Integration of Mesh Rider Radio with QGroundControl and the Flight Controller

Introduction

This guide expands on our previous application notes and guides for video streaming and wireless connectivity for Unmanned Aerial Systems. It demonstrates how a Doodle Labs Mesh Rider Radio can be used to send MAVlink telemetry data between QGroundControl (QGC) [1] and different Flight Controllers (FC) [2] running px4 [3].

An overview of the system setup is shown in Figure 1. We assume QGroundControl (QGC) is used as the GCS, and a FC is used on the UAS. There are various ways in which the Video Streamer could be implemented, and in this guide, we will assume a SBC such as the Nvidia Jetson Nano running GStreamer. We will also include the commands required to stream video directly to QGC.

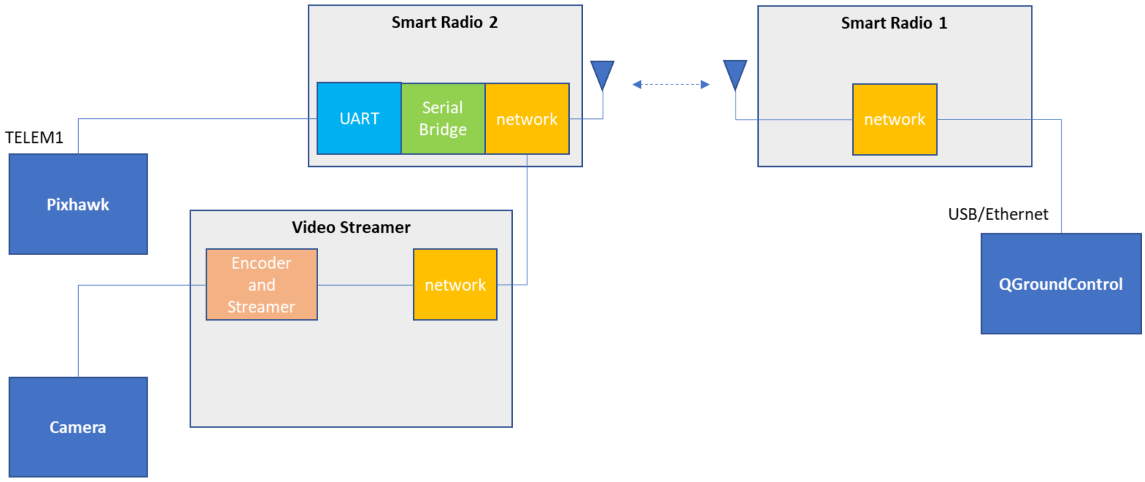


Fig. 1 System Setup

System Setup

Hardware setup with Pixhawk

By default, the Pixhawk 4 supports telemetry on serial ports TELEM1 and TELEM2. The Mesh Rider Radio UART pinout can be found in the Mesh Rider Radio Integration Guide [4]. For details on how to setup a serial to serial bridge, please consult the Doodle Labs Technical Library.

Figure 2 shows the TELEM1 pixhawk port to UART Mesh Rider Radio port wiring. UART_RX from the Mesh Rider Radio should be connected to MCU_TX on the Pixhawk 4, and UART_TX from the Mesh Rider Radio should be connected to MCU_RX on the Pixhawk 4. A ground connection is required to complete the loop. VCC_5V, MCU_RTS, and MCU_CTS can be left open.

We have only shown wiring for the -J model and the Helix models.

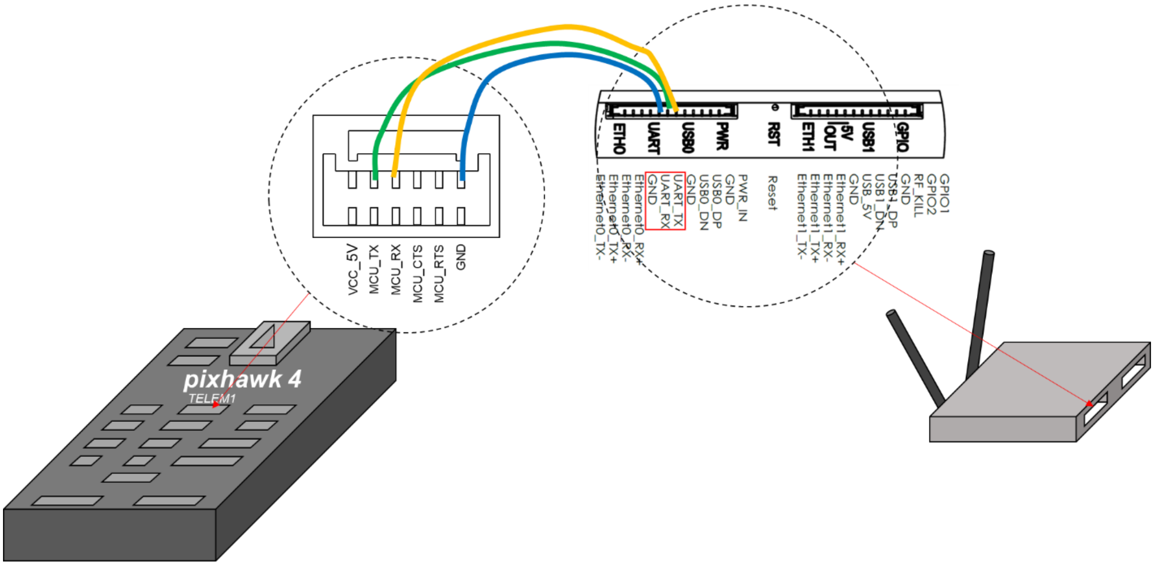


Fig. 2 Mesh Rider Radio -J model UART to Pixhawk 4 TELEM1 wiring

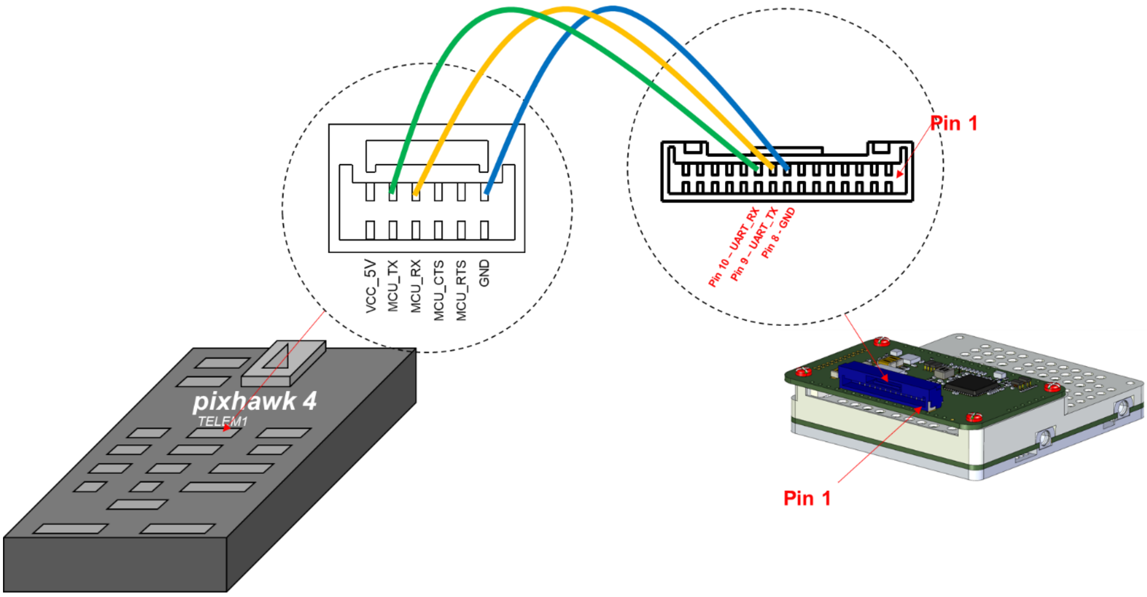


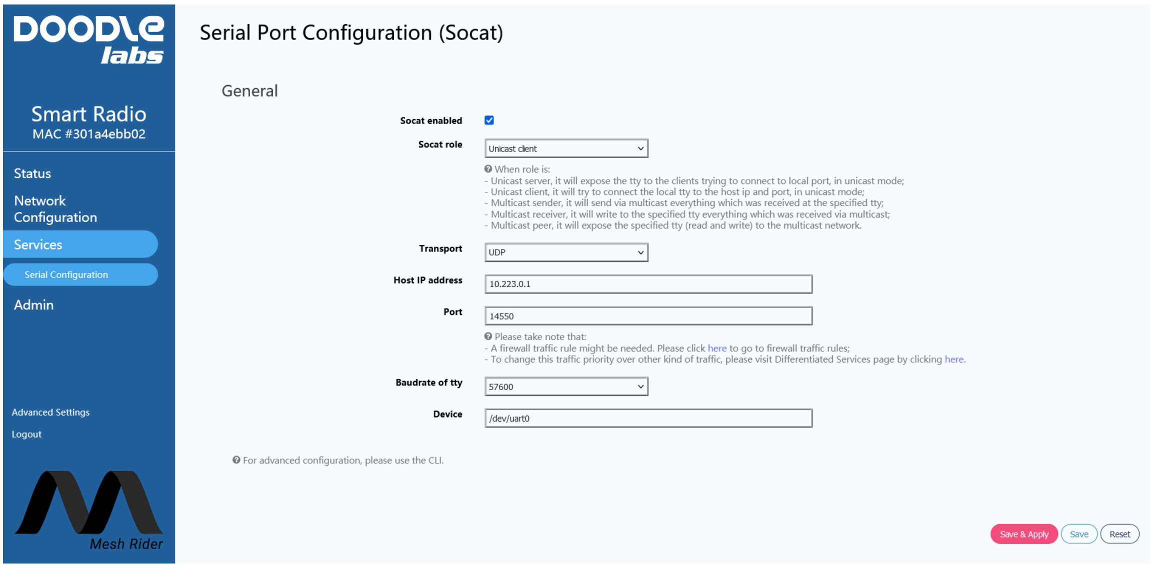
Fig. 3 Mesh Rider Radio -L model UART to Pixhawk 4 TELEM1 wiring

Mesh Rider Radio Serial Setup

In Figure 1, the serial port on Mesh Rider Radio 2 needs to be configured. We suggest you read our Serial Interface Guide for more details on how to use the serial interface (see the Doodle Labs Technical Library).

Setup using the GUI

Open up a web browser and navigate to the IP address of the Mesh Rider Radio. The Serial Configuration GUI is found in the `Services -> Serial Configuration` page. Figure 4 shows a screenshot of the page. The px4 flight controller expects a baud rate of 57600, and you can use device `/dev/uart0` as long as you don't have your own additional serial converter connected to the radio. You can either run the serial bridge as a client or a server. In client mode, you should use the IP address of the host running QGC, with UDP transport and port 14550. In server mode, `socat` will listen to an incoming connection on a port or your choice and you may use either TCP or UDP. After making your changes, click `Save & Apply`. Proceed to firewall configuration below.



Serial Port Configuration (Socat)

General

Socat enabled ☑

Socat role: Unicast client

When role is:
- Unicast server, it will expose the tty to the clients trying to connect to local port, in unicast mode;
- Unicast client, it will try to connect the local tty to the host ip and port, in unicast mode;
- Multicast sender, it will send via multicast everything which was received at the specified tty;
- Multicast receiver, it will write to the specified tty everything which was received via multicast;
- Multicast peer, it will expose the specified tty (read and write) to the multicast network.

Transport: UDP

Host IP address: 10.223.0.1

Port: 14550

Please take note that:
- A firewall traffic rule might be needed. Please click here to go to firewall traffic rules;
- To change this traffic priority over other kind of traffic, please visit Differentiated Services page by clicking here.

Baudrate of tty: 57600

Device: /dev/uart0

For advanced configuration, please use the CLI.

Fig. 4 Serial Configuration GUI

Firewall setup

By default, port 2000 is open for serial port communications. Therefore, if you setup a unicast server with UDP transport on port 2000, then no additional setup will be required.

Otherwise, you will need to open up the firewall for your desired network port. Details can be found in the Configuration Guide. Navigate to `network->firewall` in the web GUI, and click on the `Traffic Rules` tab. Under the `Open ports on router` section, add a new rule for the port you want to open (this will be port 14550 or 2000 in the examples above) and click `Save & Apply`.

Recommended Settings

We recommend using UDP transport for Command & Control as it does not establish a connection between the server and client. In the event that the drone moves in and out of range, a TCP connection will experience higher latency and significantly higher reconnection time should the connection break.

QGroundControl Setup

Detailed guides for setting up QGroundControl and Pixhawk are available online [1][2], and we won't detail the steps here. The basic steps are to install QGC on your GCS, hook it directly up to the flight controller over USB, and then program the flight control software. After the flight control software is programmed, you can setup the hardware with the Mesh Rider Radio in the link as in figure 5.

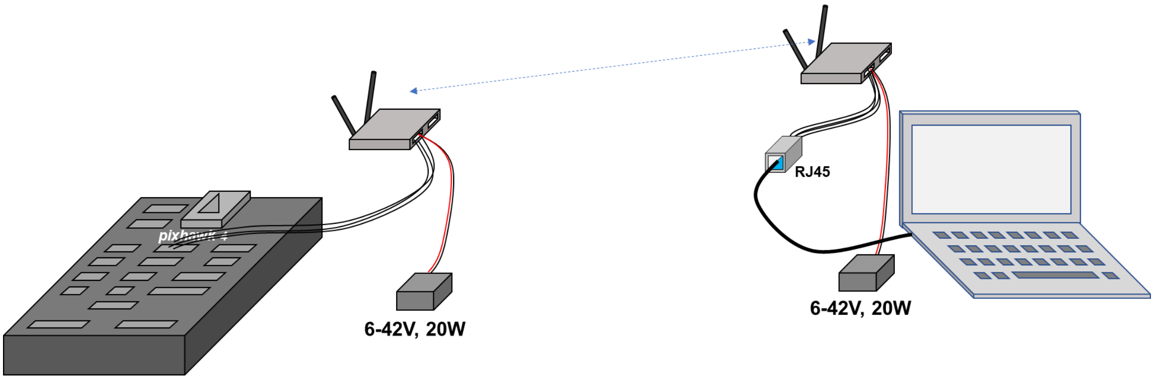


Fig. 5 Laptop to Drone Telemetry Setup

Mesh Rider Radio Optimization

The following optimizations apply to both Mesh Rider Radio 1 and Mesh Rider Radio 2 in Figure 1.

The Mesh Rider Radio Configuration Guide includes a section on Common Network Settings which we suggest you review for details on how to configure the Mesh Rider Radio. We recommend

- For the RM-2450, 2.4-GHz ISM band radio, do not use 20-MHz bandwidth, use 5/10/15 MHz bandwidths instead to avoid interference from WiFi devices.
- Use the C&C and Voice queue for telemetry (MAVLink) data. In the example above, you would map port 3000 (or 14550) to the CS6 queue. Enable "Optimize Command & Control for Voice and URLLC".
- Use the Video queue for Video. Enable "Optimize Video Streaming".

Network Setup

In Figure 1, the system running QGC, Mesh Rider Radio 2, and the Video Streamer all need to be on the same IP subnet. One easy way to do that is to run a DHCP server on one of the systems, and run DHCP clients on the other two systems. Please consult the Configuration Guide for details on how to setup a DHCP server.

Video Streamer Setup

A separate application note, "Mesh Rider Radio Video Streaming with the Nvidia Jetson Nano" (see the Doodle Labs Technical Library) details how you can go about setting up a video streamer. , A tip that in our tests, QGC ran faster with RTP stream using the GStreamer command,

```
gst-launch-1.0 nvarguscamerasrc ! 'video/x-raw(memory:NVMM),width=1920,
height=1080, framerate=30/1, format=NV12' ! nvvidconv ! omxh264enc
iframeinterval=15 control-rate=constant profile=baseline ! video/x-h264,
stream-format=byte-stream ! rtph264pay ! udpsink host=10.223.0.80 port=5000
```

Change the host to the IP address of the GCS, and choose a network port. Note that Video Streaming is optional.

Android Setup

There are three ways to connect an Android device to a Mesh Rider Radio

1. USB Reverse Tethering. This option is typically available on Smart Tablets, where it is expected that they do not have a dedicated internet connection and are getting their interconnection from another device. In this mode, the Smart Tablet acts as DHCP client, or can be assigned a static IP address. The Smart Tablet's USB port is in "host" mode and supplies 5-V to a connected device. Please make sure that the tablet is able to supply about 15W of power on its USB-C port. This mode is only supported by the Wearable and Helix Mesh Rider Radio form factors and not the Embedded or External form factors.
2. WiFi. The Wearable Mesh Rider Radio includes an auxiliary WiFi radio and this is the recommended way to connect an Android device to the Mesh Rider network.
3. USB Tethering. This option is available on most Smart Phones, In this mode, the Smart Phone starts a DHCP server at a fixed IP address. The Smart Phone's OTG port is in "device" mode and requires a 5-V input. Generally it is not possible to turn off the DHCP server or modify the IP address of the Smart Phone without rooting the device.

The recommended setup is shown in Figure 6. The Wearable Mesh Rider Radio runs a WiFi AP which is bridged to the Mesh Rider network that the Tablet can connect to. The next best option is USB Reverse Tethering as it doesn't impose any limitations on the IP addressing. Make sure to choose a Tablet which supports USB Reverse Tethering.

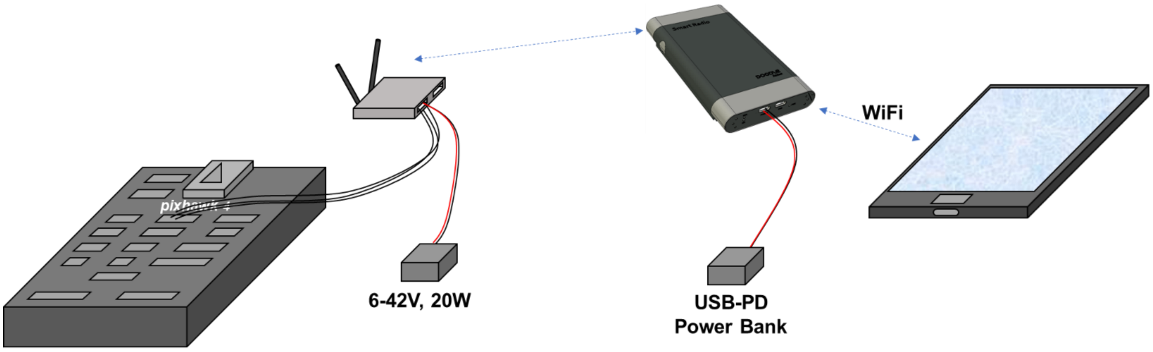


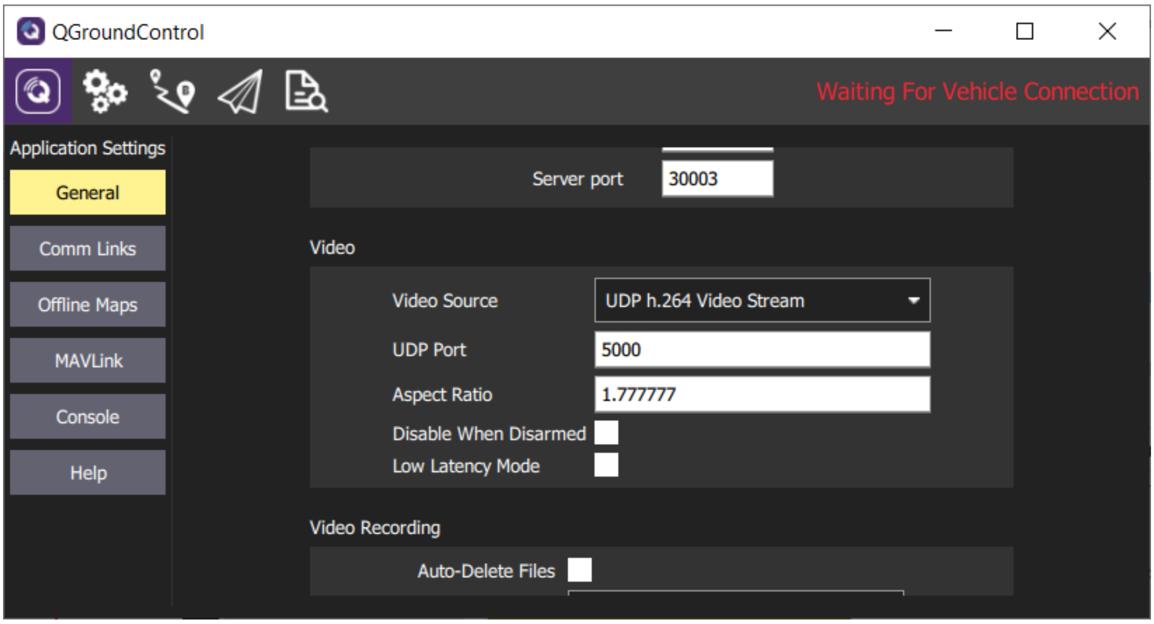
Fig. 6 QGroundControl MAVLink Verification

Testing

We tested the two setups described above with a PC running QGC in Linux, and a Tablet running QGC in Android.

Video Streaming

You can view your video stream directly in QGC by navigating to the Application Settings menu and scrolling down to the video section. Following our GStreamer Example, you would tell QGC to listen at port 5000 as shown below. You should then be able to see the video stream directly in the main window. For more details, refer to the detailed guide, "Concurrent HD Video Streaming and Low Latency C&C Tutorial" in our technical library.



Server port: 30003

Video Source: UDP h.264 Video Stream

UDP Port: 5000

Aspect Ratio: 1.777777

Disable When Disarmed □

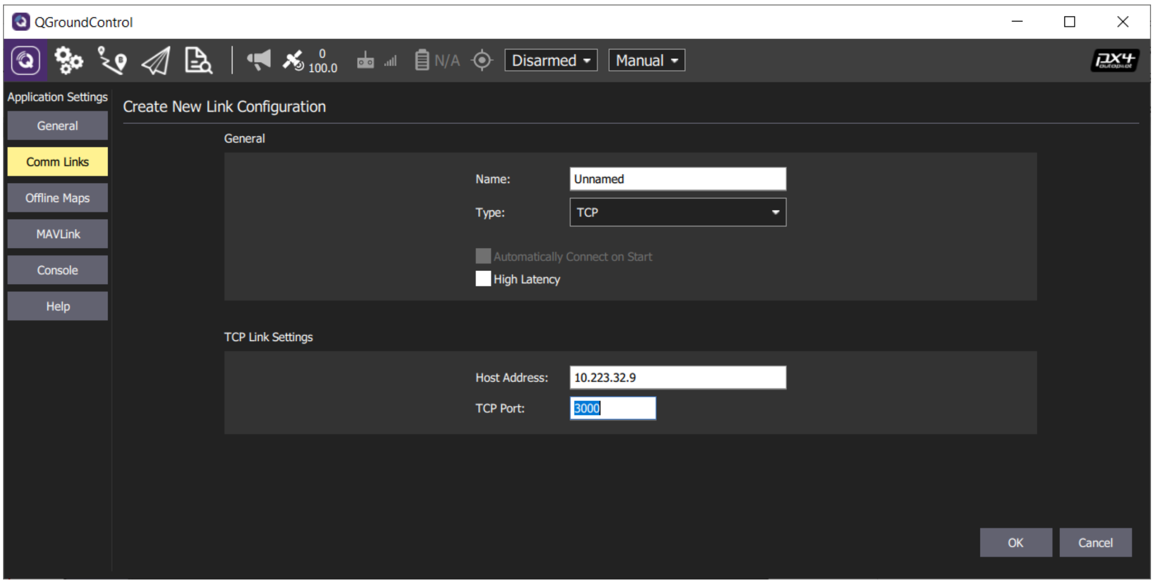
Low Latency Mode □

Auto-Delete Files □

Fig. 7 QGroundControl Video Setup

Mavlink Test

In QGroundControl, add a new Comm Link as shown in Figure 6. Change the host address to the IP address of the Mesh Rider Radio connected to the drone, and the network port to the listening port used by the Mesh Rider Radio. Click *OK*, and then *Connect*.



Create New Link Configuration

General

Name: Unnamed

Type: TCP

□ Automatically Connect on Start

□ High Latency

TCP Link Settings

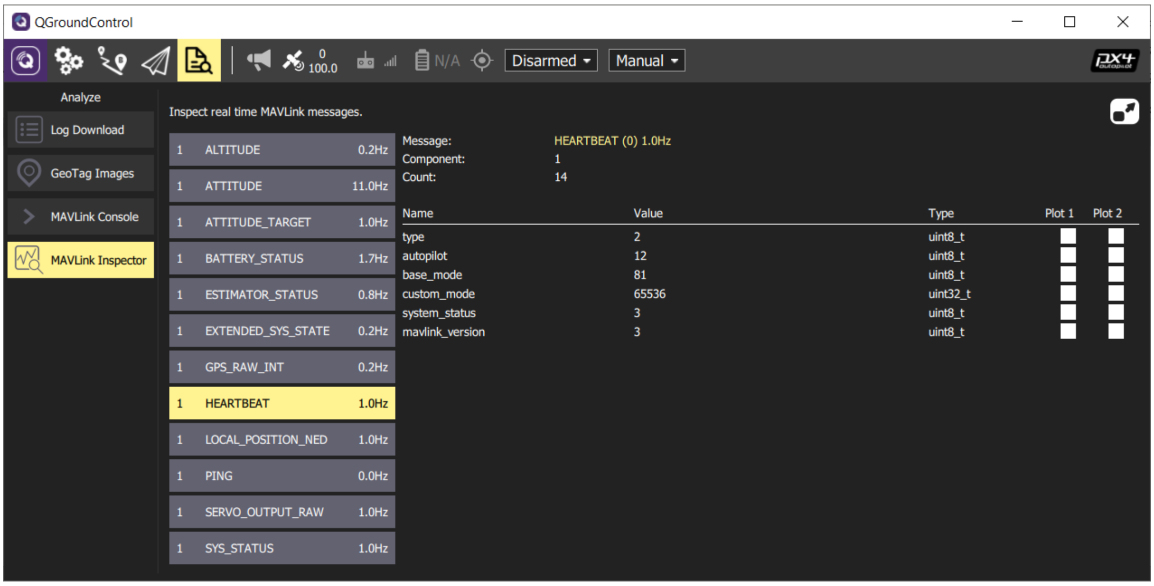
Host Address: 10.223.32.9

TCP Port: 3000

Fig. 8 QGroundControl Setup

Linux/Windows

You can verify the MAVLink connection is good by going to the tab shown in Figure 9, and inspecting the MAVLink heartbeat.



Inspect real time MAVLink messages.

1	ALTITUDE	0.2Hz
1	ATTITUDE	11.0Hz
1	ATTITUDE_TARGET	1.0Hz
1	BATTERY_STATUS	1.7Hz
1	ESTIMATOR_STATUS	0.8Hz
1	EXTENDED_SYS_STATE	0.2Hz
1	GPS_RAW_INT	0.2Hz
1	HEARTBEAT	1.0Hz
1	LOCAL_POSITION_NED	1.0Hz
1	PING	0.0Hz
1	SERVO_OUTPUT_RAW	1.0Hz
1	SYS_STATUS	1.0Hz

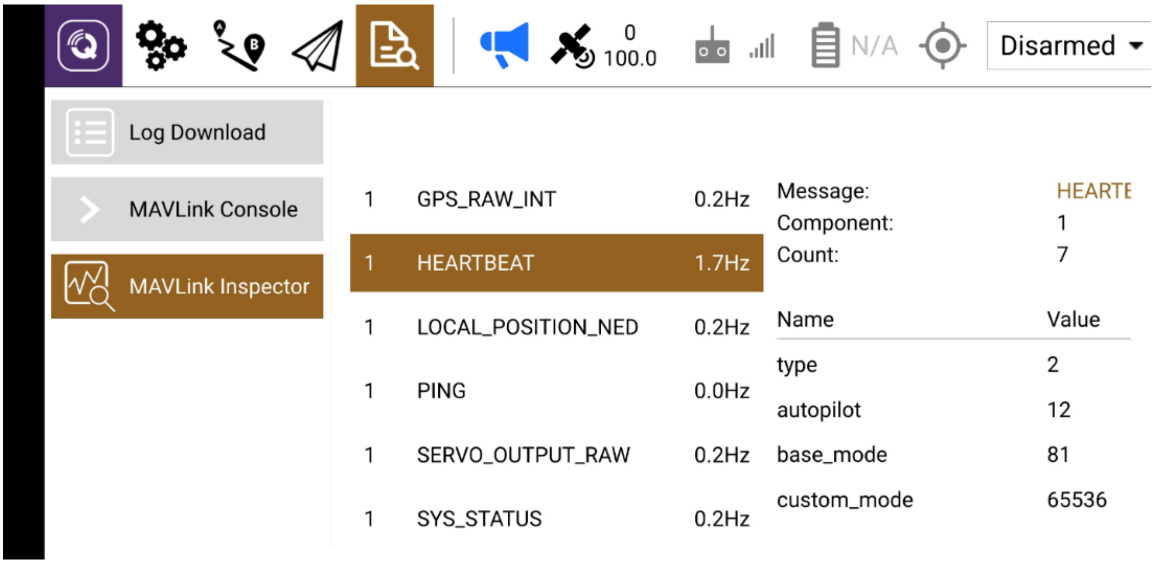
Message: HEARTBEAT (0) 1.0Hz
Component: 1
Count: 14

Name	Value	Type	Plot 1	Plot 2
type	2	uint8_t	□	□
autopilot	12	uint8_t	□	□
base_mode	81	uint8_t	□	□
custom_mode	65536	uint32_t	□	□
system_status	3	uint8_t	□	□
mavlink_version	3	uint8_t	□	□

Fig. 9 QGroundControl MAVLink Verification

Android

As far as QGC is concerned, the setup for Android is the same as for Linux. The picture below show results after a successful connection.



1	GPS_RAW_INT	0.2Hz
1	HEARTBEAT	1.7Hz
1	LOCAL_POSITION_NED	0.2Hz
1	PING	0.0Hz
1	SERVO_OUTPUT_RAW	0.2Hz
1	SYS_STATUS	0.2Hz

Message: HEARTE
Component: 1
Count: 7

Name	Value
type	2
autopilot	12
base_mode	81
custom_mode	65536

Fig. 10 QGroundControl Android MAVLink Verification

References

1. qgroundcontrol, http://qgroundcontrol.com/, 21-6-2020
2. Pixhawk 4, https://docs.px4.io/v1.9.0/en/flight_controller/pixhawk4.html, 30-March-2020
3. px4 autopilot, https://px4.io/, 21-June-2020