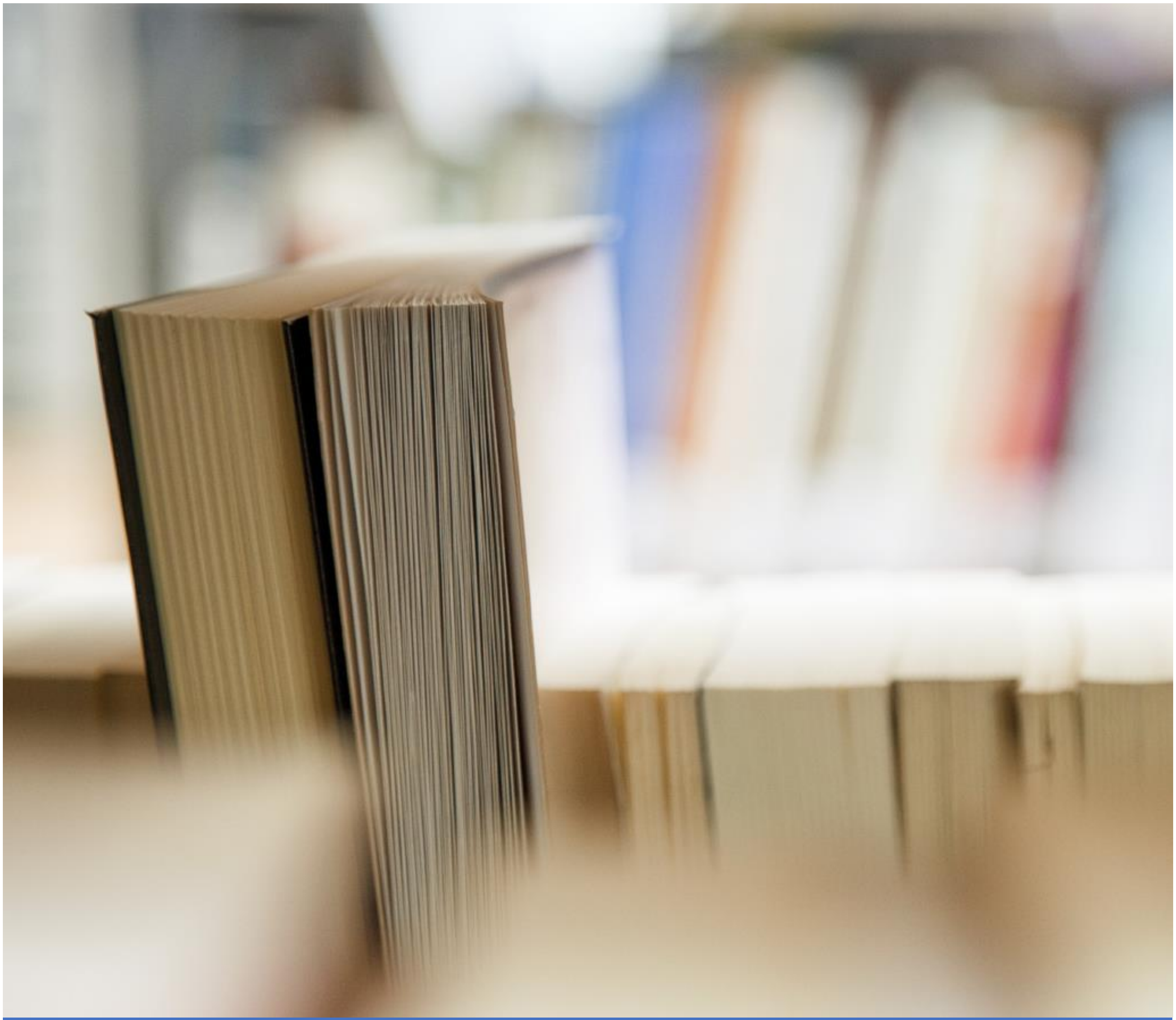


ARD VER 1.0

User Guide

Ideaxa 2/9/20 Arduino Training

IDEAXA PRESENTS

<u>ARDUINO TRAINNING KIT</u>.

<u>ARD VER 1.0</u>

<u>User Guide</u>

Table of contents –

Introduction of Kit-

This Kit is design for the beginner and advance level students. It has already some I/O peripherals listed below and easy to use for beginner.

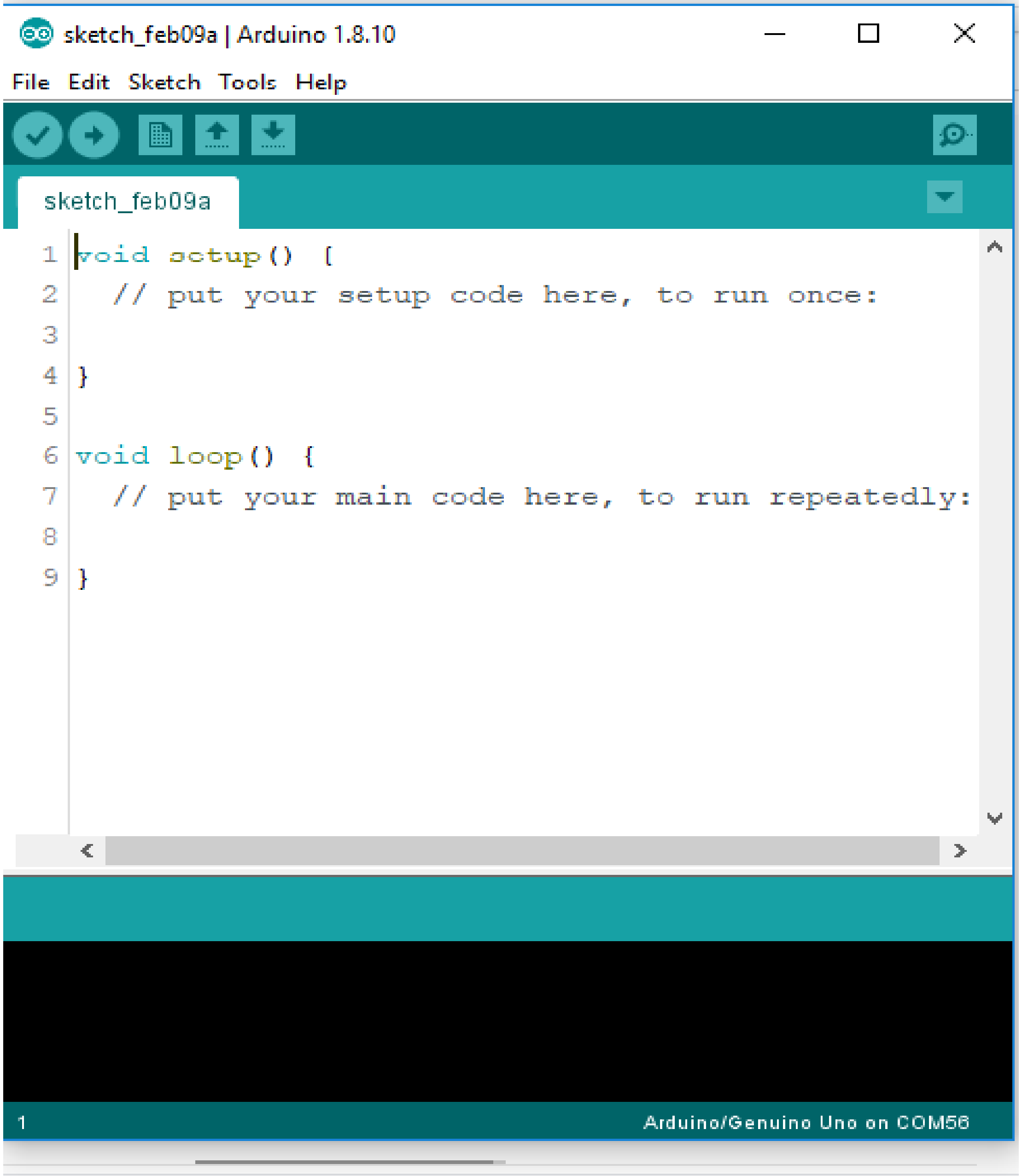
Peripherals in the kit-

Sr.no	Peripheral Name	Arduino Pin
1	Led Simple	D13
2	RGB Led	D9, D10, D11
3	Buzzer	D2
4	DHT11 (Temperature & humidity sensor)	D4
5	LDR (Light Sensor)	A0
6	TSOP 1838 (IR receiver)	D3
7	7 Segment (Common Cathode)	D9, D910, D11, D12
8	LCD 16x2 (I2C)	Address (0X27)
9	OLED Display (I2C)	Address (0X3C)
10	Buttons	D5, D6, D7, D8

Setup the software –

This kit contains **Arduino Uno R3**, For Uno R3 we need to install a software for firmware writing. The software name is **Arduino IDE**. You can download this software from below link.

- Link for Arduino IDE software - **https://www.arduino.cc/en/main/software**

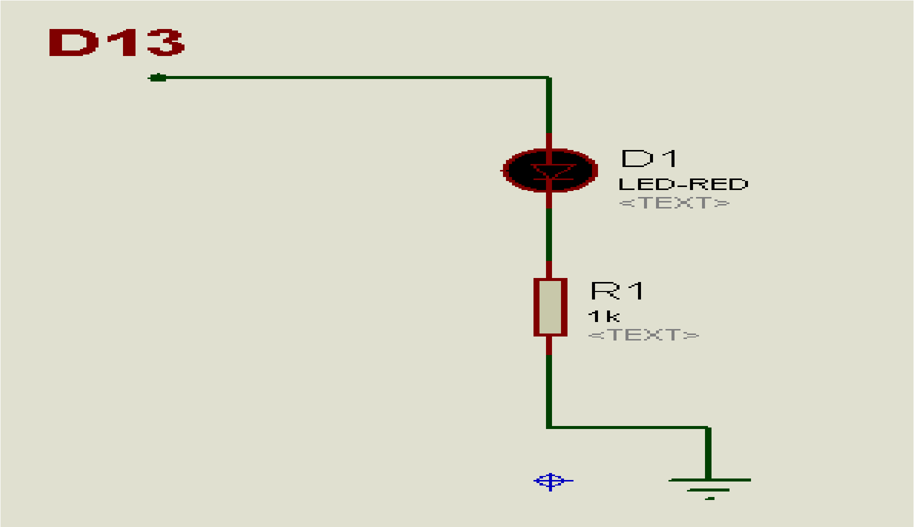


```
void setup() {
  // put your setup code here, to run once:

}

void loop() {
  // put your main code here, to run repeatedly:

}
```

1 Arduino/Genuino Uno on COM56

Led Blinking Tutorial

Circuit diagram –

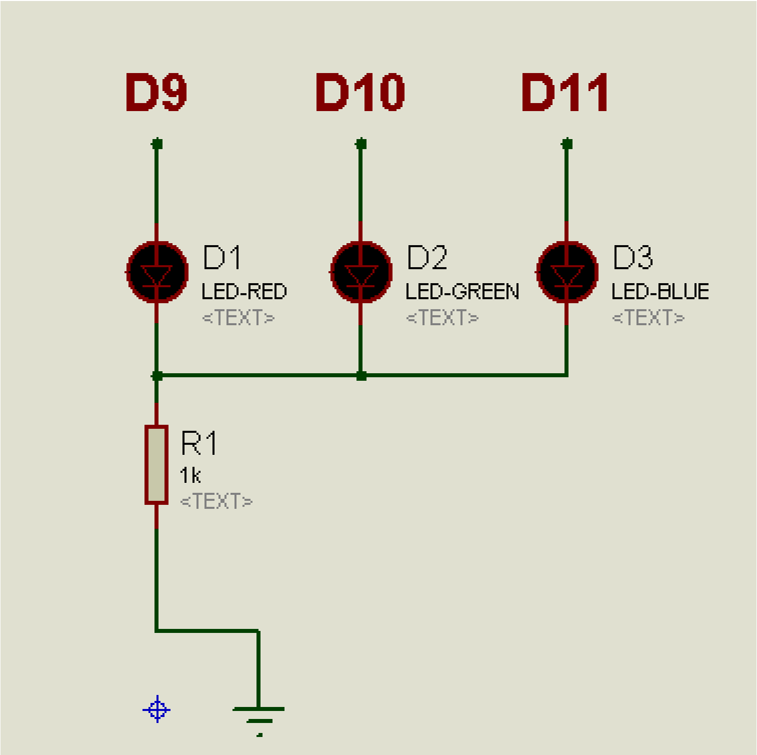


Arduino Code –

```
#define LED 13       // define led (our board has LED connected on 13 pin)
void setup() // setup loop for initialize all the devices (only one time)
{
  pinMode(LED, OUTPUT); // making LED as a output device
}
void loop() // main loop for main working task (Run again and again)
{
  digitalWrite(LED, HIGH); // Turn on the LED
  delay(1000);             // Delay for 1 sec
  digitalWrite (LED, LOW); // Turn off the LED
  delay (1000);            // Delay for 1 sec
}
```

RGB Led tutorial –

Circuit Diagram-



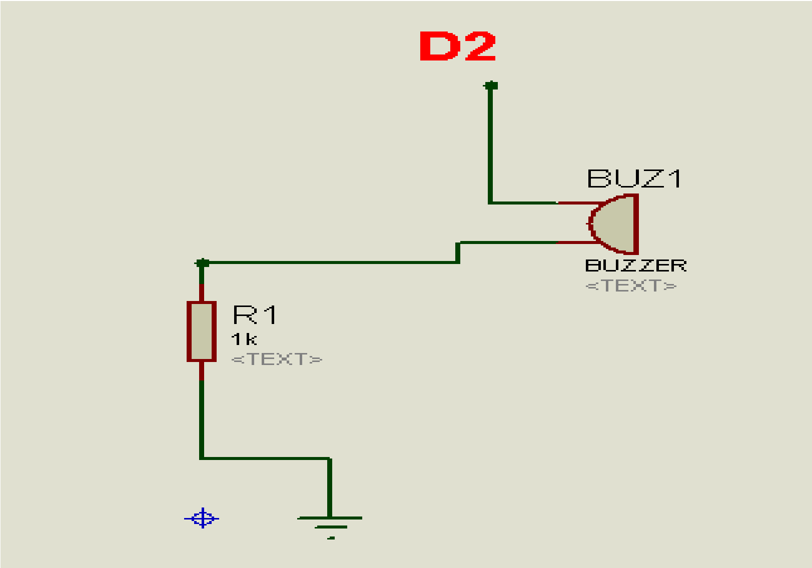
Arduino Code –

```
#define LED_R 9       // define led red (our board has LED connected on 9 pin)
#define LED_G 10      // define led green (our board has LED connected on 10 pin)
#define LED_B 11      // define led blue (our board has LED connected on 11 pin)

void setup() // setup loop for initialize all the devices (only one time)
{
  pinMode(LED_R, OUTPUT); // making LED R as a output device
  pinMode(LED_G, OUTPUT); // making LED G as a output device
  pinMode(LED_B, OUTPUT); // making LED B as a output device
}
```

```
void loop() // main loop for main working task (Run again and again)
{
 digitalWrite(LED_R, HIGH); // Turn on the LED
 delay(1000);               // Delay for 1 sec
 digitalWrite(LED_R, LOW);  // Turn off the LED
 delay(1000);               // Delay for 1 sec
 digitalWrite(LED_G, HIGH); // Turn on the LED
 delay(1000);               // Delay for 1 sec
 digitalWrite(LED_G, LOW);  // Turn off the LED
 delay(1000);               // Delay for 1 sec
 digitalWrite(LED_B, HIGH); // Turn on the LED
 delay(1000);               // Delay for 1 sec
 digitalWrite(LED_B, LOW);  // Turn off the LED
 delay(1000);               // Delay for 1 sec
}
```

Buzzer Tutorial –

Circuit Diagram –

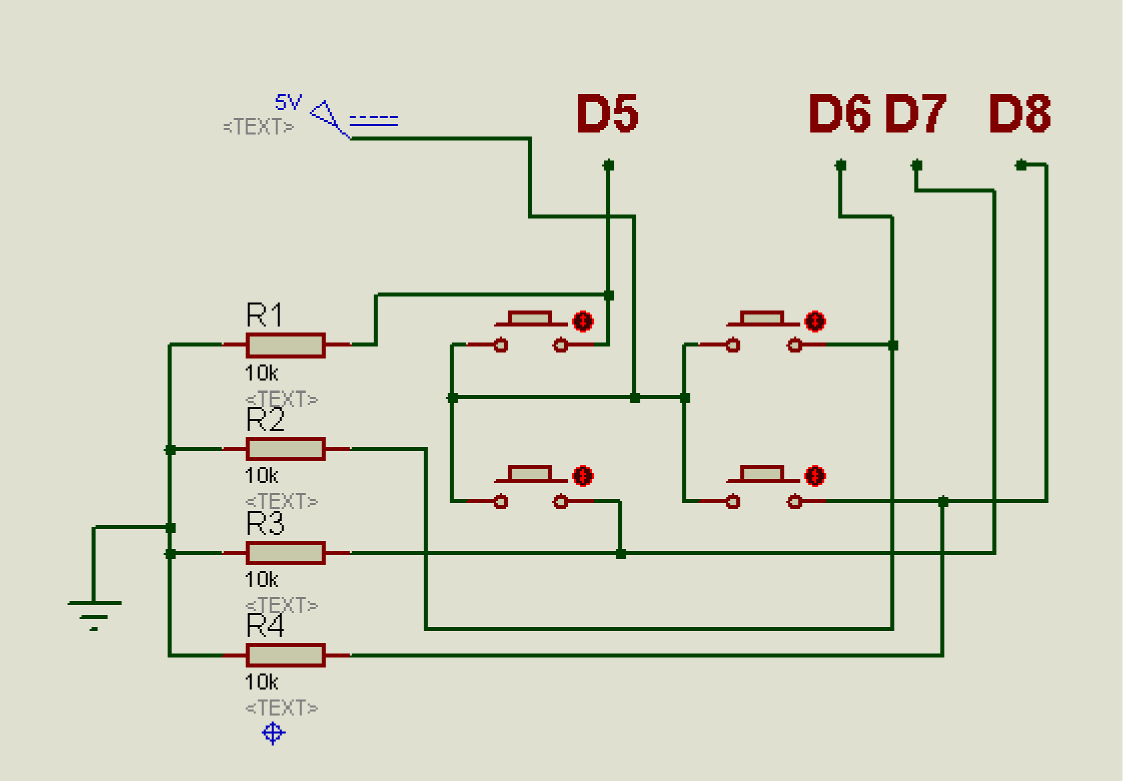


Arduino Code –

```
#define BUZZER 2       // define BUZZER (our board has BUZZER connected on 2 pin)

void setup() // setup loop for initialize all the devices (only one time)

{

  pinMode(BUZZER, OUTPUT); // making BUZZER as a output device

}

void loop() // main loop for main working task (Run again and again)

{

  digitalWrite(BUZZER, HIGH); // Turn on the BUZZER

  delay(1000);             // Delay for 1 sec

  digitalWrite(BUZZER, LOW);  // Turn off the BUZZER

  delay(1000);             // Delay for 1 sec

}
```

Buttons Tutorial –

Circuit Diagram –

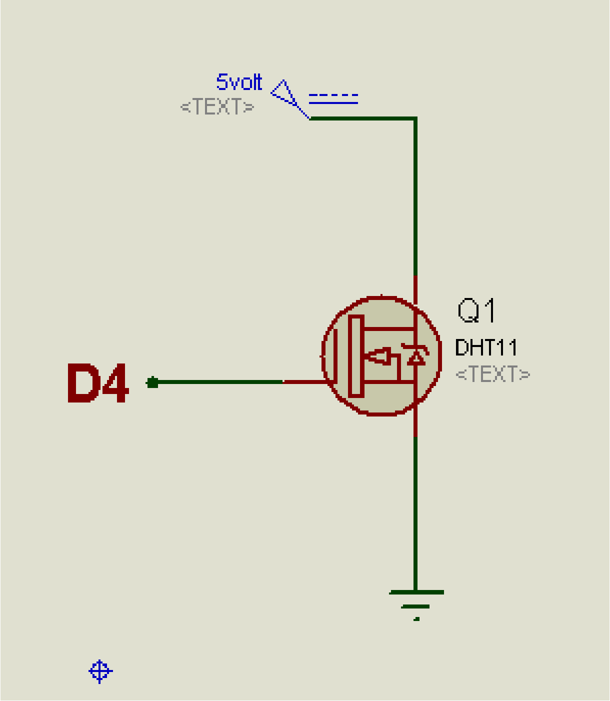


Arduino Code –

```
const int buttonPin1 = 5;     // the number of the pushbutton pin 5
const int buttonPin2 = 6;     // the number of the pushbutton pin 6
const int buttonPin3 = 7;     // the number of the pushbutton pin 7
const int buttonPin4 = 8;     // the number of the pushbutton pin 8

int buttonState1 = 0;         // variable for reading the pushbutton on pin 5
int buttonState2 = 0;         // variable for reading the pushbutton on pin 6
int buttonState3 = 0;         // variable for reading the pushbutton on pin 7
int buttonState4 = 0;         // variable for reading the pushbutton on pin 8
void setup() {
  // initialize the pushbuttons pin as an input:
  pinMode(buttonPin1, INPUT);
  pinMode(buttonPin2, INPUT);
  pinMode(buttonPin3, INPUT);
  pinMode(buttonPin4, INPUT);
  Serial.begin(9600); // init serial for print button data
}

void loop() {
  // read the state of the pushbutton value:
  buttonState1 = digitalRead(buttonPin1);
  buttonState2= digitalRead(buttonPin2);
  buttonState3 = digitalRead(buttonPin3);
  buttonState4 = digitalRead(buttonPin4);

  // when we press button state go high
  if (buttonState1 == HIGH)
  {
    Serial.println("BUTTON 1 PRESSED");
    delay(100);
  }
  if (buttonState2 == HIGH)
  {
    Serial.println("BUTTON 2 PRESSED");
    delay(100);
  }
  if (buttonState3 == HIGH)
  {
    Serial.println("BUTTON 3 PRESSED");
    delay(100);
  }
  if (buttonState4 == HIGH)
  {
    Serial.println("BUTTON 4 PRESSED");
    delay(100);
  }
  else {
    // do nothing
  }
}
```

DHT11 Tutorial –

Circuit diagram –



Arduino Code –

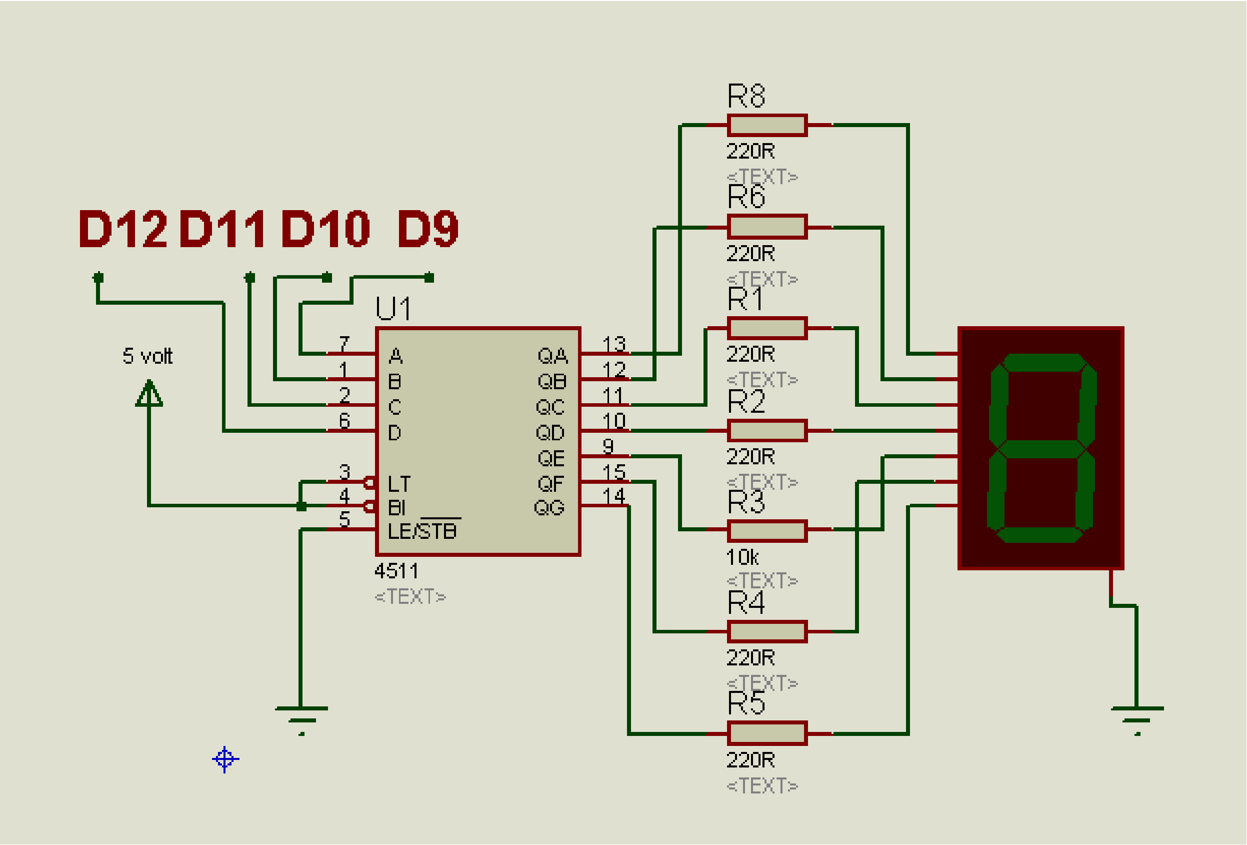
```
// DHT Temperature & Humidity Sensor
// Unified Sensor Library Example
// Written by Tony DiCola for Adafruit Industries
// Released under an MIT license.

// REQUIRES the following Arduino libraries:
// - DHT Sensor Library: https://github.com/adafruit/DHT-sensor-library
// - Adafruit Unified Sensor Lib: https://github.com/adafruit/Adafruit_Sensor

#include <Adafruit_Sensor.h>
```

```
#include <DHT.h>

#include <DHT_U.h>

#define DHTPIN 4 // define dhtpin

#define DHTTYPE DHT11 // define dht type

DHT_Unified dht(DHTPIN, DHTTYPE);

uint32_t delayMS;

void setup() {

  Serial.begin(9600);

  // Initialize device.

  dht.begin();

  Serial.println(F("DHTxx Unified Sensor Example"));

  // Print temperature sensor details.

  sensor_t sensor;

 dht.temperature().getSensor(&sensor);

 Serial.println(F("------------------------------------"));

 Serial.println(F("Temperature Sensor"));

Serial.print (F("Sensor Type: "));

 Serial.println(sensor.name);

Serial.print(F("Driver Ver:  "));

Serial.println(sensor.version);

Serial.print(F("Unique ID:   "));

Serial.println(sensor.sensor_id);

Serial.print(F("Max Value:   "));

Serial.print(sensor.max_value);
Serial.println(F("°C"));
 Serial.print(F("Min Value:   "));
Serial.print(sensor.min_value);
Serial.println(F("°C"));
Serial.print(F("Resolution:  "));
Serial.print(sensor.resolution); Serial.println(F("°C"));
  Serial.println(F("------------------------------------"));
  // Print humidity sensor details.
  dht.humidity().getSensor(&sensor);
  Serial.println(F("Humidity Sensor"));
  Serial.print  (F("Sensor Type: "));
Serial.println(sensor.name);
  Serial.print  (F("Driver Ver:  "));
Serial.println(sensor.version);
  Serial.print  (F("Unique ID:   "));
 Serial.println(sensor.sensor_id);
  Serial.print  (F("Max Value:   "));
Serial.print(sensor.max_value); Serial.println(F("%"));
  Serial.print  (F("Min Value:   "));
Serial.print(sensor.min_value); Serial.println(F("%"));
  Serial.print  (F("Resolution:  "));
Serial.print(sensor.resolution); Serial.println(F("%"));
  Serial.println(F("------------------------------------"));
  // Set delay between sensor readings based on sensor details.
```

```
delayMS = sensor.min_delay / 1000;
}
void loop() {
 // Delay between measurements.
 delay(delayMS);
 // Get temperature event and print its value.
 sensors_event_t event;
 dht.temperature().getEvent(&event);
 if (isnan(event.temperature)) {
  Serial.println(F("Error reading temperature!"));
 }
 else {
  Serial.print(F("Temperature: "));
  Serial.print(event.temperature);
  Serial.println(F("°C"));
 }
 // Get humidity event and print its value.
 dht.humidity().getEvent(&event);
 if (isnan(event.relative_humidity)) {
  Serial.println(F("Error reading humidity!"));
 }
 else {
  Serial.print(F("Humidity: "));
  Serial.print(event.relative_humidity);Serial.println(F("%")); }}
```

7 Segment tutorial –

Circuit diagram –



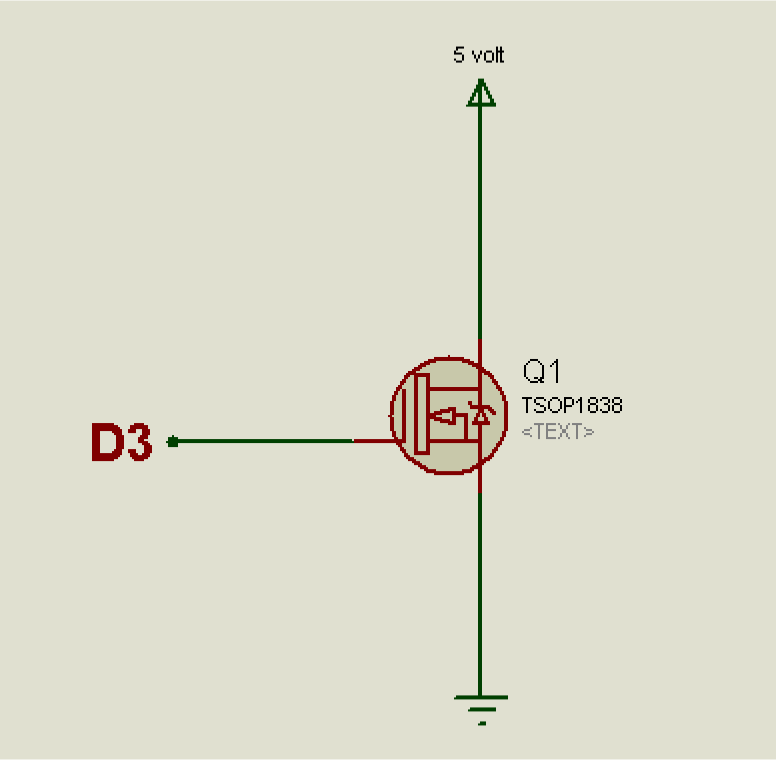
Arduino Code –

```
const int A=9; // 7 segment driver cd4511 pins

const int B=10;

const int C=11;

const int D=12;

void setup() {

  pinMode(A, OUTPUT); //LSB

  pinMode(B, OUTPUT);

  pinMode(C, OUTPUT);

  pinMode(D, OUTPUT); //MSB
```

```
}
void loop()
{
digitalWrite(D, LOW); // 0 digit
digitalWrite(C, LOW);
digitalWrite(B, LOW);
digitalWrite(A, LOW);
delay(1000);
digitalWrite(D, LOW); // 1 digit
digitalWrite(C, LOW);
digitalWrite(B, LOW);
digitalWrite(A, HIGH);
delay(1000);
digitalWrite(D, LOW); // 2 digit
digitalWrite(C, LOW);
digitalWrite(B, HIGH);
digitalWrite(A, LOW);
delay(1000);
digitalWrite(D, LOW); // 3 digit
digitalWrite(C, LOW);
digitalWrite(B, HIGH);
digitalWrite(A, HIGH);
delay(1000);
digitalWrite(D, LOW);   // 4 digit
digitalWrite(C, HIGH);
digitalWrite(B, LOW);
digitalWrite(A, LOW);
delay(1000);
digitalWrite(D, LOW);   // 5 digit
digitalWrite(C, HIGH);
digitalWrite(B, LOW);
digitalWrite(A, HIGH);
delay(1000);
digitalWrite(D, LOW);   // 6 digit
digitalWrite(C, HIGH);
digitalWrite(B, HIGH);
digitalWrite(A, LOW);
delay(1000);
digitalWrite(D, LOW);   // 7 digit
digitalWrite(C, HIGH);
digitalWrite(B, HIGH);
digitalWrite(A, HIGH);
delay(1000);
digitalWrite(D, HIGH); // 8 digit
digitalWrite(C, LOW);
digitalWrite(B, LOW);
digitalWrite(A, LOW);
delay(1000);
digitalWrite(D, HIGH);  // 9 digit
digitalWrite(C, LOW);
digitalWrite(B, LOW);
digitalWrite(A, HIGH);
delay(1000);
}
```

TSOP tutorial -

Circuit diagram –



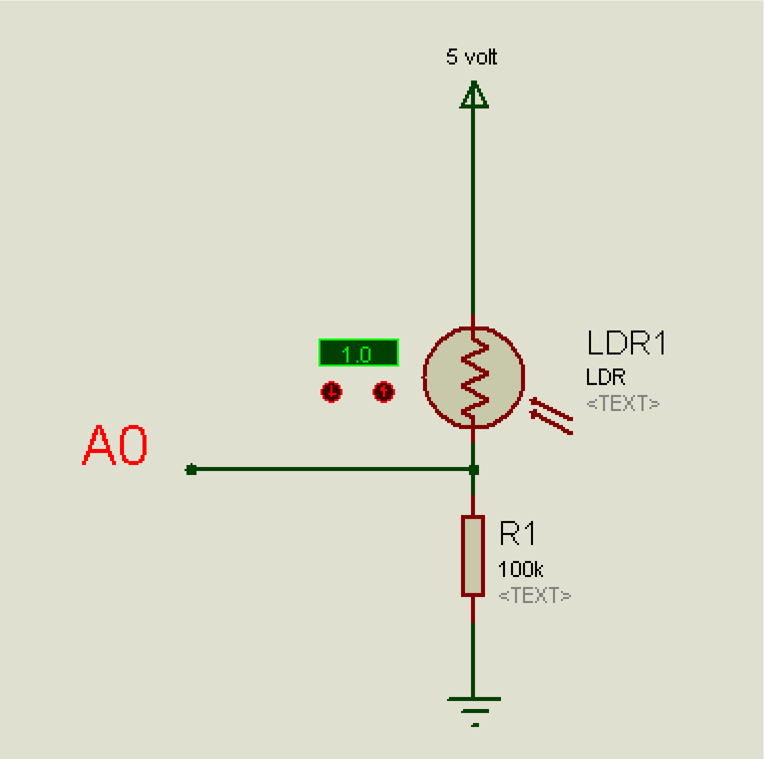
Arduino Code –

```
// this example is for read the value of each button IR code by tsop1838
#include <IRremote.h> // install library for IR TSOP 1838
int RECV_PIN = 3;  // our board has TSOP connected on pin 3
IRrecv irrecv(RECV_PIN); // object of IRremote class
decode_results results;
```

```
void setup() {
  Serial.begin(9600);
  irrecv.enableIRIn();// enabling the IR pin
}
void loop() {
  if (irrecv.decode(&results)) {
    Serial.println(results.value, HEX);
    irrecv.resume();
  }
  delay(100);
}
```

Light Sensor tutorial –

Circuit diagram –



Arduino Code –

```
int sensorPin = A0;    // select the input pin for the LIGHT SENSOR
int sensorValue = 0;  // variable to store the value coming from the sensor
void setup() {
  // declare the Serial for print data
  Serial.begin(9600);
}
void loop() {
  // read the value from the sensor:
  sensorValue = analogRead(sensorPin);
  Serial.println(sensorValue);// print the data from light sensor.
  delay(100);
}
```

LCD 16X2 tutorial –

NOTE - LCD is connected to I2C pins of Arduino UNO (A5,A4).

Arduino code –

```
// install required library
#include <LiquidCrystal_I2C.h>
LiquidCrystal_I2C lcd(0x27,16,2);  // set the LCD address to 0x27 for a 16 chars and 2 line display(address is 0x27 and a  16x2 lcd attached)
void setup()
{
  lcd.init();   // initialize the lcd
  // Print a message to the LCD.
  lcd.backlight();
  lcd.setCursor(0,0); // set cursor to 1st row
  lcd.print("Hello, world!"); // print data in 1st row
  lcd.setCursor(2,1); // set cursor to 2nd row
  lcd.print("IDEAXA");// print data in 2nd row

}
void loop()
{
//you can write your code here
}
```

OLED display tutorial –

NOTE – OLED display 1306 128x32 is connected to I2C pins of Arduino Uno (A5,A4).

Arduino code –

The code for this tutorial is bit large, you can go to our **GITHUB** page for this code.

Operate Kit with external Power –

You can operate kit with battery or a power adopter available in the market. We have added a screw terminal for battery and a dc power jack for adaptor you can use a power supply from 6 to 15 volt for this kit.

Parameters of the board -

The Board parameter of the power is below listed.

- Max 15-volt, Min 6 volt
- Operating temperature is -10 to 55 degree Celsius
- Chip Used ATMEGA328
- Frequency 16mhz
- Digital I/O 16 pins
- Analog input pins 6
- Dc current per pin 20mA
- Flash memory 32KB
- SRAM 2KB
- EEPROME 1KB
- Board measurement 23x17 cm

Extra shield use in the kit.

We can use any other modules, sensors, peripherals in the kit. For this we give a dedicated space and Arduino headers. You can use them. whenever you want to use some extra modules with this kit make sure to disable all non-required on-board modules.

Enabling & Disabling the onboard module –

We have dedicated jumpers for enabling and disabling the onboard modules.

- For enabling each module, you have to join the terminal with a jumper socket.
- For disabling each module, you have to remove the jumper socket from the board.

Required Library for all on board modules –

- DHT sensor (https://github.com/adafruit/DHT-sensor-library)
- IR remote (https://github.com/z3t0/Arduino-IRremote)
- Adafruit sensor (https://github.com/adafruit/Adafruit_Sensor)
- LCD 16x2 I2c (https://github.com/fdebrabander/Arduino-LiquidCrystal-I2C-library)
- Adafruit GFX (https://github.com/adafruit/Adafruit-GFX-Library)
- Adafruit SSD1306 (https://github.com/adafruit/Adafruit_SSD1306)

References –

- Cad soft Eagle
- www.Arduino.cc
- www.ladyada.com
- www.tmeeducation.com
- www.electronicshub.org
- www.hackr.io
- www.programmingelectronics.com/