Alejandra Sandoval
Jonathan Westerfield

Laboratory #2: USING GPIO AS INPUT AND OUTPUT

Objective: 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 Raspberry Pi's GPIO interface.

Questions to explore:

1. **What is RPI 3 CPU clock Rate?**

   1.4 GHz 64/32-bit quad-core ARM Cortex-A53

2. **In this lab, we have used a loop to continuously check the signal from the input source(s). Is there a way to read input from an external button or switch other than simply polling the signal?**

   We can use a hardware interrupt and call the polling routine.

3. **We used GPIO pins to read input from the Button switches. If you replace the input switch by an analog-out 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.**

   No, it would be different because it would need to know the exact number of the temperature not just a high input. You can probably read all the temperature values and then compare them and find out the values from there.

CODE

```
import RPi.GPIO as GPIO
import LEDLibrary
from SevSeg import SevSeg
from time import sleep

# Pin 20 is the button pin

LEDLibrary.setupLEDs()
Segment = SevSeg()
GPIO.setwarnings(False)
GPIO.setmode(GPIO.BCM)
GPIO.setup(20, GPIO.IN, pull_up_down = GPIO.PUD_UP)

while True:
    LEDLibrary.LED1Green()
    LEDLibrary.LED2Red()

    inputState = GPIO.input(20)
    if(inputState == False):
        print("Button Has Been Pressed!!!!")
```

```
        LEDLibrary.LED1Blue()
        sleep(0.5)
        LEDLibrary.LED1Off()
        LEDLibrary.LED2Red()
        sleep(0.5)
        LEDLibrary.LED1Blue()
        LEDLibrary.LED2Red()
        sleep(0.5)
        LEDLibrary.LED1Off()
        LEDLibrary.LED2Red()
        sleep(0.5)
        LEDLibrary.LED1Blue()
        LEDLibrary.LED2Red()
        sleep(0.5)
        LEDLibrary.LED1Off()
        LEDLibrary.LED2Red()
        sleep(1)
        LEDLibrary.LED2Green()
        LEDLibrary.LED1Red()
        Segment.seg9()
        sleep(1)
        Segment.seg8()
        sleep(1)
        Segment.seg7()
        sleep(1)
        Segment.seg6()
        sleep(1)
        Segment.seg5()
        sleep(1)
        Segment.seg4()
        LEDLibrary.LED2Blue()
        LEDLibrary.LED1Red()
        sleep(1)
        Segment.seg3()
        sleep(1)
        Segment.seg2()
        sleep(1)
        Segment.seg1()
        sleep(1)
        LEDLibrary.LED1Green()
        LEDLibrary.LED2Red()
        sleep(4)
        LEDLibrary.LED1Off()
        LEDLibrary.LED2Off()
        sleep(15)
```

----- HELPER LIBRARY TO TURN ON 7 LED SEGMENT

```
import RPi.GPIO as GPIO

"""This file is used to control a seven segment display"""
```

Alejandra Sandoval
Jonathan Westerfield

```
class SevSeg:

	def __init__(self):

	# The pins for each bar on the seven segment display
		self.gPinA = 13
		self.gPinB = 19
		self.gPinC = 18
		self.gPinD = 23
		self.gPinE = 24
		self.gPinF = 25
		self.gPinG = 12
		self.gPinH = 16

		# Setup which pins are which
		GPIO.setwarnings(False)
		GPIO.setmode(GPIO.BCM)

		GPIO.setup(self.gPinA, GPIO.OUT, initial=GPIO.LOW)
		GPIO.setup(self.gPinB, GPIO.OUT, initial=GPIO.LOW)
		GPIO.setup(self.gPinC, GPIO.OUT, initial=GPIO.LOW)
		GPIO.setup(self.gPinD, GPIO.OUT, initial=GPIO.LOW)
		GPIO.setup(self.gPinE, GPIO.OUT, initial=GPIO.LOW)
		GPIO.setup(self.gPinF, GPIO.OUT, initial=GPIO.LOW)
		GPIO.setup(self.gPinG, GPIO.OUT, initial=GPIO.LOW)
		GPIO.setup(self.gPinH, GPIO.OUT, initial=GPIO.LOW)

	def pinA(self, turnOn=0):
		if turnOn:
			GPIO.output(self.gPinA, GPIO.HIGH)
		else:
			GPIO.output(self.gPinA, GPIO.LOW)

	def pinB(self, turnOn=0):
		if turnOn:
			GPIO.output(self.gPinB, GPIO.HIGH)
		else:
			GPIO.output(self.gPinB, GPIO.LOW)

	def pinC(self, turnOn=0):
		if turnOn:
			GPIO.output(self.gPinC, GPIO.HIGH)
		else:
			GPIO.output(self.gPinC, GPIO.LOW)

	def pinD(self, turnOn=0):
		if turnOn:
			GPIO.output(self.gPinD, GPIO.HIGH)
		else:
			GPIO.output(self.gPinD, GPIO.LOW)

	def pinE(self, turnOn=0):
		if turnOn:
```

```
                GPIO.output(self.gPinE, GPIO.HIGH)
        else:
                GPIO.output(self.gPinE, GPIO.LOW)

    def pinF(self, turnOn=0):
        if turnOn:
                GPIO.output(self.gPinF, GPIO.HIGH)
        else:
                GPIO.output(self.gPinF, GPIO.LOW)

    def pinG(self, turnOn=0):
        if turnOn:
                GPIO.output(self.gPinG, GPIO.HIGH)
        else:
                GPIO.output(self.gPinG, GPIO.LOW)

    def pinH(self, turnOn=0):
        if turnOn:
                GPIO.output(self.gPinH, GPIO.HIGH)
        else:
                GPIO.output(self.gPinH, GPIO.LOW)

    # Make the number 0
    def seg0(self):
        # 0 is: A, B, C, D, E, F
        self.pinA(1)
        self.pinB(1)
        self.pinC(1)
        self.pinD(1)
        self.pinE(1)
        self.pinF(1)
        self.pinG()
        self.pinH()

    def seg1(self):
        # 1 is: B, C
        self.pinA()
        self.pinB(1)
        self.pinC(1)
        self.pinD()
        self.pinE()
        self.pinF()
        self.pinG()
        self.pinH()

    def seg2(self):
        #2 is: A, B, D, E, G, H
        self.pinA(1)
        self.pinB(1)
        self.pinC()
        self.pinD(1)
        self.pinE(1)
        self.pinF()
        self.pinG(1)
```

```
            self.pinH(1)

        def seg3(self):
            #3 is: A, B, C, D, G, H
            self.pinA(1)
            self.pinB(1)
            self.pinC(1)
            self.pinD(1)
            self.pinE()
            self.pinF()
            self.pinG(1)
            self.pinH(1)

        def seg4(self):
            # 4 is: B, C, F, G, H
            self.pinA()
            self.pinB(1)
            self.pinC(1)
            self.pinD()
            self.pinE()
            self.pinF(1)
            self.pinG(1)
            self.pinH(1)

        def seg5(self):
            # 5 is: A, C, D, F, G, H
            self.pinA(1)
            self.pinB()
            self.pinC(1)
            self.pinD(1)
            self.pinE()
            self.pinF(1)
            self.pinG(1)
            self.pinH(1)

        def seg6(self):
            # 6 is: A, C, D, E, F, G, H
            self.pinA(1)
            self.pinB()
            self.pinC(1)
            self.pinD(1)
            self.pinE(1)
            self.pinF(1)
            self.pinG(1)
            self.pinH(1)

        def seg7(self):
            # 7 is: A, B, C
            self.pinA(1)
            self.pinB(1)
            self.pinC(1)
            self.pinD()
            self.pinE()
            self.pinF()
```

```
                self.pinG()
                self.pinH()

        def seg8(self):
                # 8 is: All of them
                self.pinA(1)
                self.pinB(1)
                self.pinC(1)
                self.pinD(1)
                self.pinE(1)
                self.pinF(1)
                self.pinG(1)
                self.pinH(1)

        def seg9(self):
                # 9 is: A, B, C, F, G, H
                self.pinA(1)
                self.pinB(1)
                self.pinC(1)
                self.pinD()
                self.pinE()
                self.pinF(1)
                self.pinG(1)
                self.pinH(1)
```

HELPER LIBRARY FOR LED LIGHTS ----------

```
import RPi.GPIO as GPIO

"""
LED 1 (NOT CONNECTED TO THE SEVEN SEGMENT DISPLAY):
        #4 - red
        #17 - green
        #27 - blue

LED 2 (CONNECTED TO THE SEVEN SEGMENT DISPLAY)
        #22 - red
        #5 - green
        #6 - blue
"""

def setupLEDs():
        GPIO.setwarnings(False)
        GPIO.setmode(GPIO.BCM)

        #LED 1
        GPIO.setup(4, GPIO.OUT, initial=GPIO.LOW)
        GPIO.setup(17, GPIO.OUT, initial=GPIO.LOW)
        GPIO.setup(27, GPIO.OUT, initial=GPIO.LOW)

        # LED 2
        GPIO.setup(22, GPIO.OUT, initial=GPIO.LOW)
```

```
        GPIO.setup(5, GPIO.OUT, initial=GPIO.LOW)
        GPIO.setup(6, GPIO.OUT, initial=GPIO.LOW)

# Make LED 1 red
def LED1Red():
        GPIO.output(4, GPIO.HIGH)
        GPIO.output(17, GPIO.LOW)
        GPIO.output(27, GPIO.LOW)

# Make LED 1 blue
def LED1Blue():
        GPIO.output(4, GPIO.LOW)
        GPIO.output(17, GPIO.LOW)
        GPIO.output(27, GPIO.HIGH)

# Make LED 1 green
def LED1Green():
        GPIO.output(4, GPIO.LOW)
        GPIO.output(17, GPIO.HIGH)
        GPIO.output(27, GPIO.LOW)

# Turn off LED 1
def LED1Off():
        GPIO.output(4, GPIO.LOW)
        GPIO.output(17, GPIO.LOW)
        GPIO.output(27, GPIO.LOW)

# Make LED 2 red
def LED2Red():
        GPIO.output(22, GPIO.HIGH)
        GPIO.output(5, GPIO.LOW)
        GPIO.output(6, GPIO.LOW)

# Make LED 2 blue
def LED2Blue():
        GPIO.output(22, GPIO.LOW)
        GPIO.output(5, GPIO.LOW)
        GPIO.output(6, GPIO.HIGH)

# Make LED 2 green
def LED2Green():
        GPIO.output(22, GPIO.LOW)
        GPIO.output(5, GPIO.HIGH)
        GPIO.output(6, GPIO.LOW)

# Turn off LED 2
def LED2Off():
        GPIO.output(22, GPIO.LOW)
        GPIO.output(5, GPIO.LOW)
        GPIO.output(6, GPIO.LOW)
```