Issa Odeh
CPE 406
Assignment # 2
02/28/2021

Assignment Description:

In this assignment, we used the term “Breathing LED” which means that an LED that is off will then turn on gradually and turn off gradually, like how we breath. To control the brightness of an LED to create a breathing light we use PWM. Pulse-Width Modulation (PWM) is a method for using digital signals to control analog circuits. This method is highly effective. PWM uses digital pins to send certain freq. of square waves which outputs either high or low. The total amount of time of each set of high levels and low levels is fixed which is called the period. When the output is high, it is called “pulse width” and the duty cycle can be identified as the percentage of the ratio of pulse duration. The longer the output of high levels last, the longer the duty cycle. The longer the PWM duty cycle is, the higher the output power. In this project we will implement a circuit using PWM “Breathing LED light”.

Problems Encountered:

Only problem I had was when I was playing with the speeds, I could not figure out which number is which but after reading the tutorial pdf and its description it helped and I was able to successfully about to make the time change in my circuit.

Lessons Learned:

In this lab I learned about PWM (Pulse-Width Modulation) and how to implement it in python code. It was good to play around with the frequencies and timing to see what the LED would do.

Description of Completed Lab:

```
import RPi.GPIO as GPIO
import time

LedPin = 12     # define the LedPin

def setup():
    global p
    GPIO.setmode(GPIO.BOARD)        # use PHYSICAL GPIO Numbering
    GPIO.setup(LedPin, GPIO.OUT)    # set LedPin to OUTPUT mode
    GPIO.output(LedPin, GPIO.LOW)   # make ledPin output LOW level to
turn off LED

    p = GPIO.PWM(LedPin, 500)       # set PWM Frequence to 500Hz
    p.start(0)                      # set initial Duty Cycle to 0

def loop():
    while True:
        for dc in range(0, 101, 1):   # make the led brighter
            p.ChangeDutyCycle(dc)     # set dc value as the duty cycle
            time.sleep(0.01)
        time.sleep(1)
        for dc in range(100, -1, -1): # make the led darker
            p.ChangeDutyCycle(dc)     # set dc value as the duty cycle
            time.sleep(0.01)
        time.sleep(1)

def destroy():
    p.stop() # stop PWM
    GPIO.cleanup() # Release all GPIO

if __name__ == '__main__':     # Program entrance
    print ('Program is starting ... ')
    setup()
    try:
        loop()
    except KeyboardInterrupt:  # Press ctrl-c to end the program.
        destroy()
```

in the above circuit, the LED is connected to the GPIO 18 port. In the code the LedPin is set to be at pin 12 and is expected to output mode according to the corresponding chart for pin designations. Then a PWM was created and set frequency to 1000HZ. The two for loops are used to control the LED and can control the speed. The first loop outputs a power signal to the ledPin PWM from 0 – 100% and the second loop outputs a power signal to the ledPin from 100 – 0%.

This link will take you to the video for the code above:

```
import RPi.GPIO as GPIO
import time

LedPin = 12     # define the LedPin

def setup():
    global p
    GPIO.setmode(GPIO.BOARD)       # use PHYSICAL GPIO Numbering
    GPIO.setup(LedPin, GPIO.OUT)   # set LedPin to OUTPUT mode
    GPIO.output(LedPin, GPIO.LOW)  # make ledPin output LOW level to
turn off LED

    p = GPIO.PWM(LedPin, 500)      # set PWM Frequence to 500Hz
    p.start(0)                     # set initial Duty Cycle to 0

def loop():
    while True:
        for dc in range(0, 101, 5):   # make the led brighter
            p.ChangeDutyCycle(dc)     # set dc value as the duty cycle
            time.sleep(0.01)
        time.sleep(1)
        for dc in range(100, -1, -5): # make the led darker
            p.ChangeDutyCycle(dc)     # set dc value as the duty cycle
            time.sleep(0.01)
        time.sleep(1)

def destroy():
    p.stop() # stop PWM
    GPIO.cleanup() # Release all GPIO

if __name__ == '__main__':     # Program entrance
    print ('Program is starting ... ')
    setup()
    try:
        loop()
    except KeyboardInterrupt:  # Press ctrl-c to end the program.
        destroy()
```

In the above code, it is similar to the first one that is posted above. The only difference is I changed the speed of the LED. I changed the two for loop “for dc in range (0,101,1) changed to (0,101,5) and the other for loop from (100, -1, -1) to (100, -1, -5)”. In the first video it is clear that the LED turns on slower compared to the second video which is located below the text.

This link will take you to the video for the code above: