOTING

I do not know the IP address of the sensor
Login to the router and check the list of DHCP clients. The sensor should be in the list. If you don't have access to the router, use nmap (available in most Linux distros, https://nmap.org/)

Execute this command:

```
nmap -sn 192.168.100.0/24
```

Where 192.168.100.0/24 means it will scan the local network 192.168.100.x. Change to your own IP range accordingly. For instance, if your own IP address is 192.168.1.5, use 192.168.1.1/24 instead.

An example result is:

```
Starting Nmap 7.70 ( https://nmap.org ) at 2024-01-07 20:50 GMT
Nmap scan report for Archer (192.168.100.1)
Host is up (0.00048s latency).
Nmap scan report for N300A-IP (192.168.100.151)
Host is up (-0.092s latency).
Nmap scan report for ds221000000230 (192.168.100.185)
Host is up (-0.10s latency).
Nmap scan report for 192.168.100.199
Host is up (0.00039s latency).
Nmap done: 256 IP addresses (4 hosts up) scanned in 10.09 seconds
```

Here, the sensor ds221000000230 is found and has IP address 192.168.100.185

The sensor outputs data initially, but stops after a while
The sensor uses MQTT. If the MQTT broker is restarted, or the publisher, subscriber can't access the MQTT broker, there will be no output data anymore. Also, the sensor reboots every week, if the MQTT broker is installed on the sensor, the MQTT subscriber may not be able to reconnect successfully after such reboot. Try to restart the MQTT subscriber, the sensor or MQTT broker to find the cause. In some situations a firewall may drop connections to the MQTT broker. (Note: the sensor does not have a firewall installed, because it does not have unnecessarily applications installed.)

How can I increase the operating temperature range of the ds230/ds240 receiver?
The temperature range of -20 degree Celsius to +50 degrees Celsius has been calculated by looking at the specifications of the main components and using a safety margin. Very likely it will operate beyond this temperature range as well, but this is not easy to evaluate. The ds230/ds240 receiver consists of multiple components. Most of them have a specified operating temperature of -20 degree Celsius to +70 degree Celsius. An exception are the WLAN adapters as they are certified from 0 degree Celsius to +70 degree Celsius by the manufacturer. (Although it is expected that they will also work at lower temperatures.) The temperature in the enclosure needs to be within this temperature range. When it is cold, the electronic components will generate heat and as a result inside the enclosure it will be warmer than outside. In firmware *20240108-1539* and newer, by default, the sensor will generate extra heat by generating an extra artificial CPU load if the CPU temperature is lower as +25 degree Celsius. (Typically, the ambient temperature would be 0 degree Celsius in this case). As a result the temperature in the enclosure should rise. This should ensure more reliable operation when the inside temperature is near -20 degree Celsius. In summer, when the ambient temperature is +40 degree Celsius, there is 30 degree margin as the electronic

components allow up to 70 degree Celsius operating temperature. A fan on top of the CPU will make sure there is airflow inside the enclosure to prevent hot spots.

The sensor hangs during a (weekly) reboot
Typically, the PoE connection is not able to deliver enough power or the Ethernet cable is of bad quality or is too long. Please change PoE router/injector and/or change Ethernet cable to resolve.

MQTT

No MQTT data is published to the MQTT broker
In firmware *20240108-1539* and newer, the sensor creates a file created that shows the MQTT status: /tmp/mqtt.log

View the file:
`cat /tmp/mqtt.log`

Normal operation:
The last line of this file is: `INFO MQTT: connecting and publishing data`

Errors:

- The last line of this file is: `ERROR MQTT: could not publish message, error code [x] ([y])`
  This means that the MQTT connection was setup, but publishing failed. It is the return value of the *mosquitto_publish* function: https://mosquitto.org/api/files/mosquitto-h.html#mosquitto_publish
  [x] indicates the error code and [y] a string that describes this error.
- The last line of this file is: `ERROR MQTT: Out of memory.`
  This means that the mosquitto_new function failed to allocate memory. This should never happen, unless there is no free memory available. *mosquitto_new* function: https://mosquitto.org/api/files/mosquitto-h.html#mosquitto_new
- The last line of this file is: `ERROR MQTT: setting user name password failed with result: [x] ([y])`
  This means that the user name, password could not be set by the *mosquitto_username_pw_set* function: https://mosquitto.org/api/files/mosquitto-h.html#mosquitto_username_pw_set
  [x] indicates the error code and [y] a string that describes this error.
- The last line of this file is: `ERROR MQTT: setting TLS (SSL) failed with result: [x] ([y])`
  This means that *mosquitto_tls_set* function to configure an encrypted SSL/TLS connection failed: https://mosquitto.org/api/files/mosquitto-h.html#mosquitto_tls_set
  [x] indicates the error code and [y] a string that describes this error.
- The last line of this file is: `ERROR MQTT: unable to connect: [x] ([y])`
  This means that sensor could not connect to the MQTT broker: https://mosquitto.org/api/files/mosquitto-h.html#mosquitto_connect_async
  [x] indicates the error code and [y] a string that describes this error.
- The last line of this file is: `ERROR MQTT: unable to start loop: [x] ([y])`
  This means that sensor could not start the MQTT publisher loop: https://mosquitto.org/api/files/mosquitto-h.html#mosquitto_loop_start
  [x] indicates the error code and [y] a string that describes this error.

7 Warranty

The product has a two-year warranty period, starting at the date of receiving the product. Outside warranty are issues like crash damage, improper use, (extreme) weather conditions that damages the product. The product is eligible for future firmware updates as described in the Chapter Firmware update.

8 MORE INFORMATION

If you need more information, please contact us at info@bluemark.io or by phone: +31 53 711 2104.

All contact information can be found at the *DroneScout* contact page:
https://dronescout.co/contact/