Hardware Coding

(Haroldas Tamosauskas)

**To run the Raspberry Pi Camera Module 3 NoIR:**

The hardware would require an update for the Raspberry Pi in order to make it work properly, best case scenario is to update the Pi fully using these commands:

|  |
| --- |
| **sudo apt update**  **sudo apt full-upgrade**  **sudo rpi-update**  **sudo reboot** |

Afterwards, it is best to disable the legacy camera inside the Interface Options, then afterwards rebooting it and using the camera to take a picture:

|  |
| --- |
| **sudo raspi-config**  **sudo reboot**  **libcamera-still -o image.jpg -t 5000** |

**(5000 represents it takes 5 seconds to take the picture)**

**To run the Mini USB Microphone:**

This hardware just requires using this code to record for 5 seconds (As long as you updated Raspberry Pi before):

|  |
| --- |
| **arecord -D plughw:1,0 -d 5 test\_audio.wav** |

* **arecord: a command line utility for linux to record audio**
* **-D: Specifies the device**
* **Plughw:1,0: the name of the device (Mini USB microphone)**
* **-d 5: will record for 5 seconds**
* **Test\_audio.wav: will output as “Test\_audio.wav”**

**To run the Mini USB Microphone to the mini–External USB Stereo speaker:**

Before writing the code for connecting the mini USB microphone to the mini-External USB stereo speaker, it is best to update the Raspberry Pi Zero to support “sounddevice”, Here is how to do it:

|  |
| --- |
| **python3 -m pip install --upgrade pip (To check if pip is up to date)**  **python3 -m pip install sounddevice** |

Create the python file (for this case, it will be referred to as “microphone\_to\_speaker\_code.py”):

|  |
| --- |
| **nano live\_camera\_feed.py**  **vim live\_camera\_feed.py** |

Write the following code:

|  |
| --- |
| **import sounddevice as sd**  **# Specifies the devices IDs**  **input\_device = 1  # Microphone**  **output\_device = 0  # Speaker**  **# Sampling rate for your device**  **SAMPLE\_RATE = 16000**  **CHANNELS = 1**  **# Size of each audio block**  **blocksize = 1024**  **# Function to process audio input and send it to the output in real-time**  **def audio\_callback(indata, outdata, frames, time, status):**  **if status:**  **print(status)**  **outdata[:] = indata  # Copy input data to output**  **print("Start speaking... Press Ctrl+C to stop.")**  **# Sets up the audio stream**  **try:**  **# Opens a stream to read from microphone and write to speaker**  **with sd.Stream(samplerate=SAMPLE\_RATE, channels=CHANNELS, dtype='int16') as stream:**  **while True:**  **# Reads and writes the audio in chunks**  **data, \_ = stream.read(SAMPLE\_RATE // 10)  # Reads 1/10th of a second of audio**  **stream.write(data)**  **except KeyboardInterrupt:**  **print("\nExiting...")**  **except Exception as e:**  **print(f"Error: {e}")** |
|  |

Be sure to press, esc, : and the wq to save the code.

To change the audio (Optional):

|  |
| --- |
| **(For this particular device, the control\_name is PCM)**  **amixer -c 0 set <control\_name> 50% -for 50% volume** |

Now run the code and it should be able to detect the audio from the microphone to the speaker (Press ctrl + c to exit)

|  |
| --- |
| **python3 microphone\_to\_speaker\_code.py** |

**To run the Raspberry Pi Camera Module 3 NoIR live on Raspberry Pi zero:**

**(code gotten from https://www.youtube.com/watch?v=U7yVpYv3gxQ)**

In order to run the live feed of the Raspberry Pi Camera module 3 on the Raspberry Pi zero, this is the code to use:

|  |
| --- |
| **libcamera-hello**  **libcamera-hello -t 0 (for a constant live feed)** |

In case that it does not detect the camera, it is best to enable legacy camera in:

|  |
| --- |
| **sudo raspi-config** |

Then later go to Interface Options > Legacy Camera, enable it and do:

|  |
| --- |
| **sudo reboot** |

If it still doesn't detect the camera, go to this config file:

|  |
| --- |
| **sudo nano /boot/config.txt** |

Be sure to add or uncomment these files:

* **camera\_auto\_detect=1**
* **dtoverlay=vc4-kms-v3d**

Also be sure to comment out these files:

* **#start\_x=1**
* **#gpu\_mem=128**

Ctrl+o and enter to save and exit, then ctrl+x to exit editor. Perform the sudo reboot and run libcamera-hello.

**To test the pir motion sensor:**

In order to test the pir motion sensor inside the Raspberry Pi zero, first thing you need to do is make a python file (for this case “motion\_test.py”):

|  |
| --- |
| **nano motion\_test.py** |

Then afterwards enter the file using vim motion\_test.py (or sudo vim motion\_test.py if there is not permission), then enter the following code:

|  |
| --- |
| **import RPi.GPIO as GPIO**  **import time**  **PIR\_PIN = 17**  **GPIO.setmode(GPIO.BCM)**  **GPIO.setup(PIR\_PIN, GPIO.IN)**  **print("Monitoring action")**  **try:**  **while True:**  **if GPIO.input(PIR\_PIN):**  **print("Motion!")**  **else:**  **print("No motion")**      **time.sleep(0.5)**  **except KeyboardInterrupt:**  **print("\nExiting program.")**  **finally:**  **GPIO.cleanup()** |

* **PIR\_PIN = 17 (The motion sensor is connected to GPIO 17 pin, could be changed to something else)**
* **if GPIO.input(PIR\_PIN): (If the motion is detected, print “Motion!”)**
* **time.sleep(0.5) (used to delay the detection)**
* **except KeyboardInterrupt: (If the user presses ctrl+c, it will exit and say “Exiting program.)**
* **GPIO.cleanup() (Resets all pins to inputs)**

Then run the code using the following command to run the code:

|  |
| --- |
| **sudo python3 motion\_test.py** |

If everything should work, it should read “Motion!” if there is motion, and “No motion” if there is no motion.