Lab 2: Using GPIO as input/output

Demo Due: **By the End of lab section on Sept 22nd(50%)**

Report Due: 10:00a.m. Sept 18th(50%)

Learning Objectives: In this lab, you are going to learn how to involve user's control of some actuation, read some digital data and output digital data for actuation through RPi's GPIO interface. Input data can be manipulated inside Pi before writing the outputs.

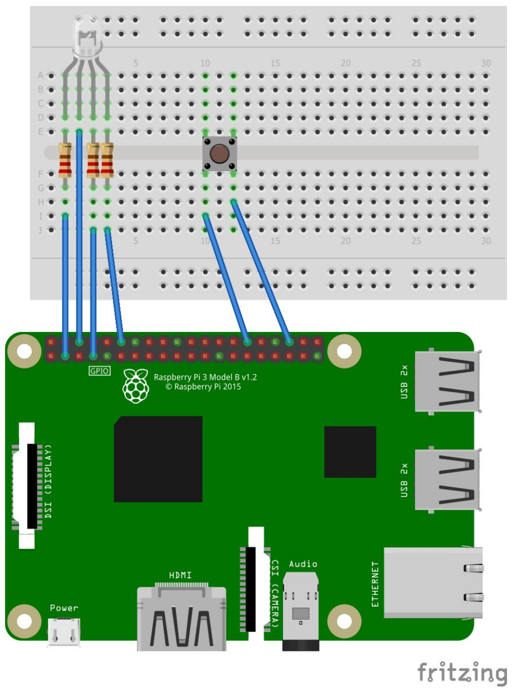
Components Needed:

Pi 3 Board, RGB LEDs, Press button switch for user control, 8-bit DIP switches, LED Matrix Display, resistors and appropriate wires.

Lab Activities:

1. Interface 3-input RGB LED as output device and 4-pin single press button switch to control LED activation. The switch and LED are to be interfaced through GPIO like the last lab. The switch will wait for user inputs to make the LED glow.
   Refer: http://www.elinux.org/RPi_GPIO_Interface_Circuits to make sure you know how to interface switch/resistor/diode etc.

Circuit Design: (A tentative schematic is shown below. You will determine appropriate pins, resistors, power supplies for your design.)



In lab activity: During the lab week Sept 11th, write an assembly language program to enable the LED to blink (1 sec on/off) with user input. Display of particular color light in LED when user presses the button once. The cycle of blinking colors repeats.

Report: *Give the final working circuit design as part of the report and the assembly language program that was used for blinking.*

2. Now consider 8-bit DIP switch as the input source and LED matrix as the output source interfaced through GPIO. Set a combination of bits (0, 1) by toggling switches in 8-bit DIP switch as the data input. For this you will connect the input pins to either VCC(1 or high) or GND(0 or low) through appropriate resistors. The Pi 3 shall be programmed in either Python to read an input value (between 00-FF (hex)) and display that number on the LED matrix. Make sure your program will output logical 1 or 0 to appropriate GPIO pins of the LED matrix. Note that the Pi and LED are to be protected using resistors. **Demonstration:** Write a python program, that take input from the DIP switch, and display the correct character on the LED matrix.

Report: *Draw the schematic details of your working circuit and Python program that you have implemented.*

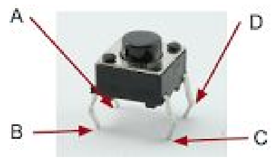
Questions to Explore:
In your report, write down the discussion of the following questions:

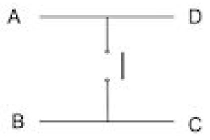
1. Discuss what is the advantage/disadvantage using assembly over C/Python.
2. In lab 2's activities 1 & 2, we have used a loop to keep checking on the signal from the input source(s). Is there an alternative way to read input from external button or switch than simply pulling the signal?
3. We used GPIO pins to read input from the DIP switch. If you replace the input switch by a temperature sensor(model:**TMP36**), can you read temperature value from the sensor same way as you did from the switch? If yes, explain. If no, provide a solution to correctly reading data from a temperature sensor TMP36.

Helpful Hints:

1. Press Button

   A press button is a very common component we used to control electronic devices. We usually use them as a switch to connect/disconnect circuits. First of all, you will need to understand the four pins on the press button:

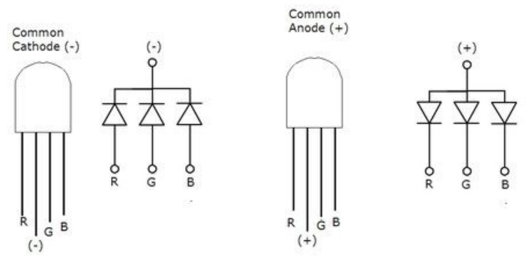




   When you're pressing the button, AD will be connected BC, which will connect the circuit. **To protect your raspberry Pi, a resistor is suggested to be placed in the circuit.** Also, the input pin on the Pi will be floating when the button is not pressed (because there are no complete circuits attached to that pin). To deal with the pin's voltage floating, it may be beneficial to put a pull-down resistor attached to the input pin if the other side of the button is attached to positive voltage. If the button is attached to ground, then a pull-up resistor should be used. **Make sure the resistance is high enough that the input pin can go to the correct voltage when the button is pressed**

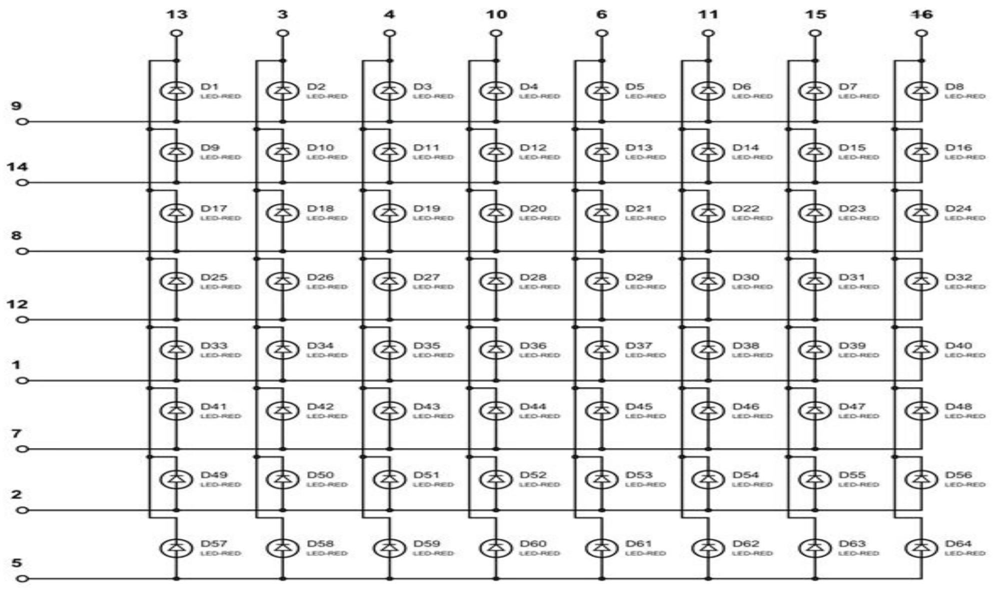
(preferably significantly higher than the resistor you placed in series with the switch).

2. RGB LED: A RGB(Red, Blue, Green) LED produces three different colors of light when you connect the power source to the different pins. We provide a RGB led have the same hardware schema as the image below. We have a common Cathode(-), which means, the second node should connect to the ground while the rest of them connect to the output pins on the Pi. A positive voltage on the output pin will cause the corresponding LED to light up.



Always Remember to connect a resistor to each node! The voltage for each pin is around 2-3.2 V. The red pin will get burned immediately if you connect it directly into a pin. The blue and green pins can both stay on for several second before they will generate lots of heat and get burned (if connected directly to the pins).

3. LED Matrix
Suggest reading:
https://circuitdigest.com/microcontroller-projects/control-8x8-led-matrix-with-raspberry-pi
An 8 by 8 LED matrix is a group of 64 LEDs in the form of matrix. There are 8 positive pins and 8 negative pins. A design schema for the matrix looks like this:



However the pin on the real Matrix is very confusing. Please read the suggest reading to figure out the correct pin order. In order to led D1, we need to provide positive voltage on pin 9 and ground pin 13. The max voltage that each LED can tolerant is no more than 3.3 V. **Always connect a Resistor to protect your device and LED Matrix.**