

Interfacing Raspberry-pi with APM flight controller

Akshit Gandhi
Keyur Rakholiya

June 30, 2016

Contents

1	Objective	3
2	Prerequisites	3
3	Hardware Requirement	3
4	Procedure	3
4.1	Connecting to RPi with an SSH/Telnet client	4
4.2	Install the required packages on the Raspberry Pi	6
4.3	Disable the OS control of the serial port	6
4.4	Testing the connection	7
4.5	Configure MAVProxy to always run	9
5	References	9

1 Objective

Topic: Interfacing Raspberry-pi with APM(ARDUPILOT) flight controller
APM is a flight controller which performs the stabilization of quadcopter by varying the signal to the motors based upon Radio control input. It has different flight modes which we can set from remote. It also supports different sensors like GPS ,Sonar etc. In this tutorial we will learn to how to setup a communication between the and APM flight controller, so that we can run/write our python scripts on R-PI and execute them on our drone. We will be going into details of each step.

2 Prerequisites

It's better if you first go through the official documentation on ardupilot's website on connecting R-PI with APM.[1], it will be beneficial also for this tutorial and also for later tutorials.

3 Hardware Requirement

Drone or Quadcopter, Raspberry pi, telemetry port connector for connecting APM to RPI and few jumpers(M-F).

4 Procedure

This tutorial explains how to connect and configure a Raspberry Pi (RPi) so that it is able to communicate with a APM flight controller using the MAVLink protocol[2] over a serial connection

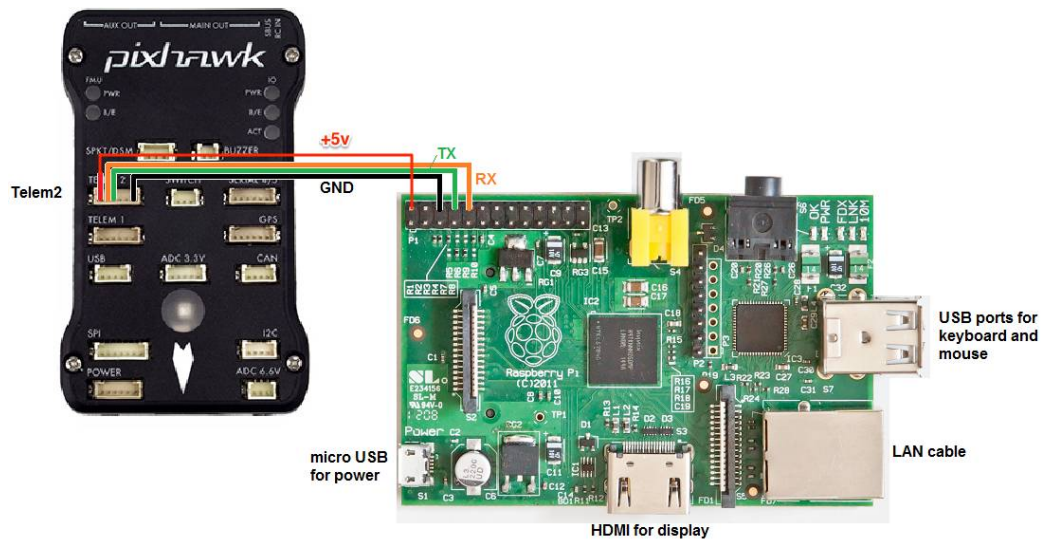


Figure 1: Connection diagram.

The picture shown above is for pixhawk controller but the same connection works for APM also.

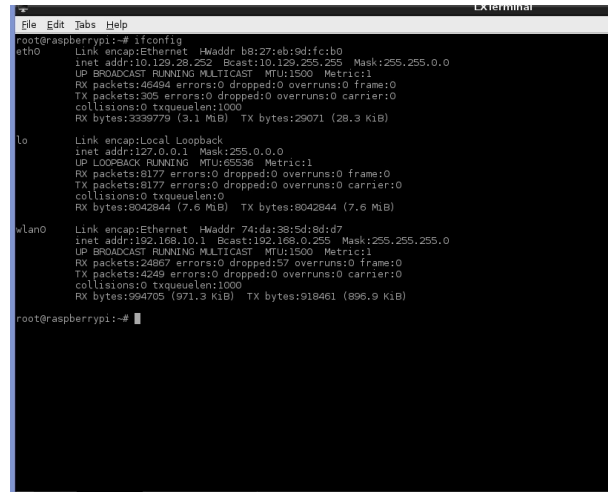
4.1 Connecting to RPi with an SSH/Telnet client

NOTE: These steps assume that you have already set-up your RPi so that it is running Raspbian.

To avoid the requirement to plug a keyboard, mouse and HDMI screen into your RPi it is convenient to be able to connect from your Desktop/Laptop computer to your RPi using an SSH/Telnet client such as PuTTY.

- Connect the RPi to your local network by one of the following methods:
 - Connecting an Ethernet LAN cable from the RPi board to your Ethernet router, or
 - Use a USB dongle to connect your RPi to the local wireless network, or
 - Connect the Ethernet LAN cable between the RPi and your desktop/laptop computer. Open the control panels Network and Sharing Center, click on the network connection through which your desktop/laptop is connected to the internet, select properties and then in the sharing tab, select Allow other networks to connect through this computers Internet connection
- Determine the RPi's IP address:

- If you have access to the RPi's terminal screen (i.e. you have a screen, keyboard, mouse connected) you can use the `/sbin/ifconfig` command.



```

root@raspberrypi:~# ifconfig
eth0:  Link encap:Ethernet  HWaddr b8:27:eb:9d:fc:b0
        inet addr:10.129.28.252  Bcast:10.129.255.255  Mask:255.255.0.0
        UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
        RX packets:46494 errors:0 dropped:0 overruns:0 frame:0
        TX packets:305 errors:0 dropped:0 overruns:0 carrier:0
        collisions:0 txqueuelen:1000
        RX bytes:3339779 (3.1 MiB)  TX bytes:29071 (28.3 KiB)

lo:    Link encap:Local Loopback
        inet addr:127.0.0.1  Mask:255.0.0.0
        UP LOOPBACK RUNNING  MTU:65536  Metric:1
        RX packets:8177 errors:0 dropped:0 overruns:0 frame:0
        TX packets:8177 errors:0 dropped:0 overruns:0 carrier:0
        collisions:0 txqueuelen:0
        RX bytes:8042844 (7.6 MiB)  TX bytes:8042844 (7.6 MiB)

wlan0: Link encap:Ethernet  HWaddr 74:da:38:5d:8d:d7
        inet addr:192.168.10.1  Bcast:192.168.0.255  Mask:255.255.255.0
        UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
        RX packets:24867 errors:0 dropped:57 overruns:0 frame:0
        TX packets:4249 errors:0 dropped:0 overruns:0 carrier:0
        collisions:0 txqueuelen:1000
        RX bytes:994705 (971.3 KiB)  TX bytes:1918461 (896.9 KiB)

root@raspberrypi:~#

```

Figure 2: ip address on terminal.

- If your Ethernet router has a web interface it may show you the IP address of all connected devices
- Connect with Putty:

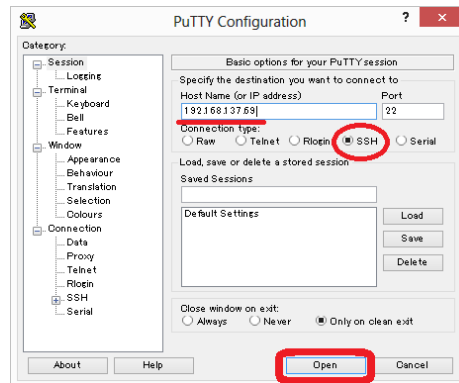


Figure 3: Connect screen with Putty.

If all goes well you should be presented with the regular login prompt to which you can enter the username/password which defaults to pi/raspberry

4.2 Install the required packages on the Raspberry Pi

Log into the RPi board (default username password is pi/raspberry) and check that its connection to the internet is working. If there's some problem then troubleshooting guide can be found on the documentation site <http://ardupilot.org/dev/docs/raspberry-pi-via-mavlink.html>

After the internet connection is confirmed to be working install these packages:

- `sudo apt-get update`
- `sudo apt-get install screen python-wxgtk2.8 python-matplotlib python-opencv python-pip python-numpy python-dev libxml2-dev libxslt-dev`
- `sudo pip install pymavlink`
- `sudo pip install mavproxy`

4.3 Disable the OS control of the serial port

Use the Raspberry Pi configuration utility for this.

Type:

- sudo raspi-config
- And in the utility, select Advanced Options:

[illegible]

Figure 4: RasPiConfiguration Utility: Serial Settings: Advanced Options.

- And then Serial to disable OS use of the serial connection:

```

===== Raspberry Pi Software Configuration Tool (rpi-software) =====
A) Overview You may need to configure overrides if black bars are present on display
B) hostname Set the name for this Pi on a network
C) Memory split Change the amount of memory made available to the GPU
D) SWP Enable/disable read/write access to the swap partition using SWP
E) Device Tree Enable/disable the use of Device Tree
F) I2C Enable/disable I2C bus access (e.g. for PiFaced)
G) SPI Enable/disable SPI bus access (e.g. for PiFaced)
H) UART Enable/disable automatic loading of UART kernel module
I) General Enable/disable General System Settings from the General category
A) update Update the tool to the latest version
B) update Update the tool to the latest version

<Select>                                <Back>

```

Reboot the Raspberry Pi when you are done.

4.4 Testing the connection

To test the RPi and APM are able to communicate with each other first ensure the RPi and APM are powered, then in a console on the RPi type:

- `sudo -s`
- `mavproxy.py --master=/dev/ttyAMA0 --baudrate 57600 --aircraft My-Copter`

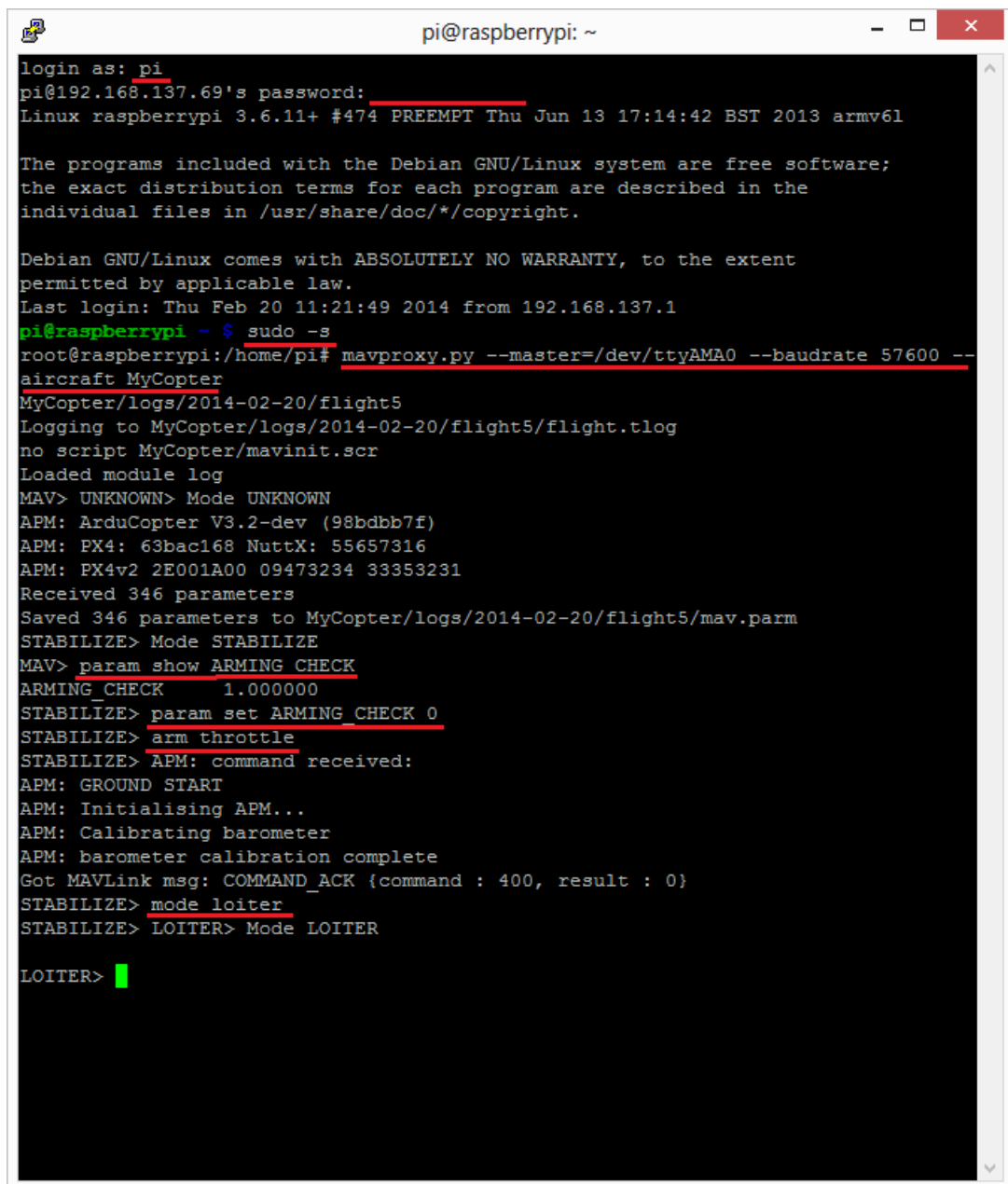
rasberry pi's default port is connected with console input/output. so use this port we have to disable it first. now we have connected our device via UART port of R-Pi. so `"/dev/ttyAMA0"` is a port for serial communication between R-pi and APM.

You configure BaudRate as bits per second. The transferred bits include the start bit, the data bits, the parity bit (if used), and the stop bits. However, only the data bits are stored.

The baud rate is the rate at which information is transferred in a communication channel. In the serial port context, "9600 baud" means that the serial port is capable of transferring a maximum of 9600 bits per second. If the information unit is one baud (one bit), the bit rate and the baud rate are identical. If one baud is given as 10 bits, (for example, eight data bits plus two framing bits), the bit rate is still 9600 but the baud rate is 9600/10, or 960. You always configure BaudRate as bits per second. Therefore, in the previous example, set BaudRate to 9600.

Note: In the above command it's MyCopter and My-Copter

- Once MAVProxy has started you should be able to type in the following command to display the `ARMING_CHECK` parameters value
 - `param show ARMING_CHECK`
 - `param set ARMING_CHECK 0`
 - `arm throttle`
- * And then Serial to disable OS use of the serial connection:



```
login as: pi
pi@192.168.137.69's password: 
Linux raspberrypi 3.6.11+ #474 PREEMPT Thu Jun 13 17:14:42 BST 2013 armv6l

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Thu Feb 20 11:21:49 2014 from 192.168.137.1
pi@raspberrypi ~$ sudo -s
root@raspberrypi:/home/pi# mavproxy.py --master=/dev/ttyAMA0 --baudrate 57600 --
aircraft MyCopter
MyCopter/logs/2014-02-20/flight5
Logging to MyCopter/logs/2014-02-20/flight5/flight.tlog
no script MyCopter/mavinit.scr
Loaded module log
MAV> UNKNOWN> Mode UNKNOWN
APM: ArduCopter V3.2-dev (98bdbb7f)
APM: PX4: 63bac168 NuttX: 55657316
APM: PX4v2 2E001A00 09473234 33353231
Received 346 parameters
Saved 346 parameters to MyCopter/logs/2014-02-20/flight5/mav.parm
STABILIZE> Mode STABILIZE
MAV> param show ARMING_CHECK
ARMING_CHECK 1.000000
STABILIZE> param set ARMING_CHECK 0
STABILIZE> arm throttle
STABILIZE> APM: command received:
APM: GROUND START
APM: Initialising APM...
APM: Calibrating barometer
APM: barometer calibration complete
Got MAVLink msg: COMMAND_ACK {command : 400, result : 0}
STABILIZE> mode loiter
STABILIZE> LOITER> Mode LOITER

LOITER> █
```

Figure 5: Terminal screen.

NOTE: If you get an error about not being able to find log files or if this example otherwise doesnt run properly, make sure that you havent accidentally assigned these files to another username, such as Root.

Entering the following at the Linux command line will ensure that all files belong to the standard Pi login account:

```
* sudo chown -R pi /home/pi
```

4.5 Configure MAVProxy to always run

To setup MAVProxy to start whenever the RPi is restarted open a terminal window and edit the `/etc/rc.local` file, adding the following lines just before the final `exit 0` line:

```
(  
date  
echo $PATH  
PATH=$PATH:/bin:/sbin:/usr/bin:/usr/local/bin  
export PATH  
cd /home/pi  
screen -d -m -s /bin/bash mavproxy.py --master=/dev/ttyAMA0 --baudrate 57600 --aircraft MyCopter  
) > /tmp/rc.log 2>&1  
exit 0
```

To open `/etc/rc.local` file type in `sudo nano /etc/rc.local` and type the code below and save it in the file

If you wish to connect to the MAVProxy application that has been automatically started you can log into the RPi and type: `sudo screen -x`

It is also worth noting that MAVProxy can do a lot more than just provide access to your APM. By writing python extension modules for MAVProxy you can add sophisticated autonomous behaviour to your vehicle. A MAVProxy module has access to all of the sensor information that your APM has, and can control all aspects of the flight.

5 References

- [1]<http://ardupilot.org/dev/docs/raspberry-pi-via-mavlink.html>
- [2]<http://qgroundcontrol.org/mavlink/start> <http://ardupilot.org/dev/docs/raspberry-pi-via-mavlink.html>

Everything mentioned above bit by bit is from the official documentation.