7. Interfacing of basic components (LED) with Raspberry Pi.

Name: Abrar Inamdar

Roll no: 44528

Batch: B2

```
import time
from gpiozero import LED

led1 = LED(8)
led2 = LED(10)
led3 = LED(9)
led4 = LED(11)
while True:
        try:
                led1.off()
                time.sleep(0.5)
                led1.on()
                led2.off()
                time.sleep(0.5)
                led2.on()
                led3.off()
                time.sleep(0.5)
                led3.on()
                led4.off()
                time.sleep(0.5)
                led4.on()
                time.sleep(0.5)
        except KeyboardInterrupt:
                print("closing")
                exit()
```

8. Interfacing of relay buzzer with Raspberry Pi.

Name: Abrar Inamdar

Roll no: 44528

Batch: B2

```
import time
import RPi.GPIO as GPIO
TRUE = 1
buzzer = 4
GPIO.setmode(GPIO.BCM)
GPIO.setup(buzzer,GPIO.OUT)
def buzzerState(val):
        GPIO.output(buzzer,val)
        try:
   while TRUE:
        buzzerState(1)
        time.sleep(1)
        buzzerState(0)
        time.sleep(1)

# If CTRL+C is pressed the main loop is broken
except KeyboardInterrupt:
   RUNNING = False
   print "\Quitting"
# Actions under 'finally' will always be called
finally:
   # Stop and finish cleanly so the pins
   # are available to be used again
   GPIO.cleanup()
```

9. Interfacing of IR sensor with Raspberry Pi.

Name: Abrar Inamdar

Roll no: 44528

Batch: B2

```
import time
import RPi.GPIO as GPIO
RUNNING = True
HIGH  = 1
LOW  = 0
DetectPin = 4
led = 8

def InitSystem():
        GPIO.setmode(GPIO.BCM)
        GPIO.setup(DetectPin,GPIO.IN,pull_up_down=GPIO.PUD_UP)
        GPIO.setup(led,GPIO.OUT)
        return

def DetectPerson():
        while True:
                input_state = GPIO.input(DetectPin)
                time.sleep(0.3)
                if input_state == 0:
                        return LOW
                else:
                        return HIGH
try:
        print ("\nCounting using IR LED\n")
```

```
        print  ("-----------------------------------------------\n")
        InitSystem()
        count =0;
        while RUNNING:
                state = DetectPerson()
                if state == LOW:
                        count+=1
                        print ("person count =%d" %count)
                        GPIO.output(led,LOW)
                        time.sleep(1)
                        GPIO.output(led,HIGH)

# If CTRL+C is pressed the main loop is broken
except KeyboardInterrupt:
    RUNNING = False


# Actions under 'finally' will always be called
finally:
    # Stop and finish cleanly so the pins
    # are available to be used again
    GPIO.cleanup()
```

10. Interfacing of Ultrasonic sensor with Raspberry Pi.

Name: Abrar Inamdar

Roll no: 44528

Batch: B2

```
import RPi.GPIO as GPIO
import time

#GPIO Mode (BOARD / BCM)
GPIO.setmode(GPIO.BCM)

#set GPIO Pins
GPIO_TRIGGER = 27
GPIO_ECHO = 18

#set GPIO direction (IN / OUT)
GPIO.setup(GPIO_TRIGGER, GPIO.OUT)
GPIO.setup(GPIO_ECHO, GPIO.IN)

def distance():
    # set Trigger to HIGH
    GPIO.output(GPIO_TRIGGER, True)

    # set Trigger after 0.01ms to LOW
    time.sleep(0.00001)
    GPIO.output(GPIO_TRIGGER, False)

    StartTime = time.time()
    StopTime = time.time()
```

```
    # save StartTime
    while GPIO.input(GPIO_ECHO) == 0:
        StartTime = time.time()

    # save time of arrival
    while GPIO.input(GPIO_ECHO) == 1:
        StopTime = time.time()

    # time difference between start and arrival
    TimeElapsed = StopTime- StartTime
    # multiply with the sonic speed (34300 cm/s)
    # and divide by 2, because there and back
    distance = (TimeElapsed * 34300) / 2

    return distance

if __name__ == '__main__':
    try:
        while True:
            dist = distance()
            print ("Measured Distance = %.1f cm" % dist)
            time.sleep(1)

        # Reset by pressing CTRL + C
    except KeyboardInterrupt:
        print("Measurement stopped by User")
        GPIO.cleanup()
```

A) Interfacing LED with Arduino with variable duty cycle.

Name: Abrar Inamdar

Roll no: 44528

Batch: B2

```
int del = 5; int a = 0;
void setup()
{
  pinMode(3, OUTPUT); // LED control pin is 3, a PWM capable pin
}
void loop()
{
  for (a = 0 ; a< 256 ; a++)
  {
  analogWrite(3, a);
  delay(del);
  }
  for (a = 255 ; a >= 0 ; a--)
  {
    analogWrite(3, a); delay(del);
  }
  delay(200);
}
```

B) Interfacing of RGB LED with Arduino and display all possible colours.

Name: Abrar Inamdar

Roll no: 44528

Batch: B2

```
int redPin= 9;
int greenPin = 6;
int bluePin = 5;
void setup()
{
pinMode(redPin, OUTPUT);
pinMode(greenPin, OUTPUT);
pinMode(bluePin, OUTPUT);
}
void loop()
{
  int i,j,k;
  for(i=0;i<255;i++)
  {
    analogWrite(redPin, i);
    analogWrite(greenPin, 128);
    analogWrite(bluePin, 128);
    delay(1000);
  }
  for(j=0;j<255;j++)
  {
    analogWrite(greenPin, j);
    analogWrite(bluePin, 128);
    analogWrite(redPin, 128);
    delay(1000);
  }
```

```
  for(k=0;k<255;k++)
  {
    analogWrite(bluePin, k);
    analogWrite(redPin, 128);
    analogWrite(greenPin, 128);
    delay(1000);
  }
}
```

2.Interfacing of LDR with Arduino and program for displaying the light Intensity

Name: Abrar Inamdar

Roll no: 44528

Batch: B2

```
int ldrPin = A0; // Analog input pin for LDR int ldrValue; // Variable to store LDR
value
void setup()
{
pinMode(ldrPin,INPUT);
Serial.begin(9600); // Initialize serial communication for debugging
}
void loop()
{
  int readValue;
  float realValue;
  readValue = analogRead(ldrPin);
  realValue = (5.0/1024.0)*readValue;
  Serial.println(realValue);
}
```

3. Interfacing of DC motor with Arduino and speed control using PWM.

Name: Abrar Inamdar

Roll no: 44528

Batch: B2

```
const int ENA = 6; // PWM-enabled digital pin connected to ENA of the motor driver
const int IN1 = 7; // Digital pin connected to IN1 of the motor driver
const int IN2 = 8; // Digital pin connected to IN2 of the motor driver
void setup()
{
pinMode(ENA, OUTPUT);
pinMode(IN1, OUTPUT);
pinMode(IN2, OUTPUT);
}
void loop()
{
// Rotate the motor in one direction (forward)
analogWrite(ENA, 255);// Adjust the value (0-255) to control the motor speed
digitalWrite(IN1, HIGH);
digitalWrite(IN2, LOW);

delay(2000);
// Rotate the motor in one direction (backward)
analogWrite(ENA, 255); // Adjust the value (0-255) to control the motor speed
digitalWrite(IN1, LOW);
digitalWrite(IN2, HIGH);

delay(2000);
}
```

4.Interfacing of 16x2 LCD with Arduino and displaying information.

Name: Abrar Inamdar

Roll no: 44528

Batch: B2

```
#include <LiquidCrystal.h>
const int rs = 12, en = 11, d4 = 5, d5 = 4, d6 = 3, d7 = 2;
LiquidCrystal lcd(rs, en, d4, d5, d6, d7);
void setup() {
  // set up the LCD's number of columns and rows:
  lcd.begin(16, 2);
  // Print a message to the LCD.
  lcd.print("hello, world!");
}
void loop() {
  // Turn off the blinking cursor:
  lcd.noBlink();
  delay(3000);

  // Turn on the blinking cursor:

  lcd.blink();

  delay(3000);
}
```

5.Interfacing of LM35 with Arduino and display the temperature data on LCD.

Name: Abrar Inamdar

Roll no: 44528

Batch: B2

```
#define sensorPin A0
void setup() {
Serial.begin(9600);
}
void loop() {                    // Get a reading from the temperature sensor:
int reading = analogRead(sensorPin);          //Convert digital data into analog by multiplying by 5000 and dividing by 1024
float voltage = reading * (5000 / 1024.0);    // Convert the voltage into the temperature in degree Celsius: float
float temperatureC = voltage / 10;
float temperatureF=(temperatureC*1.8)+32;          // Converting to Fahrenheit// Print the temperature in Celsius into the Serial Monitor:
Serial.print("Temperature in Celsius = ");
Serial.print(temperatureC);
Serial.println("C");                    // Print the temperature in Celsius into the Serial Monitor:
Serial.print("Temperature in Fahrenheit = ");
Serial.print(temperatureF);
Serial.println("F"); Serial.print("\n");
delay(1000);                    // wait a second between readings
}
```

6.Interfacing of Bluetooth module with Arduino and sending commands wirelessly

Name: Abrar Inamdar

Roll no: 44528

Batch: B2

```
int LEDpin=13;


void setup()
{
  Serial.begin(9600);
  pinMode(LEDpin,OUTPUT);
}
void loop()
{
  if (Serial.available()>0)
  {
    char data=Serial.read();


    switch(data)
    {
      case 'a':
digitalWrite(LEDpin,HIGH);break;
      case 'd':
digitalWrite(LEDpin,LOW);break;
      case 'r': for(int i=0; i<10; i++)
{digitalWrite(LEDpin,HIGH); delay(500);
```

```
digitalWrite(LEDpin,LOW); delay(500);}
break;
     default : break;
   }
 }
}
```