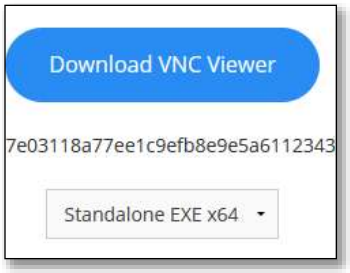
Raspberry Pi Bookworm Getting Started

Contents

Raspberry Pi Install

1. Use Raspberry Pi Imager to Create a MicroSD card image.
    a. Choose OS: Raspberry Pi OS Bookworm 64-bit
    b. Choose Storage: MicroSD adapter
    c. Click the Gear at the bottom right
    d. Set hostname
    e. Enable SSH → Use password authentication
    f. Set username and password: pi Password01
    g. Configure wireless LAN
    h. Set local settings
    i. Write file to MicroSD card
2. Insert MicroSD card → turn on Pi.
3. Use an IP scanner to find the IP address.
4. Go to https://www.chiark.greenend.org.uk/~sgtatham/putty/latest.html
5. Download the **PuTTY** client.
6. Start the **PuTTY** client. Type in the **IP address** of the Pi → Click **Open**.
7. **Accept** the **PuTTY Security Alert**.
8. Login as: **pi** Password: **Password01**
9. Type: **sudo raspi-config**
    a. **Interface Options → VNC** → Select **Yes.**
    b. **Interface Options -> I2C →** Select **Yes.**
10. Select **Finish** to exit

RealVNC Viewer



RealVNC viewer allows us to remotely control the Pi in headless mode over the network. Headless mode is when there isn't a keyboard, video, or mouse.

1. Go to https://www.realvnc.com/en/connect/download/viewer/
2. Download the VNC Viewer Standalone EXE
3. This is a self executing program, it does not need an installation
4. Double Click the downloaded file.
5. Type in the IP address of your robot → Click **Connect**.

If RealVNC doesn't work, try TigerVNC.

TigerVNC Viewer

1. https://sourceforge.net/projects/tigervnc/
2. Download the latest **vncviewer64.exe** file.
3. This is a self executing program, it does not need an installation
4. Double Click the downloaded **VNC Viewer**.
5. Type in the IP address of your robot → Click **Connect**.

Multiple SSID's

GUI

1. On the taskbar → Click the Network Connection icon.
2. **Advanced Options → Edit Connections**
3. Add or Edit a connection.

CLI

You can also use the command line **nmtui** for text based editing of the network connection.

1. sudo nmtui
2. Edit a connection

3. Add
4. Wi-Fi
5. Add your Wi-Fi SSID information.
6. Ok
7. Press Esc to exit.

Disable Onboard Wi-Fi

Any external Wi-Fi antenna will have better signal strength and range. To use an external Wi-Fi antenna only, disable the internal Wi-Fi.

```
# Edit this file with nano
sudo nano /boot/firmware/config.txt
# Add this line to the end of the file and save it
dtoverlay=disable-wifi
```

After you have disabled the on-board Wi-Fi, you must always plug a Wi-Fi adapter into a USB port.

Set to 802.11N only for range. This will disable the shorter range A.

1. On the taskbar → Click the Network Connection icon.
2. **Advanced Options → Edit Connections**
3. Wireless → Band → B/G

Turn Off Wi-Fi Power Saving

If your Pi experiences Wi-Fi connection issues, turn off the power saving.

```
# Show the connection name
sudo nmcli connection show
# Change connection name
sudo nmcli conn modify <connection-name> connection.id <new-connection-name>
# Turn on power saving
sudo nmcli connection modify <connection-name> 802-11-wireless.powersave 2
# Show all options
sudo nmcli connection show <connection-name>
# Set WPA2 key
sudo nmcli con modify <connection-name> wifi-sec.psk new_password
# Start wireless connection, ask for WPA key
sudo nmcli --ask con up <connection-name>
```

Update Raspberry Pi OS

At a terminal:

```
# Update the apt package list
sudo apt update
# Upgrade all packages -y no prompt
sudo apt upgrade -y
# Remove any packages that are not needed
sudo apt autoremove
```

Configure Raspberry Pi OS (Optional)

1. Right Click Desktop → **Desktop Preferences** → **Layout**: No image
    a. Colour: Choose a colour you like.
2. **Change Clock Display:** Right Click Clock, Configure Plugin
    a. Clock Format: **%I:%M %p**
3. **Add Temperature Monitor:** Right Click Task Bar → Add/Remove Plugins.
    a. **Add** → **CPU Temp** → **Add to right**.

Email IP on Boot

1. Create a Code folder → **home/pi/Code**

2. Copy **startup_mailer.py** to this folder
3. Open a terminal.
4. Type in the following to make the script executable.

```
chmod 744 /home/pi/Code/startup_mailer.py
```

5. There should not be any errors if the command was successful.
6. Test the script with the following command.

```
python3 /home/pi/Code/startup_mailer.py
```

7. In a few moments, you should receive an email with your Raspberry Pi IP address.

Run startup_mailer.py Script on Startup

1. At the terminal, type in the following command to access the Raspbian scheduler. (Don't add sudo)

```
crontab -e
```

2. Press **Enter** to edit the file with nano.
3. Cursor to the bottom of the file. (The mouse will not work.)
4. Enter the following information. (**sleep 15** waits 15 seconds after startup to run the script.)

```
@reboot sleep 15 && python3 /home/pi/Code/startup_mailer.py
```

5. Type **CTRL+S** to Save the file.
6. Press **CTRL+X** to Exit nano.
7. Type **sudo reboot**
8. You should receive an email with your IP address.

Wi-Fi Signal Strength

The **iwconfig** command will give you a snapshot of Wi-Fi quality.

wavemon will monitor Wi-Fi signal strength in real time.

```
sudo apt update
# Install wavemon
sudo apt install wavemon
# Run wavemon
wavemon
# Quit wavemon
q
```

Show all AP's in the area

```
# Show all AP's in the area.
sudo iwlist wlan0 scan | egrep "Cell|ESSID|Signal|Rates"
```

Signal Strength

The higher the signal strength, the more reliable the connection and higher speeds are possible. The signal strength is specified as -dBm (decibels related to one milliwatt).

Values between 0 and -100 are possible, with more being better. -51 dBm is a better signal strength than -60 dBm.

The value 0 is not realistic. Even -30 dBm is hard to reach, and you must stand almost directly next to the access point.

Some guidance on how to read the results:

- -50 dBm is considered an excellent signal strength.
- -67 dBm is said to be the minimum signal strength for reliable and relatively fast packet delivery.
- -70 dBm is the minimum signal strength for reliable packet delivery.
- -80 dBm is the minimum value for a basic connection. However, packet delivery is no longer necessarily reliable.
- -90 dBm is already very close to the basic noise. Here a connection probably does not work anymore.

Link Quality

A network can have very good signal strength without good link quality.

This is how much of the data you send and receive will make it to the destination in good condition.

The quality indicator includes data like Bit Error Rate (BER), i.e., the number of bit errors in received bits that have been altered due to noise, interference, distortion, or bit synchronization errors. Others are Signal-to-Noise and Distortion Ratio (SINAD).

It is measured in percentage or on a scale of up to 70. So you will see a value like "60/70".

Unlike signal strength, it is somewhat harder to say which values are still considered to be ok.

If the value is low and your signal strength is high, you may have interference from, e.g., kitchen appliances or other electronic devices. Moving them further away may improve the link quality.

Frequency

Another interesting indicator is the Wi-Fi frequency.

This shows if your Raspberry Pi connects to the slower and longer range 2.4 GHz network, or the faster but shorter range 5 GHz version.

Speedtest

Install speedtest-cli from Speedtest.net

```
sudo apt update
sudo apt dist-upgrade -y

# Install support software
sudo apt install apt-transport-https gnupg1 dirmngr lsb-release

# Install keychain to download speed-cli, this command is on one line.
curl -L https://packagecloud.io/ookla/speedtest-cli/gpgkey | gpg --dearmor |
sudo tee /usr/share/keyrings/speedtestcli-archive-keyring.gpg >/dev/null

sudo apt update

sudo apt install speedtest-cli

speedtest
```

Find Ports with ls dev/tty

Pi serial port is ttyAMA0

```
sudo ls /dev/tty*
```

This will list all of the terminal interfaces.

Find all serial ports

```
dmesg | grep tty
```

Cron Scheduled shutdown

```
# Edit root crontab
sudo crontab -e
# Shutdown at 11 pm each day.
0 23 * * /usr/sbin/shutdown -h now
```

Hardware Information

```
# The top command will show memory, cpu, processes, etc
top
# This one has colors and is prettier
htop
# Quit
q
```

```
cat /proc/cpuinfo
# Display the current CPU temperature
vcgencmd measure_temp
```

```
# Displays detailed information about the hardware
sudo apt update
sudo apt install lshw
sudo lshw
sudo lshw -short
```

~~Bluetooth and Serial Port~~

~~vcgencmd measure_clock isp / v3d~~

~~vcgencmd measure_clock core~~

~~mini uart settings~~

```
# Set cpu core to 250 or 500
core_freq=250
# or
force_turbo=1
```

~~# Enable Bluetooth to use miniuart~~
~~dtoverlay=miniuart-bt~~

~~another~~

~~force_turbo=1~~
~~gpu_freq=250~~
~~gpu_freq_min=250~~

~~Turn Off HDMI~~

~~vcgencmd display_power 0 turns off the screen~~

~~vcgencmd display_power 1 turns on the screen~~

Enable snaps on Raspberry Pi and install btop

Snaps are applications packaged with all their dependencies to run on all popular Linux distributions from a single build. They update automatically and roll back gracefully.

Snaps are discoverable and installable from the Snap Store, an app store with an audience of millions.

On a Raspberry Pi running the latest version of Raspbian snap can be installed directly from the command line:

```
sudo apt update
sudo apt install snapd
```

You will also need to reboot your device:

```
sudo reboot
```

To install btop, simply use the following command:

```
sudo snap install btop
```

Filesystem Checks and Repair

The Linux filesystem can be damaged under various circumstances, e.g., system crash, power loss, disconnected disk, accidentally overwritten i-node, etc. Thus it is a good idea to check the integrity of the filesystem regularly to minimize the risk of filesystem corruption.

The Linux filesystem can be damaged under various circumstances, e.g., system crash, power loss, disconnected disk, accidentally overwritten i-node, etc. It is a good idea to check the integrity of the filesystem regularly to minimize the risk of filesystem corruption. Add the following to `/boot/cmdline.txt`:

```
fsck.mode=force
```

Make sure that file remains all one line. Parameters should be separated with spaces.

You'll probably notice `fsck.repair=yes` is already there; these are not the same thing. From `man systemd-fsck` (these are parameters that are passed on by the kernel to init, i.e., systemd):

fsck.mode=

One of "auto", "force", "skip". Controls the mode of operation. The default is "auto", and ensures that file system checks are done when the file system checker deems them necessary. "force" unconditionally results in full file system checks. "skip" skips any file system checks.

fsck.repair=

One of "preen", "yes", "no". Controls the mode of operation. The default is "preen", and will automatically repair problems that can be safely fixed. "yes " will answer yes to all questions by fsck and "no" will answer no to all questions.

To do a filesystem check on the next reboot, do the following

```
sudo touch /forcefsck
```

Once you create an empty file named `forcefsck` in the root directory, it will force filesystem check the next time you boot up. After successful booting, `/forcefsck` will automatically be removed.

An alternative is to shut down the system with the `-F` option like this:

```
sudo shutdown -r -F now
```

Static IP

We now need to plug this information into the Pi's network configuration file using a text editor. I always use nano text editor. . .

sudo nano /etc/network/interfaces

Simply change the line that reads:

iface eth0 inet dhcp to **iface eth0 inet static**

Then directly below this line enter the following (Please Note. **You will need your own addresses we gathered in Part B, more details below**). . . .

address 192.168.9.30
netmask 255.255.255.0
network 192.168.9.0

broadcast 192.168.9.255
gateway 192.168.9.1

CTRL X to save and exit