GOVERNMENT OF TAMILNADU

DIRECTORATE OF TECHNICAL EDUCATION, CHENNAI

NAAN MUDHALVAN SCHEME (TNSDC) SPONSORED

STUDENTS DEVELOPMENT PROGRAMME

ON

IoT AND ITS APPLICATIONS

HOST INSTITUTION

xxxxx

COIMBATORE – 04

TRAINING PARTNER

ENTHU TECHNOLOGY SOLUTIONS INDIA PVT LTD

DATE:

Name	Roll No

Table Of Contents

S no	Title	Page no

1	Abstract	
2	Introduction	
3	Hardware and Software Requirements	
4	Block Diagram	
5	Code	
6	Output Results	
7	Cloud Output	
8	Conclusion	

Abstract

In the pursuit of sustainable energy solutions, solar power systems have become increasingly prevalent. To ensure the efficiency and reliability of these systems, continuous monitoring of solar irradiance is essential. This project presents an IoT-based solar power monitoring system that leverages the BH1750 light intensity sensor and the ESP32 microcontroller to measure and transmit real-time solar irradiance data.

The BH1750 sensor, known for its precision in measuring light intensity, is integrated with the ESP32, a versatile microcontroller with built-in Wi-Fi and Bluetooth capabilities. The sensor captures the intensity of sunlight, which is a critical parameter in determining the performance of solar panels. The ESP32 processes this data and transmits it to a cloud-based platform for remote monitoring and analysis.

The controller system is further enhanced by integrating it with Wi-Fi connectivity to ThingzMate Cloud, allowing for remote monitoring and control. This connection enables real-time updates on the status of the solar power monitoring sysytem

This system enables real-time monitoring of solar panel performance, providing insights into the efficiency of energy conversion and identifying potential issues such as shading or panel degradation. The data can be accessed remotely via a web interface or mobile application, allowing users to make informed decisions on system maintenance and optimization.

By utilizing IoT technology, this solar power monitoring system offers an affordable, scalable, and efficient solution for enhancing the management of solar energy resources. This project not only contributes to the optimization of solar power systems but also supports the broader goal of promoting renewable energy adoption.

Introduction

As the global demand for renewable energy sources continues to grow, solar power has emerged as one of the most promising and sustainable alternatives to traditional fossil fuels. Solar panels convert sunlight into electrical energy, providing a clean and renewable source of power. However, the efficiency and performance of solar panels can be affected by various factors, including the intensity of sunlight, shading, dust accumulation, and panel degradation. To maximize the efficiency and lifespan of solar power systems, it is crucial to continuously monitor these factors.

This project focuses on developing an IoT-based solar power monitoring system that uses the BH1750 light intensity sensor and the ESP32 microcontroller. The BH1750 sensor is a highly accurate device that measures the intensity of sunlight, which directly influences the amount of electrical energy generated by solar panels. The ESP32, known for its low power consumption and built-in Wi-Fi capabilities, serves as the core of the monitoring system, processing the data from the BH1750 sensor and transmitting it to a remote server for real-time analysis.

The integration of IoT technology into solar power monitoring offers significant advantages. By enabling remote monitoring, this system allows users to track the performance of their solar panels from anywhere in the world using a web interface or mobile application. This real-time data access helps in identifying issues such as reduced efficiency due to shading or dirt, allowing for timely maintenance and optimization of the solar power system.

Hardware and Software Requirements

Hardware Requirements

1.ESP32 Microcontroller

2.LED

3.BreadBoard

4.USB Cable

5.Jumper Wires

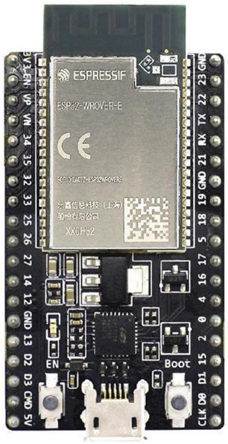
6.BH1750 Sensor

Software Requirements

1.Arduino IDE

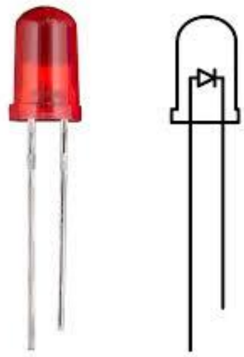
2.Thingzmate Cloud

ESP32 Microcontroller



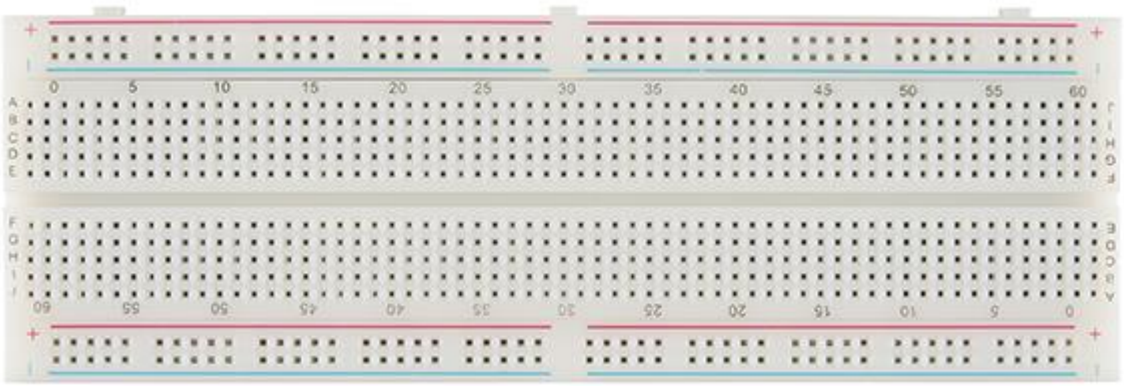
The ESP32 microcontroller is a powerful and versatile chip used in this project to control the four-way traffic light system. It features integrated Wi-Fi and Bluetooth capabilities, allowing for seamless communication with the ThingzMate Cloud for remote monitoring and control. With its dual-core processor and multiple GPIO pins, the ESP32 efficiently handles the timing and sequencing of the traffic lights, making it ideal for real-time IoT applications.

LED



In this project, standard LEDs are used to represent the traffic lights at the four-way intersection. These LEDs are controlled by the ESP32 microcontroller, which turns them on and off according to the programmed traffic light sequence. The LEDs provide a simple yet effective way to visualize the traffic signals, making them essential for testing and demonstrating the functionality of the traffic light controller system.

BreadBoard



The breadboard is an essential component in this project, used to prototype the circuit connections for the four-way traffic light controller system. It allows for easy placement and rearrangement of the LEDs, resistors, and connections to the ESP32 microcontroller without the need for soldering. The breadboard's flexibility makes it ideal for quickly testing and modifying the circuit design as the project develops.

USB-Cable

The USB cable is a critical tool in this project, used to connect the ESP32 microcontroller to a computer for power supply, programming, and debugging. It enables the transfer of code and data between the development environment and the microcontroller, facilitating the upload of firmware and real-time communication during the development process. The USB connection also allows for serial monitoring, providing valuable insights into the system's performance and behavior.

Jumper Wires

Jumper wires are essential in this project, used to connect the ESP32 microcontroller to the components on the breadboard. These wires provide a flexible and reliable way to link the microcontroller's GPIO pins to the LEDs, resistors, and other circuit elements, enabling proper signal and power flow. Their ease of use allows for quick modifications and testing during the prototyping stage.

BH1750 Sensor

The BH1750 is a digital light intensity sensor that measures ambient light in lux, which is a unit of illumination. It is commonly used in applications where accurate measurement of light intensity is required, such as in smart lighting, solar energy monitoring, and other IoT-based systems. The sensor is particularly valued for its simplicity, accuracy, and low power consumption.

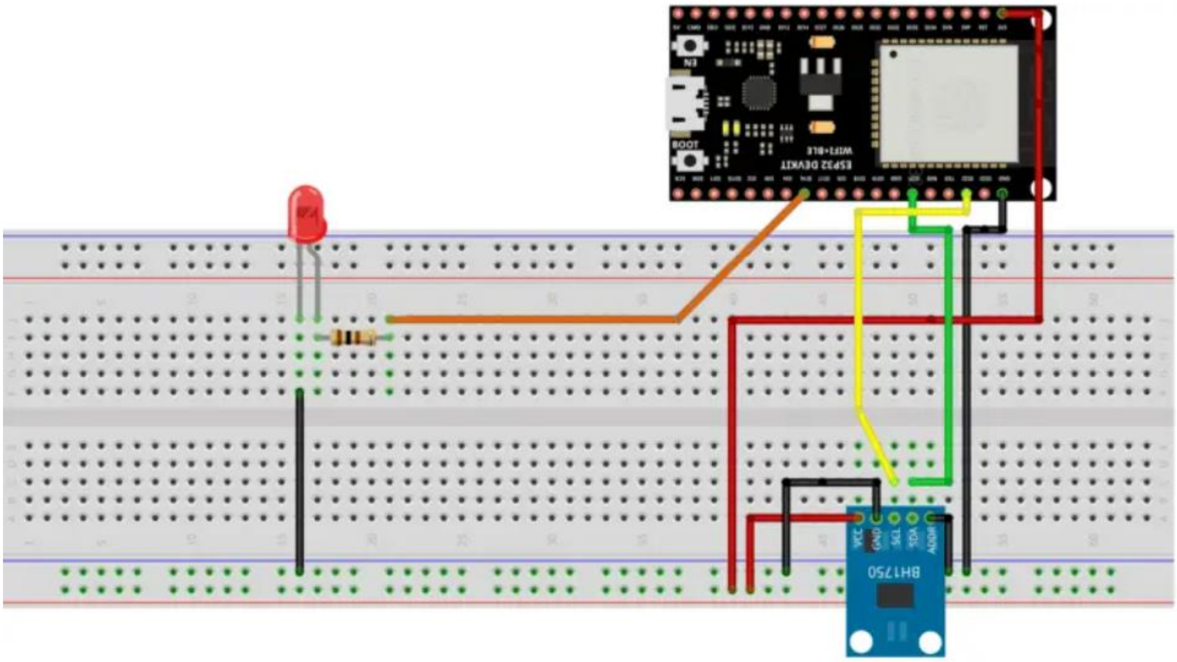
Arduino IDE

Arduino IDE is the primary development environment used in this project for programming and uploading code to the ESP32 microcontroller. It provides an easy-to-use interface for writing, compiling, and debugging the code that controls the four-way traffic light system. With its extensive library support and compatibility with ESP32, the Arduino IDE streamlines the development process, allowing for efficient code iteration and testing.

Thingzmate Cloud

ThingzMate enables real-time simulation of a 4-way traffic light control system, allowing you to monitor and manage traffic lights via cloud connectivity.It provides tools for configuring traffic light sequences, ensuring accurate simulation of traffic flow and light transitions.With ThingzMate, you can easily visualize and control traffic light statuses, optimizing traffic management and ensuring efficient simulation scenarios.

Block Diagram



Code

```
#include <WiFi.h>

#include <HTTPClient.h>

#include <BH1750.h>


// Replace these with your ThingzMate settings

const char *serverUrl = "https://console.thingzmate.com/api/v1/device-types/esp333/devices/esp33/uplink"; // Replace with your server endpoint

String AuthorizationToken = "Bearer 2b920840c605d9648538408aa1817b01";
```

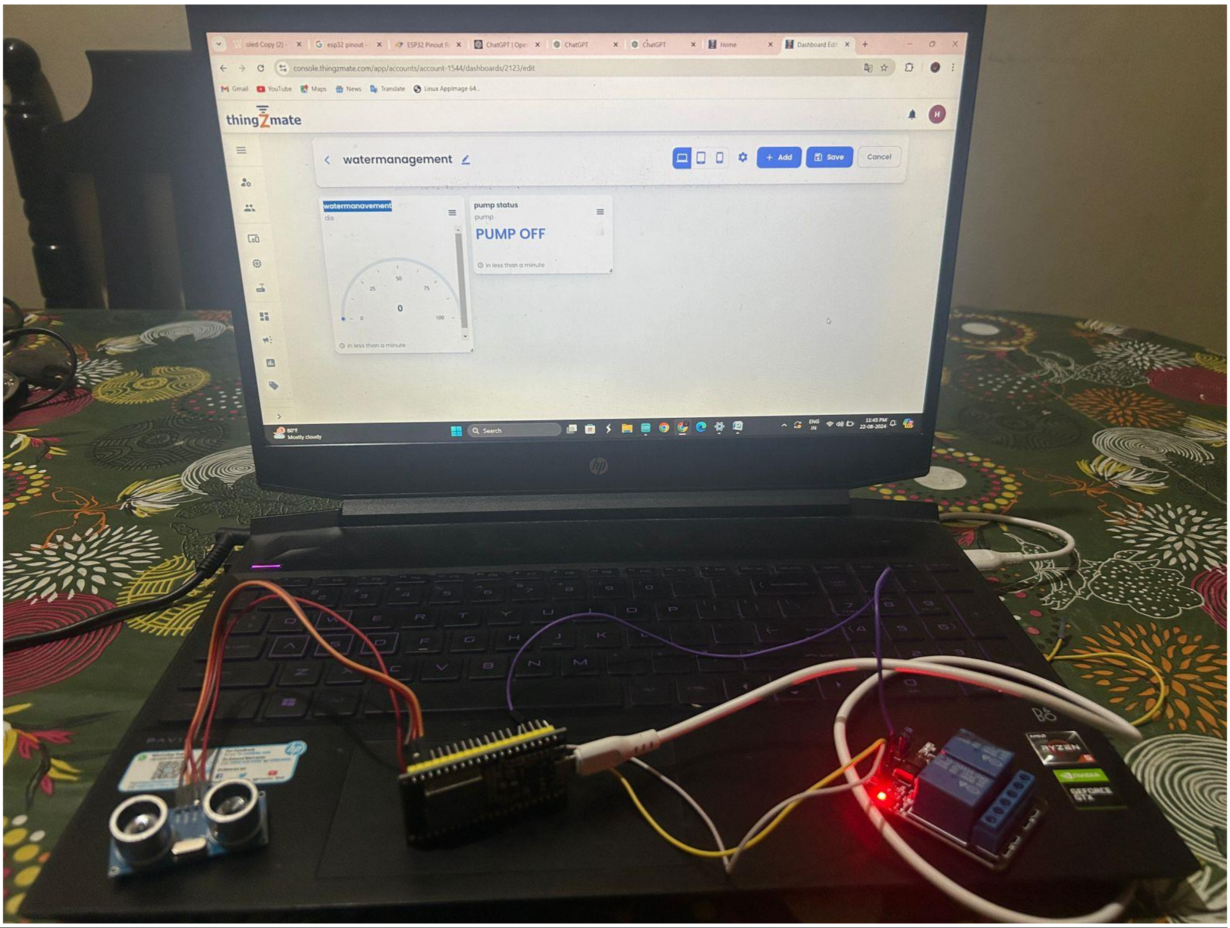
```
// Replace these with your WiFi credentials
const char* ssid = "Harish63";
const char* password = "123456789";

// Define the GPIO pins for the relay
BH1750 lightMeter;
int Relay = 26;

void setup() {
  Serial.begin(115200);
  Wire.begin(); // Initialize I2C communication

  // Initialize the BH1750 sensor
  if (lightMeter.begin()) {
    Serial.println("BH1750 sensor initialized successfully.");
  } else {
    Serial.println("Error initializing BH1750 sensor. Please check wiring.");
    while (1);
  }

  lightMeter.configure(BH1750::CONTINUOUS_HIGH_RES_MODE); // Set to continuous high resolution mode

  pinMode(Relay, OUTPUT);
  connectToWiFi();
}
```

```
void connectToWiFi() {
  WiFi.begin(ssid, password);

  Serial.println("Connecting to WiFi...");

  int attempts = 0;
  while (WiFi.status() != WL_CONNECTED && attempts < 20) {  // Try for 10 seconds (500ms * 20)
    delay(500);
    Serial.print(".");
    attempts++;
  }

  if (WiFi.status() == WL_CONNECTED) {
    Serial.println("Connected to WiFi");
    Serial.print("IP Address: ");
    Serial.println(WiFi.localIP());
  } else {
    Serial.println("Failed to connect to WiFi");
    // Optional: Add code to handle a failed connection (e.g., reset the device, or enter deep sleep)
  }
}

void loop() {
  // Only run the main logic if connected to WiFi
  if (WiFi.status() == WL_CONNECTED) {
    // Read the light level
```

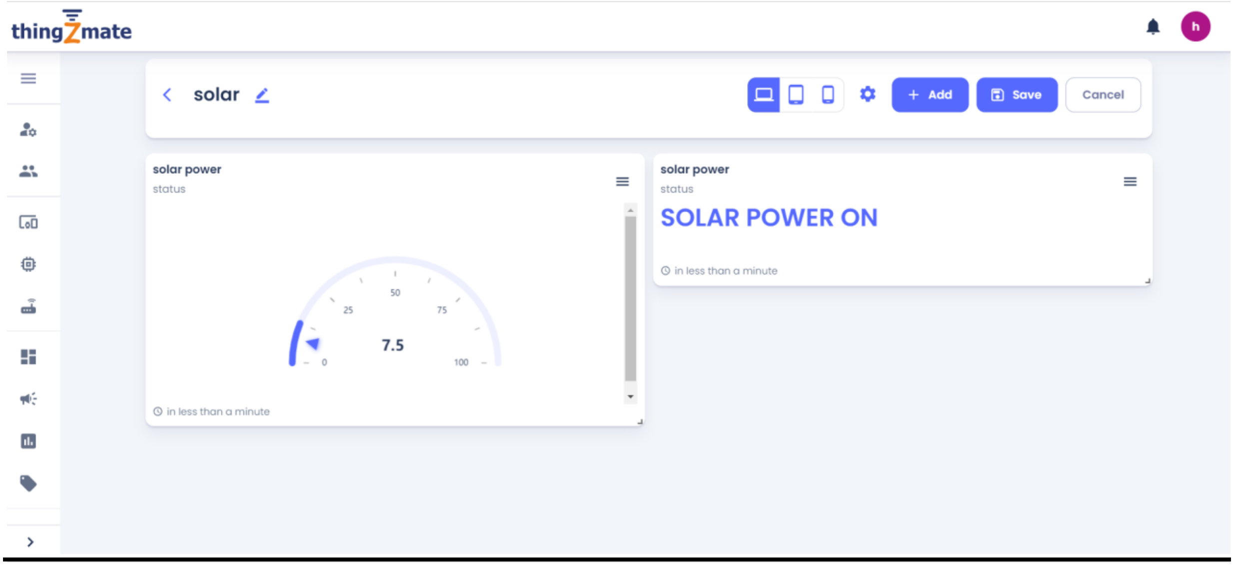
```
    float lux = lightMeter.readLightLevel(); // Read light level in lux

    Serial.print("Light: ");
    Serial.print(lux);
    Serial.println(" lx");

    String status;

    if (lux > 10) {
      status = "SOLAR POWER ON";
      Serial.println(status);
      digitalWrite(Relay, HIGH);
    } else {
      status = "SOLAR POWER OFF";
      Serial.println(status);
      digitalWrite(Relay, LOW);
      delay(1000);
    }

    // Send data to ThingzMate
    sendToCloud(status, lux);
  } else {
    Serial.println("WiFi not connected, attempting to reconnect...");
    connectToWiFi();
  }
}
```

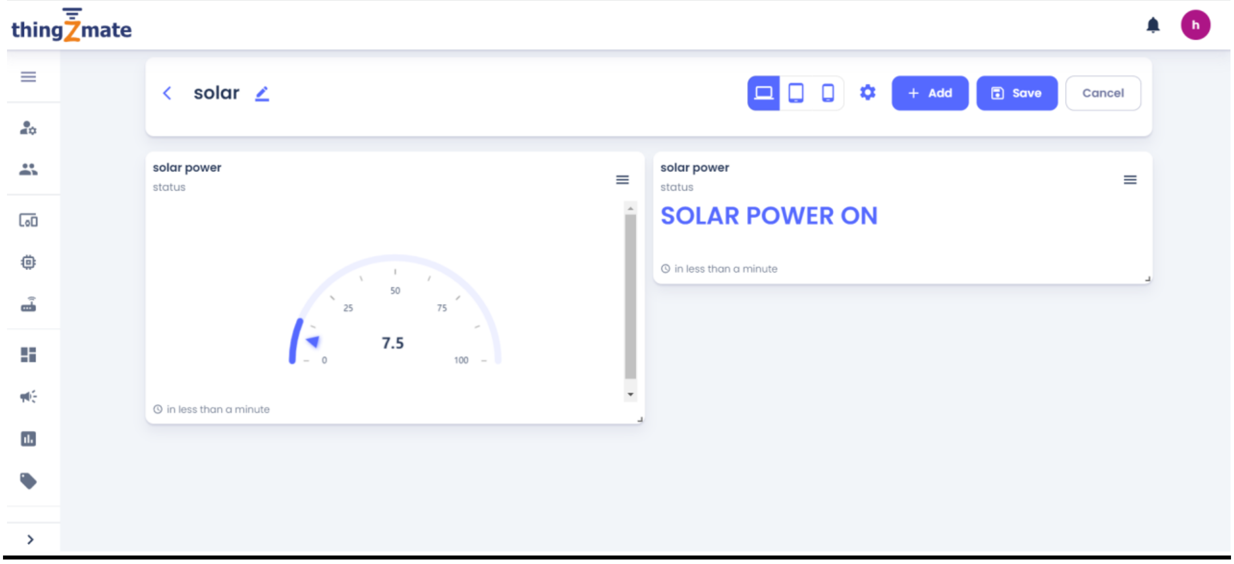
```
void sendToCloud(String status, float lux) {
  HTTPClient http;
  http.begin(serverUrl);
  http.addHeader("Content-Type", "application/json");
  http.addHeader("Authorization", AuthorizationToken);

  // Create JSON payload with the status and light level
  String payload = "{\"status\":\"" + status + "\",\"Light\":\"" + String(lux, 2) + "\"}";

  // Send POST request
  int httpResponseCode = http.POST(payload);

  if (httpResponseCode > 0) {
    String response = http.getString();
    Serial.println("HTTP Response code: " + String(httpResponseCode));
    Serial.println("Response: " + response);
  } else {
    Serial.print("Error code: ");
    Serial.println(httpResponseCode);
  }

  http.end();
}
```

Output Results



Cloud Output





Conclusion

The development of an IoT-based solar power monitoring system using the BH1750 light sensor, ESP32 microcontroller, and Thingzmate cloud platform has successfully demonstrated a reliable and efficient way to monitor solar energy performance in real-time. The system provides accurate and timely data on solar irradiance, allowing users to optimize solar panel orientation and maintenance schedules, ultimately improving energy harvest and system efficiency.