Embedded lab

Diodes

lab 2 - interfacing LED

lab 3 - varying intensity

Lab 4 - interrupt programming

lab 5 - 7 segment display

lab 6 - PIR sensor

lab 7 - ultra sonic sensor and buzzer

lab 8 - temperature sensor

lab 9 - LCD

lab 2 - interfacing LED

```
import RPi.GPIO as GPIO
import time //or from time import sleep
GPIO.setmode(GPIO.BOARD)
GPIO.setwarnings(False)
GPIO.setup(21,GPIO.OUT)
print "LED on"
GPIO.output(21,GPIO.HIGH) //GPIO.output(21,GPIO.HIGH,initial = GPIO.LOW)
time.sleep(10)
print "LED off"
GPIO.output(21,GPIO.LOW)
```

small leg - ground

large - gpio

lab 3 - varying intensity

```
import RPI.GPIO as GPIO

import time

GPIO.setmode(GPIO.Board)

GPIO.setup(11, GPIO.OUT)

p = GPIO.PWM(11, 100) //11 is the pin number, and 100 is the max range of PWM.

p.start(0) //Starting point of the PWM signal, you can select any value between 0 to 100.

while True:

for x in range (0, 100, 1): //Increasing brightness of LED from 0 to 100

p.ChangeDutyCycle(x)

time.sleep(0.1)

for x in range (100, 0, -1): //fading brightness of LED from 100 to 0

p.ChangeDutyCycle(x)

time.sleep(0.1)
```

Lab 4 - interrupt programming

```
import RPi.GPIO as GPIO
from time import sleep
GPIO.setmode(GPIO.BOARD)
GPIO.setwarnings(False)
BUTTON_PIN = 16
GPIO.setup(BUTTON_PIN, GPIO.IN, pull_up_down=GPIO.PUD_UP)

GPIO.set(3,GPIO.OUT,initial=GPIO.LOW)
While True:
  GPIO.wait_for_edge(24, GPIO.FALLING)
  print("lights on")
  GPIO.output(3,GPIO.HIGH)
  GPIO.wait_for_edge(24, GPIO.RISING)
  print("lights off")
  GPIO.output(3,GPIO.LOW)


button one leg - ground and one gpio

difference between falling and rising
```

lab 5 - 7 segment display

common anode - power

The individual segments are illuminated by applying a ground, logic “0” or “LOW”

common cathode - ground

The individual segments are illuminated by the application of a “HIGH”, or logic “1”

```
import RPi.GPIO as GPIO
import time
import os, sys
GPIO.setwarnings(False)
GPIO.setmode(GPIO.BOARD)
#setup output pins
GPIO.setup(35, GPIO.OUT)      //GPIO19
GPIO.setup(12, GPIO.OUT)      //GPIO18
GPIO.setup(36, GPIO.OUT)      //GPIO16
GPIO.setup(33, GPIO.OUT)      //GPIO13
GPIO.setup(32, GPIO.OUT)      //GPIO12
GPIO.setup(38, GPIO.OUT)      //GPIO20
GPIO.setup(40, GPIO.OUT)      //GPIO21

#define 7 segment digits
digitclr=[1,1,1,1,1,1,1]
digit0=[0,0,0,0,0,0,1]
digit1=[1,0,0,1,1,1,1]
digit2=[0,0,1,0,0,1,0]
digit3=[0,0,0,0,1,1,0]
digit4=[1,0,0,1,1,0,0]
digit5=[0,1,0,0,1,0,0]
digit6=[0,1,0,0,0,0,0,]
digit7=[0,0,0,1,1,1,1]
digit8=[0,0,0,0,0,0,0]
digit9=[0,0,0,1,1,0,0,]
gpin=[35,12,36,33,32,38,40]
#routine to clear and then write to display
def digdisp(digit):
for x in range (0,7):
GPIO.output(gpin[x], digit[x])
#routine to display digit from 0 to 9
digdisp(digit0)
time.sleep(1)
digdisp(digit1)
time.sleep(1)
digdisp(digit2)
```

```
time.sleep(1)
digdisp(digit3)
time.sleep(1)
digdisp(digit4)
time.sleep(1)
digdisp(digit5)
time.sleep(1)
digdisp(digit6)
time.sleep(1)
digdisp(digit7)
time.sleep(1)
digdisp(digit8)
time.sleep(1)
digdisp(digit9)
time.sleep(1)
#tidy up
GPIO.cleanup()
import sys
sys.exit()
```

lab 6 - PIR sensor

gnd - connect to ground

out - check normal pin number (not the gpio number)

vcc - power

```
import RPi.GPIO as GPIO
PIR_input = 8
green = 19
red = 23

GPIO.setwarnings(False)
GPIO.setmode(GPIO.BOARD)
GPIO.setup(PIR_input,GPIO.IN)
GPIO.setup(red,GPIO.OUT,intial = gpio.LOW)
GPIO.setup(green,GPIO.OUT,intial = gpio.LOW)

while True:
  if(GPIO.input(PIR_input)):
    GPIO.output(red, GPIO.HIGH)
    GPIO.output(green, GPIO.LOW)
  else
    GPIO.output(red, GPIO.LOW)
    GPIO.output(green, GPIO.HIGH)

GPIO.cleanup()
```

lab 7 - ultra sonic sensor and buzzer

vcc - 3volt

trig - normal pin number

echo - normal pin number

gnd - ground

-

connect to ground

s - gpio number

```
import RPi.GPIO s GPIO
import time
from gpiozero import Buzzer
try:
    GPIO.setwarnings(False)
    GPIO.setmode(GPIO.BCM)
    PIN_TRIGGER = 7
    PIN_ECHO = 11
    buzzer = Buzzer(22)
    GPIO.setup(PIN_TRIGGER, GPIO.OUT)
    GPIO.setup(PIN_ECHO, GPIO.IN)
    while True:
            GPIO.output(PIN_TRIGGER, GPIO.LOW)
            GPIO.output(PIN_TRIGGER, GPIO.HIGH)
            time.sleep(0.00001)
            GPIO.output(PIN_TRIGGER, GPIO.LOW)
            while GPIO.input(PIN_ECHO)==0:
                        pulse_start_time=time.time()
            while GPIO.input(PIN_ECHO)==1:
                        pulse_end_time=time.time()
            pulse_duration = pulse_end_time - pulse_start_time
            distance = round(pulse_diration * 17150,2)
            print("Distance:  ", distance, "  cm")
            if(distance<10)
                        for i in range(10):
                                    buzzer.on()
```

```
                                    time.sleep(0.001)
                                    buzzer.off()
                                    time.sleep(0.001)
finally:
    GPIO.cleanup()
```

lab 8 - temperature sensor

vcc - power

dt - gpio number (use 4)

gnd - ground

selection.py

```
import sqlite3
conn = sqlite3.connect('th.db')
data = conn.execute('select * from lab8)
for row in data:
  print('temp_c = ',row[0])
  print('temp_f = ',row[1])
  print('humidity = ',row[2])
```

insertion.py

```
import sqlite3
conn = sqlite3.connect('th.dt')
conn.execute('create table lab8 (tempc varchar(255), tempf varchar(255), humidity(255))')
conn.close()
```

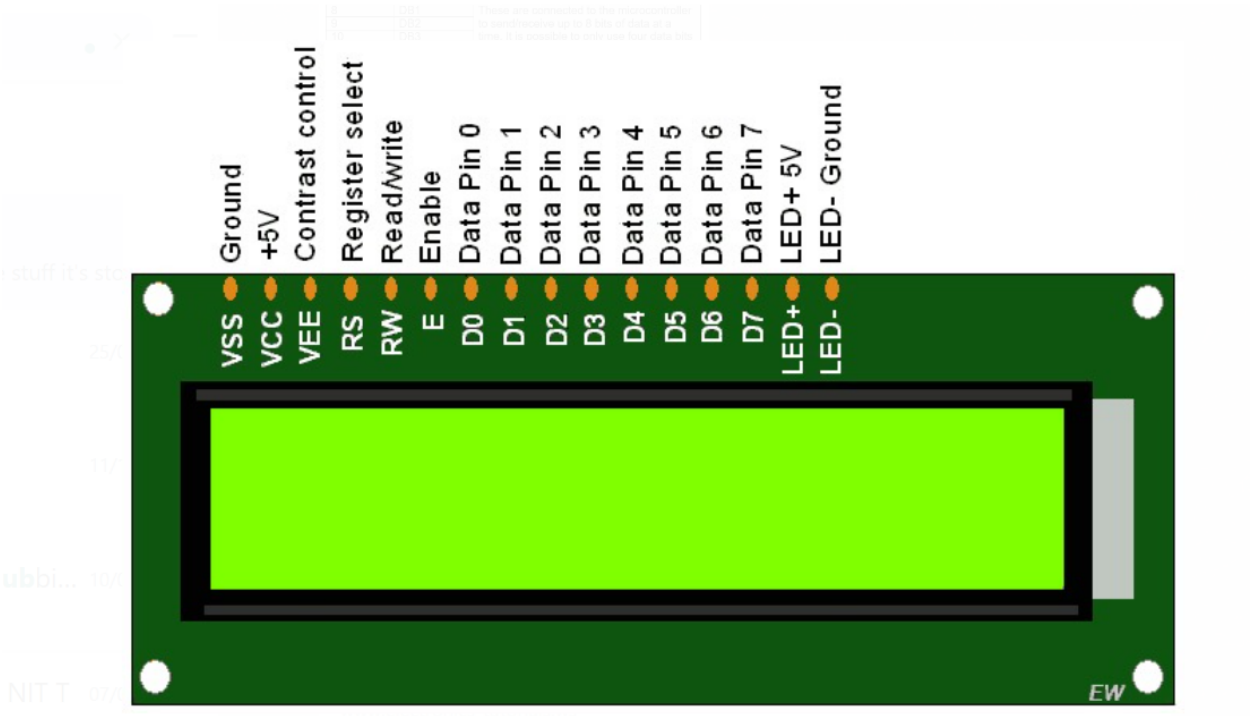
adafruit.py

```
import time
import board
import adafruit_dht
import sqlite3
dht_device = adafruit_dht.DHT22(board.D4,use_pulseio = False)
```

```
conn = sqlite3.connect('th.db')
c = conn.cursor()

while True:
    try:
      temp_c = dht_device.temperature
      temp_f = temp_c*(9/5)+32
      humidity = dht_device.humidity
      print('Temp: {:1f}F/{:1f}c'.format(temp_f, temp_c))
      print('Humidity : {:1f}'.format(humidity))
      c.execute(f"insert into lab8 values ({temp_c}, {temp_f},{humidity})")
      conn.connect()
    finally:
      print(' ')
conn.close()
```

lab 9 - LCD



VEE - ground

RS - gpio (gpio number)

RW - ground

E - gpio (gpio number)

d0,d1,d2,d3

d4,d5,d6,d7 - used some gpio (gpio number)

LED+ - 5V

LED- - ground

```
import time
import board as b
import digitalio as dio
import adafruit_character_lcd.character as lcd
```

```
lcd_col = 16
lcd_row = 2
rs = dioDigitalInOut(b.D7)
en = dioDigitalInOut(b.D8)
d4 = dioDigitalInOut(b.D11)
d5 = dioDigitalInOut(b.D25)
d6 = dioDigitalInOut(b.D24)
d7 = dioDigitalInOut(b.D10)
backlight = dioDigitalInOut(b.D4)

lcd = clcd.Character_LCD_Mono(rs,en,d7,d6,d5,d4,backlight)
lcd.cursor_position(0,0)
lcd.message = "Hey"
lcd.cursor_position(4,0)
lcd.message = "there."
time.sleep(3)
lcd.cursor_position(0,0)
lcd.message = "How you doing?"
time.sleep(3)
lcd.clear()
```