Lab 1: Sensors

Noah Buchanan
Derek Yocum
CS 4043: Formal Languages
University of Arkansas - Fort Smith
Fall 2021

September 25, 2021

1. The numbers and letters simply help you identify individual holes in the breadboard.
2. The plus sign indicates voltage and the minus sign indicates ground.
3. The rows are connected horizontally but not across the bridge in the middle and they are connected vertically on the plus and minus sign strips or the voltage and ground strips.
4. Jumper wires are simply to connect holes on the breadboard.

Lab1ButtonLed.py source code

```
#!/usr/bin/env python3
import RPi.GPIO as GPIO
import time



BtnPin = 11
Gpin   = 13
Rpin   = 12

def setup():
        print(”setup method called”)
        GPIO.setmode(GPIO.BOARD)        # Numbers GPIOs by physical location
        GPIO.setup(Gpin, GPIO.OUT)      # Set Green Led Pin mode to output
        GPIO.setup(Rpin, GPIO.OUT)      # Set Red Led Pin mode to output
        GPIO.setup(BtnPin, GPIO.IN, pull_up_down=GPIO.PUD_UP)
# Set BtnPin's mode is input, and pull up to high level(3.3V)
```

```
        GPIO.add_event_detect(BtnPin, GPIO.BOTH, callback=detect, bouncetime=200

def Led(x):
        print("button pressed")
        print(x)
        if(x==0):
                GPIO.output(Rpin,0)
                GPIO.output(Gpin,0)
        while True:
                if not GPIO.input(BtnPin):
                        GPIO.output(Rpin, 1)
                        time.sleep(3)
                        GPIO.output(Rpin, 0)
                        GPIO.output(Gpin,0)
                        time.sleep(1)
                else:
                        break
def detect(chn):
        print("detect method called")
        Led(GPIO.input(BtnPin))

def loop():
        print("loop method called")
        while True:
                pass

def destroy():
        print("destroy method called")
        GPIO.output(Gpin, GPIO.HIGH)       # Green led off
        GPIO.output(Rpin, GPIO.HIGH)       # Red led off
        GPIO.cleanup()                     # Release resource

if __name__ == '__main__':     # Program start from here
        setup()
        try:
                loop()
        except KeyboardInterrupt:  # When 'Ctrl+C' is pressed, the child program
executed.
                destroy()
```

Lab1Led.py source code

```
#!/usr/bin/env python3
import RPi.GPIO as GPIO
import time

BtnPin = 11
Gpin   = 13
Rpin   = 12

def setup():
        print(”setup method called”)
        GPIO.setmode(GPIO.BOARD)        # Numbers GPIOs by physical location
        GPIO.setup(Gpin, GPIO.OUT)      # Set Green Led Pin mode to output
        GPIO.setup(Rpin, GPIO.OUT)      # Set Red Led Pin mode to output
        GPIO.setup(BtnPin, GPIO.IN, pull_up_down=GPIO.PUD_UP)
# Set BtnPin's mode is input, and pull up to high level(3.3V)

def loop():
        print(”loop method called”)
        while True:
                print(”printing loop”)
                GPIO.output(Rpin,1)
                time.sleep(3)
                GPIO.output(Gpin,0)
                GPIO.output(Rpin,0)
                time.sleep(1)

def destroy():
        print(”destroy method called”)
        GPIO.output(Gpin, GPIO.HIGH)        # Green led off
        GPIO.output(Rpin, GPIO.HIGH)        # Red led off
        GPIO.cleanup()                      # Release resource

if __name__ == '__main__':      # Program start from here
        setup()
        try:
                loop()
        except KeyboardInterrupt:  # When 'Ctrl+C' is pressed, the child program
executed.
                destroy()
```