Smart Thermostat System Using Arduino Mega and ESP32

Rotaru Cristian

January 16, 2025

Contents

1 Problem Statement

The goal of this project is to design and implement a smart thermostat system capable of monitoring and controlling room temperature. The system aims to:

- Measure the current temperature using an analog temperature sensor.
- Display the current temperature and desired threshold on an LCD.
- Control a heating element based on the temperature and user-defined threshold.
- Provide remote access via a Wi-Fi connection to view temperature data and modify the temperature threshold using an ESP32 web server.

The problem lies in creating a reliable, low-cost solution for temperature monitoring and heating control with both local (LCD) and remote (web interface) access.

2 The Solution

The proposed solution consists of two microcontrollers:

1. **Arduino Mega:** Handles temperature measurement, heating control, and local display on an LCD.
2. **ESP32:** Acts as a Wi-Fi-enabled web server for remote monitoring and control.

2.1 Arduino Mega Implementation

The Arduino Mega uses an analog temperature sensor to measure the room temperature and updates the LCD display with:

- Current temperature.
- Desired threshold.
- Status of the heating element (ON/OFF).

If the temperature falls below the threshold, the Arduino activates a relay to turn on the heating element. The Mega communicates with the ESP32 via serial to send current temperature data and receive updated thresholds.

Code Example:

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

  pinMode ( tempPin , INPUT );
  pinMode ( relayPin , OUTPUT );
  digitalWrite ( relayPin , LOW ) ;
  Serial1 . begin (9600) ;

  lcd . print (" Thermostat ␣ Init ");
  lcd . createChar (0 , flame );
  delay (2000) ;
}

void loop () {
  // Read sensors
  temperature = readTemperature (10 , tempPin ) ;

  // Control logic
  if ( temperature < tempThreshold && ! heatingOn ) {
    digitalWrite ( relayPin , HIGH ) ;
    heatingOn = true ;
  } else if ( temperature >= tempThreshold && heatingOn )
    {
    digitalWrite ( relayPin , LOW ) ;
    heatingOn = false ;
  }

  lcd . clear () ;
  // Display
  lcd . setCursor (0 , 0) ;
  lcd . print (" Temp :␣");
  lcd . print ( temperature , 1) ;
  lcd . print ("C␣");
  Serial1 . print (" TEMP :");
  Serial1 . print ( temperature , 1) ;
  Serial1 . print (" ,");

  if ( heatingOn ) {
    lcd . setCursor (15 ,0) ;
    lcd . write ( byte (0) );
    Serial1 . print (" HEATING :ON ,");
  }
```

```
42      else    {
43          Serial1 . print (" HEATING : OFF ,");
44      }
45
46      lcd . setCursor (0 ,        1) ;
47      lcd . print (" Desired :␣");
48      lcd . print ( tempThreshold ,         1) ;
49      lcd . print ("C␣");
50      Serial1 . print (" THRESH :");
51      Serial1 . print ( tempThreshold ,           1) ;
52      Serial1 . print ("\n");
53
54      delay (1000) ;
55  }
56
57  void    serialEvent ()        {
58      if    ( Serial1 . available () )          {
59          String     data    =   Serial1 . readStringUntil ( '\n ');
60          if    ( data . startsWith (" SET_TRESHOLD :"))          {
61              tempThreshold     =   data . substring (14) . toFloat () ;
62          }
63      }
64  }
65
66  float     readTemperature ( int       count ,    int     pin )   {
67      float     sumTemp =   0;
68      for    ( int    i    =   0;    i    <   count ;    i ++)    {
69          int    reading     =   analogRead ( pin );
70          float     voltage     =   reading     *   resolutionADC ;
71          float     tempCelsius      =   ( voltage      -    0.5)     /
                resolutionSensor         ;
72          sumTemp =   sumTemp +   tempCelsius ;
73      }
74      return     sumTemp /    ( float ) count ;
75  }
```

Listing 1: Arduino Mega Implementation

2.2 ESP32 Implementation

The ESP32 creates a Wi-Fi access point and hosts a web server. Users can connect to the ESP32 network to:

- View the current temperature, heating status, and threshold.

• Modify the threshold using interactive buttons.

Code Example:

```
// HTML Page
String createHTML() {
  String html = "<html><body>";
  html += "<meta␣http-equiv='refresh'␣content='5'>"; // Refresh every 5 seconds
  html += "<h1>Temperature␣Control</h1>";
  html += "<p>Current␣Temperature:␣" + String(temperature) + "␣C</p>";
  html += "<p>Heating:␣" + String(heating) + "␣</p>";
  html += "<p>Threshold:␣" + String(threshold) + "␣C</p>";
  html += "<button␣onclick=\"location.href='/increase'\">Increase␣Threshold</button>";
  html += "<button␣onclick=\"location.href='/decrease'\">Decrease␣Threshold</button>";
  html += "</body></html>";
  return html;
}

void setup() {
  Serial.begin(9600);
  if (!WiFi.softAP(ssid, password)) {
    while(1)
      ;
  }

  server.on("/", handleRoot);
  server.on("/increase", handleIncrease);
  server.on("/decrease", handleDecrease);
  server.begin();
}

void loop() {
  server.handleClient();
}
```

Listing 2: ESP32 Web Server Implementation

The ESP32 parses data from the Arduino Mega via serial communication and updates the web interface accordingly.

3 Circuit Diagram/Schematics

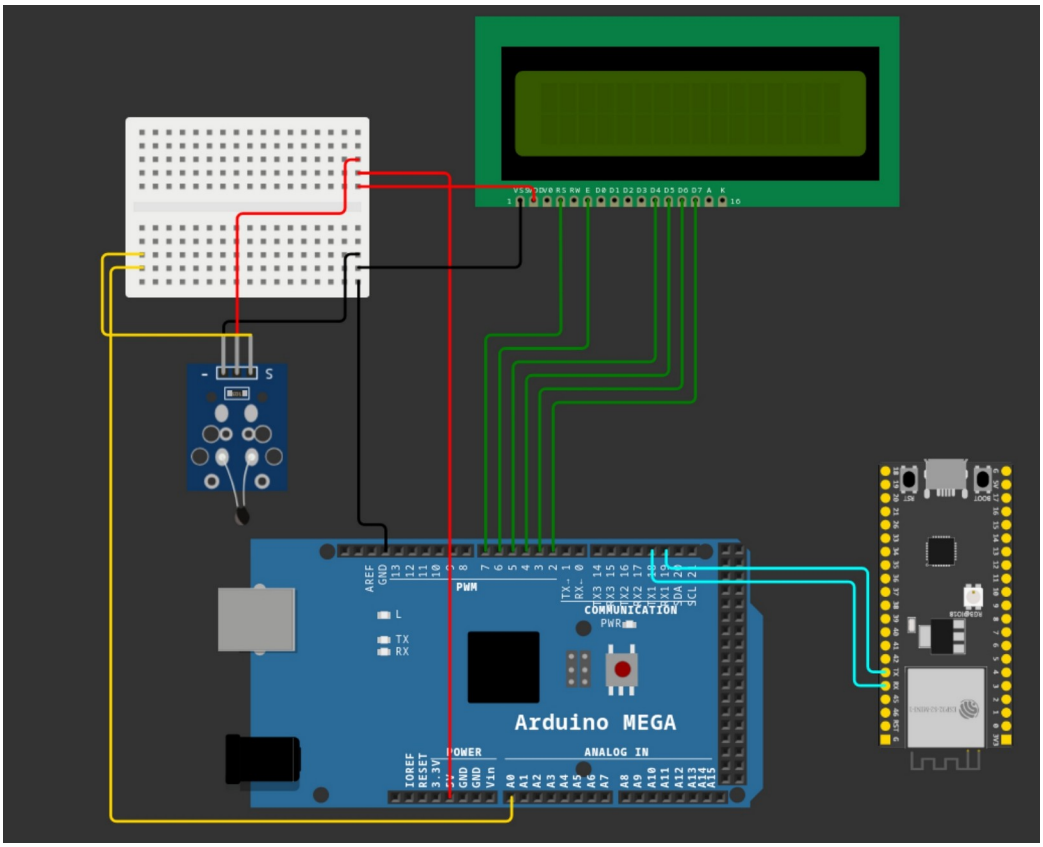

Figure 1: Circuit Diagram of the Smart Thermostat System

4 Picture of the Project

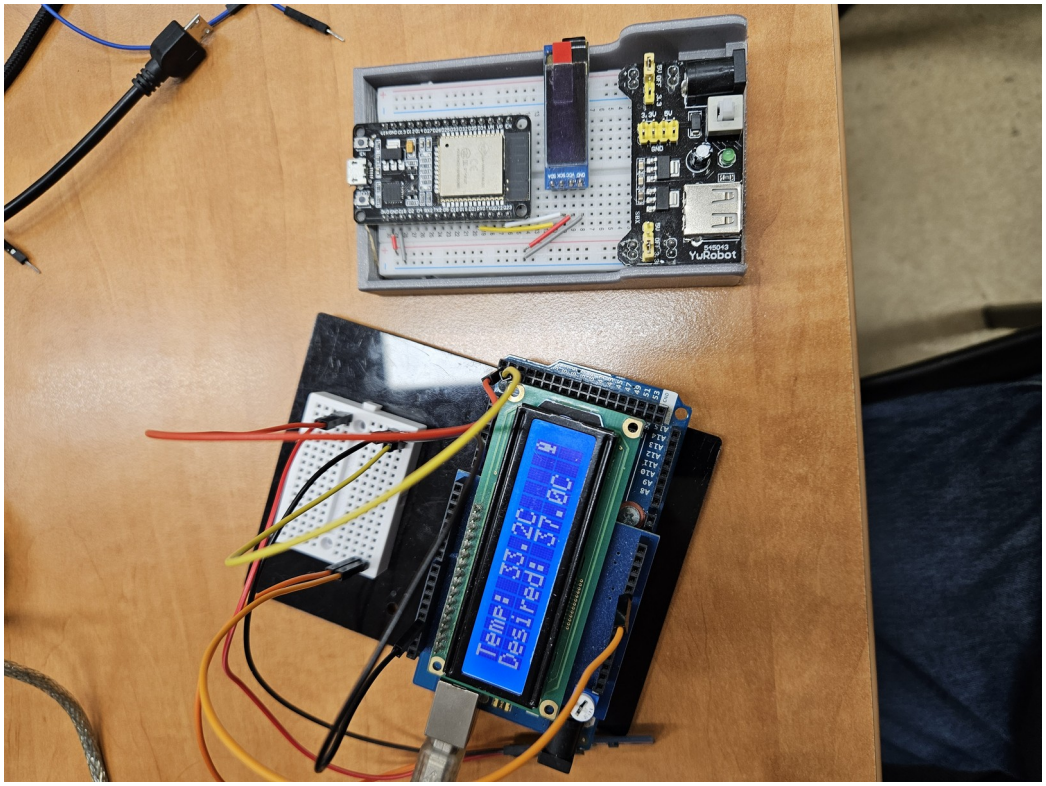

Figure 2: Smart Thermostat Prototype

5 Bibliography

- Arduino Documentation: https://www.arduino.cc/en/Reference/HomePage
- ESP32 Documentation:https://docs.espressif.com/projects/esp-idf/en/latest/