Practical no : 0 Arduino Components

Arduino Uno R3:

A programmable board you can use to build interactive circuits.

BreadBoard Small:

A half-size breadboard with 30 rows,10 columns, and two pairs of power rails.

Resistor:

Restricts the flow of electricity in a circuit, reducing the voltage and current as a result.

LED:

Light- Emitting Diode that lights up when electricity passes through it in the correct direction.

BreadBoard Mini:

A quarter-size breadboard with 17 rows and 10 columns.

7 segment Display:

A single 7-segment LED for displaying a number of letter.

LCD 16 X 2:

A Liquid Crystal Display capable of displaying two lines of 16 characters.

Pushbutton:

A switch that closes a circuit while pressed.

Temperature Sensor:

A sensor that outputs different voltages based on the ambient temperature.

Oscilloscope:

Electronic test equipment for measuring output signals.

Potentiometer:

A type of resistor whose resistance changes at the turn of a knob.

LED RGB:

A type of LED that combines Red, Blue, and Green to produce any color.

PIR Sensor:

Passive infrared motion sensor used to sense motion in front of it.

DC Motor:

A motor,which converts electrical energy into mechanical energy.

Photoresistor:

A sensor whose resistance changes based on the amount of light it senses.

Power Supply:

Electronic test equipment for supplying power to your circuit.

Relay SPDT:

A 5V SPDT power relay for switching between two circuits.

Light Bulb**:**

A 12V/ 3W incandescent light bulb.

Practical no : 0 Raspberry Pi Components

GPIO Pins (General Purpose Input/Output):

Raspberry Pi has a set of GPIO pins that allow it to interface with external hardware, such as sensors, motors, and other devices. These pins can be programmed for input (to read data) or output (to send signals).

Power Supply:

Raspberry Pi is typically powered by a 5V micro USB or USB-C power supply. It is important to provide a stable and sufficient power supply to prevent issues during operation.

MicroSD Card:

Raspberry Pi uses a microSD card as its primary storage device. The operating system (like Raspberry Pi OS) and all programs are stored on this card.

HDMI Port:

Raspberry Pi has an HDMI port that allows it to be connected to a monitor or TV. This makes it easy to use as a desktop computer or display visual outputs from projects.

USB Ports:

Multiple USB ports are available for connecting peripherals like keyboards, mice, or USB storage devices.

Ethernet/Wi-Fi:

Some models come with Ethernet ports for wired internet connectivity. Wi-Fi is built into many modern Raspberry Pi models for wireless communication.

Camera and Display Interface:

The Raspberry Pi has dedicated ports for connecting cameras (using the Camera Serial Interface, CSI) and displays (using the Display Serial Interface, DSI). This makes it suitable for visual recognition or surveillance projects.

Add-on Modules and Shields:

Common add-ons include sensors (temperature, motion, etc.), actuators (motors), and shields that expand its capabilities. For example, the Raspberry Pi can be integrated with GPS modules, RFID readers, or LCD displays for interactive projects

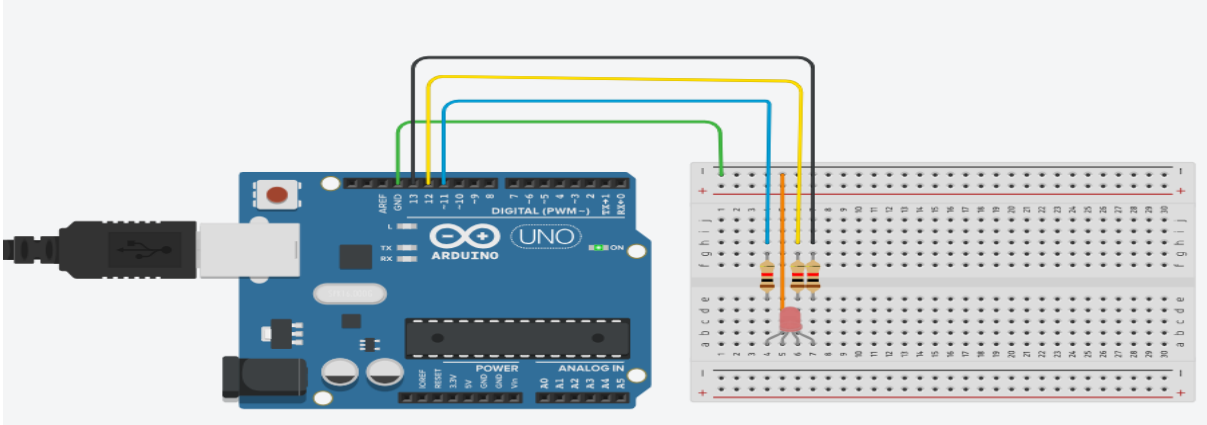
Practical No : 1 Displaying different LED patterns using Arduino

Aim : To Displaying different LED patterns using Arduino

Code :

```
void setup()
{
  pinMode(LEDR, OUTPUT);
  pinMode(LEDG, OUTPUT);
  pinMode(LEDB, OUTPUT);
}
int r = 0;
int g = 0;
int b = 0;
void loop()
{
  r=random(0,255);
  g=random(0,255);
  b=random(0,255);

  analogWrite(LEDR, r);
  analogWrite(LEDG, g);
  analogWrite(LEDB, b);
  delay(1000);
}
```

OUTPUT:



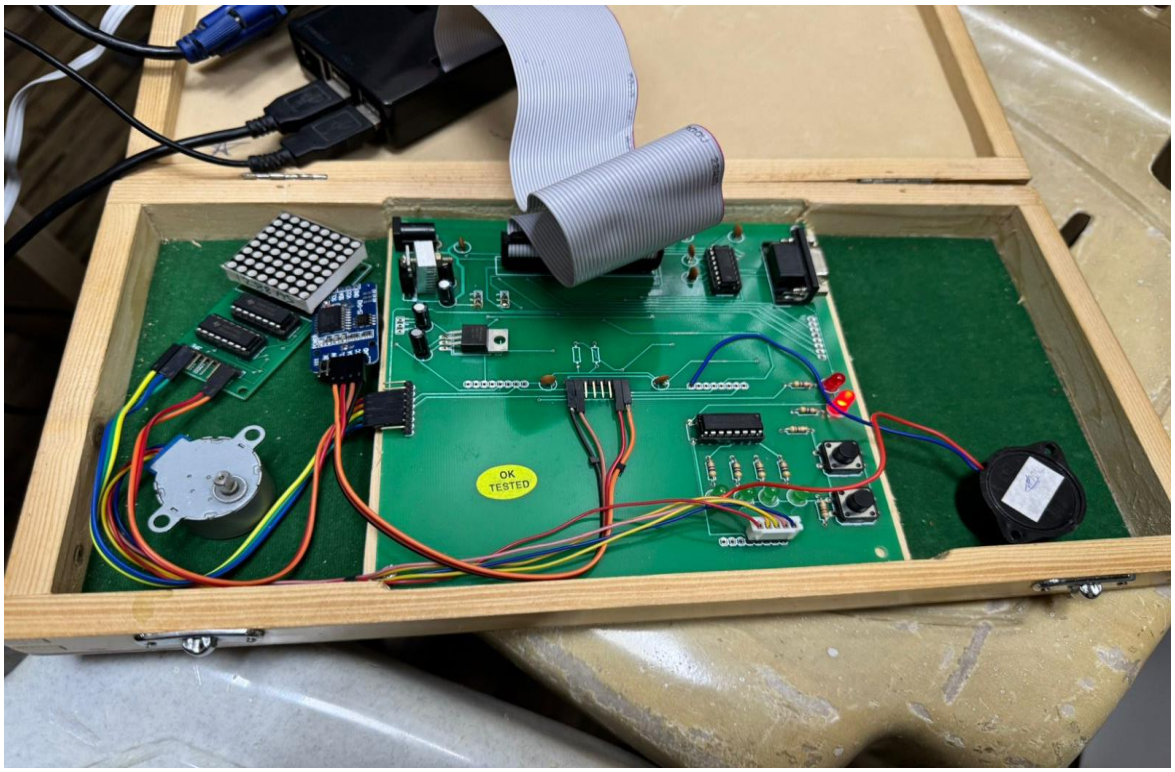
Practical No : 1 Displaying different LED patterns with Raspberry Pi.

Aim : To Displaying different LED patterns with Raspberry Pi.

Code :

```
import RPi.GPIO as GPIO
import time
GPIO.setwarnings(False)
GPIO.setmode(GPIO.BCM)
GPIO.setup(21,GPIO.OUT)
GPIO.setup(17,GPIO.OUT)
list=[21,17]
while True:
    for n in range(len(list)):
        GPIO.output(list[n],True)
        time.sleep(0.1)
        GPIO.output(list[n],False)
        time.sleep(0.2)
GPIO.cleanup()
```

OUTPUT:



Practical No : 2 Displaying Time over 4-Digit 7-Segment Display using Arduino

Aim : To Displaying Time over 4-Digit 7-Segment Display using Arduino

Code:

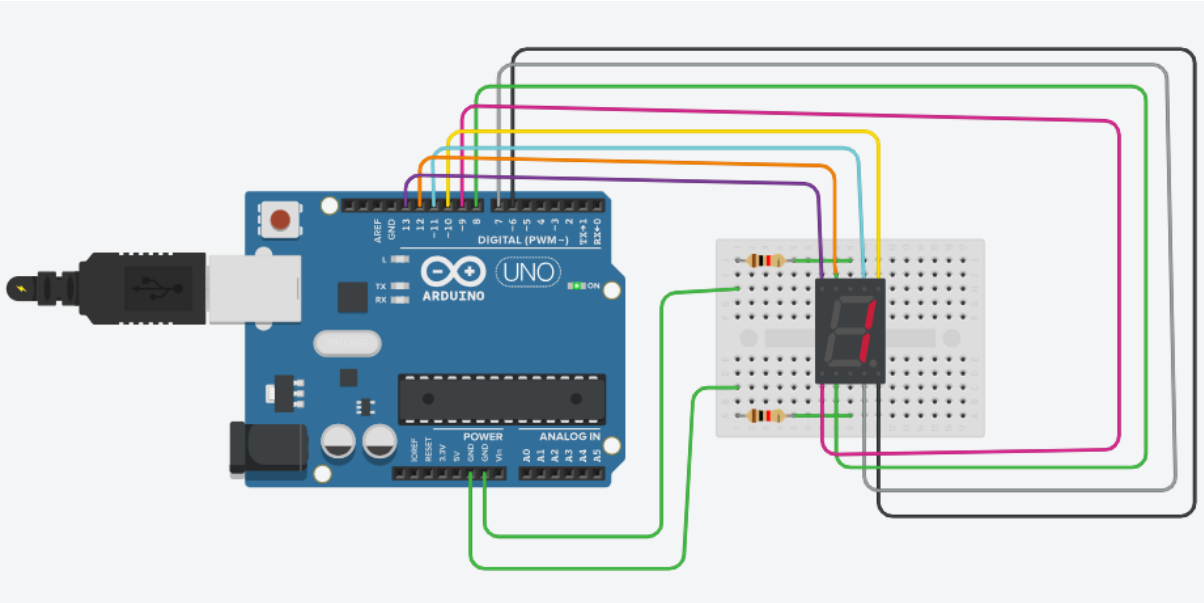
```
unsigned const int A=13;
unsigned const int B=12;
unsigned const int C=11;
unsigned const int D=10;
unsigned const int E=9;
unsigned const int F=8;
unsigned const int G=7;
unsigned const int H=6;
void setup(void)
{
  pinMode(A,OUTPUT);
  pinMode(B,OUTPUT);
  pinMode(C,OUTPUT);
  pinMode(D,OUTPUT);
  pinMode(E,OUTPUT);
  pinMode(F,OUTPUT);
  pinMode(G,OUTPUT);
pinMode(H,OUTPUT);
}
void zero(void){
  digitalWrite(A,LOW);
   digitalWrite(B,HIGH);
  digitalWrite(C,HIGH);
  digitalWrite(D,HIGH);
  digitalWrite(E,HIGH);
  digitalWrite(F,HIGH);
  digitalWrite(G,HIGH);
  digitalWrite(H,LOW);
}
void one(void){
  digitalWrite(A,LOW);
   digitalWrite(B,LOW);
  digitalWrite(C,LOW);
  digitalWrite(D,HIGH);
  digitalWrite(E,LOW);
  digitalWrite(F,LOW);
  digitalWrite(G,HIGH);
  digitalWrite(H,LOW);
}
void two(void){
```

```
  digitalWrite(A,HIGH);
   digitalWrite(B,LOW);
  digitalWrite(C,HIGH);
  digitalWrite(D,HIGH);
  digitalWrite(E,HIGH);
  digitalWrite(F,HIGH);
  digitalWrite(G,LOW);
  digitalWrite(H,LOW);
}
void three(void){
  digitalWrite(A,HIGH);
   digitalWrite(B,LOW);
  digitalWrite(C,HIGH);
  digitalWrite(D,HIGH);
  digitalWrite(E,LOW);
  digitalWrite(F,HIGH);
  digitalWrite(G,HIGH);
  digitalWrite(H,LOW);
}
void four(void){
  digitalWrite(A,HIGH);
   digitalWrite(B,HIGH);
  digitalWrite(C,LOW);
  digitalWrite(D,HIGH);
  digitalWrite(E,LOW);
  digitalWrite(F,LOW);
  digitalWrite(G,HIGH);
  digitalWrite(H,LOW);
}
void five(void){
  digitalWrite(A,HIGH);
   digitalWrite(B,HIGH);
  digitalWrite(C,HIGH);
  digitalWrite(D,LOW);
  digitalWrite(E,LOW);
  digitalWrite(F,HIGH);
  digitalWrite(G,HIGH);
  digitalWrite(H,LOW);
}
void six(void){
  digitalWrite(A,HIGH);
   digitalWrite(B,HIGH);
  digitalWrite(C,HIGH);
  digitalWrite(D,LOW);
```

```
  digitalWrite(E,HIGH);
  digitalWrite(F,HIGH);
  digitalWrite(G,HIGH);
  digitalWrite(H,LOW);
}
void seven(void){
  digitalWrite(A,LOW);
   digitalWrite(B,LOW);
  digitalWrite(C,HIGH);
  digitalWrite(D,HIGH);
  digitalWrite(E,LOW);
  digitalWrite(F,LOW);
  digitalWrite(G,HIGH);
  digitalWrite(H,LOW);
}
void eight(void){
  digitalWrite(A,HIGH);
   digitalWrite(B,HIGH);
  digitalWrite(C,HIGH);
  digitalWrite(D,HIGH);
  digitalWrite(E,HIGH);
  digitalWrite(F,HIGH);
  digitalWrite(G,HIGH);
  digitalWrite(H,LOW);
}
void nine(void){
  digitalWrite(A,HIGH);
   digitalWrite(B,HIGH);
  digitalWrite(C,HIGH);
  digitalWrite(D,HIGH);
  digitalWrite(E,LOW);
  digitalWrite(F,HIGH);
  digitalWrite(G,HIGH);
  digitalWrite(H,LOW);
}
void loop(void){
  zero();
  delay(1000);
  one();
  delay(1000);
  two();
  delay(1000);
  three();
  delay(1000);
```

```
  four();
  delay(1000);
  five();
  delay(1000);
  six();
  delay(1000);
  seven();
  delay(1000);
  eight();
  delay(1000);
  nine();
  delay(1000);
}
```

OUTPUT:



Practical No : 2 Displaying Time over 4-Digit 7-Segment Display using Raspberry Pi

Aim : Displaying Time over 4-Digit 7-Segment Display using Raspberry Pi

Code :

```
import RPi.GPIO as GPIO
import time
from table import *

SDI=25
RCLK=27
SRCLK=24

per_line=[0x7f, 0xbf, 0xdf, 0xef, 0xf7, 0xfb, 0xfd, 0xfe]

def print_msg():
    print('Program is running ...')
    print('Please press ctrl+c to end the program..')

def setup():
    GPIO.setmode(GPIO.BCM)
    GPIO.setup(SDI,GPIO.OUT)
    GPIO.setup(RCLK,GPIO.OUT)
    GPIO.setup(SRCLK,GPIO.OUT)
    GPIO.output(SDI,GPIO.LOW)
    GPIO.output(RCLK,GPIO.LOW)
    GPIO.output(SRCLK,GPIO.LOW)

def hc595_in(dat):
    for bit in range(0,8):
        GPIO.output(SDI,(1&(dat>>bit)))
        GPIO.output(SRCLK,GPIO.HIGH)
        time.sleep(0.000001)
        GPIO.output(SRCLK,GPIO.LOW)

def hc595_out():
    GPIO.output(RCLK,GPIO.HIGH)
    time.sleep(0.000001)
    GPIO.output(RCLK,GPIO.LOW)

def flash(table):
    for i in range(8):
        hc595_in(per_line[i])
        hc595_in(table[i])
        hc595_out()
```

```
        hc595_in(per_line[7])
        hc595_in(0x00)
        hc595_out()

def show(table,second):
    start=time.time()
    while True:
        flash(table)
        finish=time.time()
        if finish-start>second:
            break

def main():

charactors='AaBbCcDdEeFfGgHhIiJjKkLlMmNnOoPpQqRrSsTtUuVvWwXxYyZz1234567890'
    word='Microbyte'
    while True:
        for table in word:
            table = table.upper()
            show(table.charactors[table],1)
            time.sleep(1)
            show(table.picture['creeper'],1)
            time.sleep(1)
            show(table.picture['smile'],1)
            time.sleep(1)
        for charactor in charactors:
            print("charactor: %s",charactor)
            show(table.charactors[charactor],1)
            time.sleep(1)

def destroy():
    GPIO.cleanup()

if __name__ == '__main__':
    setup()
    try:
        main()
    except KeyboardInterrupt:
        destroy()
```

Practical No : 3 Interfacing 16X2 LCD With Arduino to display different messages.
Aim : To Interfacing 16X2 LCD With Arduino to display different messages**.**

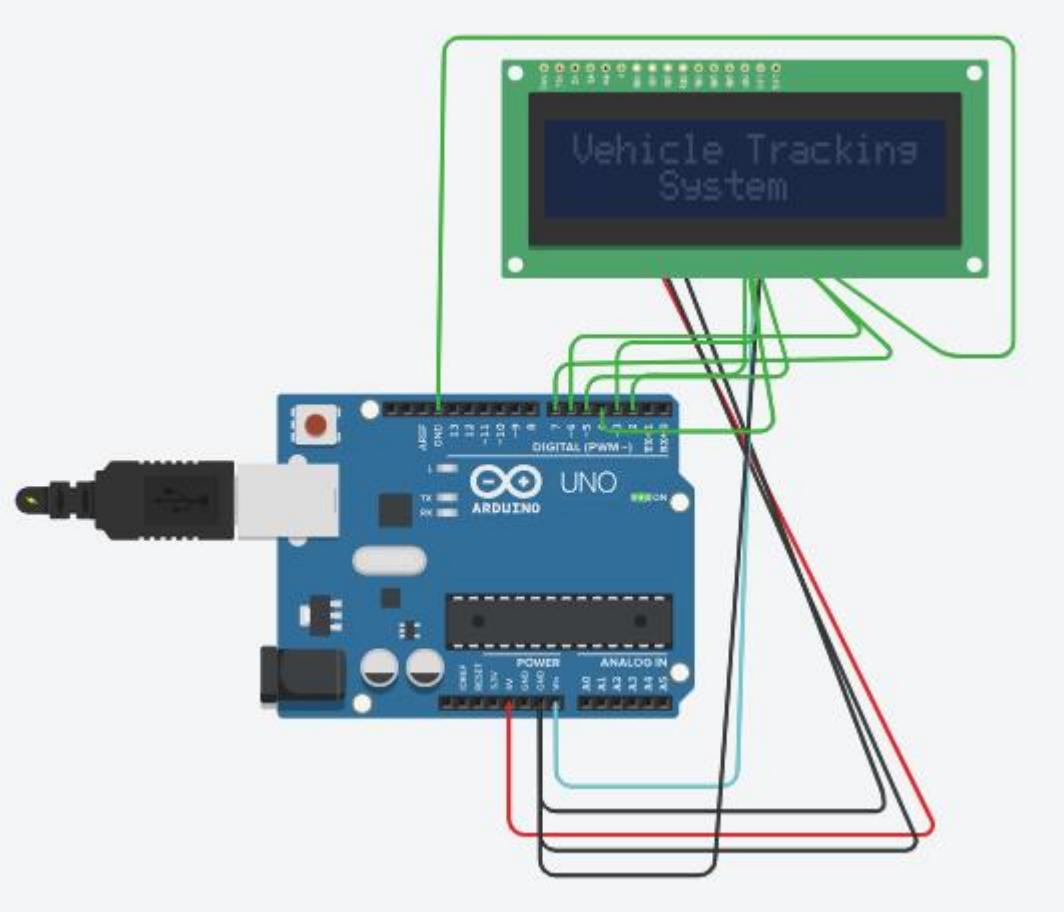
Code :

```
#include<LiquidCrystal.h>
LiquidCrystal lcd(7, 6, 5, 4, 3, 2);
//String str="";
char str[70];
String gpsString="$GPGGA,134658.00,5106.9792,N,11402.3003,W,2,09,1.0,1048.47,M,-
16.27,M,08,AAAA*60";
char *test="$GPGGA";
String latitude="No Range  ";
String longitude="No Range     ";
int temp=0,i;
boolean gps_status=0;
void setup()
{
  lcd.begin(16,2);
  Serial.begin(9600);
  lcd.print("Vehicle Tracking");
  lcd.setCursor(0,1);
  lcd.print("    System      ");
  delay(4000);
  lcd.clear();
  delay(2000);
  get_gps();
}
void loop()
{
    get_gps();
}
void get_gps()
{
   gps_status=0;
   int x=0;
```

```
while(gps_status==0)
{
 int str_lenth=81;
 latitude="";
 longitude="";
 int comma=0;
 //gpsString = '134658.00,5106.9792,N,11402.3003,W,2,09,1.0,1048.47';
 //Serial.println(gpsString);
 while(x<81)
 {
   //Serial.println(gpsString[x]);
   if(gpsString[x]==','){
   comma++;

   }
   if(comma==2)        //extract latitude from string
   latitude+=gpsString[x+1];
   else if(comma==4)        //extract longitude from string
   longitude+=gpsString[x+1];
   x++;
 }
 int l1=latitude.length();
 latitude[l1-1]=' ';
 l1=longitude.length();
 longitude[l1-1]=' ';

 lcd.clear();
 lcd.println("Lat:");
 lcd.print(latitude);
 lcd.setCursor(0,1);
 lcd.print("Long:");
 lcd.print(longitude);

 Serial.print("Lat:  ");
```

```
    Serial.print(latitude);
    Serial.print("\tLong:  ");
    Serial.println(longitude);
    i=0;x=0;
    str_lenth=0;
    delay(2000);
   }
}
```

OUTPUT :



Practical No : 3 Interfacing 16X2 LCD with Raspberry Pi to display different messages.

Aim : To Interfacing 16X2 LCD with Raspberry Pi to display different messages

Code :

```
charactors = {
   "A" : [
   0b00000000,
   0b00011000,
   0b00100100,
   0b00100100,
   0b00111100,
   0b00100100,
   0b00000000,
   0b00000000,
   ],
   "a" : [
   0b00000000,
   0b00000000,
   0b00011100,
   0b00100100,
   0b00100100,
   0b00011100,
   0b00000000,
   0b00000000,
   ],
   "B" : [
   0b00000000,
   0b00111000,
   0b00100100,
   0b00111000,
   0b00100100,
   0b00111000,
   0b00000000,
   0b00000000,
   ],
   "b" : [
   0b00000000,
   0b00100000,
   0b00111000,
   0b00100100,
   0b00100100,
   0b00111000,
   0b00000000,
   0b00000000,
   ],
```

```
"C" : [
0b00000000,
0b00011000,
0b00100000,
0b00100000,
0b00100000,
0b00011000,
0b00000000,
0b00000000,
],
"c" : [
0b00000000,
0b00000000,
0b00011000,
0b00100000,
0b00100000,
0b00011000,
0b00000000,
0b00000000,
],
"D" : [
0b00000000,
0b00111000,
0b00100100,
0b00100100,
0b00100100,
0b00111000,
0b00000000,
0b00000000,
],
"d" : [
0b00000000,
0b00000100,
0b00011100,
0b00100100,
0b00100100,
0b00011100,
0b00000000,
0b00000000,
],
"E" : [
0b00000000,
0b00111000,
0b00100000,
```

```
0b00111000,
0b00100000,
0b00111000,
0b00000000,
0b00000000,
],
"e" : [
0b00000000,
0b00000000,
0b00011000,
0b00101100,
0b00110000,
0b00011000,
0b00000000,
0b00000000,
],
"F" : [
0b00000000,
0b00111000,
0b00100000,
0b00111000,
0b00100000,
0b00100000,
0b00000000,
0b00000000,
],
"f" : [
0b00000000,
0b00001000,
0b00010000,
0b00111000,
0b00010000,
0b00010000,
0b00000000,
0b00000000,
],
"G" : [
0b00000000,
0b00011100,
0b00100000,
0b00101100,
0b00100100,
0b00011100,
0b00000000,
```

```
0b00000000,
],
"g" : [
0b00000000,
0b00000000,
0b00011100,
0b00100100,
0b00100100,
0b00011100,
0b00000100,
0b00011000,
],
"H" : [
0b00000000,
0b00100100,
0b00100100,
0b00111100,
0b00100100,
0b00100100,
0b00000000,
0b00000000,
],
"h" : [
0b00000000,
0b00100000,
0b00111000,
0b00100100,
0b00100100,
0b00100100,
0b00000000,
0b00000000,
],
"I" : [
0b00000000,
0b00111000,
0b00010000,
0b00010000,
0b00010000,
0b00111000,
0b00000000,
0b00000000,
0b00000000,
],
"i" : [
```

```
0b00000000,
0b00100000,
0b00000000,
0b00100000,
0b00100000,
0b00100000,
0b00000000,
0b00000000,
0b00000000,
],
"J" : [
0b00000000,
0b00001100,
0b00000100,
0b00000100,
0b00100100,
0b00011000,
0b00000000,
0b00000000,
],
"j" : [
0b00000000,
0b00000100,
0b00000000,
0b00000100,
0b00000100,
0b00000100,
0b00001000,
0b00000000,
],
"K" : [
0b00000000,
0b00100100,
0b00101000,
0b00110000,
0b00101000,
0b00100100,
0b00000000,
0b00000000,
],
"k" : [
0b00000000,
0b00100000,
0b00100100,
```

```
0b00101000,
0b00111000,
0b00100100,
0b00000000,
0b00000000,
],
"L" : [
0b00000000,
0b00100000,
0b00100000,
0b00100000,
0b00100000,
0b00111100,
0b00000000,
0b00000000,
],
"l" : [
0b00000000,
0b00110000,
0b00010000,
0b00010000,
0b00010000,
0b00010000,
0b00000000,
0b00000000,
],
"M" : [
0b00000000,
0b00100010,
0b00110110,
0b00101010,
0b00100010,
0b00100010,
0b00000000,
0b00000000,
],
"m" : [
0b00000000,
0b00000000,
0b00111100,
0b00101010,
0b00101010,
0b00101010,
0b00000000,
```

```
0b00000000,
],
"N" : [
0b00000000,
0b00100100,
0b00110100,
0b00101100,
0b00100100,
0b00100100,
0b00000000,
0b00000000,
],
"n" : [
0b00000000,
0b00000000,
0b00111000,
0b00100100,
0b00100100,
0b00100100,
0b00000000,
0b00000000,
],
"O" : [
0b00000000,
0b00011000,
0b00100100,
0b00100100,
0b00100100,
0b00011000,
0b00000000,
0b00000000,
],
"o" : [
0b00000000,
0b00000000,
0b00011000,
0b00100100,
0b00100100,
0b00011000,
0b00000000,
0b00000000,
],
"P" : [
0b00000000,
```

```
0b00111000,
0b00100100,
0b00111000,
0b00100000,
0b00100000,
0b00000000,
0b00000000,
],
"p" : [
0b00000000,
0b00000000,
0b00111000,
0b00100100,
0b00111000,
0b00100000,
0b00100000,
0b00000000,
],
"Q" : [
0b00000000,
0b00011000,
0b00100100,
0b00100100,
0b00100100,
0b00011000,
0b00000100,
0b00000000,
],
"q" : [
0b00000000,
0b00000000,
0b00011100,
0b00100100,
0b00011100,
0b00000100,
0b00000100,
0b00000000,
],
"R" : [
0b00000000,
0b00111000,
0b00100100,
0b00100100,
0b00111000,
```

```
0b00100100,
0b00000000,
0b00000000,
],
"r" : [
0b00000000,
0b00000000,
0b00101000,
0b00110000,
0b00100000,
0b00100000,
0b00000000,
0b00000000,
],
"S" : [
0b00000000,
0b00011100,
0b00100000,
0b00011000,
0b00000100,
0b00111000,
0b00000000,
0b00000000,
],
"s" : [
0b00000000,
0b00000000,
0b00011100,
0b00110000,
0b00001100,
0b00111000,
0b00000000,
0b00000000,
],
"T" : [
0b00000000,
0b00111000,
0b00010000,
0b00010000,
0b00010000,
0b00010000,
0b00000000,
0b00000000,
],
```

```
"t" : [
0b00000000,
0b00010000,
0b00111000,
0b00010000,
0b00010000,
0b00001000,
0b00000000,
0b00000000,
],
"U" : [
0b00000000,
0b00100100,
0b00100100,
0b00100100,
0b00100100,
0b00011000,
0b00000000,
0b00000000,
],
"u" : [
0b00000000,
0b00000000,
0b00100100,
0b00100100,
0b00100100,
0b00011100,
0b00000000,
0b00000000,
],
"V" : [
0b00000000,
0b00100100,
0b00100100,
0b00101000,
0b00101000,
0b00010000,
0b00000000,
0b00000000,
],
"v" : [
0b00000000,
0b00000000,
0b00100100,
```

```
0b00100100,
0b00101000,
0b00010000,
0b00000000,
0b00000000,
],
"W" : [
0b00000000,
0b00100010,
0b00101010,
0b00101010,
0b00101010,
0b00010100,
0b00000000,
0b00000000,
],
"w" : [
0b00000000,
0b00000000,
0b00101010,
0b00101010,
0b00010100,
0b00010100,
0b00000000,
0b00000000,
],
"X" : [
0b00000000,
0b00100100,
0b00100100,
0b00011000,
0b00100100,
0b00100100,
0b00000000,
0b00000000,
],
"x" : [
0b00000000,
0b00000000,
0b00101000,
0b00010000,
0b00010000,
0b00101000,
0b00000000,
```

```
0b00000000,
],
"Y" : [
0b00000000,
0b00100100,
0b00100100,
0b00011100,
0b00000100,
0b00011000,
0b00000000,
0b00000000,
],
"y" : [
0b00000000,
0b00000000,
0b00100100,
0b00100100,
0b00100100,
0b00011100,
0b00000100,
0b00011000,
],
"Z" : [
0b00000000,
0b00111000,
0b00001000,
0b00010000,
0b00100000,
0b00111000,
0b00000000,
0b00000000,
],
"z" : [
0b00000000,
0b00000000,
0b00111100,
0b00001000,
0b00010000,
0b00111100,
0b00000000,
0b00000000,
],
"1" : [
0b00000000,
```

```
0b00010000,
0b00110000,
0b00010000,
0b00010000,
0b00111000,
0b00000000,
0b00000000,
],
"2" : [
0b00000000,
0b00111000,
0b00000100,
0b00011000,
0b00100000,
0b00111100,
0b00000000,
0b00000000,
],
"3" : [
0b00000000,
0b00111000,
0b00000100,
0b00011000,
0b00000100,
0b00111000,
0b00000000,
0b00000000,
],
"4" : [
0b00000000,
0b00001000,
0b00011000,
0b00101000,
0b00111100,
0b00001000,
0b00000000,
0b00000000,
],
"5" : [
0b00000000,
0b00111100,
0b00100000,
0b00111000,
0b00000100,
```

```
0b00111000,
0b00000000,
0b00000000,
],
"6" : [
0b00000000,
0b00011000,
0b00100000,
0b00111000,
0b00100100,
0b00011000,
0b00000000,
0b00000000,
],
"7" : [
0b00000000,
0b00111100,
0b00000100,
0b00001000,
0b00010000,
0b00010000,
0b00000000,
0b00000000,
],
"8" : [
0b00000000,
0b00011000,
0b00100100,
0b00011000,
0b00100100,
0b00011000,
0b00000000,
0b00000000,
],
"9" : [
0b00000000,
0b00011000,
0b00100100,
0b00011100,
0b00000100,
0b00011000,
0b00000000,
0b00000000,
],
```

```
"0" : [
0b00000000,
0b00011000,
0b00100100,
0b00100100,
0b00100100,
0b00011000,
0b00000000,
0b00000000,
],
"!" : [
0b00000000,
0b00100000,
0b00100000,
0b00100000,
0b00000000,
0b00100000,
0b00000000,
0b00000000,
],
"?" : [
0b00000000,
0b00111000,
0b00000100,
0b00011000,
0b00000000,
0b00010000,
0b00000000,
0b00000000,
],
"," : [
0b00000000,
0b00000000,
0b00000000,
0b00000000,
0b00000000,
0b00010000,
0b00100000,
0b00000000,
],
"." : [
0b00000000,
0b00000000,
0b00000000,
```

```
    0b00000000,
    0b00000000,
    0b00100000,
    0b00000000,
    0b00000000,
    ],
    "<" : [
    0b00000000,
    0b00001000,
    0b00010000,
    0b00100000,
    0b00010000,
    0b00001000,
    0b00000000,
    0b00000000,
    ],
    ">" : [
    0b00000000,
    0b00100000,
    0b00010000,
    0b00001000,
    0b00010000,
    0b00100000,
    0b00000000,
    0b00000000,
    ],
}
picture = {
    "smile":[
        0b00000000,
        0b01000010,
        0b10100101,
        0b00000000,
        0b00000000,
        0b01000010,
        0b00111100,
        0b00000000,
        ],
    "creeper":[
        0b11111111,
        0b11111111,
        0b10011001,
        0b10011001,
        0b11100111,
        0b11000011,
        0b11000011,
        0b11011011,
        ],
}
```

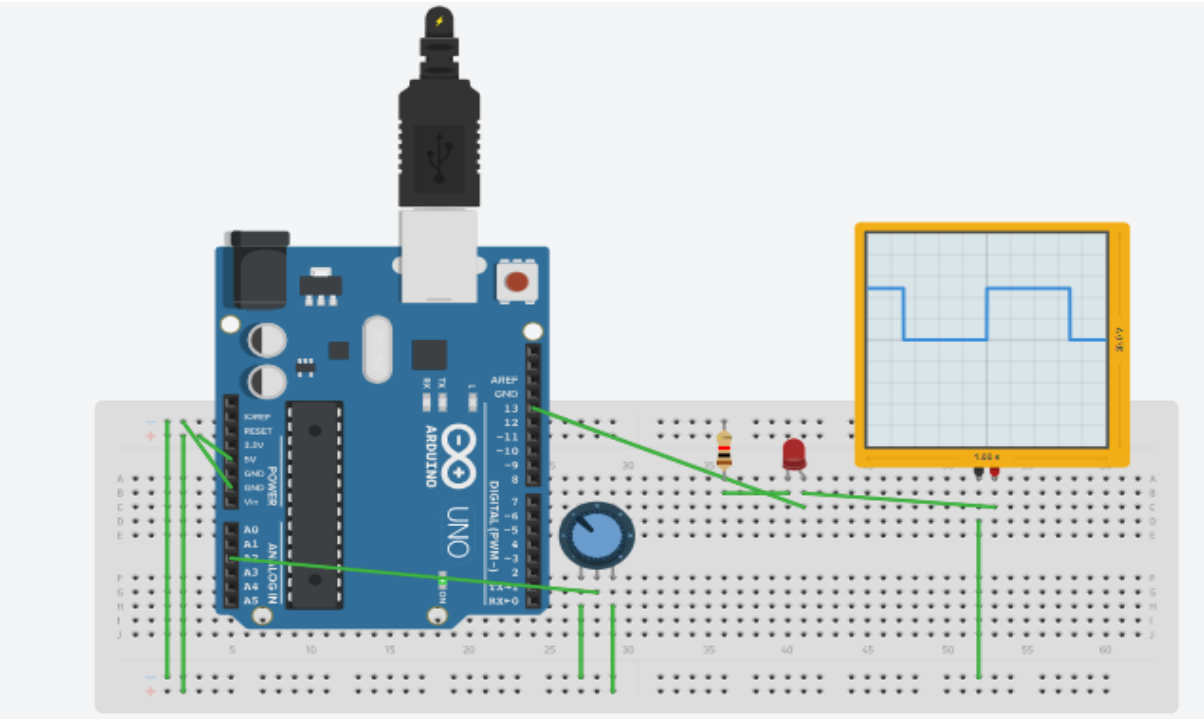
Practical No : 4 Arduino Based Oscilloscope

Aim : To Arduino Based Oscilloscope

Code :

```
int potPin=2;
int ledPin=13;
int val=10;
void setup()
{
  pinMode(ledPin, OUTPUT);
}
void loop()
{
  val = analogRead(potPin);
  digitalWrite(ledPin, HIGH);
  delay(val);
  digitalWrite(ledPin, LOW);
  delay(val);
}
```

OUTPUT :



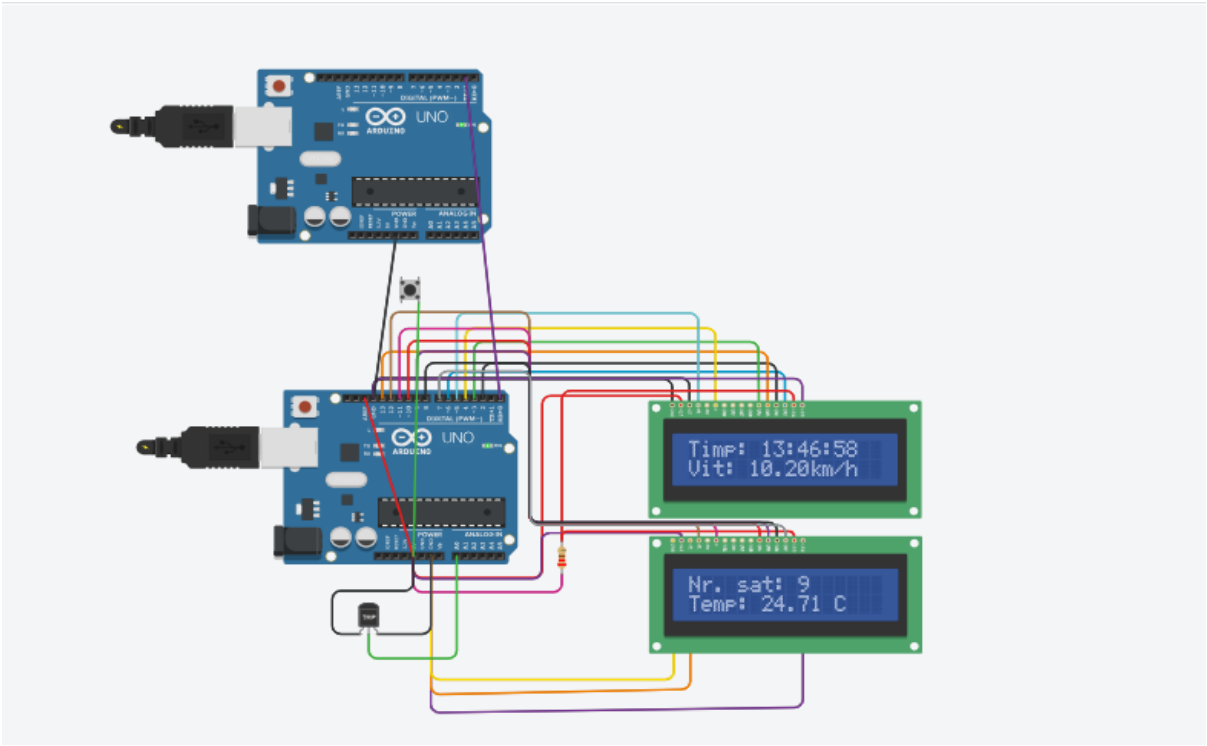
Practical No 7 Arduino GPS Module Interfacing

Aim : To Arduino GPS Module Interfacing

Code :

```
char text1[] = "$GPVTG,054.7,T,034.4,M,005.5,N,010.2,K";
char text2[] = "$GPGGA,134658.00,5106.9792,N,11402.3003,W,2,09,1.0,1048.47,M,-16.27,M,08,AAAA*60";
void setup()
{
  Serial.begin(9600);
}
void loop()
{
  delay(200);
  Serial.write(text1);
  Serial.write("/");
  delay(500);
  Serial.write(text2);
  delay(500);
}
```

OUTPUT:



Practical No : 8 IoT based Web Controlled Home Automation using Arduino

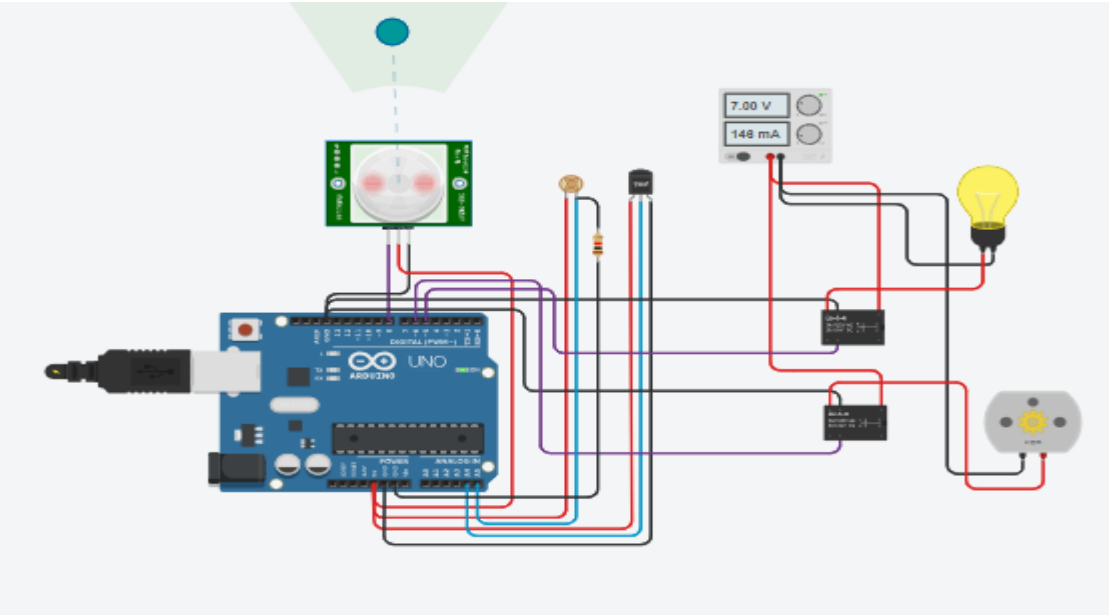
Aim : To IoT based Web Controlled Home Automation using Arduino

Code :

```
float x,y,z,temp;
void setup()
{
  pinMode(8, INPUT);
  pinMode(5, OUTPUT);
  pinMode(6, OUTPUT);
  pinMode(A5, INPUT);
  pinMode(A4, INPUT);
Serial.begin(9600);
}
void loop()
{
  x= digitalRead(8);
  y= analogRead(A5);
  z= analogRead(A4);
  Serial.println(x);
  Serial.println(y);
  Serial.println(z);
  temp = (double)z / 1024;
  temp = temp * 5;
  temp = temp - 0.5;
  temp = temp * 100;
  if ( (x>0) )
  {
    if ((y<550)&&(temp>30))
    {
      digitalWrite(5, HIGH);
      digitalWrite(6, HIGH);
    }
    else if((y<550)&&(temp<30))
    {
```

```
    digitalWrite(5, HIGH);
    digitalWrite(6, LOW);
   }
   else if((y>550)&&(temp>30))
   {
    digitalWrite(5, LOW);
    digitalWrite(6, HIGH);
   }
   else if((y>550)&&(temp<30))
   {
    digitalWrite(5, LOW);
    digitalWrite(6, LOW);
   }
  }
  else
  {
   digitalWrite(5, LOW);
   digitalWrite(6, LOW);
  }
}
```

OUTPUT:



Practical No : 10 Interfacing Arduino with RFID.

Aim : To Interfacing Arduino with RFID.

Code :

```
#include <Keypad.h>
#include <LiquidCrystal.h>
LiquidCrystal lcd(5, 4, 3, 2, A4, A5);
struct student_detail{
  String name;
  String regno;
  int status;
};
student_detail student_registered[30];
void student_registered_database()
{
  student_registered[0]={"A0","18BLC0000",0};
  student_registered[1]={"A1","18BLC0001",0};
  student_registered[2]={"A2","18BLC0002",0};
  student_registered[3]={"A3","18BLC0003",0};
  student_registered[4]={"A4","18BLC0004",0};
  student_registered[5]={"A5","18BLC0005",0};
  student_registered[6]={"A6","18BLC0006",0};
  student_registered[7]={"A7","18BLC0007",0};
  student_registered[8]={"A8","18BLC0008",0};
  student_registered[9]={"A9","18BLC0009",0};
  student_registered[10]={"A10","18BLC0010",0};
  student_registered[11]={"A11","18BLC0011",0};
  student_registered[12]={"A12","18BLC0012",0};
  student_registered[13]={"A13","18BLC0013",0};
  student_registered[14]={"A14","18BLC0014",0};
  student_registered[15]={"A15","18BLC0015",0};
  student_registered[16]={"A16","18BLC0016",0};
  student_registered[17]={"A17","18BLC0017",0};
  student_registered[18]={"A18","18BLC0018",0};
  student_registered[19]={"A19","18BLC0019",0};
```

```
  student_registered[20]={"A20","18BLC0020",0};
  student_registered[21]={"A21","18BLC0021",0};
  student_registered[22]={"A22","18BLC0022",0};
  student_registered[23]={"A23","18BLC0023",0};
  student_registered[24]={"A24","18BLC0024",0};
  student_registered[25]={"A25","18BLC0025",0};
  student_registered[26]={"A26","18BLC0026",0};
  student_registered[27]={"A27","18BLC0027",0};
  student_registered[28]={"A28","18BLC0028",0};
  student_registered[29]={"A29","18BLC0029",0};
}
int find(String regno){
  int status=-1;
  for(int i=0;i<20;i++)
  {
    if(regno==student_registered[i].regno)
    {
      status=i;
    }
  }
  return status;
}
void setup(){
   Serial.begin(9600);
   lcd.begin(16, 2);
   lcd.setCursor(0,0);
  student_registered_database();
}
void loop()
{
  int status=-1;
  if(Serial.available()>0)
  {
    String reg_no=Serial.readString();
```

```
    status=find(reg_no);
    if(status>=0 && status<=29)
    {
      lcd.clear();
      lcd.setCursor(0,0);
      lcd.print(student_registered[status].name);
      lcd.setCursor(0,1);
      lcd.print(student_registered[status].regno);
      delay(5000);
      student_registered[status].status=1;
    }
    else{
      lcd.clear();
      lcd.setCursor(0,0);
      lcd.print("not found");
      delay(5000);
    }
  }
  else{
    lcd.clear();
    lcd.setCursor(0,0);
    lcd.print("Give your regno");
    delay(5000);
  }
}
```

OUTPUT: