Internship Project Installation Guide

Giuseppe Cervone
giuseppe.cervone@edu.unife.it

1 Introduction

The following installation guide is meant to be followed while installing the current system on your network. It will break down how to set up the specific devices, which software has to be installed and the configuration parameters to follow for a complete installation.

2 Devices

2.1 Raspberry Pi

As far as choosing the best Raspberry Pi model for the job, I have chosen to use the Raspberry Pi 3. For this type of project, a machine that can be used without cooling, that has enough RAM to handle possible big data chunks and that has a fast enough CPU at a low enough cost is highly suggested. Needing for it not to be cooled removes the Raspberry Pi 4 from the equation, since it prefers having a cooling solution and the increased clock speeds won't help with serial communication limitations, on the other hand the Raspberry Pi 2 already comes with 1Gb Ram, but misses the clock speed needed for multiple programs to run at a good enough speed. This particular machine should be able to handle the data size we are planning to share and the processes we want to run on it. The machines are configured as follows:

- Raspberry Pi 3 Model B Ver1.2:
  - Quad Core 1.2GHz Broadcom BCM2837 64bit CPU
  - 1GB RAM
  - BCM43438 wireless LAN and Bluetooth Low Energy (BLE) on board
  - 100 Base Ethernet
  - 40-pin extended pinout with GPIO support
  - 4 USB 2 ports
  - 4 Pole stereo output and composite video port
  - Full size HDMI
  - CSI camera port for connecting a Raspberry Pi camera
  - DSI display port for connecting a Raspberry Pi touchscreen display
  - Micro SD port for loading your operating system and storing data
  - Upgraded switched Micro USB power source up to 2.5A
  - Raspberry Pi OS Lite (latest version)
  - 16GB microSD card

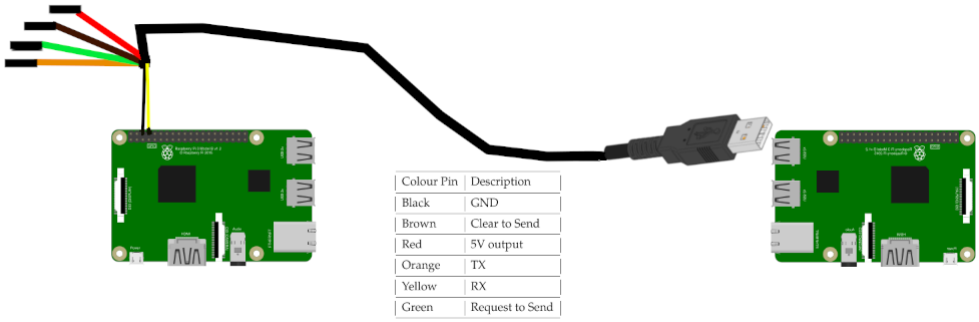
Using the RPI-Imager tool I installed the latest Raspberry PI OS image. Upon first boot up, remember that RPi login is username: pi, password: raspberry. As Raspberry Pi OS Lite is an OS without a desktop environment, we don't have a systems setting manager with a GUI, luckily we can use the utility *raspi-config* to enable a series of features useful for testing software. Using the command *sudo raspi-config* we can open the utility, once in the main menu we change these specific settings:

- In the system options:
  - Enable WIFI for SSH purposes.
  - Change username and password if needed.
- In the interface options:
  - Enable SSH. Using SSH is suggested for development on these units, as they are installs without a DE/WM(desktop environment/window manager) configuration, and thus it's best to use your preferred terminal emulator.
  - Enable Serial interface, remembering to disable login shell but enable serial interface.
- Disable bluetooth:
  - We won't be using bluetooth for the project, so we disable it, making the final product more secure.
  - We disable bluetooth by adding *dtoverlay=disable-bt* in */boot/config.txt*.
- Additional software:
  - Run *sudo apt-get install python3* to install Python in it's latest version.
  - Run *sudo apt-get install python3-pip* to install Python's package manager.
  - Run pip install pyserial to install the serial library that we will be using to have the machines comunicate.
- Run *sudo apt-get upgrade* to make sure all packages are up to date.

2.2 Serial cable

The cable used for this project is a TTL-232R-3V3 cable, which is USB-to-serial, with +3.3V TTL levels UART signals. The cable has 6-pins on one end, and USB on the other. With the cable plugged in we run the command *ls /dev/tty** on both RPi machines so we have an idea of what port we use to send or receive data on each RPi in the serial communication, for the port that's sending data it will be */dev/ttyS0*, while for the other port it should be */dev/ttyUSB0*. Raspian will always map the serial port to the alias */dev/serial0*, so it is suggested to use the alias in programming as it doesn't differentiate between Raspberry Pi Models. Here below in Figure 1 there is an image showing how the cable is meant to be plugged in, while in the Appendix 1 there is the full extract from the documentation of the cable, showing the functionality of all the cable pins in detail.

The cable pinout is peculiar as most pins are not used. The pins that get used are the ground pin and the pin that sends data over to the other RPi machine. Yellow pin is plugged in GPIO14, ground can be plugged in any of the RPi ground pins.



Colour Pin	Description
Black	GND
Brown	Clear to Send
Red	5V output
Orange	TX
Yellow	RX
Green	Request to Send

Figure 1: Pinout diagram

3 Serial interface

3.1 Diagnostic test

With the current setup ready, use this diagnostic test tocheck if the two RPi are connected correctly. In Appendix 2, and in the folder of this PDF file, we have the example code and example output that can be used to troubleshoot and check that the cable has been plugged in correctly. In case this program won't run correctly, refer to the Debugging section at the end of the file. Make sure to run the code *sndtest.py* on the machine where the GPIO pins are being used, and *rcvtest.py* has to be run on the RPi where the USB is plugged in.It's important that you start by running the command *python3 rcvtest.py*, and then run *python3 sndtest.py* on the other RPi.

4 Zabbix

4.1 Installation guide

For this installation we will be using Zabbix as a network monitoring tool. For this project I used version 5.4, running on a Apache web server and MySQL database. It is suggested to follow the installation guide found on the Zabbix website specifically for our Raspberry Pi OS version. The installation guide can be found at the link *zabbix.com/download*. Having followed the guide on both machines, we can access the frontends using the RPi IP adress on our machine. Here below we can see screenshots of how the installation frontend looks like, with the parameters to use for each column.

4.2 Configuration adjustments

- RPi private network adjustments:
  - Change the name of the host "Zabbix server" to "Zabbix server pvt"
  - Add the hosts.
- RPi external network adjustments:
  - Add the hosts you want to monitor, without filling the interface parameter. Call each host with the same name that they have on the private network.
  - Add each item for each host, configure the item as "Zabbix trapper" item types.

5 File transfer setup

5.1 Python scripts

Import the folder *send* in the home directory of the RPi on the private network. Full path should look like */home/pi/send/main.py*. Do the same for the *receive* folder on the RPi in the public network.

5.2 Cron jobs

5.2.1 Private RPi

Use command crontab -e to open up the cronjob editor, if asked to choose an editor pick the editor you prefer. Include at the end of the file the following command *cron qualcosa*. Save and close, making sure that the command has been written correctly using *crontab -l*.

5.3 Public RPi

Use command crontab -e to open up the cronjob editor, if asked to choose an editor pick the editor you prefer. Include at the end of the file the following command *cron qualcosa*. Save and close, making sure that the command has been written correctly using *crontab -l*.

6 Useful features

At this point, the system should be working in it's entirety, a series of features can be configured further from the user point of view to improve the usability.

6.1 Monitoring

tail -f

6.2 Zabbix

To make the public Zabbix interface easier to use there are a couple of customization options you can consider. For example Zabbix gives you the opportunity to configure the dashboard to include the most important parameters. Zabbix also gives you the chance to configure alarms, so as to warn you if something is not functioning. The possibility of installing an Android app to access push notification and dashboard is also available.

7 Debugging

7.1 Serial Interface

- Check that UART is enabled in */boot/config.txt* by adding *enable_UART=1*.
- Check that serial console is disabled by removing *console=serial0,115200* or *console=ttyS0,115200* in */boot/cmdline.txt*.
- Check for permissions in the dialout group.
- Check that */dev/serial0* doesn't have a getty console running on it. In case it does, it can be disabled by using the commands: *sudo systemctl stop serial-getty@ttyS0.service* and *sudo systemctl disable serial-getty@ttyS0.service*.

7.2 Zabbix

7.3 Troubleshooting

- Beware that the MySQL version in the install guide could be outdated, so I suggest updating to the newest version running the command *sudo apt-get install mysql* before creating the database host. I also suggest *mysql_secure_installation* as away to correctly set-up the MySQL installation on the machine.

7.4 File transfer

tail -f

8 Appendix

8.1 Appendix 1: Documentation extract

4.2 TTL-232R-5V and TTL-232R-3V3 Cable Signal Descriptions

Header Pin Number	Name	Type	Colour	Description
1	GND	GND	Black	Device ground supply pin.
2	CTS#	Input	Brown	Clear to Send Control input / Handshake signal.
3	VCC	Output	Red	+5V output,
4	TXD	Output	Orange	Transmit Asynchronous Data output.
5	RXD	Input	Yellow	Receive Asynchronous Data input.
6	RTS#	Output	Green	Request To Send Control Output / Handshake signal.

Table 4.1 TTL-232R-5V and TTL-232R-3V3 Cable Signal Descriptions

8.2 Appendix 2: Diagnostic output and code example

Listing 1: sndtest.py

```
#!/usr/bin/env python
import time
import serial

ser = serial.Serial(
        port='/dev/serial0',
        baudrate = 9600,
        parity=serial.PARITY_NONE,
        stopbits=serial.STOPBITS_ONE,
        bytesize=serial.EIGHTBITS,
        timeout=1
)
counter=0

while True:
        ser.write(b'Write counter: %d \n'%(counter))
        time.sleep(1)
        counter += 1
```

Listing 2: rcvtest.py

```
#!/usr/bin/env python
import time
import serial

ser = serial.Serial(
        port='/dev/ttyUSB0',
        baudrate = 9600,
        parity=serial.PARITY_NONE,
        stopbits=serial.STOPBITS_ONE,
        bytesize=serial.EIGHTBITS,
        timeout=1
)

while True:
        x=ser.readline()
        print(x)
```

Listing 3: output.txt

```
b'Write counter: 0\n'
b'Write counter: 1\n'
b'Write counter: 2\n'
b'Write counter: 3\n'
b'Write counter: 4\n'
b'Write counter: 5\n'
```