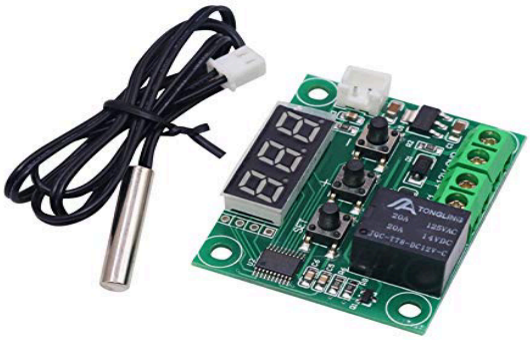
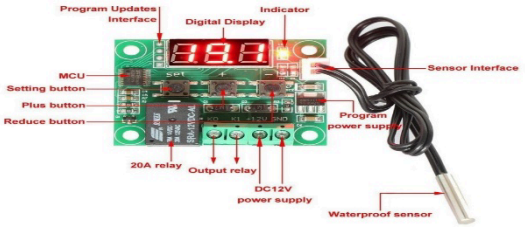
MINI PROJECT

TITLE : FIRE EXTINGUISHER USING THERMOSTAT

INTRODUCTION :

Fire extinguishers are essential safety devices designed to control or extinguish small fires in emergency situations. Understanding their working principles, types, and proper usage can be crucial in preventing the spread of fires and ensuring safety. This mini-project explores the detailed working and principles of fire extinguishers.
It is also used to find very low temperatures and gives the alarm to protect from fire explosive. And we can be ready to come down the fire.





- Temperature sensor
- Thermostat
- Buzzer
- 12v dc adapter
- Jumper cables
- Breadboard

SYSTEM DESIGN :

1. Install the thermostat in the area to be protected (e.g., a room or hallway).
2. Set the thermostat to trigger at a specific temperature (e.g., 65°C/149°F).

3. Connect the thermostat to the control panel.
4. Connect the control panel to the fire extinguisher system.
5. When the thermostat detects a temperature above the set point, it sends a signal to the control panel.
6. The control panel activates the fire extinguisher system, releasing the extinguishing agent (e.g., water or gas).
7. The system can also trigger alarms and notifications.

Objectives :
1. To understand the components and working principles of different types of fire extinguishers.
2. To identify suitable extinguishers for different classes of fires.
3. To demonstrate the correct method of using a fire extinguisher.

Application and Demonstration :

1. Identify the Type of Fire: Assess whether the fire is Class A, B, C, or K to select the appropriate extinguisher.
2. Approach Safely: Ensure a clear exit path before approaching the fire.
3. Follow PASS Method:
   - Pull the pin.
   - Aim the nozzle.
   - Squeeze the handle.
   - Sweep the nozzle

Software Code

```
#include <Wire.h>
#include <Adafruit_MLX90614.h>
#include <SoftwareSerial.h>
// Define pins
#define GSM_RX 10
#define GSM_TX 11
#define BUZZER_PIN 8
#define LIGHT_SENSOR_PIN A0
#define GAS_SENSOR_PIN A1
// Thresholds
#define TEMP_THRESHOLD 30.0   // Temperature in Celsius
#define LIGHT_THRESHOLD 500   // Light level (adjust based on your LDR)
#define GAS_THRESHOLD 300   // Gas level (adjust based on your MQ-2)
// Create sensor and GSM objects
Adafruit_MLX90614 thermistor;
SoftwareSerial gsmSerial(GSM_RX, GSM_TX);
void setup() {
  Serial.begin(9600);
  gsmSerial.begin(9600);
  pinMode(BUZZER_PIN, OUTPUT);
   if (!thermistor.begin()) {
    Serial.println("Could not find MLX90614 sensor.");
```

```
    while (1);
  }
}

void loop() {
  double temperature = thermistor.readObjectTempC();
  int lightLevel = analogRead(LIGHT_SENSOR_PIN);
  int gasLevel = analogRead(GAS_SENSOR_PIN);

  bool alert = false;

  // Check temperature threshold
  if (temperature > TEMP_THRESHOLD) {
    alert = true;
    digitalWrite(BUZZER_PIN, HIGH);
    sendSMSTemperatureAlert(temperature);
  }

  // Check light threshold
  if (lightLevel < LIGHT_THRESHOLD) {
    alert = true;
    digitalWrite(BUZZER_PIN, HIGH);
    sendSMSLightAlert(lightLevel);
  }

  // Check gas threshold
  if (gasLevel > GAS_THRESHOLD) {
    alert = true;
    digitalWrite(BUZZER_PIN, HIGH);
    sendSMSGasAlert(gasLevel);
  }

  if (!alert) {
    digitalWrite(BUZZER_PIN, LOW);
  }

  delay(5000); // Wait 5 seconds before next reading
}

void sendSMSTemperatureAlert(double temperature) {
  gsmSerial.println("AT+CMGF=1"); // Set SMS mode to text
  delay(1000);
  gsmSerial.println("AT+CMGS=\"+1234567890\""); // Replace with your number
  delay(1000);
  String message = "Temperature Alert: " + String(temperature) + " C";
  gsmSerial.println(message);
  delay(100);
  gsmSerial.println((char)26); // Send the message
```

```
  delay(1000);
}

void sendSMSLightAlert(int lightLevel) {
  gsmSerial.println("AT+CMGF=1");
  delay(1000);
  gsmSerial.println("AT+CMGS=\"+1234567890\"");
  delay(1000);
  String message = "Light Alert: Light level is " + String(lightLevel);
  gsmSerial.println(message);
  delay(100);
  gsmSerial.println((char)26);
  delay(1000);
}

void sendSMSGasAlert(int gasLevel) {
  gsmSerial.println("AT+CMGF=1");
  delay(1000);
  gsmSerial
}
```

Conclusion:

Temperature controllers and thermostats are indispensable devices for maintaining precise temperature levels in various applications. From residential comfort to industrial processes, they play a crucial role in energy efficiency, product quality, and safety.
Thermostats primarily serve residential and commercial spaces, offering basic temperature control with on/off functionality. They are user-friendly and suitable for maintaining comfortable living or working environments.
Temperature controllers provide more advanced control options, such as PID control, for precise temperature regulation in industrial and laboratory settings. They offer greater flexibility and accuracy for applications demanding stringent temperature requirements.

Advantages:

- Automated fire detection and suppression
- Quick response to potential fires
- Reduced damage and risk to people and property