

Solutions to Assignment 4 by Aether Zhou

This file is

<https://docs.google.com/document/d/1HrPdf9-wTB6-ptsPKQa7UNZuBqBIAEtdnr3rNexXug4/edit?usp=sharing>

Python filename is assignment_four_aether_zhou.py

1) Write a program that measures both analog in values and prints the results in a human readable way.

```
1 def getAnalogVolt():
2     print('AIN0 = '+str(d.getAIN(0))+ ' volts, AIN0 = '+str(d.getAIN(1))+ ' volts')
```

```
1 getAnalogVolt()
```

```
AIN0 = 0.08963199999999816 volts, AIN0 = 0.08963199999999816 volts
```

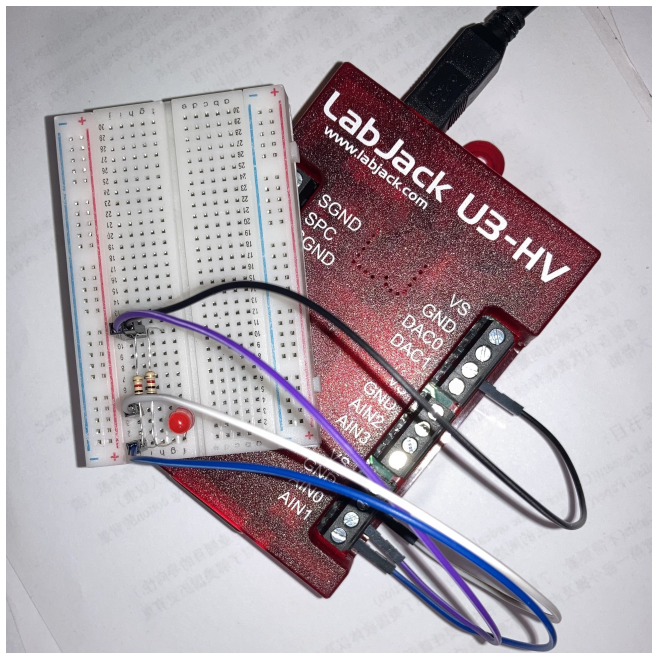
2) Write a program setDAC0(voltage) that sets the DAC0 voltage, and responds gracefully if you send it an illegal voltage (say, greater than 5 volts, or less than 0).

```
1 def setDAC0(voltage):
2     if voltage > 5 or voltage < 0:
3         print('voltage out of bounds (0 < voltage < 5)')
4     DAC0_VALUE = d.voltageToDACBits(voltage, dacNumber = 0, is16Bits = False)
5     d.getFeedback(u3.DAC0_8(DAC0_VALUE))
```

```
1 setDAC0(-1)
```

```
voltage out of bounds (0 < voltage < 5)
```

3) Make the following circuit, using your kits and the Labjack



4) use setDAC0(voltage) to turn the LED on and off manually.

```
1 def setDAC0(voltage):
2     if voltage > 5 or voltage < 0:
3         print('voltage out of bounds (0 < voltage < 5)')
4     DAC0_VALUE = d.voltageToDACBits(voltage, dacNumber = 0, is16Bits = False)
5     d.getFeedback(u3.DAC0_8(DAC0_VALUE))
```

```
1 setDAC0(-1)
```

voltage out of bounds (0 < voltage < 5)

5) Write a program called blink(n) that blinks the LED n times.

```
1 def blink(n=3):
2     while n > 0:
3         n-=1
4         setDAC0(2)
5         sleep(1)
6         setDAC0(0)
7         sleep(1)
```

```
1 blink()
```

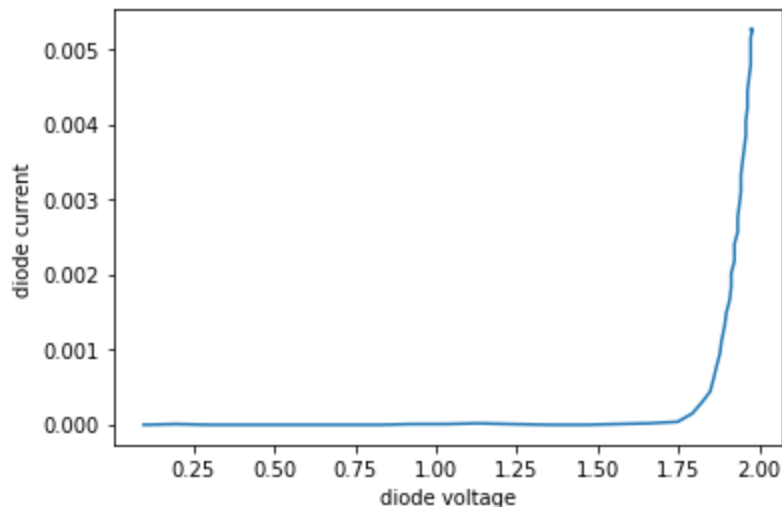
6) extend helloRobot.py to include blinker.

```
27     def blink(self, n = 3):
28         while n > 0:
29             n-=1
30             self.setDAC0(2)
31             sleep(1)
32             self.setDAC0(0)
33             sleep(1)
```

7) Now write takeIVCurve() which uses your circuit to measure the I-V curve of the diode.

```
35 def takeIVCurve(self, maxVoltage = 5, deltaVoltage = 0.1, verbose = 0):
36     setVoltages = np.arange(0,maxVoltage,deltaVoltage)
37     currents = np.zeros(len(setVoltages))
38     diodeVoltages = np.zeros(len(setVoltages))
39     for i in range(len(setVoltages)):
40         thisVoltage = setVoltages[i]
41         self.setDAC0(thisVoltage)
42         sleep(0.05)
43         topVoltage = self.labjack.getAIN(0)
44         midVoltage = self.labjack.getAIN(1)
45         diodeVoltages[i] = midVoltage;
46         currents[i] = (topVoltage - midVoltage)/self.__resistance
47
48     plt.figure('Diode I-V curve')
49     plt.plot(diodeVoltages,currents)
50     plt.xlabel('diode voltage')
51     plt.ylabel('diode current')
52     plt.show()
53     if verbose:
54
55         plt.figure('Input voltages')
56         plt.plot(setVoltages,diodeVoltages)
57         plt.xlabel('DAC setpoint')
58         plt.ylabel('measured output voltage')
59         plt.show()
```

```
1 myRobot.takeIVCurve(verbose=0)
```



8) Now use the tools you have so far to make a plot like our initial example (you have less colors, but use what you've got). Make such a plot, with appropriate colors and labels.

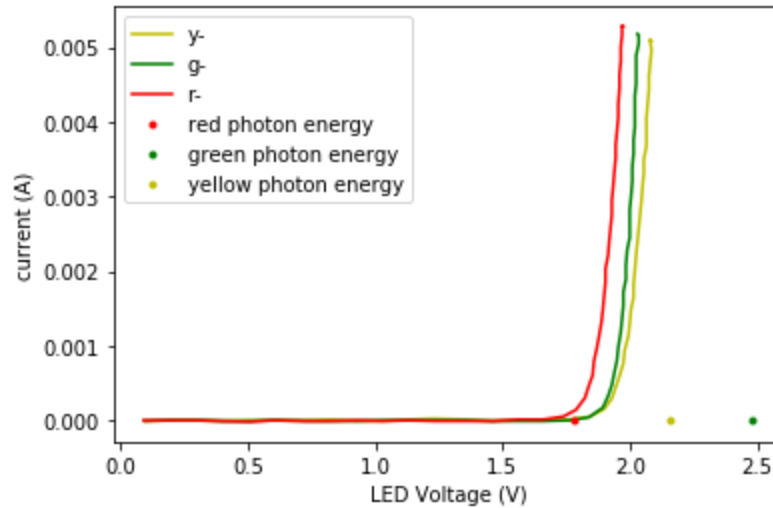
```
61     def getRGYData(self, maxVoltage = 5, deltaVoltage = 0.1):
62         trials = 3
63         dataFiles = ['y-', 'g-', 'r-']
64         while trials > 0:
65             trials -= 1
66             input('click \"Enter\" when ready')
67             setVoltages = np.arange(0, maxVoltage, deltaVoltage)
68             currents = np.zeros(len(setVoltages))
69             diodeVoltages = np.zeros(len(setVoltages))
70             for i in range(len(setVoltages)):
71                 thisVoltage = setVoltages[i]
72                 self.setDAC0(thisVoltage)
73                 sleep(0.05)
74                 topVoltage = self.labjack.getAIN(0)
75                 midVoltage = self.labjack.getAIN(1)
76                 diodeVoltages[i] = midVoltage;
77                 currents[i] = (topVoltage - midVoltage)/self.__resistance
78             self.setDAC0(0)
79             aData=open(dataFiles[trials], 'wb')
80             pickle.dump([diodeVoltages, currents], aData)
81             aData.close()
82
83
84     plt.figure("I-V diagram for LED")
85     while trials < 3:
86         aData=open(dataFiles[trials], 'rb')
87         xData, yData = pickle.load(aData)
88         aData.close()
89         plt.plot(xData, yData, dataFiles[trials], label=dataFiles[trials])
90         trials += 1
91     plt.plot(1.78, 0, 'r.', label='red photon energy')
92     plt.plot(2.48, 0, 'g.', label='green photon energy')
93     plt.plot(2.16, 0, 'y.', label='yellow photon energy')
94     plt.xlabel("LED Voltage (V)")
95     plt.ylabel("current (A)")
96     plt.legend()
97     plt.show()
```

```
1 myRobot.getRGYData()
```

click "Enter" when ready

click "Enter" when ready

click "Enter" when ready



9) Using Google and whatever else you like, find the energy of a single red photon, green photon, and yellow photon.

$E=hf$, where h is the Planck's constant, f is the frequency of photon, and E is the energy. We can plug in the threshold voltage as E to calculate the h , although this calculation is only good for the red LED used in this experiment.

Full Code:

```
import u3
from time import sleep
import matplotlib.pyplot as plt
import numpy as np
import pickle

class robot:

    def __init__(self):
        self.labjack = u3.U3()
        self.__resistance = 500

    def close(self):
        self.labjack.close()

    def printAnalogIns(self):
        AIN0value = self.labjack.getAIN(0)
        AIN1value = self.labjack.getAIN(1)
        descriptor = 'A0 = %5.3f volts, A1 = %5.3f volts ' %
(AIN0value, AIN1value)
        print(descriptor)

    def setDAC0(self, voltage = 0):
        if voltage < 0:
            print('output voltage must be >= 0')
            return
        elif voltage > 5:
            print('output voltage must be <= 5')
            return
        else:
            DAC0_VALUE = self.labjack.voltageToDACBits(voltage,
dacNumber = 0, is16Bits = False)
            self.labjack.getFeedback(u3.DAC0_8(DAC0_VALUE))

    def blink(self, n = 3):
        while n > 0:
            n-=1
            self.setDAC0(2)
            sleep(1)
            self.setDAC0(0)
            sleep(1)
```

```

def takeIVCurve(self, maxVoltage = 5, deltaVoltage = 0.1, verbose
= 0):
    setVoltages = np.arange(0,maxVoltage,deltaVoltage)
    currents = np.zeros(len(setVoltages))
    diodeVoltages = np.zeros(len(setVoltages))
    for i in range(len(setVoltages)):
        thisVoltage = setVoltages[i]
        self.setDAC0(thisVoltage)
        sleep(0.05)
        topVoltage = self.labjack.getAIN(0)
        midVoltage = self.labjack.getAIN(1)
        diodeVoltages[i] = midVoltage;
        currents[i] = (topVoltage - midVoltage)/self.__resistance

    plt.figure('Diode I-V curve')
    plt.plot(diodeVoltages,currents)
    plt.xlabel('diode voltage')
    plt.ylabel('diode current')
    plt.show()
    if verbose:

        plt.figure('Input voltages')
        plt.plot(setVoltages,diodeVoltages)
        plt.xlabel('DAC setpoint')
        plt.ylabel('measured output voltage')
        plt.show()

def getRGYData(self, maxVoltage = 5, deltaVoltage = 0.1):
    trials = 3
    dataFiles = ['y-','g-','r-']
    while trials > 0:
        trials -= 1
        input('click \"Enter\" when ready')
        setVoltages = np.arange(0,maxVoltage,deltaVoltage)
        currents = np.zeros(len(setVoltages))
        diodeVoltages = np.zeros(len(setVoltages))
        for i in range(len(setVoltages)):
            thisVoltage = setVoltages[i]
            self.setDAC0(thisVoltage)
            sleep(0.05)
            topVoltage = self.labjack.getAIN(0)
            midVoltage = self.labjack.getAIN(1)
            diodeVoltages[i] = midVoltage;
            currents[i] = (topVoltage -
midVoltage)/self.__resistance

```

```

        self.setDAC0(0)
        aData=open(dataFiles[trials],'wb')
        pickle.dump([diodeVoltages,currents],aData)
        aData.close()

plt.figure("I-V diagram for LED")
while trials < 3:
    aData=open(dataFiles[trials],'rb')
    xData, yData = pickle.load(aData)
    aData.close()
    plt.plot(xData, yData, dataFiles[trials],
label=dataFiles[trials])
    trials += 1
plt.plot(1.78, 0, 'r.', label='red photon energy')
plt.plot(2.48, 0, 'g.', label='green photon energy')
plt.plot(2.16, 0, 'y.', label='yellow photon energy')
plt.xlabel("LED Voltage (V)")
plt.ylabel("current (A)")
plt.legend()
plt.show()

myRobot = robot()
myRobot.takeIVCurve(verbose=0)
myRobot.setDAC0(0)
myRobot.getRGYData()
myRobot.close()

```