# EET 2812

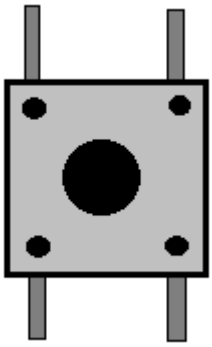
# Activity 8

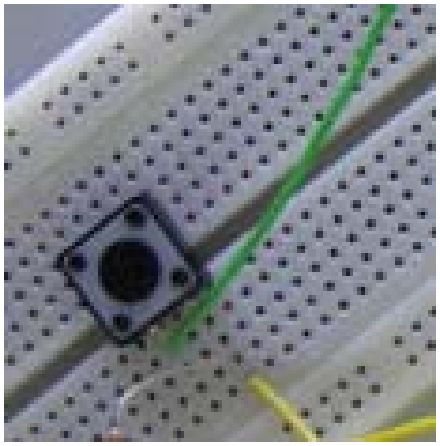
# Single Board Computer (RPi)

# S

Cuyahoga Community College

Youth Technology Academy





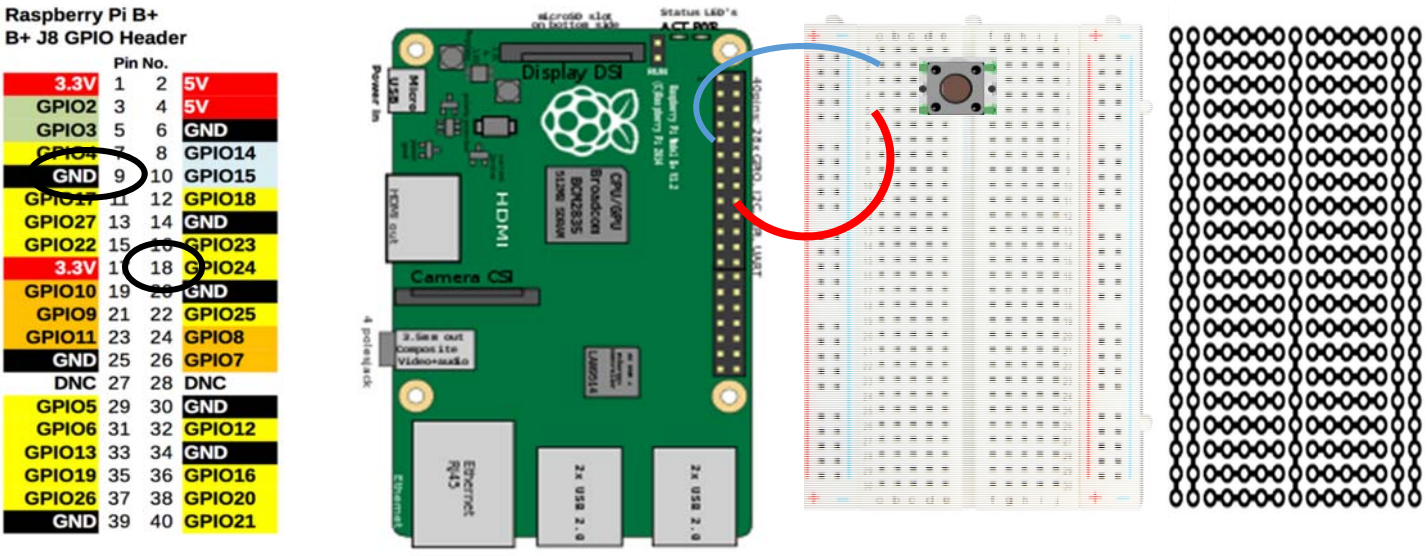
# ACTIVITY

# RPi Sensors Programming – Switch

**Overview:** Build a circuit with the RPi and an LED. Program the RPi to display a message when a button is pressed.

**Vocabulary:** RPi, circuit, LED, function library, **resistor,** pull-up resistor, pull-down resistor

**When finished with the program go to https://www.codecademy.com/learn/python and complete Lessons 1-4**



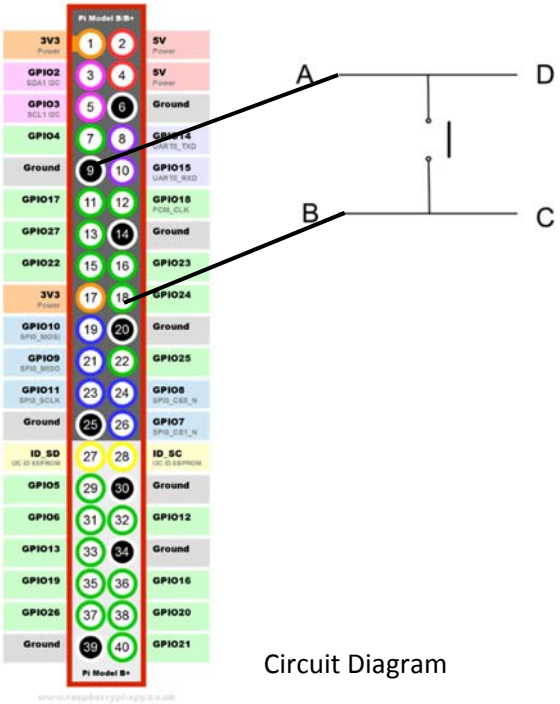

Create the circuit above using the RPi, breadboard, button and 2 wires connected to pins GND9 and 18.

Start the RPi and change your directory to my_python:

**$cd my_python**

Open nano and create a program called switch.py:

**$nano switch.py**




Circuit Diagram

Copy this code into nano to display a message when the button is pressed:
**Everything written after the # will not be read by the program – these are explanatory comments.**

```
import RPi.GPIO as GPIO                    #import the GPIO library
import time                                #import the time library
button=18
GPIO.setmode(GPIO.BOARD)                   #set the program to read the physical pin numbers (1-40)

GPIO.setup(button, GPIO.IN, pull_up_down =GPIO.PUD_UP) #make button input & activate Pull Up Resistor

while True:
        input_state = GPIO.input(button)   #look for button press
        if input_state == False:
            print('Button Pressed')
            time.sleep(0.2)                #delay
GPIO.cleanup()                             #reset pins for next program
```

**Save the program** pressing CTRL O and ENTER. Then exit nano by pressing CTRL X.

Run the program as superuser: **$sudo python switch.py** **(To exit the program press CTRL Z)**

## Discussion

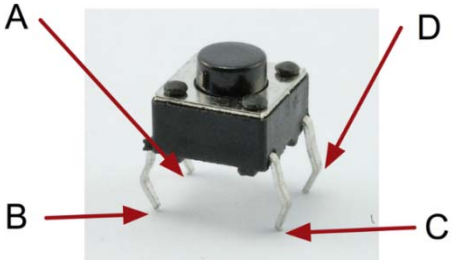
You will notice that the switch is wired so that when it is pressed, it will connect pin 18 configured as an input to GND.

Each GPIO pin has software configurable pull-up and pull-down resistors. When using a GPIO pin as an input, you can configure these resistors so that one or either or neither of the resistors is enabled, using the optional `pull_up_down` parameter to `GPIO.setup`. If this parameter is omitted, then neither resistor will be enabled. This leaves the input *floating*, which means that its value cannot be relied upon and it will drift between high and low depending on what it picks up in the way of electrical noise.
If it is set to `GPIO.PUD_UP`, the pull-up resistor is enabled; if it is set to `GPIO.PUD_DOWN`, the pull-down resistor is enabled.

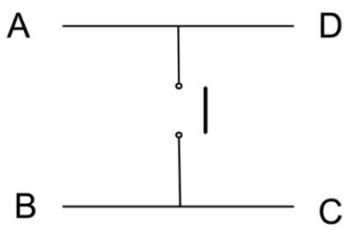
You might expect the push switch to have just two connections, which are either open or closed. While some of these tactile push switches do have just two connections, most have four.

The circuit diagram below shows how these connections are arranged in the button we are using.

Tactile Push Switch




Push Switch Circuit Diagram




(From http://razzpisampler.oreilly.com/ch07.html)

**When finished go to https://www.codecademy.com/learn/python and complete Lessons 1-4**