Literature Review

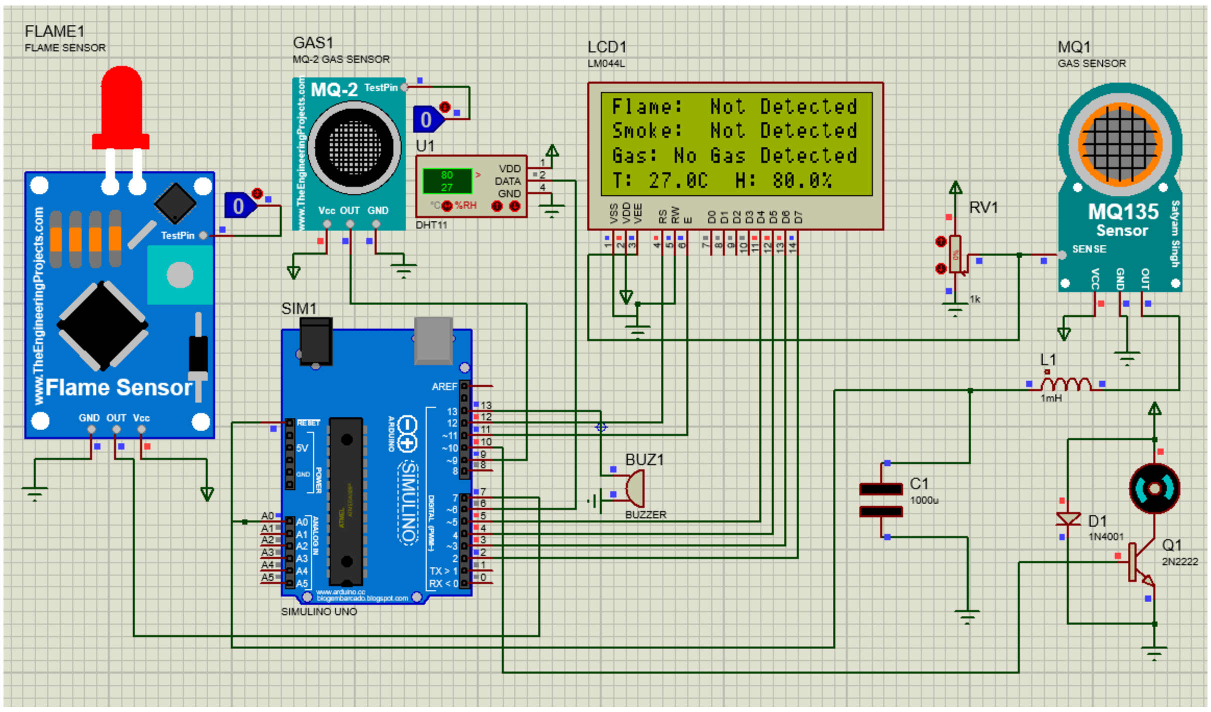
Introduction:

In an age of lightspeed computation, we are compelled to think and react at the speed of light. However, whilst digital computation reaches lightspeed, mankind's mental computation is still cumbersome and analog in nature; prone to negligence. Thus, in recent years, Internet of Things (IoT) has somewhat mitigated human error in a plethora of places like offices and homes, the latter is briefly discussed in this paper.

IoT encompasses many low-powered devices that aid in the day-to-day lives of various people and in various places. One such place is the humble kitchen, a homemade factory that sustains and maintains our biology. Just like any factory, a kitchen is a dangerous place. If not maintained or monitored, catastrophic failure is imminent.

Furthermore, aside from cooking disasters and health hazards, a poorly ventilated kitchen may damage utensils. At the heart of this topic human negligence stands solus. However, that being said, no machine can change that, but mitigate it. Hence, this paper deals with smart kitchen and alarm system, where monitoring and alarm are given priority over automation. This is because, we believe technology should be there as our co-pilot; not our pilot.

Simulation Schematic:



Working Principle of the Circuit:

Our project primarily relies on the Arduino Uno R3, for onboard control and decision-making process. Arduino Uno R3 is powered by an external 5V, 1A DC Power Adapter, connected to the AC power outlet. Furthermore, to avoid damaging the Arduino Uno R3, while also providing additional power to an add-on component (a 6V rated motor), four 1.5V AA alkaline batteries are used.

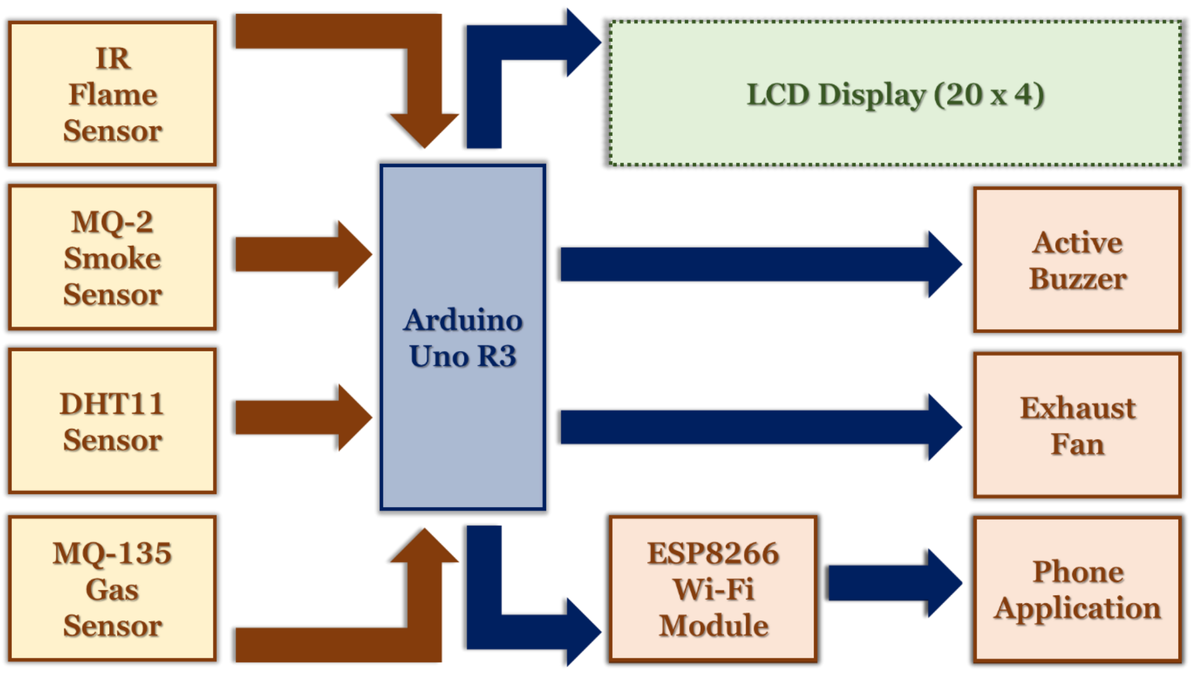
Four sensors (IR Flame sensor, MQ-2 smoke sensor, DHT11 temperature and humidity sensor, and MQ-135 gas leakage detection sensor) are connected to the Arduino Uno R3 inputs pins (7, ~9, ~6, A0) respectively. Three output components (LCD display 20x4, passive buzzer, and a 6V rated motor) are connected to the Arduino Uno R3 output pins (12, ~11, ~5, 4, ~3, 2; 13; ~10) respectively. All the sensors along with the LCD display take power from the 5V power trail of the Arduino Uno R3.

A 6V rated motor acts as an exhaust fan. When kitchen humidity exceeds 60%, Arduino Uno R3 switches a npn transistor (2N2222) which in turn turns on the motor; else it remains turned off. Moreover, to protect the motor from damage a freewheeling diode (1N4001) is used to prevent any feedback current flow to the motor. In addition to the motor, a passive buzzer is turned on, for every time a sensor detects its respective particulate. As we are using a passive buzzer, so each sensor sounds a different tone of alarm. Hence, these unique tones indicate which sensor is turned on at that time.

Our project secondarily relies on ESP8266 Wi-Fi Module and a Phone App, for off-board control and decision-making process. ESP8266 Wi-Fi Module's receive pin (RXD0) is connected to the transfer pin of Arduino Uno R3 (TX), and Arduino Uno R3's receive pin (RX) is connected to the transfer pin (TXD0) of the ESP8266 Wi-Fi Module [11].

Whenever a sensor detects its respective particulate, a signal is sent via Wi-Fi to a cloud server that in turn sends a notification to the user's Phone App. Furthermore, we intend to incorporate some level of automation, such that a user can set a timer to turn off the stove automatically. In addition to that, through the Phone App, the user can manually control the exhaust fan and the passive buzzer. This is done by taking input from the user through the Phone App. The Phone App sends a signal to a cloud server, and the cloud server in turn sends a signal via Wi-Fi to the Arduino Uno R3 to take the necessary steps to execute the user's command.

Block Diagram:



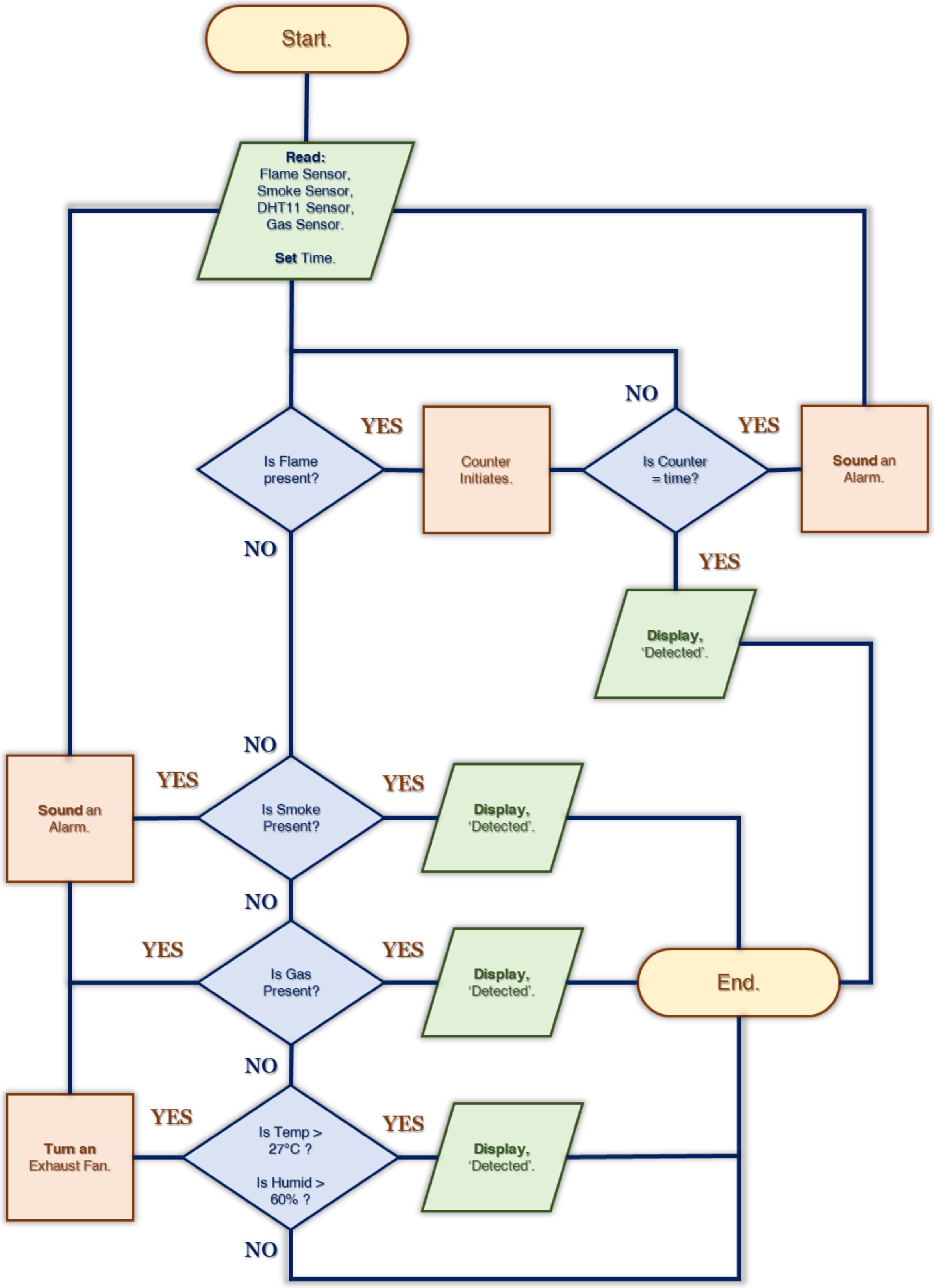
Project Limitations:

In our project we used MQ-2, MQ-135 sensors to detect smoke and gas leakage respectively [12]. These two sensors are analog sensors; they are prone to errors and often give inaccurate data. They cannot even detect the exact type of smoke or gas. Furthermore, in this project neither smoke concentration nor gas concentration is shown. This is done to keep the project as simple and cost effective as possible. Hence, this project is by no means suitable for professional cooking places like bustling bistros or restaurants. This project is more suitable for use in homes.

Also, using ESP8266 Wi-Fi Module, instead of the much-advanced ESP32 Wi-Fi & Bluetooth Module, was a point of contention for some of us. ESP32 has twice the speed of a ESP8266 Wi-Fi Module but also costs thrice the price of just one ESP8266 Wi-Fi Module. So, even though ESP8266 is extremely cost efficient it may turn out to be a bottleneck while uploading code.

On a different note, other more established projects have used a PIR sensor (passive infra-red sensor) or an ultrasonic sensor to monitor the presence of people in kitchen [1]. However, we have opted not to use such sensors as we are disinterested in such a functionality. Whilst this decreases one functionality of the project. Its absence has minimal effect on our final goals for this project.

Flowchart of the Code:



Comment & Code:

```
#include <LiquidCrystal.h>                                  // Library for LCD Display.
#include <dht.h>                                            // Library for DHT11 Sensor.

// Pin definitions.
const int Flame_Sensor_Pin = 7;                             // Connects to Arduino pin 7.
const int Smoke_Sensor_Pin = 9;                             // Connects to Arduino pin ~9.
const int Gas_Sensor_Pin = A0;                              // Connects to Arduino pin A0.
const int Buzzer_Pin = 13;                                  // Connects to Arduino pin 13.
const int Dht_Pin = 6;                                      // Connects to Arduino pin ~6.
const int Motor_Pin = 10;                                   // Connects to Arduino pin ~10.

// Typical threshold values for gas detection & humidity detection.
const int Gas_Threshold = 125;
const int Humidity_Threshold = 60;

// Initialization for LCD display & DHT11 sensor.
LiquidCrystal lcd(12, 11, 5, 4, 3, 2);  // Connects Arduino pins 12, ~11, ~5, 4, ~3, 2 to LCD
                                        // display pins RS, E, D4, D5, D6, D7 respectively.
dht DHT;

// Setup part is run once.
void setup() {
  Serial.begin(9600);                                   // Setting Baud Rate
  pinMode(Flame_Sensor_Pin, INPUT);                     // Takes input from Arduino pin 7.
  pinMode(Smoke_Sensor_Pin, INPUT);                     // Takes input from Arduino pin ~9.
  pinMode(Gas_Sensor_Pin, INPUT);                       // Takes input from Arduino pin A0.
  pinMode(Buzzer_Pin, OUTPUT);                          // Takes output from Arduino pin 13.
  pinMode(Motor_Pin, OUTPUT);                        // Takes output from Arduino pin ~10.
  lcd.begin(20, 4);                             // Defines and initiates 4 rows and 20 columns of the
                                                // LCD display.

  lcd.print("Flame: ");                           // Part of the constant UI of the LCD display.
  lcd.setCursor(0, 1);                            // Moves cursor/pointer to row = 1, column = 0.
  lcd.print("Smoke: ");                           // Part of the constant UI of the LCD display.
  lcd.setCursor(0, 3);                            // Moves cursor/pointer to row = 3, column = 0.
  lcd.print("T: ");                               // Part of the constant UI of the LCD display.
  lcd.setCursor(12, 3);                           // Moves cursor/pointer to row = 3, column = 12.
  lcd.print("H: ");                               // Part of the constant UI of the LCD display.
  lcd.setCursor(0, 2);                            // Moves cursor/pointer to row = 2, column = 0.
  lcd.print("Gas: ");                             // Part of the constant UI of the LCD display.
}

// Loop part is run multiple times.
void loop() {
  // Reads sensor values.
  int Flame_Value = digitalRead(Flame_Sensor_Pin);          // Reads digital value from
                                                            // Arduino pin 7.
  int Smoke_Value = digitalRead(Smoke_Sensor_Pin);          // Reads digital value from
                                                            // Arduino pin ~9.
  int Gas_Value = analogRead(Gas_Sensor_Pin);               // Reads digital value from
                                                            // Arduino pin A0.
  DHT.read11(Dht_Pin);                                      // Reads analog value from
                                                            // Arduino pin ~6.

  // Displays flame status.
  lcd.setCursor(8, 0);                          // Moves cursor/pointer to row = 0, column = 8.
  if (Flame_Value == HIGH) {                    // Used to compare input flame value to 1 or on state.
    lcd.print("Detected    ");              // Displays this text on the LCD display.
    tone(Buzzer_Pin, 1520);                 // Turns on buzzer and sounds an alarm with a
                                            // 1520Hz frequency.
  } else {                                  // Used to compare input flame value to 0 or off state.
    lcd.print("Not Detected");              // Displays this text on the LCD display.
    noTone(Buzzer_Pin);                     // Turns off buzzer.
  }

  // Displays smoke status.
  lcd.setCursor(8, 1);                      // Moves cursor/pointer to row = 1, column = 8.
  if (Smoke_Value == HIGH) {                // Used to compare input smoke value to 1 or on state.
    lcd.print("Detected    ");              // Display this text on the LCD display.
    tone(Buzzer_Pin, 760);                  // Turns on buzzer and sounds an alarm with a
                                            // 760Hz frequency.
  } else {                                  // Used to compare input some value to 0 or off state.
    lcd.print("Not Detected");              // Displays this text on the LCD display.
    noTone(Buzzer_Pin);                     // Turns off buzzer.
  }

  // Displays temperature & humidity.
  lcd.setCursor(3, 3);                           // Moves cursor/pointer to row = 3, column = 3.
  lcd.print(DHT.temperature, 1);                 // Displays this value on the LCD display.
  lcd.print("C");                                // Displays this text on the LCD display.

  lcd.setCursor(10, 3);                        // Moves cursor/pointer to row = 3, column = 10.
  lcd.print("H: ");                            // Displays this text on the LCD display.
  lcd.print(DHT.humidity, 1);                  // Displays this value on the LCD display.
  lcd.print("%");                              // Displays this text on the LCD display.

  if (DHT.humidity>Humidity_Threshold){      // When input humidity value is greater
```

```
                                              // than set humidity threshold value of 60%.
    digitalWrite(Motor_Pin, HIGH);            // Turns on motor.
  }else{                                      // When input humidity value is lesser
                                              // than set humidity threshold value of 60%.
    digitalWrite(Motor_Pin, LOW);             // Turns off motor.
  }

  // Displays gas status.
  lcd.setCursor(5, 2);                     // Moves cursor/pointer to row = 2, column = 5.
  if (Gas_Value > Gas_Threshold) {         // When input gas value is greater
                                           // than set threshold value of 125.
    lcd.print("Gas Detected");             // Displays this text on the LCD display.
    tone(Buzzer_Pin, 2280);                // Turns on buzzer and sounds an alarm with a
                                           // 2280Hz frequency.
  } else {                                 // When input gas value is lesser
                                           // than set threshold value of 125.
    lcd.print("No Gas Detected");          // Displays this text on the LCD display.
    noTone(Buzzer_Pin);                    // Turns off buzzer.
  }

}
```

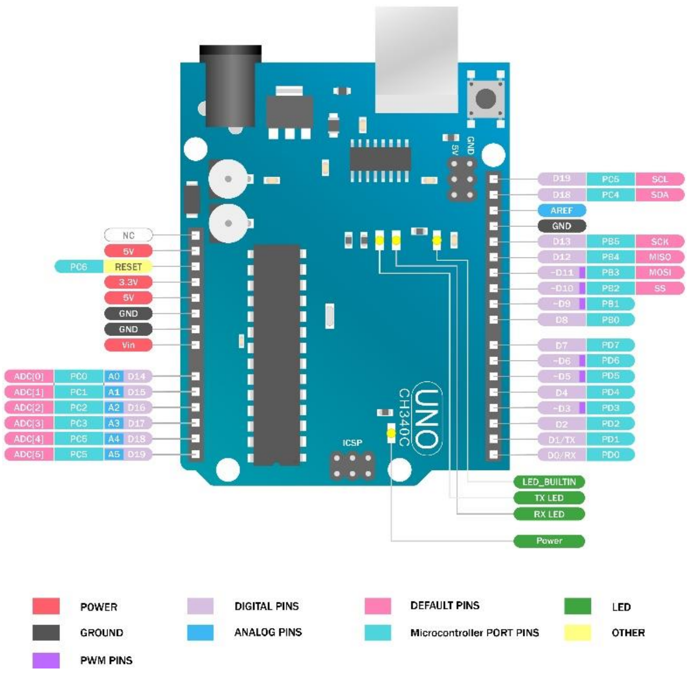
Related Projects:

i. Home Automation.
ii. Smart Appliance control.
iii. Smart Fire Alarm System.
iv. IoT Based Smart-Kitchen Security System.
v. Temperature and Humidity monitoring system.

The Physics behind the project:

The Arduino Uno R3 rests comfortably at the heart of our project. It is a PCB-mounted microcontroller that can be programmed to do simple tasks and mathematical computations [2].

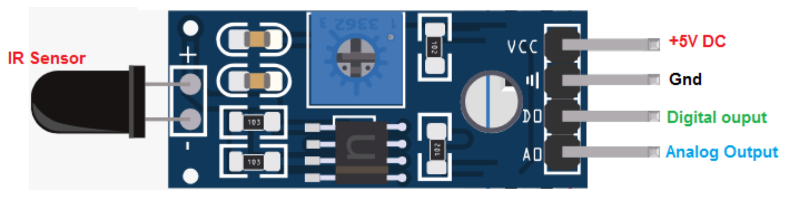
For our purposes, the Arduino Uno R3 will take-in analog input value from the sensors and convert it to digital input value through the use of its Analog-to-Digital Converter (ADC). This converted digital input value is then processed by the at ATmega328P microcontroller. The microcontroller then outputs a signal to the buzzer or the motor to turn on or off depending upon the previously taken sensor readings, and coded instructions [10].



In our project the Arduino Uno R3 will be powered through the barrel jack. Input and output will be taken from the analog pins (A0-A5) and digital pins (D0-D13 & D18-19).

The sensors used in this project are mostly depended on a change in either resistance or conductivity of an internal material or component.

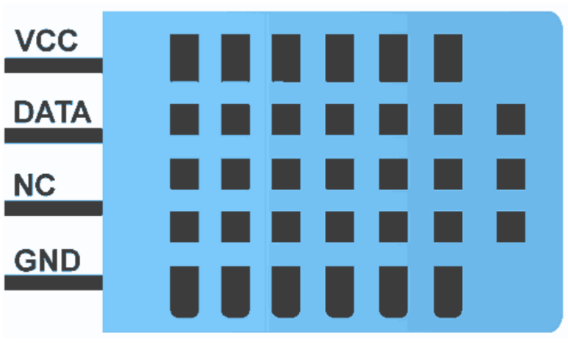
For instance, the IR Flame sensor itself has two basic components; the emitter and the detector. The emitter is simply an IR LED (Infra-red, Light Emitting Diode) and the detector is simply an IR photodiode. Photodiode is sensitive to infra-red light of a certain wavelength which is emitted by the IR LED. When infra-red light falls onto the photodiode, the resistances and the output voltages will change in proportion to the magnitude of the IR light received [3].



The IR Flame sensor module has four pins and a built-in potentiometer to calibrate flame detection accuracy. Power is provided to the module through the V_{CC} pin, Gnd pin is connected to the ground and output is taken from the D0 pin.

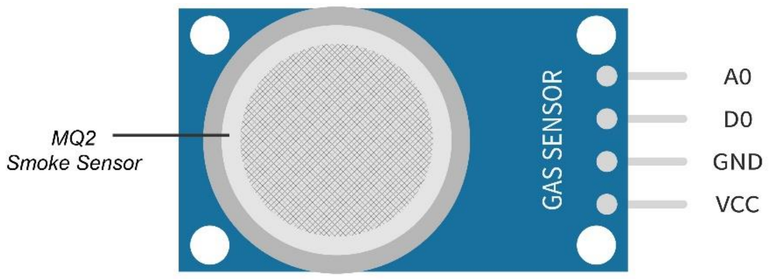
DHT11 sensor consists of a capacitive humidity sensing component coupled with a thermistor for sensing temperature. The humidity sensing capacitor has two electrodes with a moisture holding substrate as a dielectric between them. Change in the capacitance value occurs with the change in humidity levels.

For measuring temperature, this sensor uses a NTC thermistor. The term *NTC* refers to *Negative Temperature Coefficient*, which means that the resistance decreases with an increase of the temperature. To get a noticeable change in resistance value even for the smallest change in temperature, this sensor is usually made up of semiconductor ceramic or polymer [4].

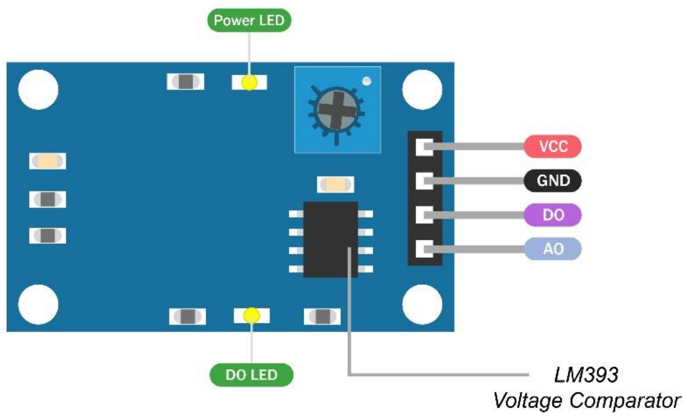


DHT11 Temperature & Humidity sensor has four pins. Power is provided to the module through the V_{CC} pin, GND pin is connected to the ground and output is taken from the DATA pin.

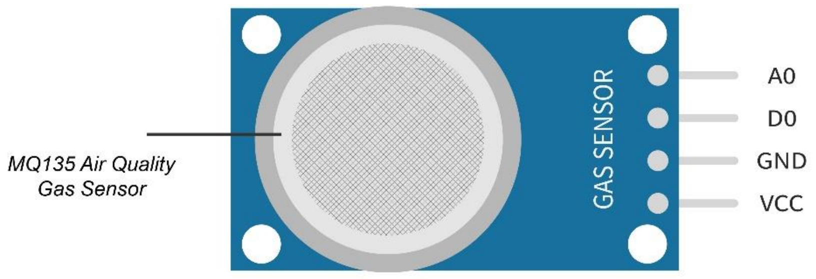
The MQ-2 sensor module consists of a sensing component attached to a PCB that amplifies and processes the signal. The sensing component is made up of a tin dioxide (SnO_2) semiconductor that is sensitive to the presence of certain gases and smokes. When gas or smoke molecules come into contact with the SnO_2 surface, the conductivity of the sensor changes, and the output voltage of the PCB is modified accordingly [5].



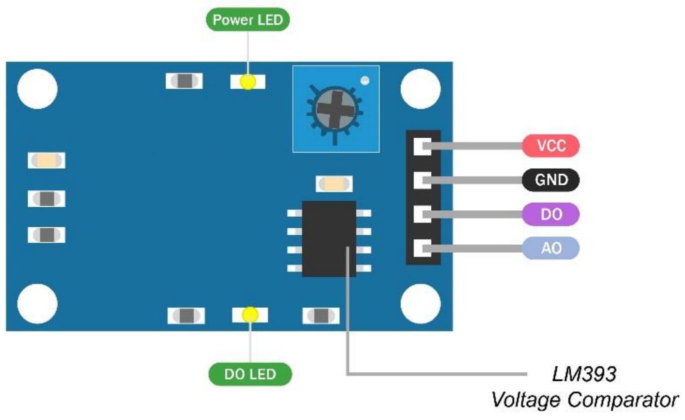
The MQ-2 smoke sensor module has four pins and a built-in potentiometer to calibrate smoke detection accuracy. Power is provided to the module through the V_{CC} pin, GND pin is connected to the ground and output is taken from the D0 pin.



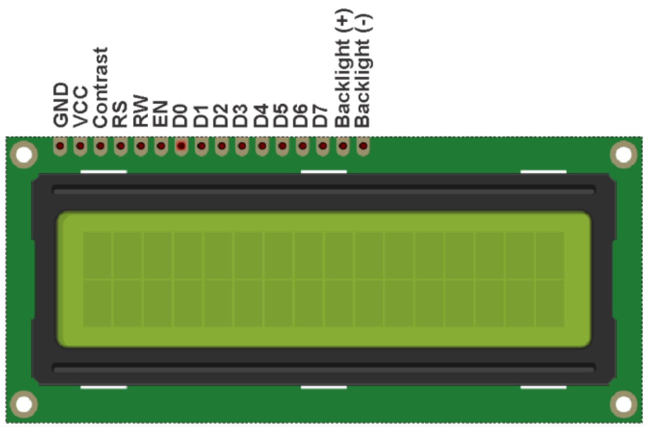
The MQ-135 alcohol sensor consists of a tin dioxide (SnO_2), a perspective layer inside aluminum oxide microtubes (measuring electrodes), and a heating element inside a tubular casing. The end face of the sensor is enclosed by a stainless-steel net and the backside holds the connection terminals. Ethyl alcohol present in the air is oxidized into acetic acid passing through the heating element. When the ethyl alcohol cascade on the tin dioxide sensing layer, the resistance decreases. By using the external load resistance, the resistance change is converted into a suitable voltage variation that is compared against a set threshold value [6].



The MQ-135 gas leakage sensor module has four pins and a built-in potentiometer to calibrate gas detection accuracy. Power is provided to the module through the V_{CC} pin, GND pin is connected to the ground and output is taken from the A0 pin.



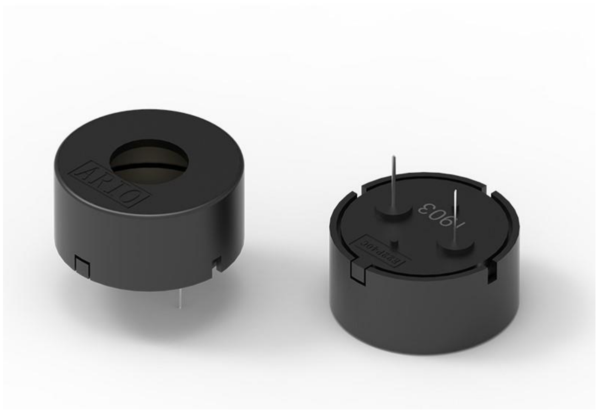
The LCD 20x4 Character Display Module operates on the basis of the physical characteristics of liquid crystals. A layer of liquid crystal material is positioned within the module between two polarizing filters. The amount of light that can travel through the polarizing filters is affected by the orientation of the crystals, which is altered when an electric current is delivered to the liquid crystal layer. The LCD module's functionality is managed by the HD44780 controller chip. It transmits data and orders to the module, whose internal circuitry interprets them to produce the desired pictures or characters on the screen. The controller chip provides a number of orders to the display in order to display characters on the screen [7].



The LCD display has numerous pins depending upon the make and model. However, we are only concerned with the D0-D7 input data pins, V_{CC} power pin, and GND or ground pin.

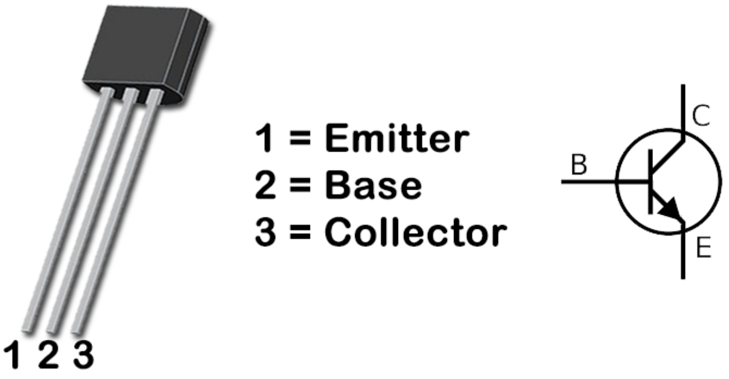
Passive buzzer is type of magnetic buzzer. There is a coil of wire inside the buzzer that is attached to its pins. In addition, the wire coil is encircled by a circular magnet. Above the circular magnet and wire coil is a thin, flexible ferromagnetic metal disc with a little metal weight fastened to the top. The metal weight and metal disc vibrate up and down as current pulses are given to the wire coil due to magnetic inductance. Sound waves are produced by the metal disk's oscillation [8].

For passive buzzers the anode and cathode are interchangeable. Hence, we can take one pin to be V_{CC} and the other to be ground.



The 2N2222 transistor consists of three layers of semiconductor material: an N-type layer sandwiched between two P-type layers. The Emitter, Base, and Collector are the names given to these levels, in that order. The features and performance of the transistor are determined by the doping levels of these layers. In its ‘off’ or inactive state, a transistor is one that is not receiving any external voltage or current. A tiny current or voltage is provided to the Base-Emitter junction of the 2N2222 transistor to turn it on. The transistor is stimulated by this input signal, which puts it in the active, or ‘on,’ state. The transistor can conduct because of the voltage drop that is produced when current enters the Base-Emitter junction [9].

The 2N2222 NPN transistor has three pins. Base pin is connected to the input. Collector pin is connected to the output and emitter pin is connected to the ground.



Bibliography

[1] (PDF) automation and monitoring smart kitchen based on internet ..., https://www.researchgate.net/publication/326349806_Automation_and_Monitoring_Smart_Kitchen_Based_on_Internet_of_Things_IoT (accessed Mar. 15, 2024).

[2] “Everything you need to know about the Arduino hardware,” What is Arduino Uno Hardware Board?, https://circuitdigest.com/article/everything-you-need-to-know-about-arduino-uno-board-hardware (accessed Mar. 15, 2024).

[3] Macfos, “IR sensor working principle and applications,” Robu.in | Indian Online Store | RC Hobby | Robotics, https://robu.in/ir-sensor-working/ (accessed Mar. 15, 2024).

[4] N. kalaburgi, “Working of DHT sensor - dht11 and DHT22,” NerdyElectronics, https://nerdyelectronics.com/working-of-dht-sensor-dht11-and-dht22/ (accessed Mar. 16, 2024).

[5] “All about MQ Series Gas Sensor,” Robocraze, https://robocraze.com/blogs/post/mq-series-gas-sensor (accessed Mar. 16, 2024).

[6] T. Agarwal, “MQ135 alcohol sensor circuit and its working ,” ElProCus, https://www.elprocus.com/mq-135-alcohol-sensor-circuit-and-working/ (accessed Mar. 16, 2024).

[7] “CART (0),” LCD 20x4 Character Display Module Yellow Background, https://techmaze.romman.store/product/99187125 (accessed Mar. 16, 2024).

[8] “Understanding difference between active and passive buzzer and how to use it with Arduino,” Circuit Digest - Electronics Engineering News, Latest Products, Articles and Projects, https://circuitdigest.com/microcontroller-projects/understanding-difference-between-active-and-passive-buzzer-with-arduino (accessed Mar. 16, 2024).

[9] C. Manager, “2N2222 transistor pinout,” ElectronicsHacks, https://electronicshacks.com/2n2222-transistor-pinout/ (accessed Mar. 16, 2024).

[10] C. A. U. Hassan et al., “Design and implementation of real-time kitchen monitoring and automation system based on internet of things,” MDPI, https://www.mdpi.com/1996-1073/15/18/6778 (accessed Mar. 16, 2024).

[11] Admin, “ESP8266 based Smart Kitchen Automation & Monitoring System,” IoT Projects Ideas, https://iotprojectsideas.com/esp8266-based-smart-kitchen-automation-monitoring-system/ (accessed Mar. 16, 2024).

[12] S. More, S. Shelar, V. Randhave, and Prof. A. Bagde, “IOT based Smart Kitchen System,” International Journal of Scientific Research in Science, Engineering and Technology, https://res.ijsrset.com/page.php?param=IJSRSET2183198 (accessed Mar. 16, 2024).