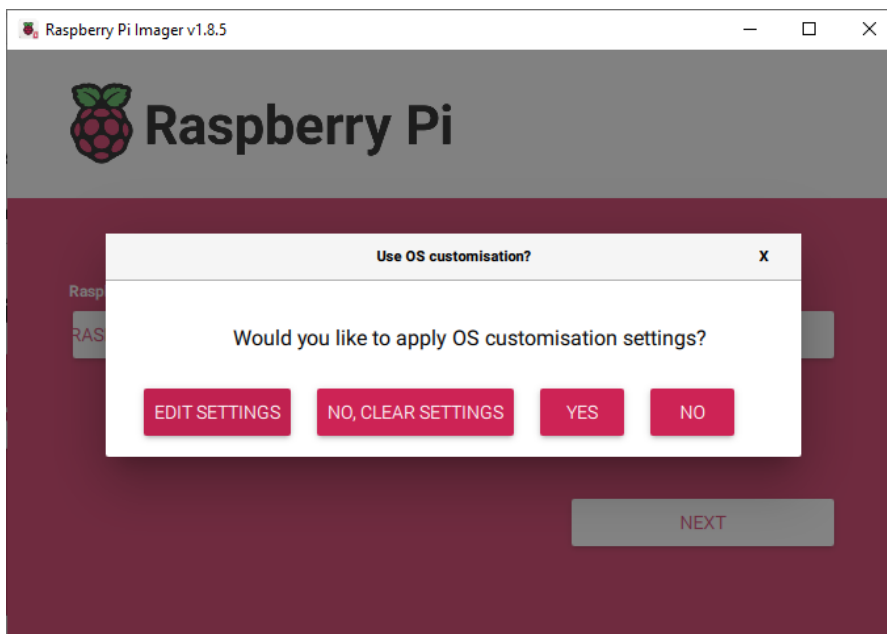
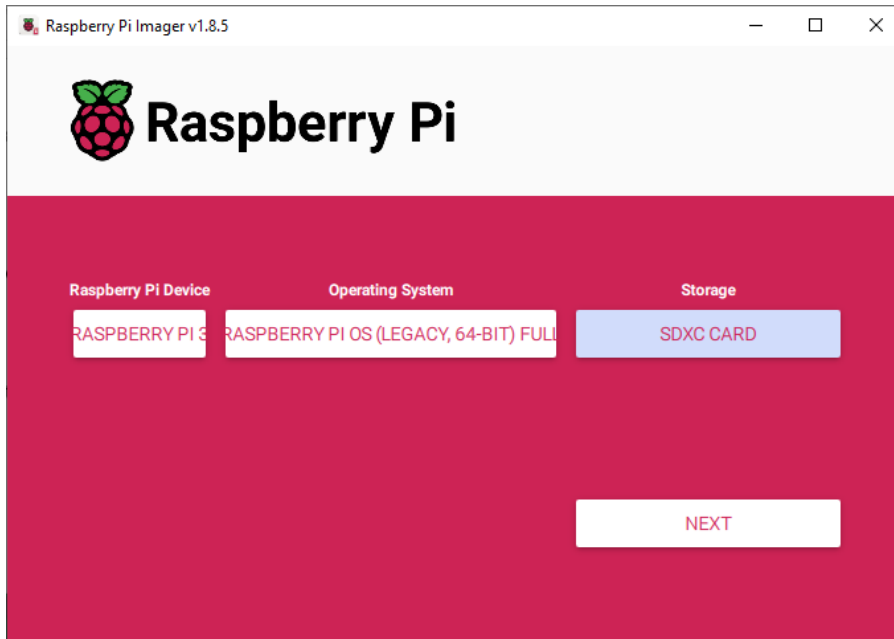


Quick Guide – How to connect and setup the components for Pi 3

Boards and Sensors

1. Raspberry Pi 3 B+ Board

- a. Obtain the SD Card from your instructor. Although you can use USB Drive for the OS, SDXC is faster and safer.
- b. Download Raspberry Pi Imager to your PC and select the following settings:



- c. Select edit settings. For the username and password, we will set it as **user** for easy access. For services, we need to enable SSH using user password authentication.

Quick Guide – How to connect and setup the components for Pi 3

The image displays three screenshots of the Raspberry Pi OS Customisation installer, showing the configuration steps for a Pi 3.

GENERAL Tab: This tab is selected in the first screenshot. It contains the following settings:

- ☒ Set hostname: `raspberrypi` .local
- ☒ Set username and password
 - Username: `USER`
 - Password: [masked]
- ☒ Configure wireless LAN
 - SSID: `your-wifi-name`
 - Password: [masked]
 - ☐ Show password ☐ Hidden SSID
 - Wireless LAN country: `GB`
- ☒ Set locale settings
 - Time zone: `Asia/Singapore`
 - Keyboard layout: `US`

A **SAVE** button is located at the bottom right.

SERVICES Tab: This tab is selected in the second screenshot. It contains the following settings:

- ☒ Enable SSH
 - ☒ Use password authentication
 - ☐ Allow public-key authentication only
 - Set authorized_keys for 'user': [text area]

A **RUN SSH-KEYGEN** button is located below the SSH settings. A **SAVE** button is at the bottom right.

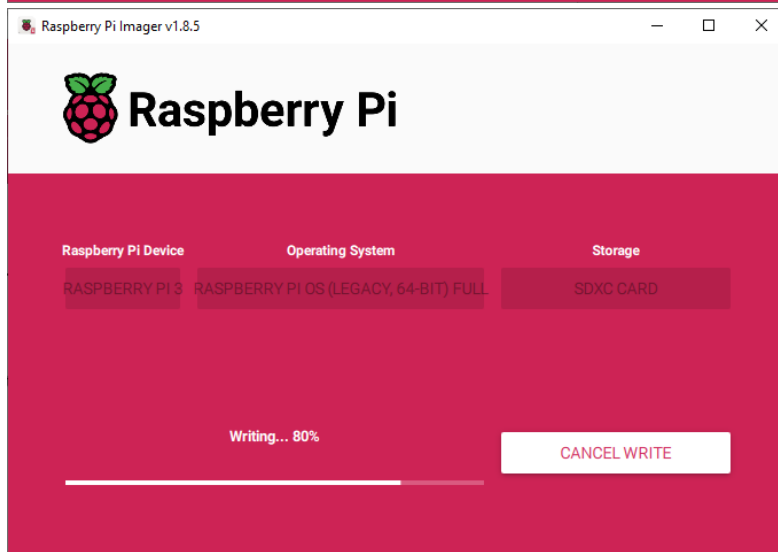
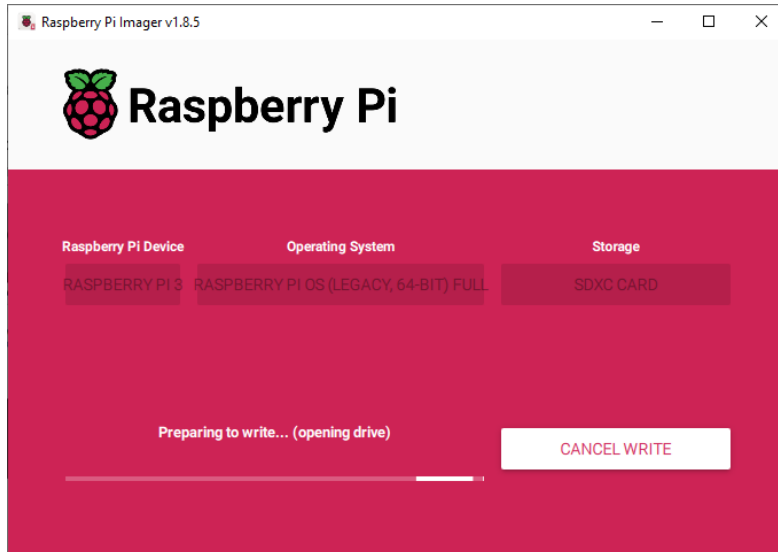
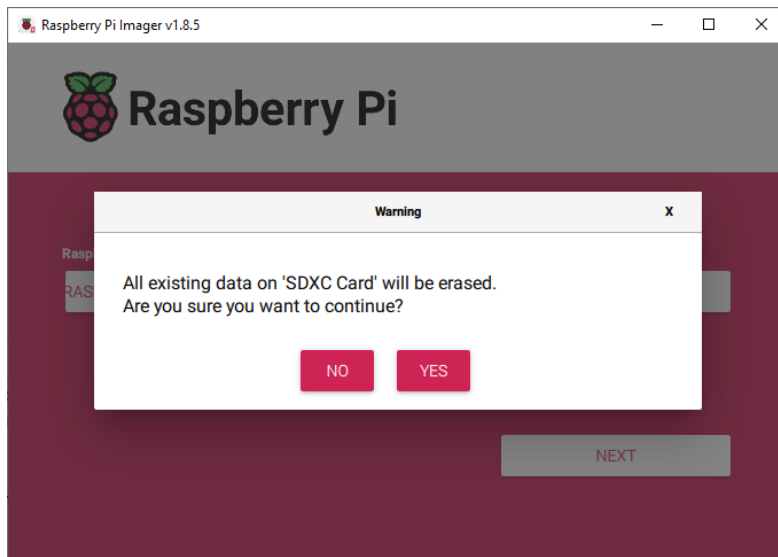
OPTIONS Tab: This tab is selected in the third screenshot. It contains the following settings:

- ☒ Play sound when finished
- ☒ Eject media when finished
- ☒ Enable telemetry

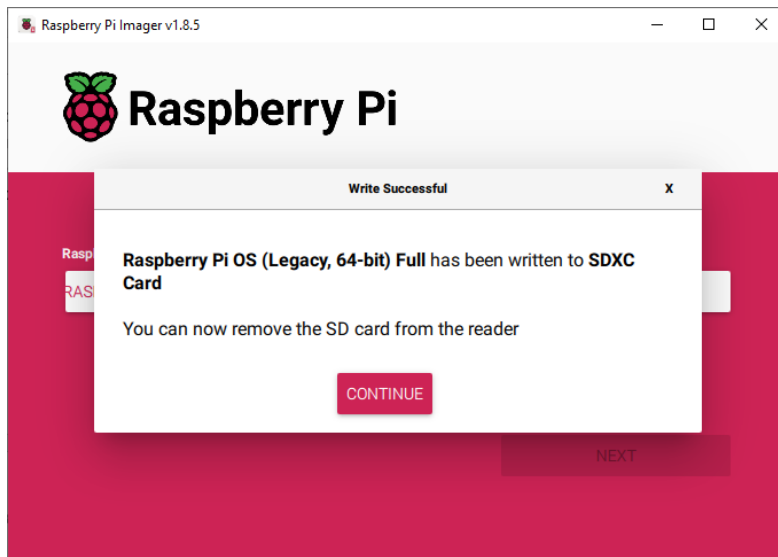
A **SAVE** button is at the bottom right.

- d. After that, select **Yes** to continue creating the OS Image on the SD Card. The SD Card will be ready in several minutes.

Quick Guide – How to connect and setup the components for Pi 3

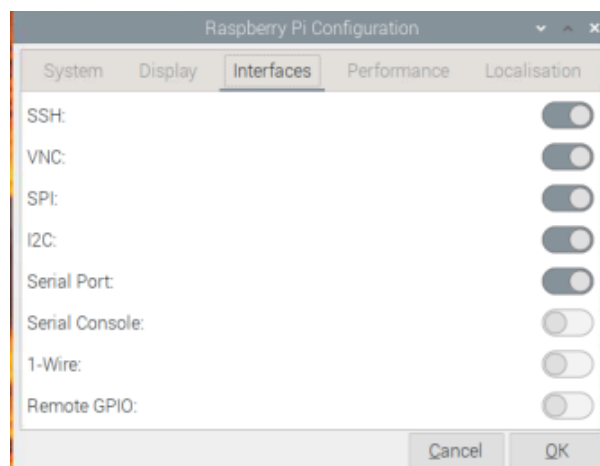


Quick Guide – How to connect and setup the components for Pi 3

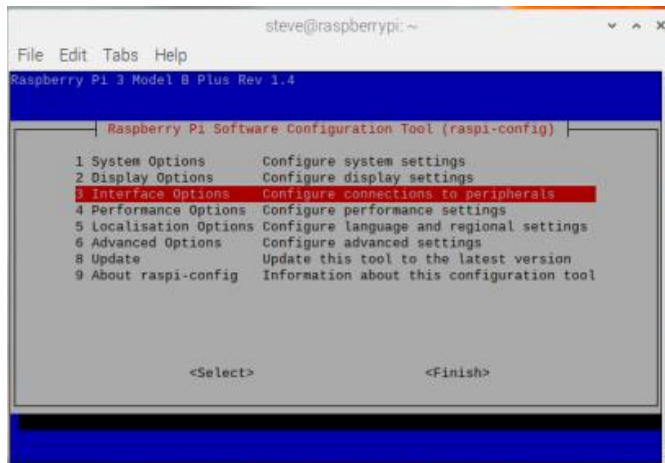


Warning: DO NOT select Format the SD Card from the Windows prompt after the completion. Just select complete from the Raspberry Pi Imager.

- e. Insert the SDXC Card and power on the board. You can use the micro-USB port to power up the board using a USB 5V phone charger that provides a minimum of 2.4A.
- f. Once power On, go to the Preferences menu and select Raspberry Pi Configuration. Make sure to enable the following Interfaces – **SSH, VNC, SPI, I2C, Serial Port**. You can also use command line **sudo raspi-config** to do the same thing.



Quick Guide – How to connect and setup the components for Pi 3

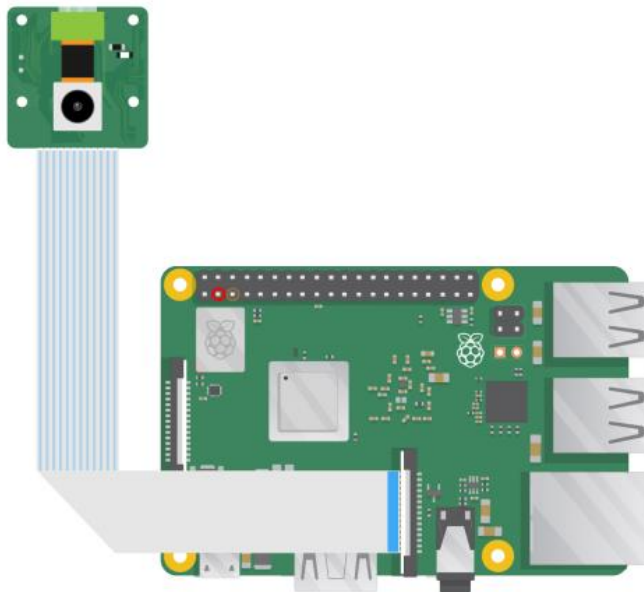


To enable graphics acceleration, navigate to **Advanced Options**, and select **Enable Glamor graphic acceleration**.

In case if you encounter errors when running the [libcamera-vid](#) tests, navigate to **Advanced Options**, **GL Driver**, and enable the **GL (Full KMS)**. To be safe, just enable it at this step.

2. Arducam 8MP IMX219

- a. Connect the flat cable to the main board like this.

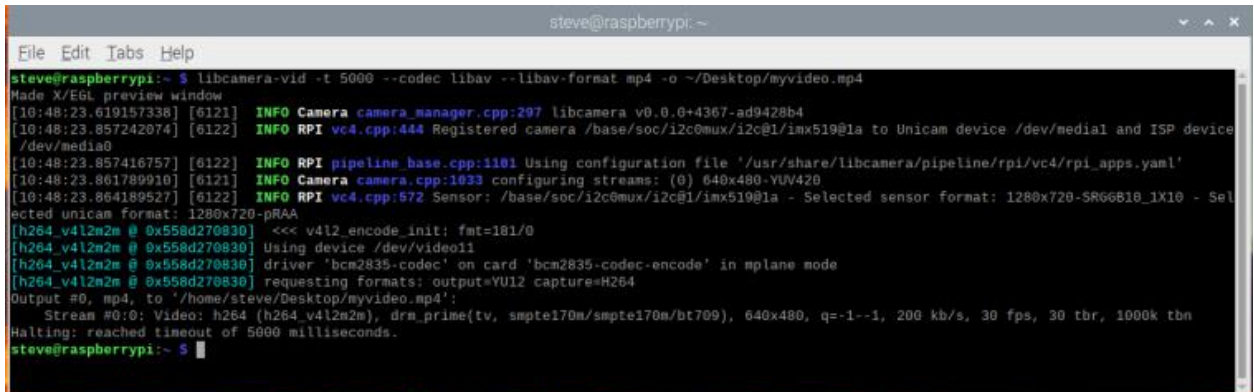


- b. Edit the `/boot/config.txt` file
 - i. Look for `camera_auto_detect = 1` and change it to 0 (zero)!
`camera_auto_detect = 0`
 - j. Add to the last line (after `[all]`)
`dtoverlay=imx219,vcm`

Quick Guide – How to connect and setup the components for Pi 3

- c. To test the camera, you can use the standard camera library from the Raspbian as follows from the terminal command line:

libcamera-vid it 5000 --codec libav --libav-format mp4 -o ~/Desktop/myvideo.mp4



```
File Edit Tabs Help
steve@raspberrypi:~$ libcamera-vid -t 5000 --codec libav --libav-format mp4 -o ~/Desktop/myvideo.mp4
Made X/EGl preview window
[10:48:23.619157338] [6121] INFO Camera camera_manager.cpp:297 libcamera v0.0.0+4367-ad9428b4
[10:48:23.857242074] [6122] INFO RPI vc4.cpp:444 Registered camera /base/soc/i2c0mux/i2c@1/imx519@1a to Unicam device /dev/media1 and ISP device /dev/media0
[10:48:23.857416757] [6122] INFO RPI pipeline_base.cpp:1101 Using configuration file '/usr/share/libcamera/pipeline/rpi/vc4/rpi_apps.yaml'
[10:48:23.861789910] [6121] INFO Camera camera.cpp:1033 configuring streams: (0) 640x480-YUV420
[10:48:23.864189527] [6122] INFO RPI vc4.cpp:572 Sensor: /base/soc/i2c0mux/i2c@1/imx519@1a - Selected sensor format: 1280x720-SRG6B18_1X10 - Selected unicam format: 1280x720-pRAA
[h264_v4l2m2m @ 0x558d270830] <<< v4l2_encode_init: fmt=181/0
[h264_v4l2m2m @ 0x558d270830] Using device /dev/video11
[h264_v4l2m2m @ 0x558d270830] driver 'bcm2835-codec' on card 'bcm2835-codec-encode' in mplane mode
[h264_v4l2m2m @ 0x558d270830] requesting formats: output=YU12 capture=H264
Output #0, mp4, to '/home/steve/Desktop/myvideo.mp4':
  Stream #0:0: Video: h264 (h264_v4l2m2m), drm_prime(tv, smpte170m/smpte170m/bt709), 640x480, q=-1--1, 200 kb/s, 30 fps, 30 tbr, 1000k tbn
Halting: reached timeout of 5000 milliseconds.
steve@raspberrypi:~$
```

- d. To use the camera in Python, you need to install several libraries. We will use the latest picamera2 library in python for our purpose. You can download the datasheet from the official Raspberry [Manual](#)
- e. Issue the command in the terminal to install the python components.
- sudo apt install -y python3-picamera2**
- f. To test the camera in Python, open Thonny from Programming menu, and type in the following code:

```
from picamera2 import Picamera2, Preview
import time
picam2 = Picamera2()
camera_config = picam2.create_preview_configuration()
picam2.configure(camera_config)
picam2.start_preview(Preview.QTGL)
picam2.start()
time.sleep(2)
picam2.capture_file("test.jpg")
```

References:

1. [For 8MP IMX219 Motorized Focus Camera - Arducam Wiki](#)
2. <https://datasheets.raspberrypi.com/camera/picamera2-manual.pdf>
3. <https://docs.arducam.com/Raspberry-Pi-Camera/Motorized-Focus-Camera/Quick-Start-Guide/IMX219-Motorized-Focus-Camera/#install-libcamera-from-arducam>

Quick Guide – How to connect and setup the components for Pi 3

g. How to write continuous monitoring Python program called *picamera2_continuous.py* ?

```
from picamera2 import Picamera2, MappedArray
from picamera2.outputs import FfmpegOutput
from picamera2.encoders import H264Encoder, Quality
import cv2, time, platform
from time import sleep
from datetime import date, datetime

#create helper function for calculating 10-minute interval
def seconds_till_x_minute(x):
    n = datetime.now()
    v = x * 60 - n.second
    return v

#main function to handle the recording
def main():
    now = datetime.now()
    filename = now.strftime("%Y-%m-%d_%H_%M_%S")
    fmt = '.mp4'
    root = '/home/steve/Desktop/Video/' #remember to change to an existing folder
    width = 1024 # default 640
    height = 768 # default 480    use main for bigger size e.g. 1024 x 768 instead of lo res
    camera_name = platform.node()
    cam = Picamera2()
    preview_config = cam.create_video_configuration(main={"size": (width, height)}, display="main")
    #lores={"size":(width,height)},display="lores" #low resolution
    #,transform=libcamera.Transform(vflip=1,hflip=1) #flips the image vertically and horizontally
    colour = (0, 255, 0)
    origin = (0, 30)
    origin_name = (width - len(camera_name)*20, 30)
    font = cv2.FONT_HERSHEY_DUPLEX
    scale = 1
    thickness = 2

    def apply_timestamp(request):
        timestamp = time.strftime("%Y-%m-%d %X")
```

Quick Guide – How to connect and setup the components for Pi 3

```
with MappedArray(request, "main") as m:

    cv2.putText(m.array, timestamp, origin, font, scale, colour, thickness)

    cv2.putText(m.array, camera_name, origin_name, font, scale, colour, thickness)

# Settings

cam.configure(preview_config)

# Afmode: 0>manual, 1=single_autofocus, 2=continuous_autofocus

cam.set_controls({"AfMode": 2, "AfTrigger": 0})
# continuous_autofocus - may report errors in the status but the picture is still fine

cam.pre_callback = apply_timestamp

cam.start()

sleep(1)

while True:

    now = datetime.now()

    filename = now.strftime("%Y-%m-%d_%H_%M_%S")

    output = FfmpegOutput(root + filename + fmt) #,audio=True #to record audio

    encoder = H264Encoder()

    quality = Quality.VERY_HIGH

    duration = seconds_till_x_minute(1) #every 1-minute

    print("record duration=" + str(duration) + " ->" + filename)

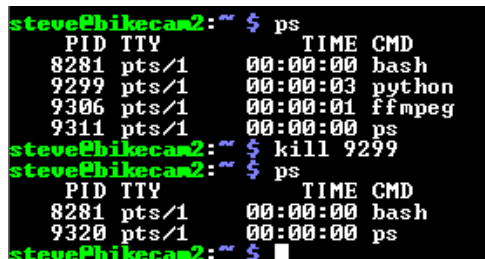
    #cam.start_recording(encoder, output, quality)

    cam.start_and_record_video(output, encoder, duration = duration, quality = quality)

main()
```

Once you have created the program, you can then schedule it to run in the operating system services or call it from the shell to run in the background as follows:

1. Create a **record.sh** text file
2. Enter the command into record.sh file ***python picamera2_continuous.py &***
3. Change the text file to executable using the command ***chmod 755 record.sh***
4. This will allow you to run the designated python program as a background process.
To run it, all you need to do is type ***./record.sh*** (remember to add the dot slash)
5. If you want to stop the process, you will have to know the process id and kill it. E.g.



```
steve@hikecan2:~$ ps
  PID TTY          TIME CMD
 8281 pts/1        00:00:00 bash
 9299 pts/1        00:00:03 python
 9306 pts/1        00:00:01 ffmpeg
 9311 pts/1        00:00:00 ps
steve@hikecan2:~$ kill 9299
steve@hikecan2:~$ ps
  PID TTY          TIME CMD
 8281 pts/1        00:00:00 bash
 9320 pts/1        00:00:00 ps
steve@hikecan2:~$
```