

List of Hardware

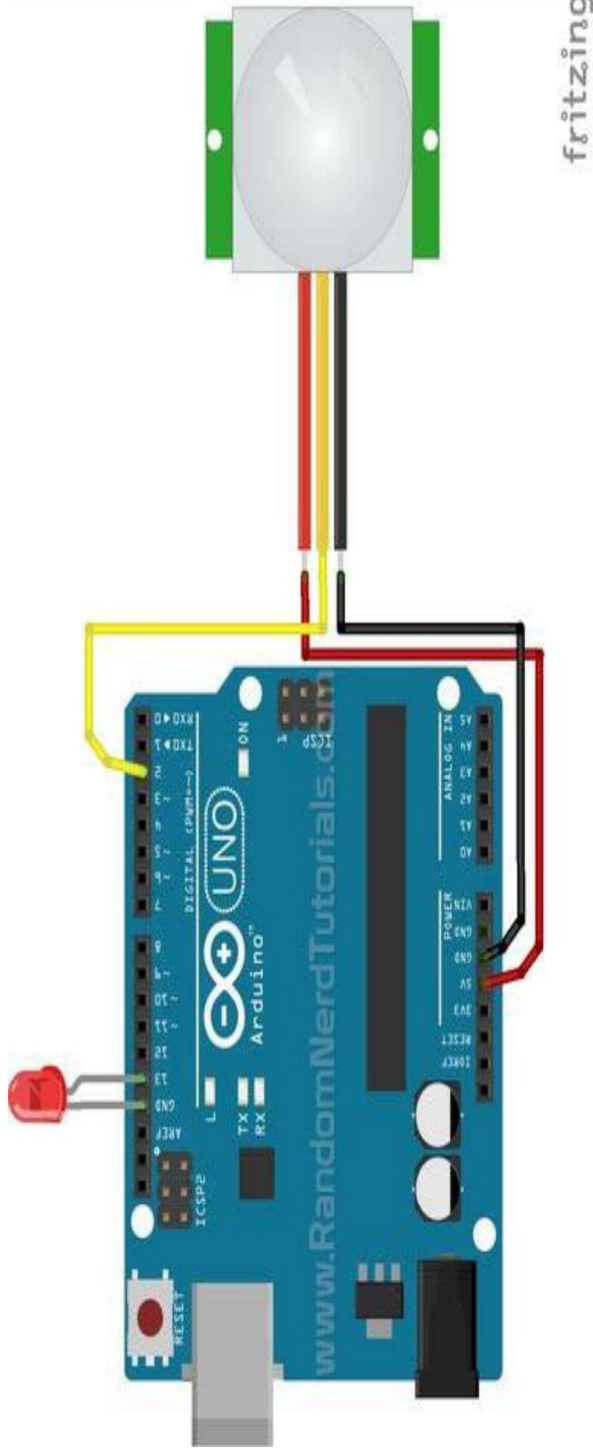
Major	Minor
PIR Sensor (Motion Sensor)	Mega Arduino Board
IR Sensor	Multimeter
Ultrasonic Sensor	Resistors
Flame Sensor	Jumper wires
Gas Sensor	Breadboard
RFID Sensor	Switches
EM-18 RFID Reader Module	
ESP8266	
GSM Module	
I2C LCD	
LCD Display	

PIR Sensor with Arduino

PIR sensor has three terminals: V_{cc} , OUT, and GND. Connect the sensor as follows:

- Connect the $+V_{cc}$ to +5v on Arduino board.
- Connect OUT to digital pin 3 on Arduino board.
- Connect GND with GND on Arduino.
- Connect LED in between the Pin no. 13 and GND.

- Circuit Magic.com



PIR SENSOR

```
//the time we give the sensor to calibrate (10-60 secs according to the datasheet)
int calibrationTime = 30;

//the time when the sensor outputs a low impulse
long unsigned int lowIn;

//the amount of milliseconds the sensor has to be low
//before we assume all motion has stopped
long unsigned int pause = 5000;

boolean locklow = true;
boolean takeLowTime;

int pirPin = 3;    //the digital pin connected to the PIR sensor's output
int ledPin = 13;

//////////////////////////
//SETUP
void setup(){
  Serial.begin(9600);
  pinMode(pirPin, INPUT);
  pinMode(ledPin, OUTPUT);
  digitalWrite(pirPin, LOW);

  //give the sensor some time to calibrate
  Serial.print("calibrating sensor ");
  for(int i = 0; i < calibrationTime; i++){
    Serial.print(".");
    delay(1000);
  }
  Serial.println(" done");
  Serial.println("SENSOR ACTIVE");
  delay(50);
}
```

PIR SENSOR (contd..)

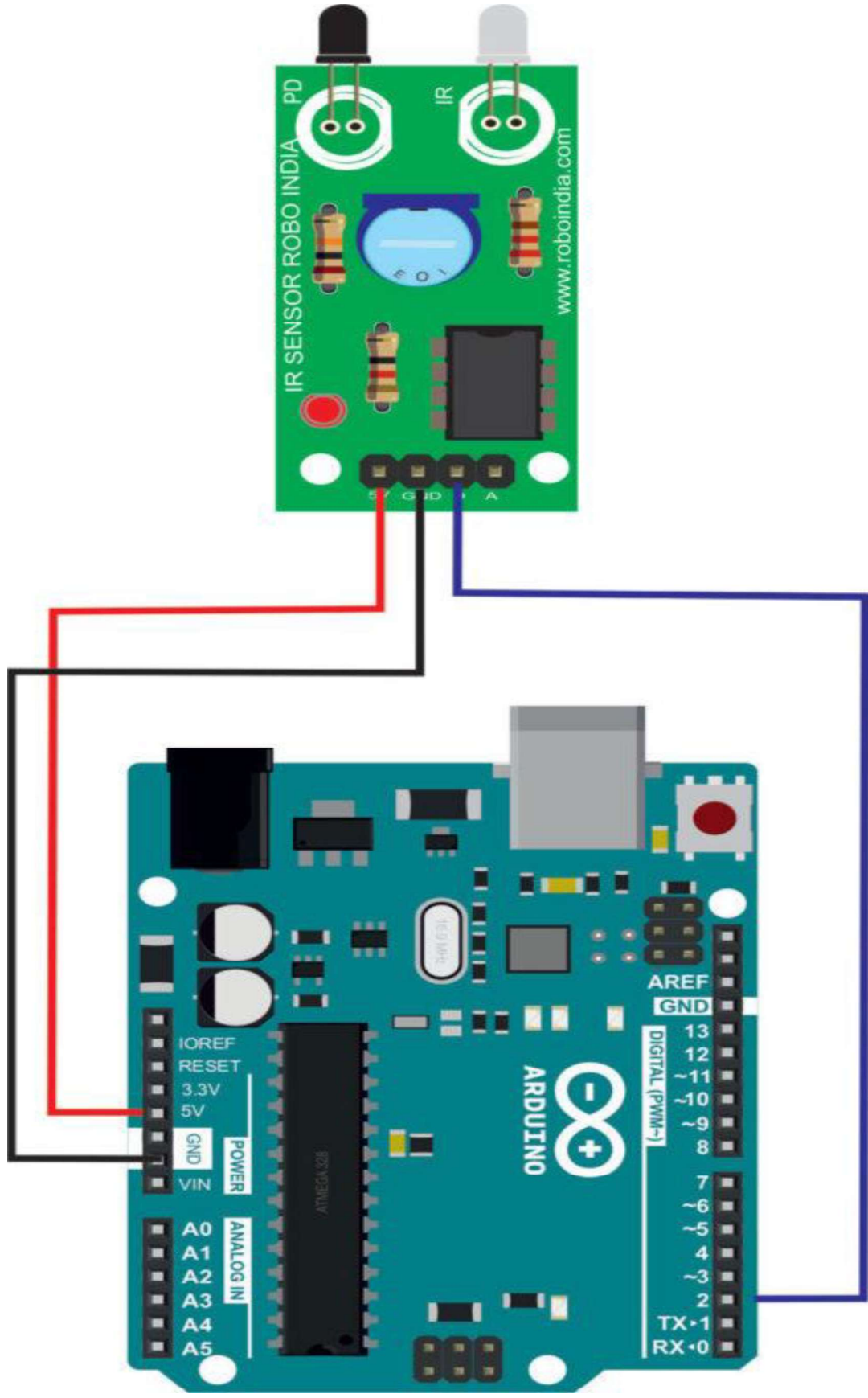
```
////////////////////////////////////
//LOOP
void loop(){

    if(digitalRead(pirPin) == HIGH){
        digitalWrite(ledPin, HIGH); //the led visualizes the sensors output pin state
        if(lockLow){
            //makes sure we wait for a transition to LOW before any further output is made:
            lockLow = false;
            Serial.println("---");
            Serial.print("motion detected at ");
            Serial.print(millis()/1000);
            Serial.println(" sec");
            delay(50);
        }
        takeLowTime = true;
    }

    if(digitalRead(pirPin) == LOW){
        digitalWrite(ledPin, LOW); //the led visualizes the sensors output pin state

        if(takeLowTime){
            lowIn = millis();
            takeLowTime = false;
        }
        //if the sensor is low for more than the given pause,
        //we assume that no more motion is going to happen
        if(!lockLow && millis() - lowIn > pause){
            //makes sure this block of code is only executed again after
            //a new motion sequence has been detected
            lockLow = true;
            Serial.print("motion ended at "); //output
            Serial.print((millis() - pause)/1000);
            Serial.println(" sec");
            delay(50);
        }
    }
}
```

IR Sensor



IR Sensor

<https://www.youtube.com/watch?v=nF8z7RcEulk>

https://www.youtube.com/watch?v=SKJGARYRGwQ&fbclid=IwAR2o8wrfZBCrg2SJAfmaGjhpLxU_vToE1yZVdIVQoWQOEQgVCULtuPvmxzQ

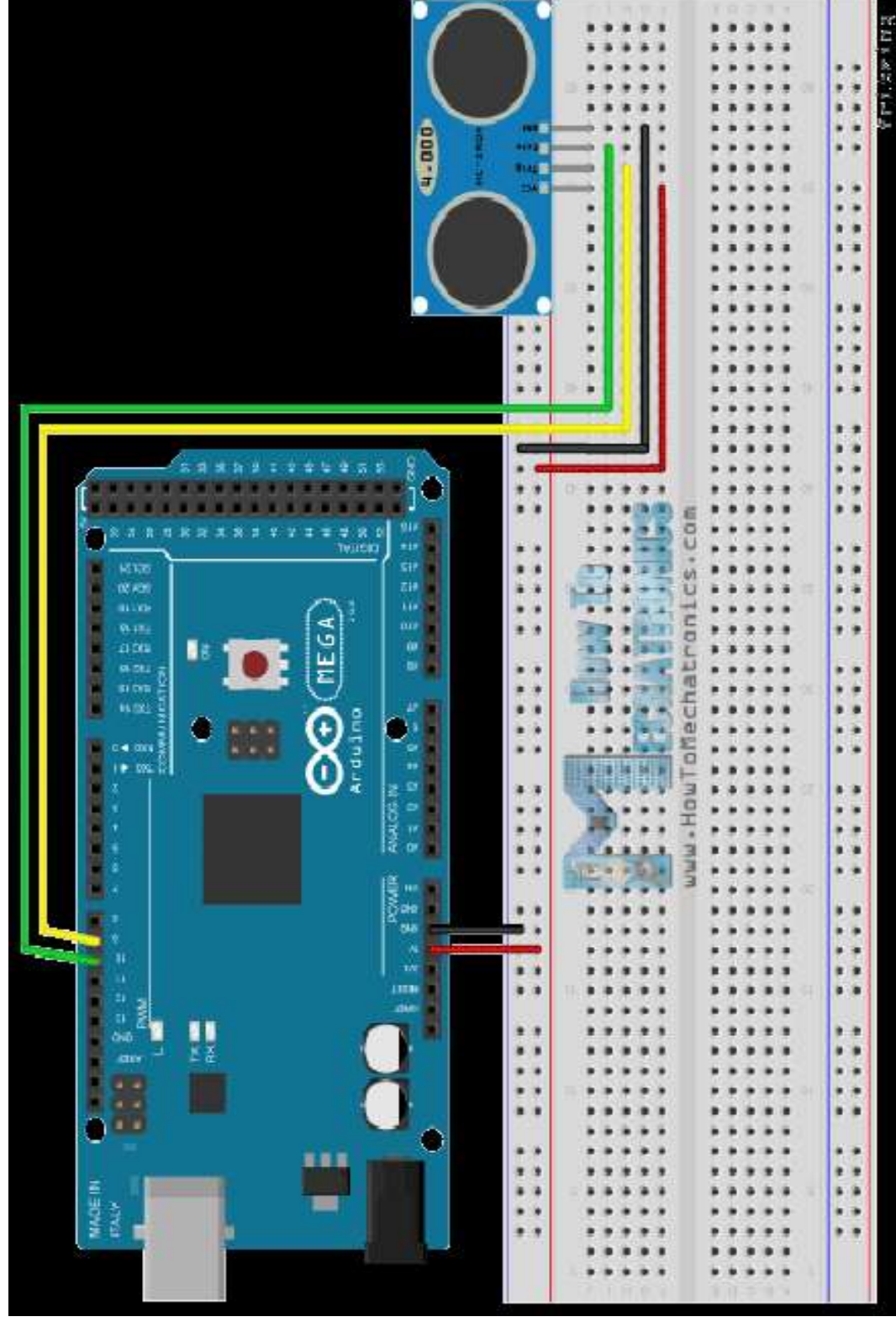
```
void setup() {  
  pinMode(7, INPUT);  
  Serial.begin(9600);  
  pinMode(13, OUTPUT);  
}  
  
void loop() {  
  Serial.print("IRSensor: ");  
  Serial.println(digitalRead(7));  
  
  if(digitalRead(7) == 0)  
  {  
    digitalWrite(13, HIGH);  
  }  
  else{  
    digitalWrite(13, LOW);  
  }  
}
```


Ultrasonic Sensor

- The transmitter transmits ultrasonic pulses & they are reflected back & gets sensed by the receiver if any obstacle lies between the path. The range of this ultrasonic is about 4 metres. There is a chance of error of +-3cm in measuring the distance. It measures the time interval between sending & receiving the pulse & then by a formula – gives us the distance.
- It has got 4 pins :-
 - VCC – connect it to 5V supply.
 - GND – connect it to ground.
 - echopin, trigpin – connect it to any digital pin (as of here, we've connected them to 9, 10).
- & the positive terminal of the LED is connected to pin number 7 on Arduino .

<http://mechstuff.com/connection-interfacing-programming-of-ultrasonic-sensor-hc-sr04/>

Ultrasonic Sensor



Ultrasonic sensor code

```
int trigPin = 9;
int echoPin = 10;
int led = 7;

void setup() {
  Serial.begin(9600);
  pinMode(led, OUTPUT);
  pinMode(trigPin, OUTPUT);
  pinMode(echoPin, INPUT);
  // put your setup code here, to run once:
}

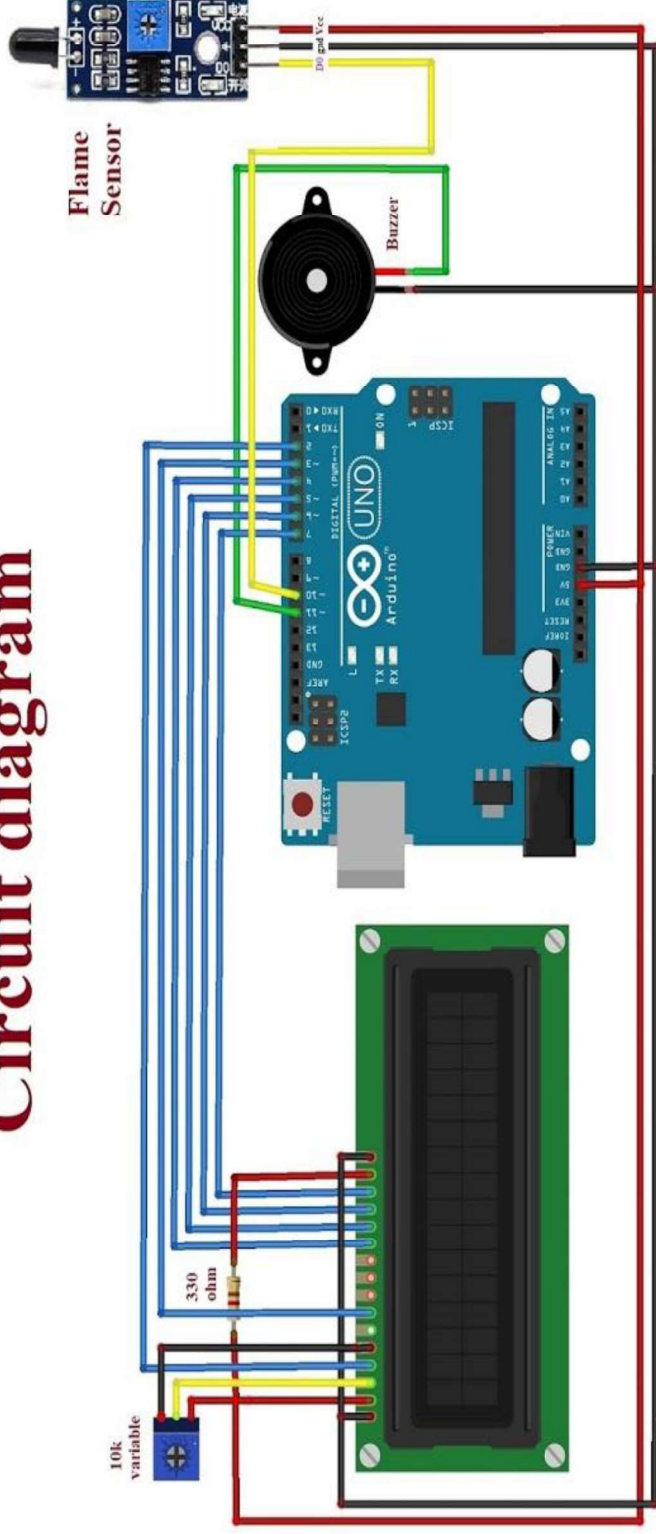
void loop() {
  long duration, distance;
  digitalWrite(trigPin, HIGH);
  delayMicroseconds(1000);
  digitalWrite(trigPin, LOW);
  duration=pulseIn(echoPin, HIGH);
  distance =(duration/2)/29.1;
  Serial.println(distance);
  Serial.println("CM");
  delay(10);

  if (distance<=10)
  {
    digitalWrite(led, HIGH);
  }
  else if (distance>10)
  {
    digitalWrite(led, LOW);
  }
}
```

Flame Sensor

- <https://www.youtube.com/watch?v=OglhextacLk>

Circuit diagram



Flame Sensor

```
#include <LiquidCrystal.h>
LiquidCrystal lcd(2, 3, 4, 5, 6, 7);

#define flamePin 10
#define buzzerPin 11

void setup() {
  Serial.begin(9600);
  lcd.begin(16, 2);

  pinMode(buzzerPin, OUTPUT);
  pinMode(flamePin, INPUT);

  lcd.setCursor(0, 0);
  lcd.print("Calibrating");
  for(int i = 0; i < 15; i++) {
    if (i--4)
    {
      lcd.setCursor(0, 1);
      lcd.print(".");
    }
    else lcd.print(".");
    delay(500);
  }
  lcd.setCursor(11, 1);
  lcd.print("Done");
  delay(1000);
  lcd.clear();
  lcd.setCursor(1, 0);
  lcd.print("Sensor Active");
  delay(1500);
  lcd.clear();
}

void loop() {
  int Flame = digitalRead(flamePin);

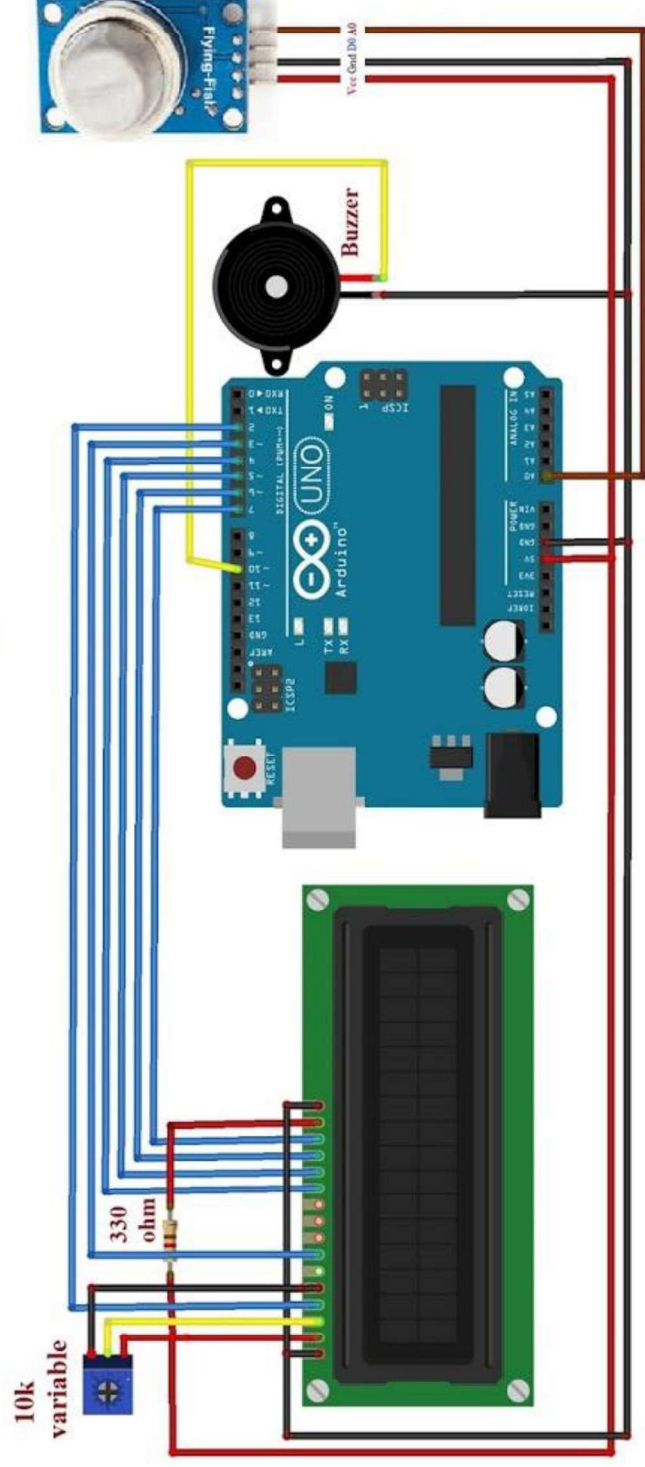
  if (Flame == LOW)
  {
    digitalWrite(buzzerPin, HIGH);
    lcd.setCursor(0, 0);
    lcd.print(" Flame : ");
    lcd.print("Flame");
    lcd.setCursor(0, 1);
    lcd.print(" is Detected");
    Serial.print(Flame);
    Serial.print("\t");
    Serial.print("Flame is Detected");
  }
  else if (Flame == HIGH)
  {
    digitalWrite(buzzerPin, LOW);
    lcd.setCursor(0, 0);
    lcd.print("Flame : ");
    lcd.print("No Flame");
    Serial.print(Flame);
    Serial.print("\t");
    Serial.println("No Flame");
  }

  delay(300);
  lcd.clear();
}
```

Gas Sensor

<https://www.youtube.com/watch?v=pCECQEZ147E>

Circuit Diagram



Gas Sensor

```
#include <LiquidCrystal.h>
LiquidCrystal lcd(2, 3, 4, 5, 6, 7);
#define buzzerPin 10
#define gasPin A0

void setup() {
  Serial.begin(9600);
  lcd.begin(16, 2);

  pinMode(buzzerPin, OUTPUT);

  lcd.setCursor(0, 0);
  lcd.print("Calibrating");
  for(int i = 0; i < 10; i++){
    if (i==4)
    {
      lcd.setCursor(0, 1);
      lcd.print(".");
    }
    else lcd.print(".");
    delay(500);
  }
  lcd.setCursor(5, 1);
  lcd.print("done");
  delay(1000);
  lcd.clear();
  lcd.setCursor(1, 0);
  lcd.print("SENSOR ACTIVE");
  delay(1500);
  lcd.clear();
}
```

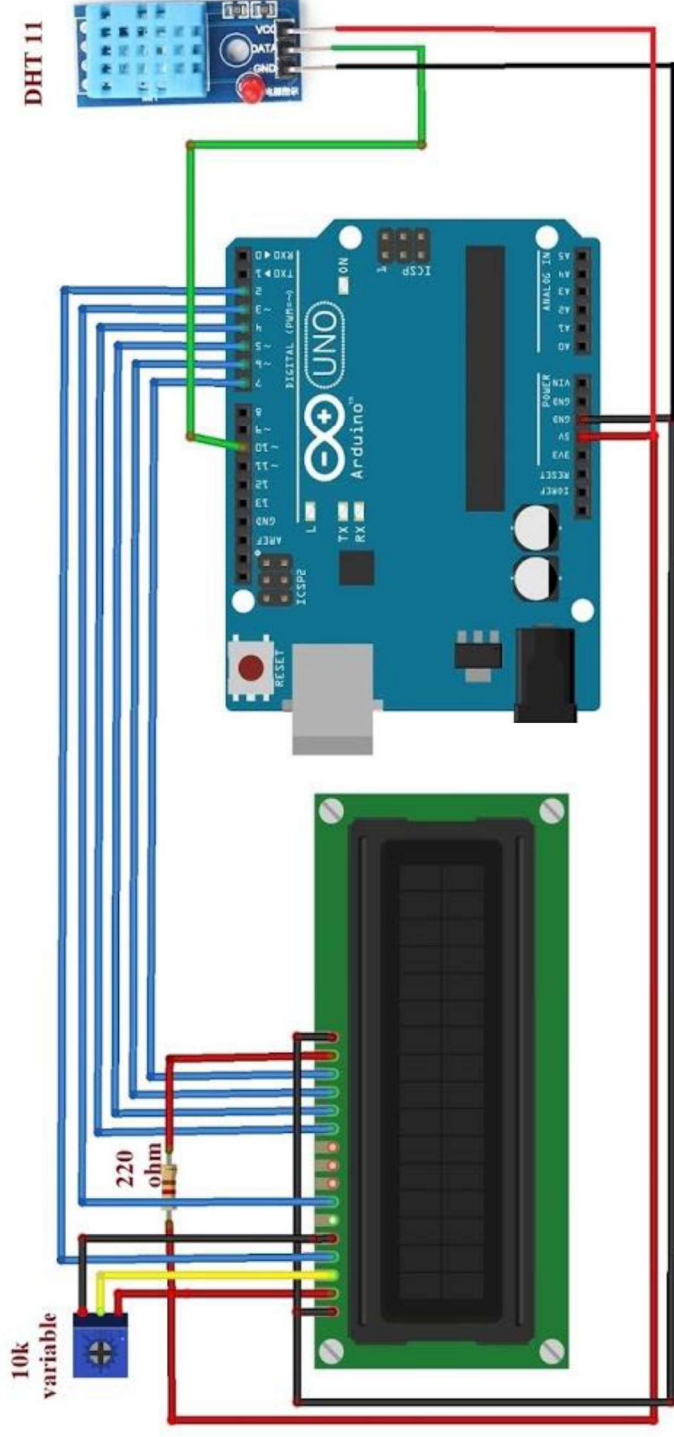
```
void loop() {
  int gasSensor = analogRead(gasPin);

  if (gasSensor > 350)
  {
    digitalWrite(buzzerPin, HIGH);
    lcd.setCursor(0, 0);
    lcd.print("Value : ");
    lcd.print(gasSensor);
    Serial.print(gasSensor);
    Serial.print("\t");
    lcd.setCursor(0, 1);
    Serial.println("Gas is Detected");
    lcd.print("Gas is Detected");
    delay(300);
    lcd.clear();
  }
  else if (gasSensor < 350)
  {
    digitalWrite(buzzerPin, LOW);
    lcd.setCursor(0, 0);
    lcd.print("Value : ");
    lcd.print(gasSensor);
    Serial.print(gasSensor);
    Serial.print("\t");
    lcd.setCursor(0, 1);
    Serial.println("No Gas");
    lcd.print("No Gas");
    delay(300);
  }
}
```


Temperature and Humidity Sensor (DHT11)

<https://www.youtube.com/watch?v=G VyabySFkFI>

Circuit Diagram



Temperature and Humidity Sensor (DHT11)

```
#include <Adafruit_Sensor.h>
#include <DHT.h>
#include <DHT_U.h>
#include <LiquidCrystal.h>
const int rs = 2, en = 3, d4 = 4, d5 = 5, d6 = 6, d7 = 7;
LiquidCrystal lcd(rs, en, d4, d5, d6, d7);

#define DHTPIN 10 // Pin which is connected to the DHT sensor.

#define DHTTYPE DHT11 // DHT 11

DHT_Unified dht(DHTPIN, DHTTYPE);

uint32_t delayMS;

void setup() {
  Serial.begin(9600);
  lcd.begin(16, 2);
  // Initialize device.
  dht.begin();
  Serial.println("DHTxx Unified Sensor Example");
  // Print temperature sensor details.
  sensor_t sensor;
  dht.temperature().getSensor(&sensor);

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

  delayMS = sensor.min_delay / 1000;

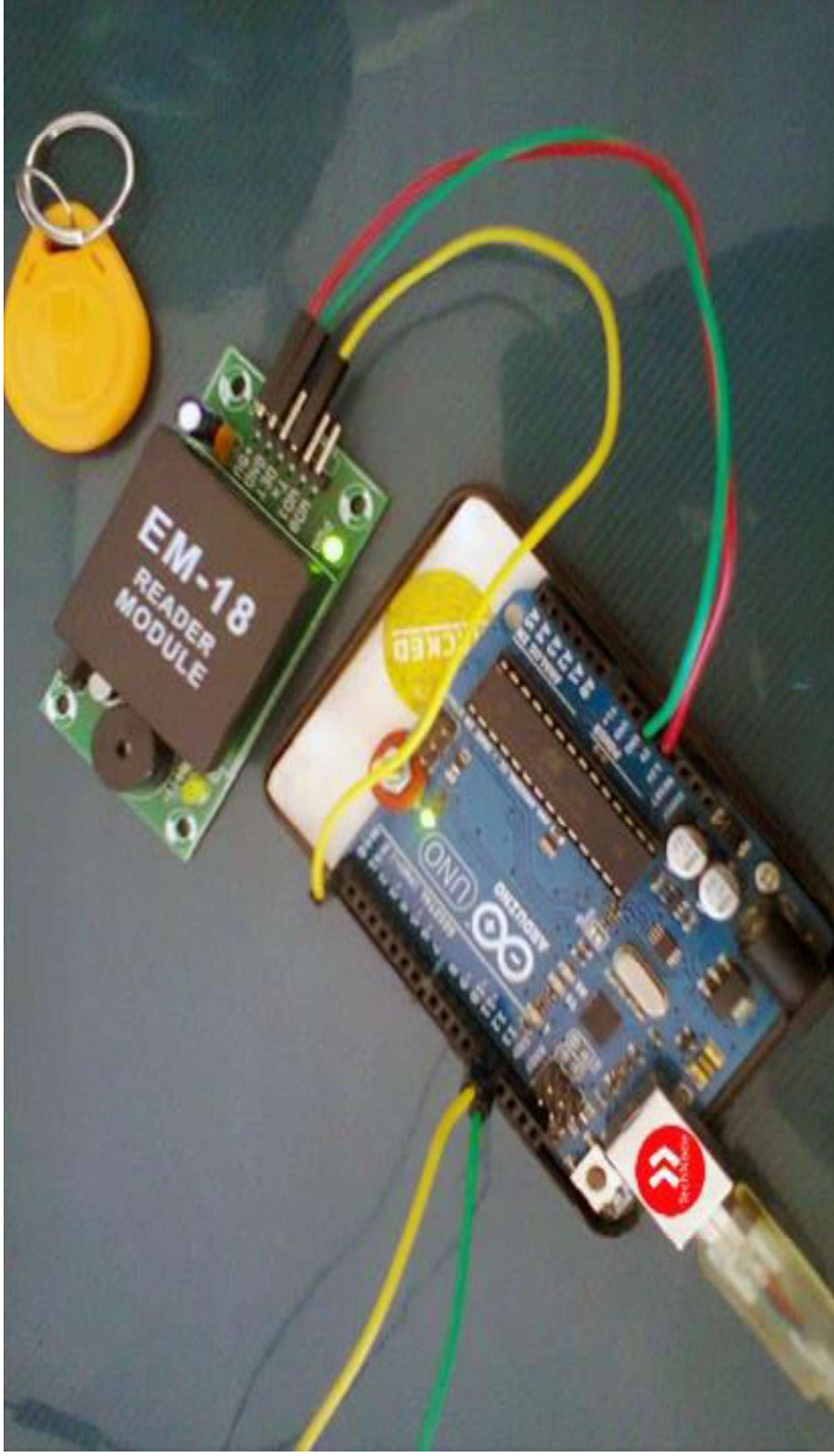
  lcd.setCursor(0, 0);
  lcd.print("Calibrating");
  for(int i = 0; i < 10; i++){
    if (i==4)
    {
      lcd.setCursor(0, 1);
      lcd.print(".");
    }
    else lcd.print(".");
    delay(500);
  }
  lcd.setCursor(5, 1);
  lcd.print("done");
  delay(1000);
  lcd.clear();
  lcd.setCursor(1, 0);
  lcd.print("SENSOR ACTIVE");
  delay(1500);
}
```

Temperature and Humidity Sensor (DHT11)

```
void loop() {
  // Delay between measurements.
  delay(delayMS);
  lcd.clear();
  // Get temperature event and print its value.
  sensors_event_t event;
  dht.temperature().getEvent(&event);
  if (!isnan(event.temperature)) {
    Serial.println("Error reading temperature!");
  }
  else {
    lcd.setCursor(0, 0);
    lcd.print("Temp : ");
    Serial.print("Temperature: ");
    Serial.print(event.temperature);
    lcd.print(event.temperature);
    Serial.println(" °C");
    lcd.write(0xdf); // for degree sign
    lcd.print("°C ");
  }
  // Get humidity event and print its value.
  dht.humidity().getEvent(&event);
  if (!isnan(event.relative_humidity)) {
    Serial.println("Error reading humidity!");
  }
  else {
    lcd.setCursor(0, 1);
    lcd.print("Humidity: ");
    Serial.print("Humidity: ");
    Serial.print(event.relative_humidity);
    lcd.print(event.relative_humidity);
    lcd.print("%");
    Serial.println("%");
  }
}
```

EM-18 RFID Reader Module

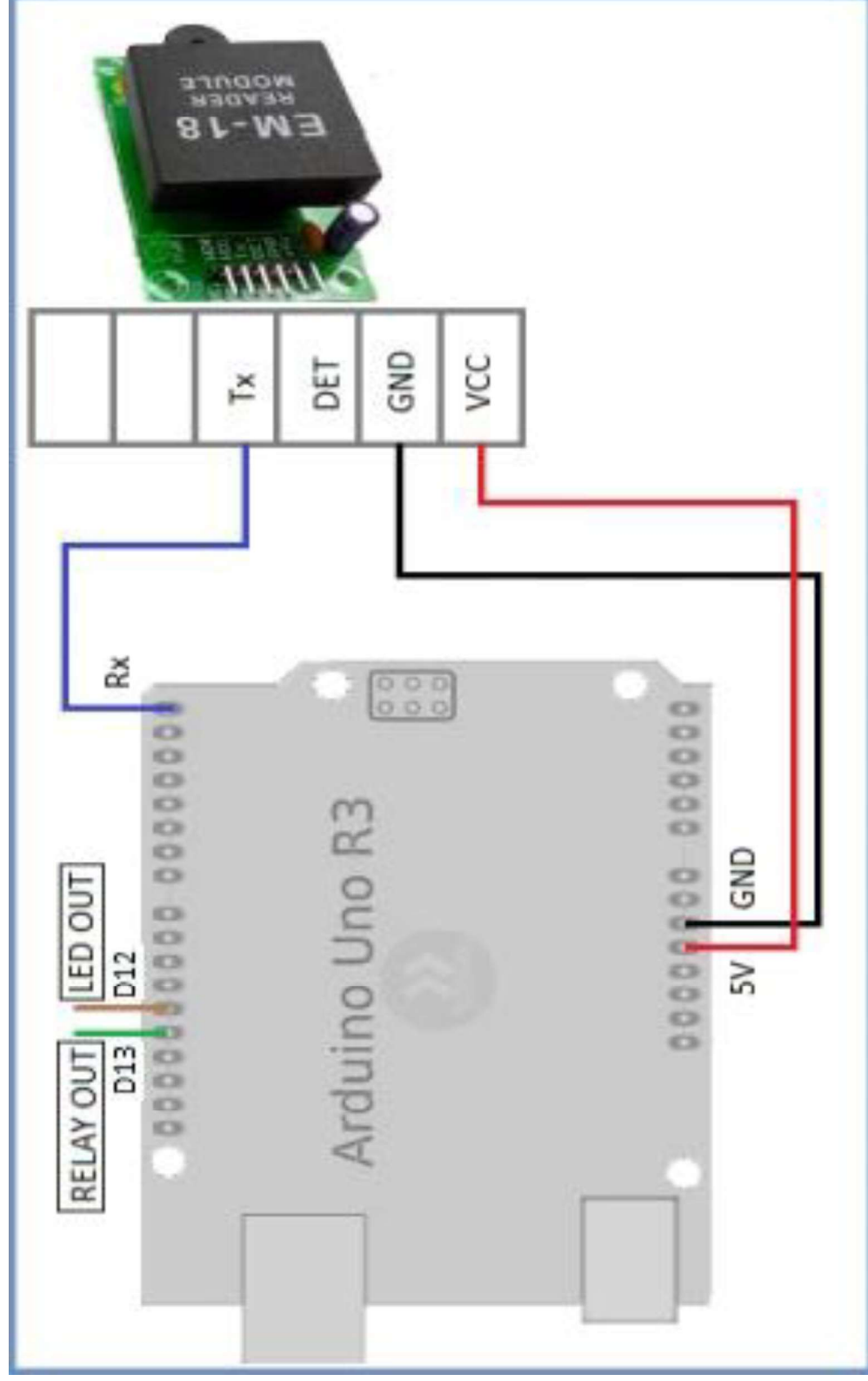
<https://www.electroschematics.com/arduino-rfid-access-control-em-18/>



EM-18 RFID Reader Module

```
int count = 0;
void setup()
{
    Serial.begin(9500);
}
void loop()
{
    if(Serial.available())
    {
        count = 0; // Reset count to zero// Keep reading Byte by Byte from the Buffer till the
        {
            char input = Serial.read();
            Serial.print(input);
            count++; //
            delay(5); //
        }
        Serial.println();
        Serial.print("Tag Length : ");
        Serial.print(count);
        Serial.println(" Bytes");
    }
}
```


EM-18 RFID Reader Module (Access Control)



EM-18 RFID Reader Module (Access Control)

```
/* Arduino Simple RFID Access Control
Using EM-18 RFID Reader Module
Prefatory Code For Novices
An inspired design. Thanks to Internet!
T.K.Hareendran
Project designed & tested at TechNode on 17.02.2014
https://www.electroschematics.com */
#define RELAYPIN 13
#define WARNLEDPIN 12
char tag[] = "51005F46642C"; // Replace with your own Tag ID
char input[12];
int count = 0;
boolean flag = 0;
void setup()
{
    Serial.begin(9600);
    pinMode(RELAYPIN,OUTPUT);
    pinMode(WARNLEDPIN,OUTPUT); //WRONG TAG INDICATOR
}
void loop()
{
    if(Serial.available())// Check if there is incoming data in the RFID Reader Serial Buffer.
    {
        count = 0; // Reset the counter to zero
        /* Keep reading Byte by Byte from the Buffer till the RFID Reader Buffer is empty
        or till 12 Bytes (the ID size of our Tag) is read */
        while(Serial.available() && count < 12)
        {
            input[count] = Serial.read(); // Read 1 Byte of data and store it in the input[
            count++; // increment counter
            delay(5);
        }
    }
}
```

EM-18 RFID Reader Module (Access Control)

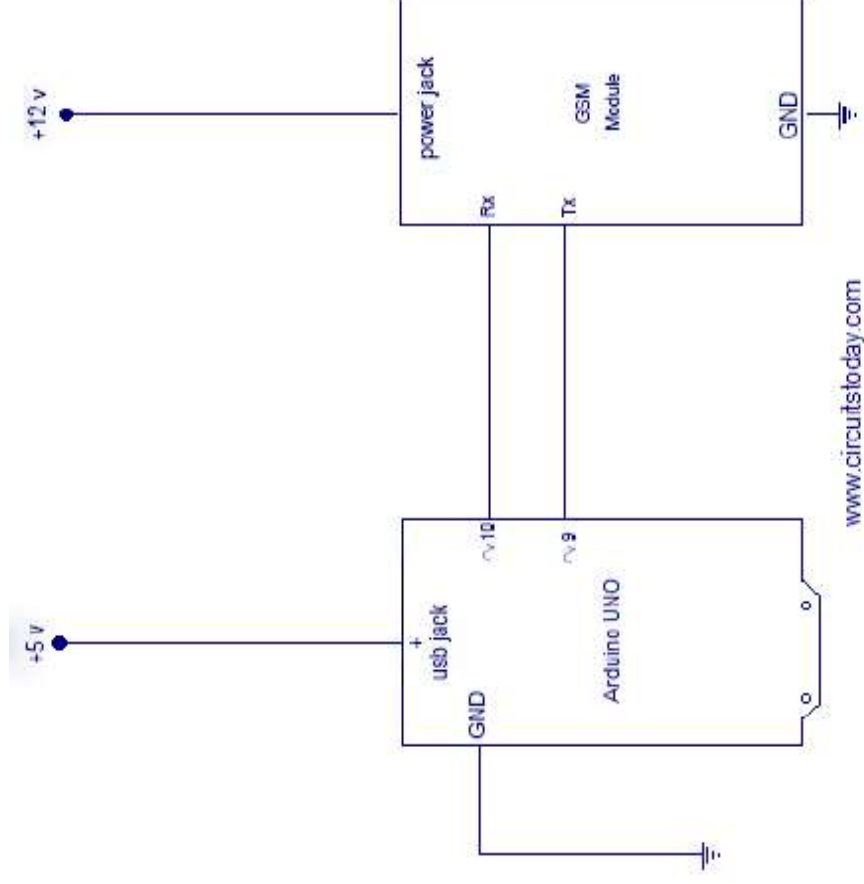
```
if(count == 12) //
{
    count = 0; // reset counter variable to 0
    flag = 1;
    /* Iterate through each value and compare till either the 12 values are
    all matching or till the first mismatch occurs */
    while(count<12 && flag !=0)
    {
        if(input[count]==tag[count])
            flag = 1; // everytime the values match, we set the flag variable to 1
        else
            flag= 0;
        /* if the ID values don't match, set flag variable to 0 and
        stop comparing by exiting the while loop */
        count++; // increment i
    }
}
if(flag == 1) // If flag variable is 1, then it means the tags match
{
    Serial.println("Access Allowed!");
    digitalWrite(RELAYPIN,HIGH);
    delay (5000);
    digitalWrite (RELAYPIN,LOW);
}
else
{
    Serial.println("Access Denied"); // Incorrect Tag Message
    digitalWrite(WARNLEDPIN,HIGH);
    delay(5000);
    digitalWrite(WARNLEDPIN,LOW);
}
/* Fill the input variable array with a fixed value 'F' to overwrite
all values getting it empty for the next read cycle */
for(count=0; count<12; count++)
{
```

EM-18 RFID Reader Module (Access Control)

```
}  
for(count=0; count<12; count++)  
{  
    input[count]='F';  
}  
count = 0; // Reset counter variable  
}
```

GSM800A with Arduino

<https://www.youtube.com/watch?v=sZxCpPYpkyY>



GSM800A with Arduino

<https://www.youtube.com/watch?v=sZxCpPYpkyY>

```
#include <SoftwareSerial.h>
SoftwareSerial mySerial(9, 10);

char msg;
char call;

void setup()
{
    mySerial.begin(9600);    // Setting the baud rate of GSM Module
    Serial.begin(9600);      // Setting the baud rate of Serial Monitor (Arduino)
    Serial.println("GSM SIM800A BEGIN");
    Serial.println("Enter character for control option:");
    Serial.println("h : to disconnect a call");
    Serial.println("i : to receive a call");
    Serial.println("s : to send message");
    Serial.println("r : to receive message");
    Serial.println("c : to make a call");
    Serial.println("e : to redial");
    Serial.println();
    delay(100);
}
```

GSM800A with Arduino

```
void loop()
{
    if (Serial.available()>0)
    switch(Serial.read())
    {
        case 's':
            SendMessage();
            break;
        case 'c':
            MakeCall();
            break;
        case 'h':
            HangupCall();
            break;
        case 'e':
            RedialCall();
            break;
        case 'i':
            ReceiveCall();
            break;
        case 'r':
            ReceiveMessage();
            break;
    }
}

void loop()
{
    if (mySerial.available()>0)
    Serial.write(mySerial.read());
}

void SendMessage()
{
    mySerial.println("AT+CMGF=1"); //Sets the GSM Module in Text Mode
    delay(1000); // Delay of 1000 milli seconds or 1 second
    mySerial.println("AT+CMGS="+Yxxxxxxxxxx"\r"); // Replace x with mobile number
    delay(1000);
    mySerial.println("sim800a sms");// The SMS text you want to send
    delay(100);
    mySerial.println((char)26);// ASCII code of CTRL+Z
    delay(1000);
}

void ReceiveMessage()
{
    mySerial.println("AT+CMML=2,2,0,0,0");// AT Command to recieve a live SMS
    delay(1000);
    if (mySerial.available()>0)
    {
        msg=mySerial.read();
        Serial.print(msg);
    }
}
```

GSM800A with Arduino

```
void MakeCall()
{
    mySerial.println("ATD+YYxxxxxxxxxx0;"); // ATDxxxxxxxxxx; -- watch out here for semicolon at the end!!
    Serial.println("Calling "); // print response over serial port
    delay(1000);
}

void HangupCall()
{
    mySerial.println("ATH");
    Serial.println("Hangup Call");
    delay(1000);
}

void ReceiveCall()
{
    mySerial.println("ATA");
    delay(1000);
    {
        call=mySerial.read();
        Serial.print(call);
    }
}

void RedialCall()
{
    mySerial.println("ATDL");
    Serial.println("Redialing");
    delay(1000);
}
```

I2C LCD

Now, with only 3 pins from microcontroller, you can display message on this LCD. Compared to parallel LCD which required at least 6 pins of I/O, this LCD offer more cost effective solution. The LCD display is four lines by 20 characters and provides basic text wrapping so that your text looks right on the display.

<https://www.instructables.com/id/How-to-Use-I2C-Serial-LCD-20X4-Yellow-Backlight/>



I2C Library

For this tutorial, it is necessary to download and install the "LiquidCrystal_I2C" library. LiquidCrystal_I2C is a library of Arduino which enables serial LCD 20x4 connect with Arduino. To be able to interface the serial LCD with Arduino, you will have to download this library and save it into your Arduino's libraries.

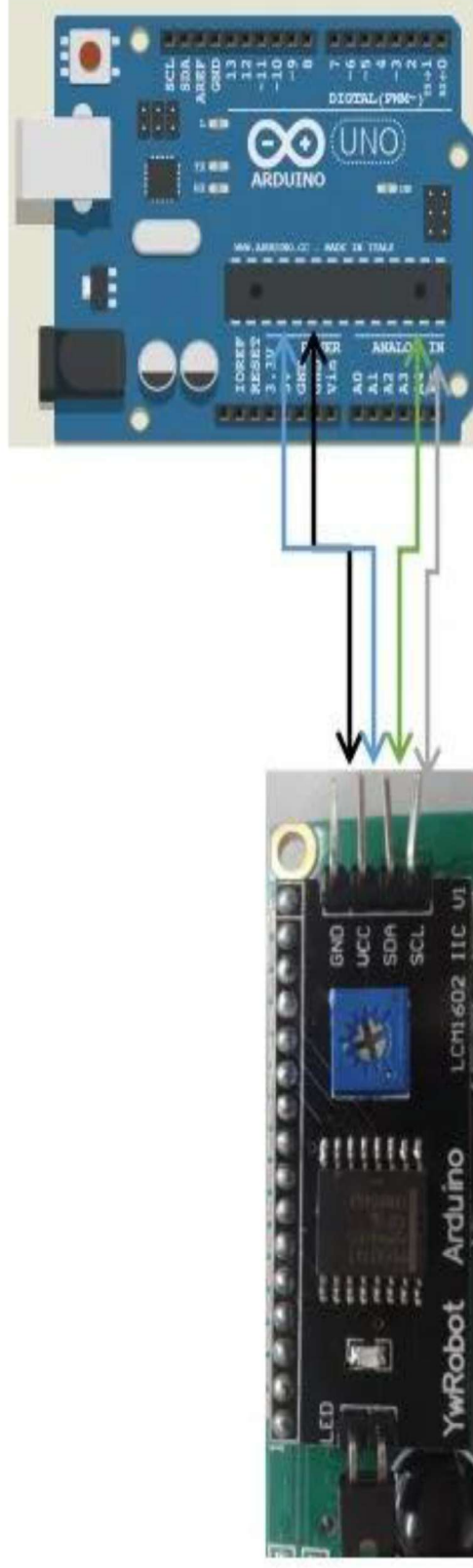
Download the LiquidCrystal_I2C file below > Go to document > Save the file into your Arduino Uno Library folder. Refer the image above for your references.

[Download](#)

(<https://cdn.instructables.com/ORIG/FHO/9LI0/J2UPIBMD/FHO9LI0J2UPIBMD.rar>)

I2C LCD (Contd...)

VCC -5V
GND - GND
SDA-A4
SCL-A5



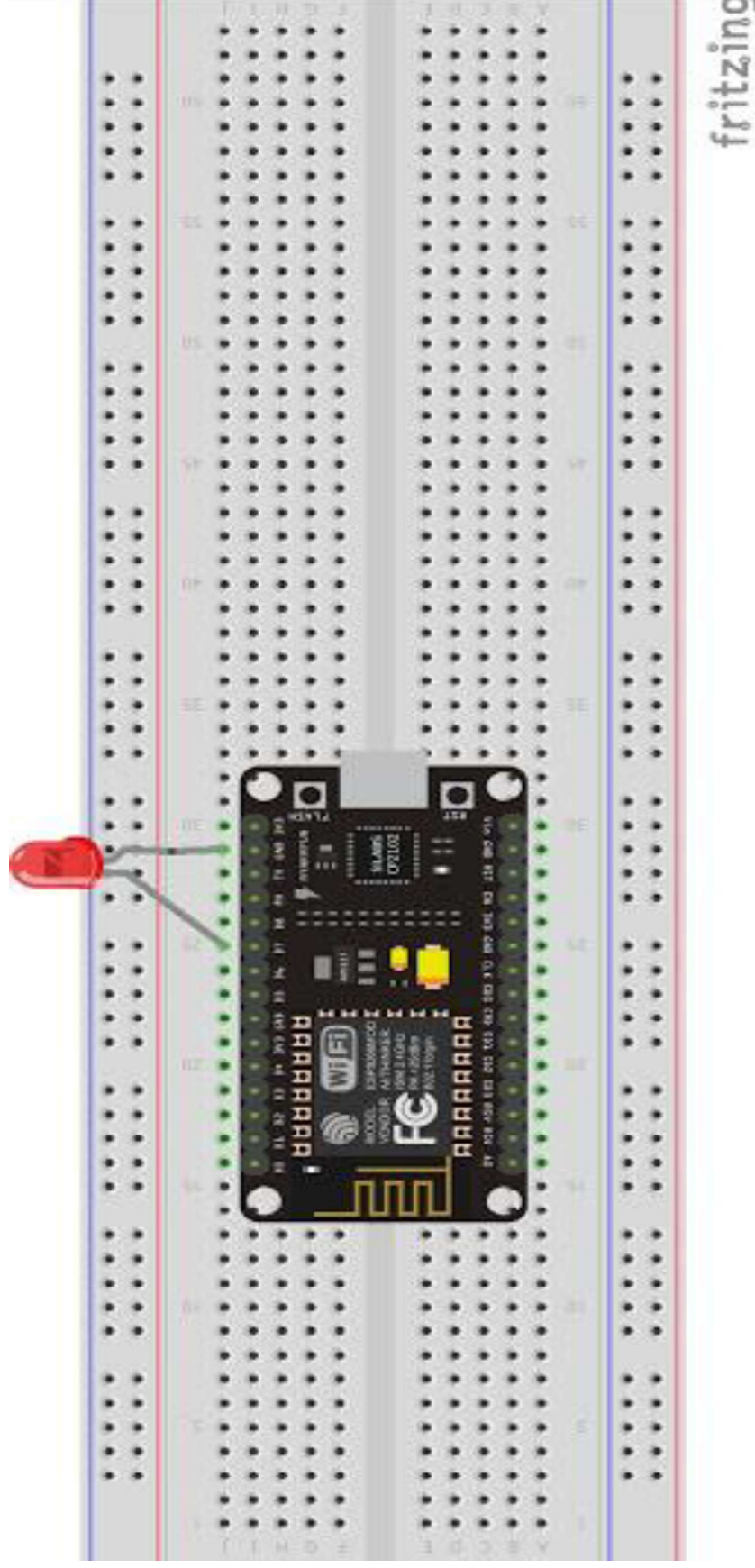
I2C LCD (Contd...)

```
#include "Wire.h" // For I2C
#include "LCD.h" // For LCD
#include "LiquidCrystal_I2C.h" // Added library*
//Set the pins on the I2C chip used for LCD connections
//ADDR, EN, R/W, RS, D4, D5, D6, D7
LiquidCrystal_I2C lcd(0x27,2,1,0,4,5,6,7); // 0x27 is the default I2C bus address of the backpack-see article
void setup()
{
    // Set off LCD module
    lcd.begin(16,2); // 16 x 2 LCD module
    lcd.setBacklightPin(3,POSITIVE); // BL, BL_POL
    lcd.setBacklight(HIGH);
    lcd.print("Hello, World!");
    lcd.setCursor(0,1);
    lcd.print("Good Day");
}

void loop()
{
}
}
```

Node MCU or ESP8266

<https://www.instructables.com/id/Quick-Start-to-Nodemcu-ESP8266-on-Arduino-IDE/>



Node MCU or ESP8266

For Program go to the mentioned link:

<https://www.youtube.com/watch?v=3gOKrMAz7WE>

<https://github.com/amphancm/ESP8266WiFiControl>

<https://www.youtube.com/watch?v=-6Nb5kL43GY>

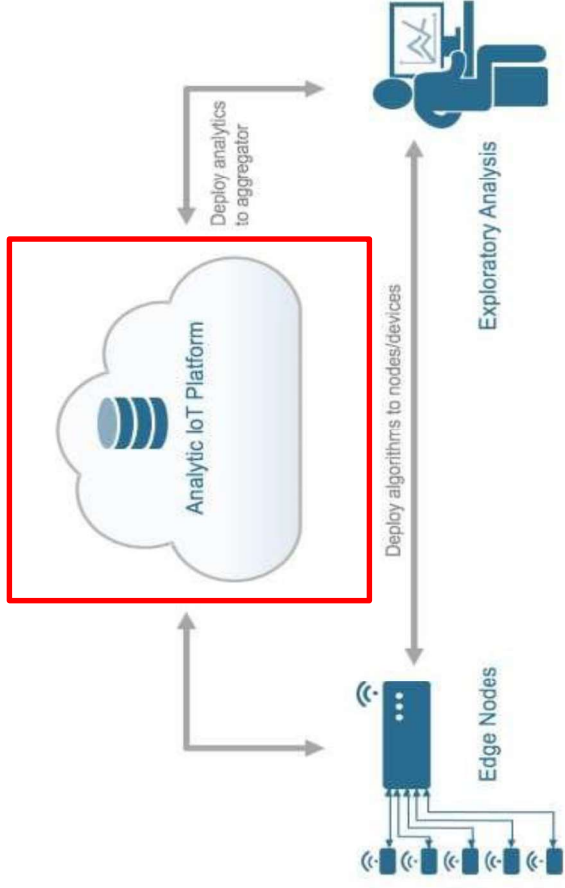
<https://www.youtube.com/watch?v=BZOfiPUyZqc>

<https://www.youtube.com/watch?v=5SvRoIROPxA>

ThingSpeak

What is ThingSpeak?

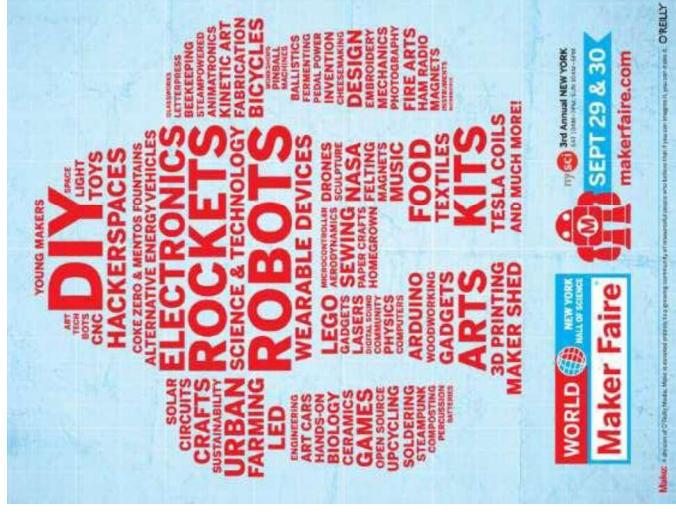
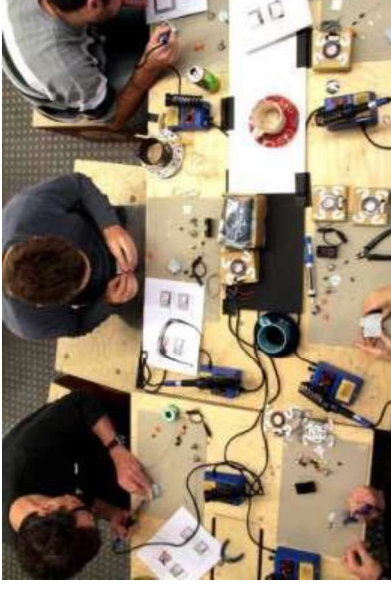
- Analytic IoT platform
 - Collect data from sensors, “things”
 - Visualize data instantly
 - Has more than 60,000 users
- Analyze data
 - MATLAB integration allows users to run scheduled code on data coming into ThingSpeak
- Act on data
 - E.g. send a tweet when the temperature in your backyard reaches 32 degrees



Who is ThingSpeak for?

- Makers
- Academics
- Engineers and scientists

<https://thingspeak.com/>



ThingSpeak

Channels

Apps

Blog

Support

Account

Sign Out

New Channel

Help

ThingSpeak Channel

Channels store all the data that a ThingSpeak application collects. Each channel includes eight fields that can hold any type of data, plus three fields for location data and one for status data. Once you collect data in a channel, you can use ThingSpeak apps to analyze and visualize it.

Channel Settings

Channel Name: Enter a unique name for the ThingSpeak channel.

Description: Enter a description of the ThingSpeak channel.

Field#: Check the box to enable the field, and enter a field name. Each ThingSpeak channel can have up to 8 fields.

Metadata: Enter information about channel data, including JSON, XML, or CSV data.

Tags: Enter keywords that identify the channel. Separate tags with commas.

Latitude: Specify the position of the sensor or thing that collects data in decimal degrees. For example, the latitude of the city of London is 51.5072.

Name

Description

Field 1

Field Label 1

☒

Field 2

☐

Field 3

☐

Field 4

☐

Field 5

☐

Field 6

☐

Field 7

☐

...

☐

For any new data, first login and create a channel in ThingSpeak

Channels have read and write API keys and can be public or private

A channel is made up of 8 fields and can store 8 streams of data (Temp, Humidity, etc.)

Channels can be updated at a maximum rate of once every 15 seconds

[ts.pdf \(iitd.ac.in\)](https://ts.pdf(iitd.ac.in))

Set up ThingSpeak channel

Private ViewPublic ViewChannel SettingsData Import / ExportAPI Keys

Channel Settings

Percentage complete85%

Channel ID82845

Name

Performance Trend

Description

Visualizing a performance trend for data collected using the MATLAB Performance Testing Framework

Field 1

Platform

☒

Field 2

Host

☒

Field 3

Version

☒

Field 4

Name

☒

Field 5

Run Identifier

☒

Field 6

Measured Time

☒

Field 7

☐

Help

ThingSpeak Channel

Channels store all the data that a ThingSpeak application collects. Each channel includes eight fields that can hold any type of data, plus three fields for location data and one for status data. Once you collect data in a channel, you can use ThingSpeak apps to analyze and visualize it.

Channel Settings

- Channel Name: Enter a unique name for the ThingSpeak channel.
- Description: Enter a description of the ThingSpeak channel.
- Field#: Check the box to enable the field, and enter a field name. Each ThingSpeak channel can have up to 8 fields.
- Metadata: Enter information about channel data, including JSON, XML, or CSV data.
- Tags: Enter keywords that identify the channel. Separate tags with commas.
- Latitude: Specify the position of the sensor or thing that collects data in decimal degrees. For example, the latitude of the city of London is 51.5072.
- Longitude: Specify the position of the sensor or thing that collects data in decimal degrees. For example, the longitude of the city of London is -0.1275.
- Elevation: Specify the position of the sensor or thing that collects data in meters. For example, the elevation of the city of London is 35.052.
- Make Public: If you want to make the channel publicly available, check this box.

Post performance data to our ThingSpeak channel

