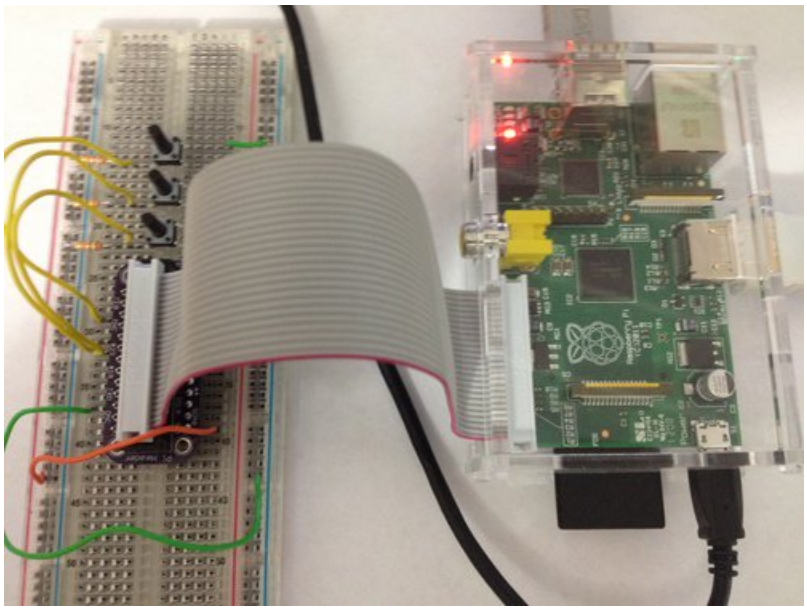
adafruit learning system

Playing sounds and using buttons with Raspberry Pi

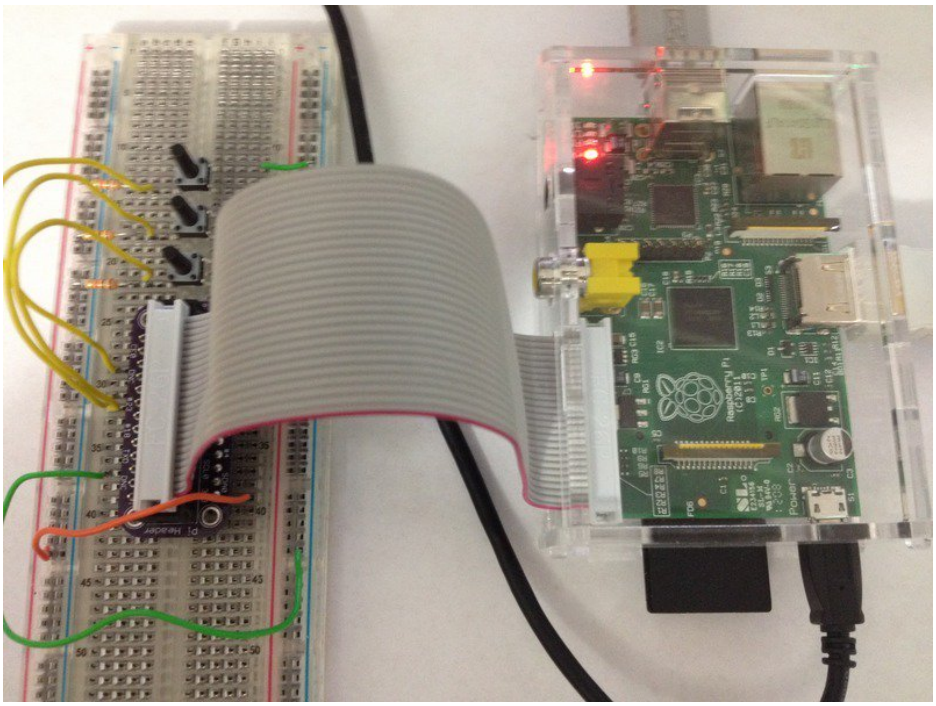
Created by Michael Sklar



Last updated on 2018-12-09 12:09:53 AM UTC

Guide Contents

Overview



This guide describes two different python based utilities to trigger audio file playback based on tactile button presses. Both of them can be installed on a modern Raspberry Pi

1. **RPi.GPIO** - works with all Raspberry Pi Hardware
2. **CircuitPython on Raspberry Pi** - Works with newer Raspberry Pi Hardware v2, v3 and Zero. This is the same interface we use with our SAMD21 based microcontrollers and is the recommended version to use.

If you have not already used the Raspberry Pi as a input device this guide will show you how to wire up the buttons to the GPIO pins and access their state from a python script.

Pi Prep

Prerequisite Pi Setup!

Your Pi will need to be running the latest version of Raspbian.

Here's the quick-start for people with some experience:

1. Download the latest Raspbian or Raspbian Lite (https://adafru.it/fQi) to your computer
2. Burn the Raspbian image to your MicroSD card (https://adafru.it/dDL) using your computer
3. Re-plug the SD card into your computer (don't use your Pi yet!) and set up your wifi connection by editing supplicant.conf (https://adafru.it/yuD)
4. Activate SSH support (https://adafru.it/yuD)
5. Plug the SD card into the Pi
6. If you have an HDMI monitor we recommend connecting it so you can see that the Pi is booting OK
7. Plug in power to the Pi - you will see the green LED flicker a little. The Pi will reboot while it sets up so wait a good 10 minutes
8. If you are running Windows on your computer, install Bonjour support so you can use **.local** names, you'll need to reboot Windows after installation (https://adafru.it/lPE)
9. You can then **ssh** into **raspberrypi.local** (https://adafru.it/jvB)

Update Your Pi and Python

Run the standard updates:

```
sudo apt-get update
```

```
sudo apt-get upgrade
```

and

```
sudo pip3 install --upgrade setuptools
```

Update all your python 3 packages with

```
pip3 freeze - local | grep -v '^\-e' | cut -d = -f 1 | xargs -n1 pip3 install -U
```

and

```
sudo bash
```

```
pip3 freeze - local | grep -v '^\-e' | cut -d = -f 1 | xargs -n1 pip3 install -U
```

Install RPi.GPIO

The RPi.GPIO python module offers easy access to the general purpose IO pins on all versions of the Raspberry Pi. The following commands will make sure your copy of Raspbian is up to date and then install a python3 compatible version of RPi.GPIO.

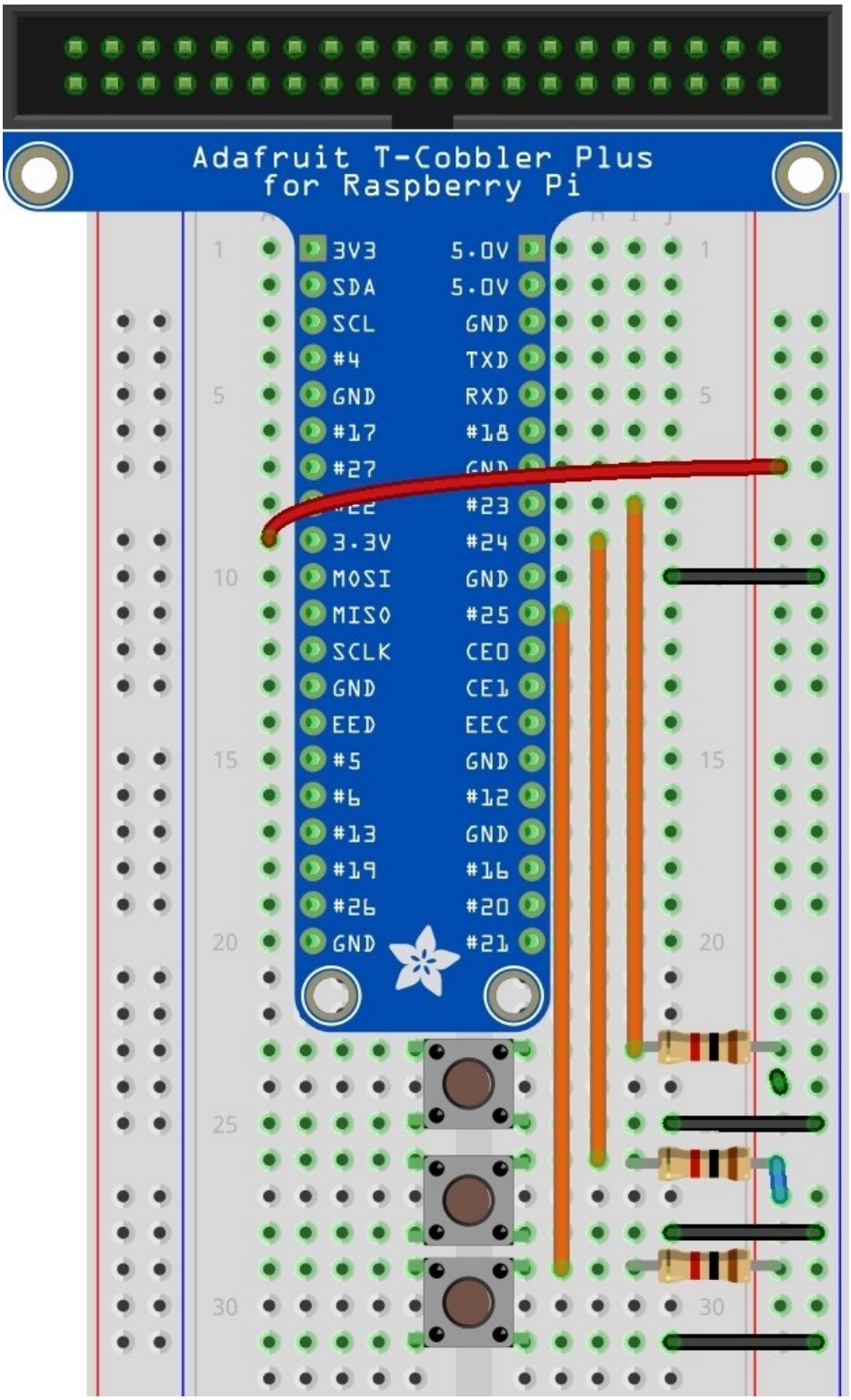
```
sudo apt-get update
sudo apt-get -y install python-dev python3-rpi.gpio
```

Circuit Diagram

There are two different button configurations described below.

1. The RPi.GPIO uses 1k resistors connected to the 3.3v rail
2. The CircuitPython code uses software pull ups.

CONFIG #1 - RPi.GPIO with 1k Pull Up Resistors

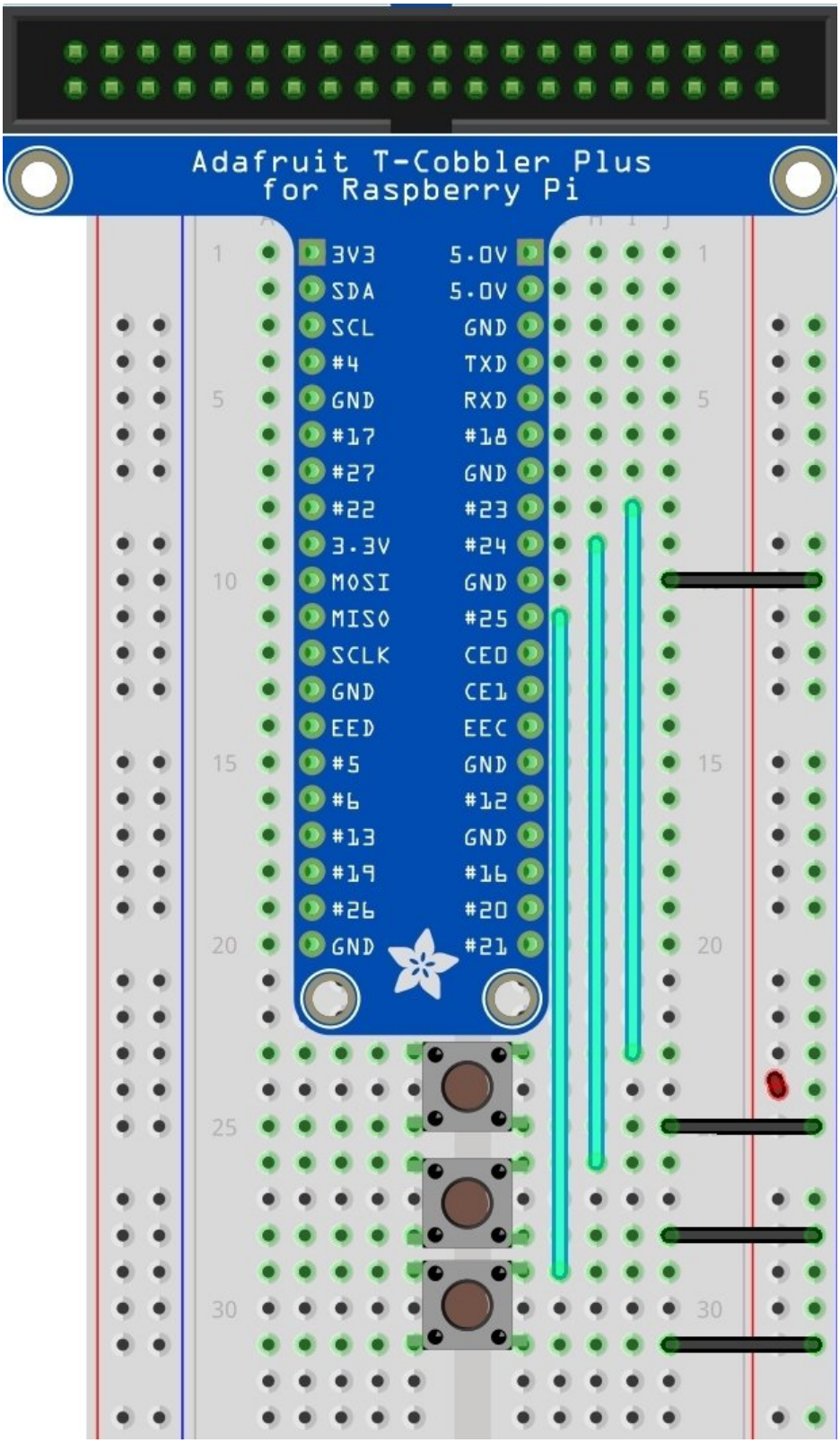


Important things to note:

- The Adafruit Pi Cobbler (http://adafru.it/914) is being used; for the A+/B+/Pi 2, the Pi Cobbler Plus (http://adafru.it/2029) or the Pi T-Cobbler Plus (https://adafru.it/xgf) should be used with any Raspberry Pi after version 1.
- The Adafruit Clear Full sized breadboard (http://adafru.it/239) is being used
- (3) 10k pull-up resistors

- (3) Momentary push-button switches (http://adafru.it/367)
- GPIO pins 23, 24, and 25
- Cobbler 3.3v rail is tied to positive breadboard
- Cobbler gnd rail is tied to negative breadboard

Config #2 CircuitPython T-Cobbler



Note that we are not using pull up resistors and do not need to connect to the 3.3v rail. We can do direct pin to button and button to GND connections with CircuitPython configuration.

RPi.GPIO Code

Now for a bit of Python. There are two example files you can download to your Pi and execute with python3. The first is an script that plays a different audio bell when when each button is pressed. The second code example is a jukebox which allows one to navigate through the files, play a selected sample and stop playback using the same three buttons.

Be sure to download the following MP3 files so that you can hear the samples playing.

1. temple-bell.mp3 (https://adafru.it/DfT)
2. temple-bell-bigger.mp3 (https://adafru.it/DfU)
3. temple-bell-huge.mp3 (https://adafru.it/DfV)

To get the MP3 audio bell files and code examples on your Pi you can right click to copy the link. Then use the '**wget**' from your Pi's command line to download each file.

```
wget https://github.com/adafruit/Adafruit_Learning_System_Guides/blob/master/Playing_Sounds_and_Using_But
```

```
import os
from time import sleep

import RPi.GPIO as GPIO

GPIO.setmode(GPIO.BCM)
GPIO.setup(23, GPIO.IN)
GPIO.setup(24, GPIO.IN)
GPIO.setup(25, GPIO.IN)

while True:
    if not GPIO.input(23):
        os.system('omxplayer temple-bell.mp3 &')

    if not GPIO.input(24):
        os.system('omxplayer temple-bell-bigger.mp3 &')

    if not GPIO.input(25):
        os.system('omxplayer temple-bell-huge.mp3 &')

    sleep(0.25)
```

To run the above code. Download the file and execute this command:

```
sudo python3 RPi.GPIO-raspi-audio-button.py
```

```
from os import listdir
import subprocess
from time import sleep

import RPi.GPIO as GPIO

GPIO.setmode(GPIO.BCM)
GPIO.setup(23, GPIO.IN)
GPIO.setup(24, GPIO.IN)
GPIO.setup(25, GPIO.IN)

mp3_files = [ f for f in listdir('.') if f[-4:] == '.mp3' ]

if not len(mp3_files) > 0:
    print("No mp3 files found!")

print('--- Available mp3 files ---')
print(mp3_files)
print('--- Press button #1 to select mp3, button #2 to play current. ---')

index = 0
while True:
    if not GPIO.input(23):
        index += 1
        if index >= len(mp3_files):
            index = 0
        print("--- " + mp3_files[index] + " ---")

    if not GPIO.input(24):
        subprocess.Popen(['omxplayer', mp3_files[index]])
        print('--- Playing ' + mp3_files[index] + ' ---')
        print('--- Press button #3 to clear playing mp3s. ---')
        sleep(1)

    if not GPIO.input(25):
        subprocess.call(['killall', 'omxplayer'])
        print('--- Cleared all existing mp3s. ---')

    sleep(0.25)
```

To run the above code. Download the file and execute this command:

```
sudo python3 RPi.GPIO-raspi-simple-jukebox.py
```

Here, we get a list of mp3 files in the current directory, and then set up the following control scheme:

- **button #1** (GPIO 23) clicks through the list of available mp3s
- **button #2** (GPIO 24) plays the currently selected mp3
- **button #3** (GPIO 25) runs a command that kills any existing mp3 player processes, stopping whatever mp3s are playing right now

CircuitPython Code

Now for a bit of Python. There are two example files you can download to your Pi and execute with python3. The first is an script that plays a different audio bell when when each button is pressed. The second code example is a jukebox which allows one to navigate through the files, play a selected sample and stop playback using the same three buttons.

Be sure to download the following MP3 files so that you can hear the samples playing.

1. temple-bell.mp3 (https://adafru.it/DfT)
2. temple-bell-bigger.mp3 (https://adafru.it/DfU)
3. temple-bell-huge.mp3 (https://adafru.it/DfV)

To get the MP3 audio bell files and code examples on your Pi you can right click to copy the link. Then use the '**wget**' from your Pi's command line to download each file.

```
wget https://github.com/adafruit/Adafruit_Learning_System_Guides/blob/master/Playing_Sounds_and_Using_But
```

```
# This script requires a Raspberry Pi 2, 3 or Zero. Circuit Python must
# be installed and it is strongly recommended that you use the latest
# release of Raspbian.

import time
import os
import board
import digitalio

print("press a button!")

button1 = digitalio.DigitalInOut(board.D23)
button1.direction = digitalio.Direction.INPUT
button1.pull = digitalio.Pull.UP

button2 = digitalio.DigitalInOut(board.D24)
button2.direction = digitalio.Direction.INPUT
button2.pull = digitalio.Pull.UP

button3 = digitalio.DigitalInOut(board.D25)
button3.direction = digitalio.Direction.INPUT
button3.pull = digitalio.Pull.UP

while True:

    # omxplayer -o local <file>
    # omxplayer -o hdmi <file>
    # omxplayer -o both <file>
    if not button1.value:
        os.system('omxplayer temple-bell.mp3 &')

    if not button2.value:
        os.system('omxplayer temple-bell-bigger.mp3 &')

    if not button3.value:
        os.system('omxplayer temple-bell-huge.mp3 &')

    time.sleep(.25)
```

To run the above code. Download the file and execute this command:

```
sudo python3 CircuitPython-raspi-audio-button.py
```

```
# This script requires a Raspberry Pi 2, 3 or Zero. Circuit Python must
# be installed and it is strongly recommended that you use the latest
# release of Raspbian.

import time
from os import listdir
import subprocess
import board
import digitalio

button1 = digitalio.DigitalInOut(board.D23)
button1.direction = digitalio.Direction.INPUT
button1.pull = digitalio.Pull.UP

button2 = digitalio.DigitalInOut(board.D24)
button2.direction = digitalio.Direction.INPUT
button2.pull = digitalio.Pull.UP

button3 = digitalio.DigitalInOut(board.D25)
button3.direction = digitalio.Direction.INPUT
button3.pull = digitalio.Pull.UP

mp3_files = [ f for f in listdir('.') if f[-4:] == '.mp3' ]

if not len(mp3_files) > 0:
    print("No mp3 files found!")

print('--- Available mp3 files ---')
print(mp3_files)
print('--- Press button #1 to select mp3, button #2 to play current. ---')

index = 0
while True:
    if not button1.value:
        index += 1
        if index >= len(mp3_files):
            index = 0
        print("--- " + mp3_files[index] + " ---")

    if not button2.value:
        subprocess.Popen(['omxplayer', mp3_files[index]])
        print('--- Playing ' + mp3_files[index] + ' ---')
        print('--- Press button #3 to clear playing mp3s. ---')
        time.sleep(0.25)

    if not button3.value:
        subprocess.call(['killall', 'omxplayer'])
        print('--- Cleared all existing mp3s. ---')

    time.sleep(0.25)
```

To run the above code. Download the file and execute this command:

```
sudo python3 CircuitPython-raspi-simple-jukebox.py
```

Here, we get a list of mp3 files in the current directory, and then set up the following control scheme:

- **button #1** (GPIO 23) clicks through the list of available mp3s
- **button #2** (GPIO 24) plays the currently selected mp3
- **button #3** (GPIO 25) runs a command that kills any existing mp3 player processes, stopping whatever mp3s are playing right now