Project Report

IoT-Based Mica Miner Safety Helmet

Names of Teammates: Amisha Balwani, Anwesha Prakash

Roll Numbers: 2022UEI2829, 2022UEI2802

Subject Name: IoT Workshop

Subject Code: EIECC19

Guide: Dr. Dhananjay V. Gadre

Contents

IoT-Based Mica Miner Safety Helmet

Project Report

1 Problem Statement

Mica mining, particularly in **Jharkhand, India**, is plagued with severe hardships, especially for the tribal communities who are mostly engaged in this practice. These miners, including *young children and women*, often work in unsafe, unregulated, and primitive mining conditions.

Key challenges faced by mica miners include:

- **Toxic gas exposure:** Miners often work in narrow pits where toxic gases accumulate, posing serious respiratory risks and sometimes leading to fatal incidents due to undetected gas inhalation.
- **Mine collapses and falls:** The fragile structure of these mines, often dug without professional reinforcement, makes them highly susceptible to collapses. Sudden slips, falls, or roof collapses can trap or injure miners severely.
- **High humidity and temperature:** Working deep underground causes severe thermal discomfort, dehydration, and other health issues due to extreme temperature and humidity fluctuations.
- **No location tracking:** When miners are trapped or lost inside the mines, rescuers have no viable way to determine their location due to lack of GPS-enabled safety equipment.
- **Lack of real-time monitoring:** There is little to no remote surveillance or health safety monitoring of miners, resulting in delayed emergency responses.

Figure: A mica mine in Jharkhand, India

These issues underline the urgent need for a cost-effective, real-time safety system tailored to the unique requirements of mica mining operations.

2 Proposed System Overview

This project proposes an **IoT-based safety helmet** designed to safeguard mica miners through real-time monitoring of environmental parameters and miner status.

Main Components Used:

- **ESP32-WROOM Module:** A microcontroller with in-built WiFi capability. It is ideal for our use case since mica mines in Jharkhand are *relatively shallow*, making WiFi-based data transmission feasible without needing expensive long-range technologies like LoRaWAN.
- **MQ135 Gas Sensor:** Monitors air quality and detects hazardous gases that may build up in confined mining spaces.
- **MPU6050 Accelerometer + Gyroscope:** Used for **fall detection** and motion analysis, which can help identify sudden collapses or unconsciousness.
- **NEO-6M GPS Module:** Tracks the miner's real-time location. In case of emergencies, the system transmits coordinates to rescuers.
- **DHT11 Sensor:** Measures temperature and humidity inside the mine, enabling thermal monitoring.

- **Active Buzzer:** Can be triggered remotely by the monitoring officer to alert the miner or others in case of *emergency (SOS)* situations.
- **Dual LEDs:** Two LEDs are placed on the helmet, *varying brightness levels based on temperature and gas concentration*, providing visual alerts to the miner.
- **Lock-In Switch:** Once pressed by the miner in distress, this button sends locked GPS coordinates to the cloud. Simultaneously, a notification is sent to the monitoring personnel.

3 Circuit Diagram

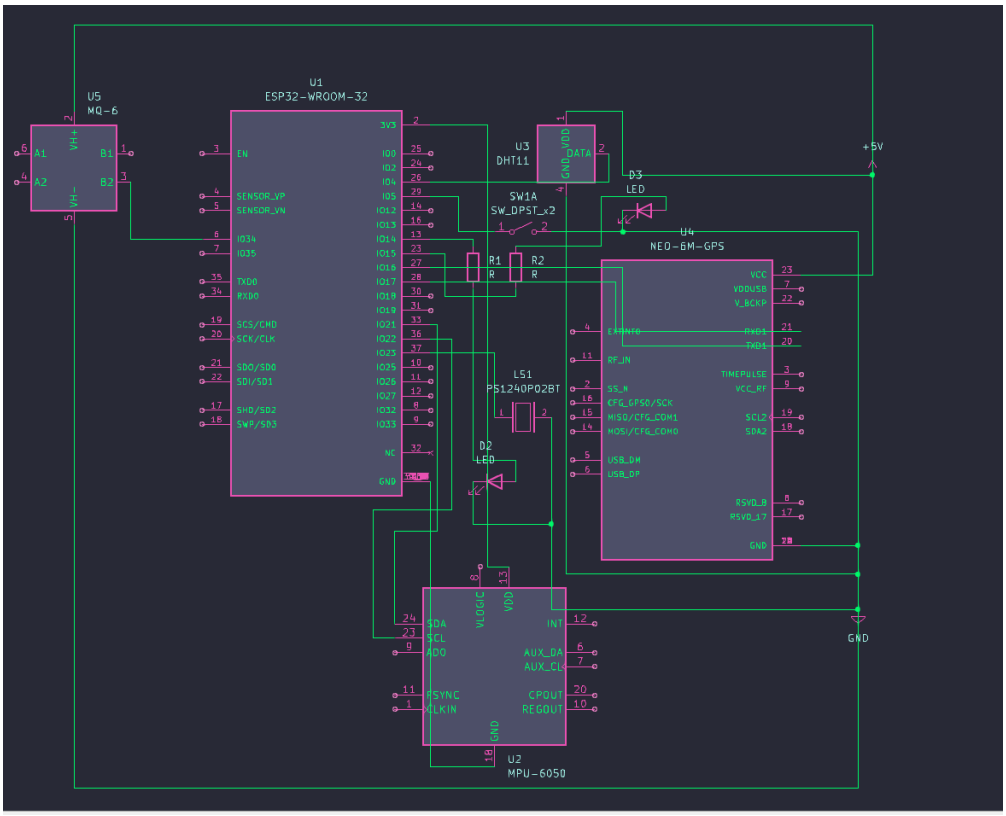


Figure: Circuit diagram of IoT Miner Helmet system

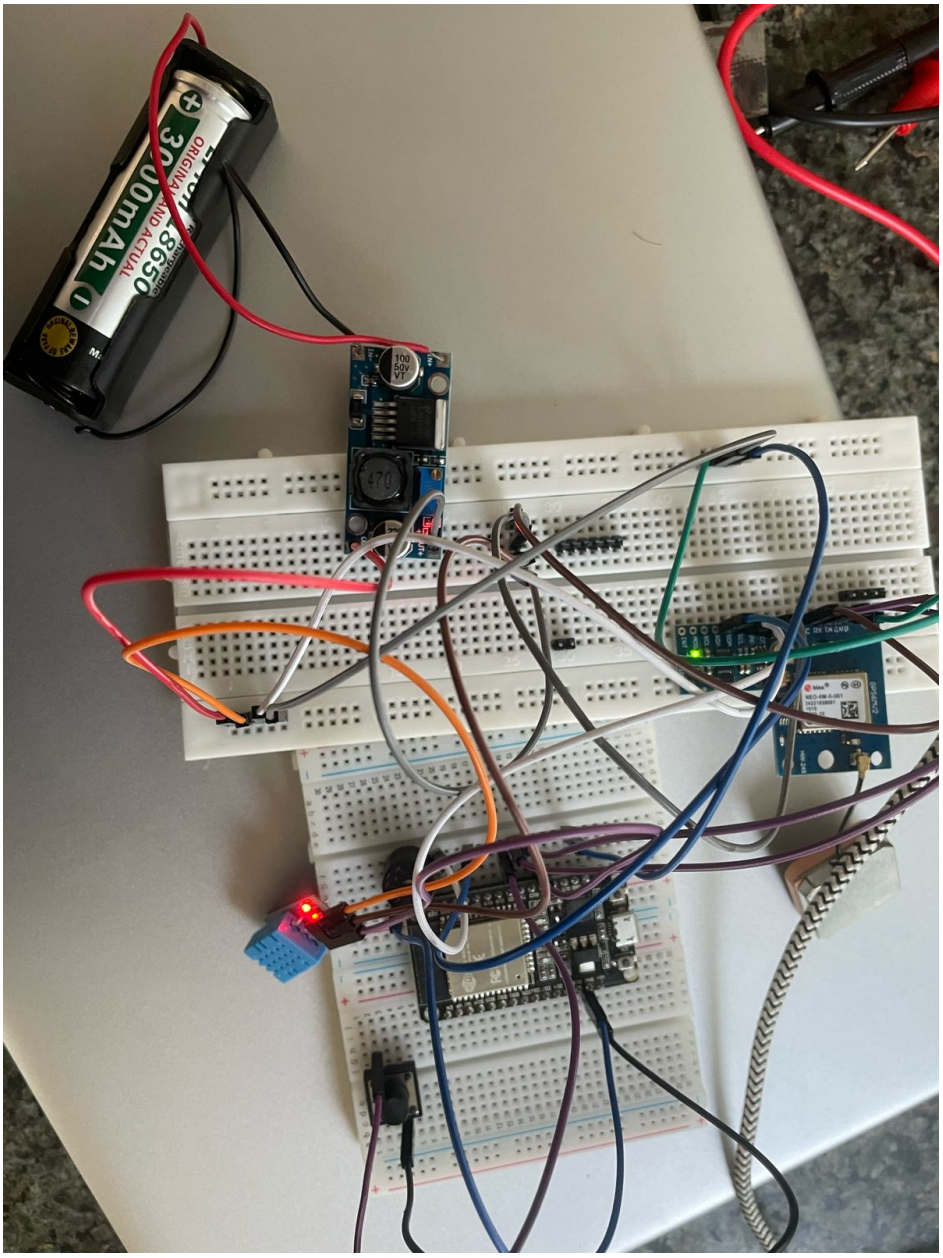


Figure: Primitive circuit on breadboard. Note that power supply is provided using a buck converter that was later designed into a proper power supply as later discussed in the document.

4 Platforms

4.1 Blynk IoT Platform

Blynk is a robust and versatile Internet of Things (IoT) platform designed to allow remote device management, real-time sensor monitoring, and interactive control via both mobile

and web dashboards. It supports seamless integration with microcontrollers like the ESP32 and allows data transmission over various protocols such as WiFi, GSM, and Ethernet.

In our project, Blynk plays the role of the **cloud connectivity and visualization layer**, receiving real-time inputs from sensors and relaying important alerts to the person monitoring the system. The miner's helmet transmits environmental readings like temperature, gas levels, and motion data to the Blynk cloud, where it is visualized using intuitive widgets.

Key functionalities leveraged in our project include:

- **Real-time dashboards** for sensor value monitoring.
- **Notification services** to alert in case of hazardous gas levels, abnormal temperatures, or falls.
- **Location tracking** using the GPS coordinates sent from the miner's helmet.
- **Remote control** for triggering alarms like the active buzzer from the monitoring system via a virtual button.

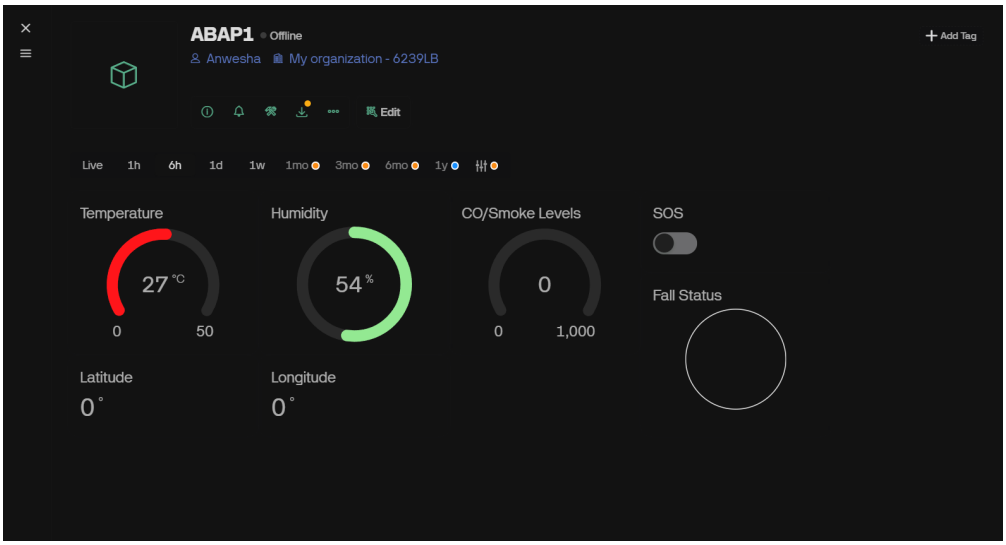


Figure: Blynk Dashboard

4.2 PlatformIO on Visual Studio Code

To program and manage the firmware of the ESP32 microcontroller, we used **PlatformIO**, an open-source embedded development ecosystem integrated with Visual Studio Code (VS

Code). PlatformIO provides a professional-grade development environment with tools tailored for IoT and embedded systems projects.

Advantages of using PlatformIO:

- **Multi-board and cross-platform support** — making it flexible for future scalability.
- **Integrated Library Manager** that makes it easy to include and manage dependencies like Blynk, TinyGPS++, and sensor libraries.
- **Auto-completion and IntelliSense** features that enhance coding speed and accuracy.
- **Integrated terminal and serial monitor** for debugging and real-time data visualization.
- **Faster compilation and modular code organization** compared to traditional Arduino IDE.

Using PlatformIO helped us maintain a structured codebase, manage libraries efficiently, and debug hardware interactions with greater ease. Its integration with VS Code allowed seamless collaboration and version control through Git.

5 C++ Code for Implementation

5.1 platformio.ini

```
[env:hornbill32dev]
platform = espressif32
board = hornbill32dev
framework = arduino
monitor_speed = 115200
upload_speed = 921600

lib_deps =
  adafruit/Adafruit Unified Sensor@^1.1.9
  adafruit/DHT sensor library@^1.4.4
  mikalhart/TinyGPSPlus@^1.0.3
  bblanchon/ArduinoJson@^6.21.2
  jrowberg/i2cdevlib@^1.0.3
  electroniccats/MQUnifiedsensor@^1.1.1

build_flags =
  -D DHTPIN=4
  -D DHTTYPE=DHT11
```

Listing 1: PlatformIO Configuration File

5.2 main.cpp

```
#define BLYNK_TEMPLATE_ID "TMPL3xLy0gxWr"
#define BLYNK_TEMPLATE_NAME "ABAPmines"
#define BLYNK_AUTH_TOKEN "aaVaAlXo9iAgPH4WR-poBge85kveiTHD"

// #include <WiFi.h>
// #include <BlynkSimpleEsp32.h>
#include <TinyGPS++.h>
#include <HardwareSerial.h>
#include <DHT.h>
#include <Wire.h>
#include <MPU6050.h>

// DHT11
#define DHTPIN 4
#define DHTTYPE DHT11
```

```
16      DHT dht(DHTPIN, DHTTYPE);
17
18      // MPU6050
19      MPU6050 mpu;
20      const int jerkThreshold = 32000;
21
22      // Buzzer
23      #define BUZZER_PIN 23
24
25      // Switch
26      #define SWITCH_PIN 5
27
28      // Gas sensor
29      #define GAS_SENSOR_1 34
30      #define GAS_LED 13
31
32      // Temperature LED
33      #define TEMP_LED 15
34      int ledBrightness = 128;
35      float previousTemp = -100.0;
36
37      // Switch debounce
38      bool lastSwitchState = HIGH;
39      unsigned long lastDebounceTime = 0;
40      const unsigned long debounceDelay = 50;
41
42      bool sosActivated = false;
43
44      // GPS Setup
45      static const int RXPin = 16, TXPin = 17;
46      static const uint32_t GPSBaud = 9600;
47      TinyGPSPlus gps;
48      HardwareSerial gpsSerial(2);  // Use UART2
49
50      void activateBuzzer();
51      void deactivateBuzzer();
52      void checkFall();
53
54      void setup() {
55          Serial.begin(115200);
56          gpsSerial.begin(GPSBaud, SERIAL_8N1, RXPin, TXPin);
57
58          dht.begin();
```

```
        Wire.begin(21, 22);

        mpu.initialize();
        if (mpu.testConnection()) {
            Serial.println("MPU6050 Connected");
        } else {
            Serial.println("MPU6050 Connection Failed!");
        }

        pinMode(BUZZER_PIN, OUTPUT);
        digitalWrite(BUZZER_PIN, LOW);

        pinMode(SWITCH_PIN, INPUT_PULLUP);
        pinMode(GAS_SENSOR_1, INPUT);
        pinMode(GAS_LED, OUTPUT);
        digitalWrite(GAS_LED, LOW);

        ledcAttachPin(TEMP_LED, 0);
        ledcSetup(0, 5000, 8);
        ledcWrite(0, ledBrightness);

        Serial.println("System Initialized...");
    }

    void loop() {
        // Blynk.run(); // if using Blynk

        // Read and encode GPS data
        while (gpsSerial.available() > 0) {
            gps.encode(gpsSerial.read());
        }

        // GPS connection status
        if (gps.satellites.isValid()) {
            Serial.print("Satellites connected: ");
            Serial.println(gps.satellites.value());
        } else {
            Serial.println("Searching for GPS satellites...");
        }

        // GPS location
        if (gps.location.isValid()) {
            Serial.print("Latitude: ");
```

```
            Serial.print(gps.location.lat(), 6);
            Serial.print(" | Longitude: ");
            Serial.println(gps.location.lng(), 6);
        } else {
            Serial.println("Waiting for valid GPS fix...");
        }

        // DHT sensor
        float temperature = dht.readTemperature();
        float humidity = dht.readHumidity();

        if (!isnan(temperature) && !isnan(humidity)) {
            Serial.print("Temperature: "); Serial.print(temperature);
               Serial.println(" \u00B0C");
            Serial.print("Humidity: "); Serial.print(humidity); Serial.
               println(" %");

            if (previousTemp != -100.0 && abs(temperature - previousTemp)
               >= 0.5) {
                if (temperature > previousTemp) {
                    ledBrightness = min(255, ledBrightness + 20);
                } else {
                    ledBrightness = max(0, ledBrightness - 20);
                }
                ledcWrite(0, ledBrightness);
                Serial.print("LED Brightness Updated: ");
                Serial.println(ledBrightness);
            }
            previousTemp = temperature;
        }

        // Gas sensor
        int gas1 = analogRead(GAS_SENSOR_1);
        Serial.print("Gas Sensor Reading: ");
        Serial.println(gas1);

        if (gas1 > 500) {
            digitalWrite(GAS_LED, HIGH);
        } else {
            digitalWrite(GAS_LED, LOW);
        }

        // Fall detection
```

```
142         checkFall();
143
144         // Switch handling with debounce
145         bool currentSwitchState = digitalRead(SWITCH_PIN);  // Read switch
                state
146
147         if (currentSwitchState == HIGH) {  // If the switch is pressed (
                LOW)
148             /*float lat = gps.location.isValid() ? gps.location.lat() :
                    0.0;
149             float lon = gps.location.isValid() ? gps.location.lng() : 0.0;
150
151             Serial.print("LOCKED IN! Lat: "); Serial.print(lat, 6);
152             Serial.print(" Lon: "); Serial.println(lon, 6);
153             // Activate the buzzer when switch is pressed */
154             Serial.print("LOCKED IN");
155         }
156
157         delay(200);
158     }
159
160     void checkFall() {
161         int16_t gx, gy, gz;
162         mpu.getRotation(&gx, &gy, &gz);
163
164         Serial.print("Gyro (X,Y,Z): ");
165         Serial.print(gx); Serial.print(", ");
166         Serial.print(gy); Serial.print(", ");
167         Serial.println(gz);
168
169         if (abs(gx) > jerkThreshold || abs(gy) > jerkThreshold || abs(gz)
                > jerkThreshold) {
170             Serial.println("Jerk Detected: Possible Fall!");
171             // Optional: activateBuzzer();
172         }
173     }
174
175     void activateBuzzer() {
176         ledcAttachPin(BUZZER_PIN, 1);
177         ledcSetup(1, 2000, 8);
178         ledcWrite(1, 128);
179         Serial.println("Buzzer ON!");
180     }
```

```
181
182     void deactivateBuzzer() {
183         ledcWrite(1, 0);
184         Serial.println("Buzzer OFF.");
185     }
```

Listing 2: ESP32 Code for Miner Safety Helmet

6 Power Supply Circuit

In this circuit, we have designed a basic regulated power supply using Li-Ion batteries, a voltage regulator, filtering capacitors, and an indicator LED. The goal is to convert the unregulated 7.4V from batteries into a stable 5V DC output using the 7805 voltage regulator IC.

- Two **3.7V, 3000mAh Li-Ion batteries** are connected in series to obtain a total voltage of **7.4V**. This configuration increases the voltage while keeping the capacity (Ah) the same.
- A **switch** is introduced in the positive line of the battery output to allow manual control over the power supply. The negative terminal of the switch is connected to the common ground of the system.
- The output from the switch is fed to the **input pin of the IC 7805**, a linear voltage regulator that steps down and regulates the voltage to a constant **5V DC**.
- To minimize voltage ripple and filter out high-frequency noise, we use two **electrolytic capacitors**:
  - A **1000μF, 25V capacitor** is connected at the **input side** of the 7805. This helps stabilize the incoming voltage and smooth out fluctuations from the battery.
  - A **470μF, 5V capacitor** is connected at the **output side** of the 7805. It filters the output voltage, reducing noise and improving the stability of the regulated 5V.
- A **Light Emitting Diode (LED)** is connected at the output to serve as a visual indicator that shows whether the circuit is providing voltage.
- A **current-limiting resistor** is connected in series with the LED to prevent excessive current from flowing through the LED, which could otherwise damage it. The resistor ensures the LED operates within safe current limits, typically 10–20mA.

This setup ensures a clean and regulated 5V power supply suitable for powering sensors in our project so as to not overburden ESP32 with the duty of providing power to the sensors. The 7805 voltage regulator is a popular choice for such applications due to its simplicity, reliability, and ease of use. It can handle input voltages up to 35V and provides a maximum output current of 1A, making it suitable for low-power applications like this one.

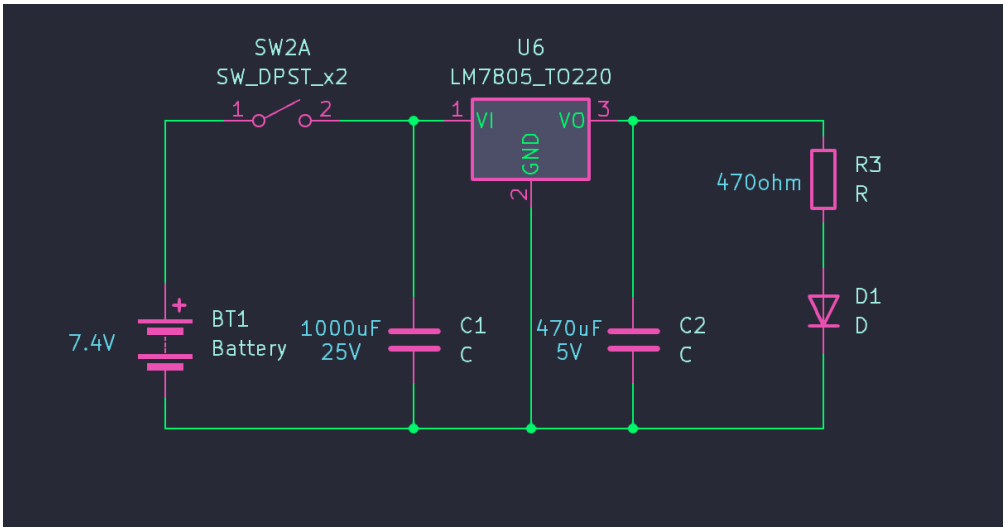


Figure: Circuit diagram of Voltage regulator for 5V supply

7 Soldering the Circuit on FR4 Zero PCB

FR4 Zero PCB boards, often referred to as general-purpose boards or stripboards, are made from high-grade fiberglass-reinforced epoxy laminate sheets. These boards are **non-conductive**, **heat resistant**, and contain pre-drilled holes with standard spacing (0.1 inch apart), making them ideal for prototyping electronic circuits in a compact and organized manner.

Why FR4 Zero Board Was Used:

- It provides a **rigid and durable base** for permanent soldering.
- Unlike breadboards, the components stay **firmly fixed** and are less prone to disconnection during movement.
- It supports **custom wiring** and tracks creation via solder bridges and jumper wires.
- The board is compact and **space-efficient**, suitable for wearable projects like a helmet.

Step-by-Step Soldering Process Followed:

1. **Component Planning and Layout:** A rough sketch of the circuit was drawn and the layout was first tested on a breadboard. Once validated, all components like the ESP32 board, gas sensor (MQ135), MPU6050, GPS module, DHT11, buzzer, and LEDs were carefully arranged on the FR4 zero board to minimize jumper wires and make the design compact.

2. **Pin Marking and Spacing:** Each pin of the components was marked with a permanent marker and care was taken to ensure spacing for heat dissipation, especially around the ESP32 module.

3. **Soldering Tools Used:**

   - *25W soldering iron with fine tip*
   - *Good quality lead solder wire with flux core*
   - *Desoldering pump for corrections*
   - *Tweezers and wire cutter for trimming leads*

4. **Soldering Process:**

   - The tip of the soldering iron was first tinned with solder to ensure heat conductivity.
   - Each component was inserted into the board and soldered by heating the pad and lead simultaneously before applying solder wire.
   - Extra care was taken not to overheat sensitive components like the MPU6050 and DHT11.
   - A minimal amount of solder was used to avoid shorts, and any excess was removed with a desoldering pump.

5. **Wire Connections and Jumpers:** For non-adjacent connections, insulated copper jumper wires were used and routed neatly on the back side of the PCB. Some connections were reinforced with header pins.

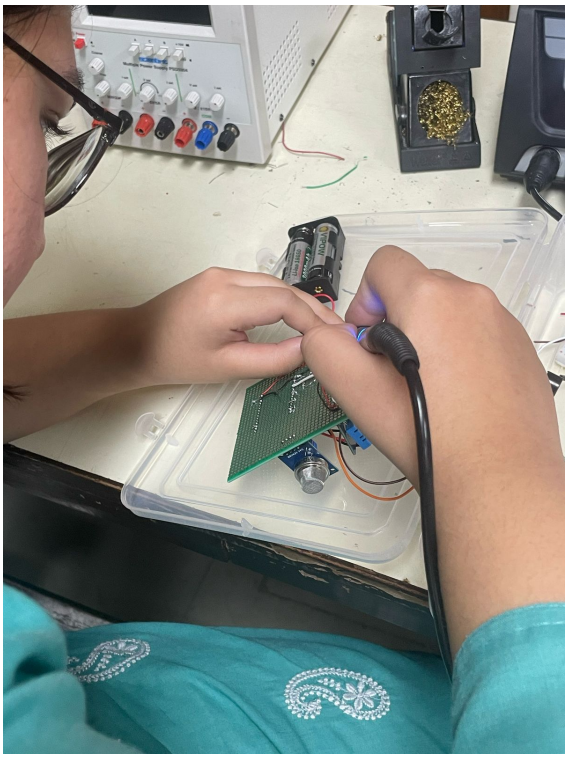
6. **Testing and Continuity Checks:** A multimeter was used to check continuity between all critical joints. Faulty joints were reheated and fixed. The circuit was powered using a regulated supply to test for correct operation and absence of shorts.

7. **Fixation and Final Assembly:** The board was trimmed to fit the helmet profile. The assembled board was then secured inside a protective enclosure, mounted firmly to avoid vibration or damage during movement.

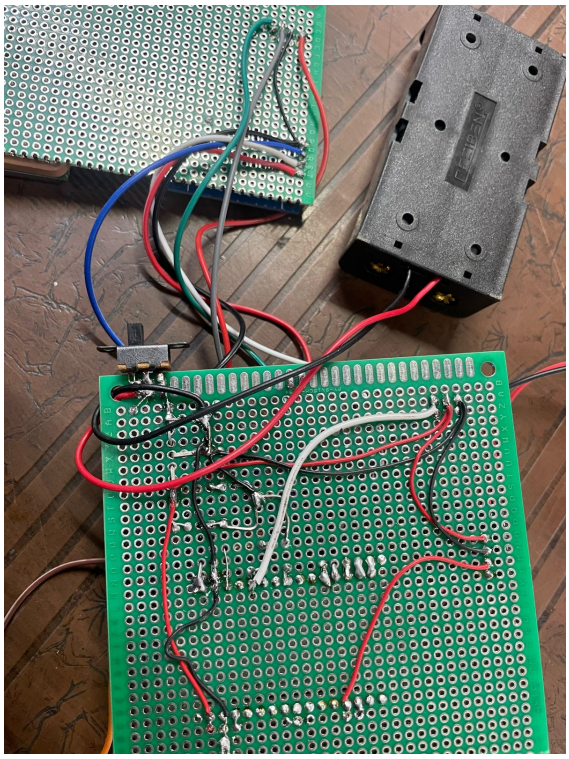
Challenges Faced:

- Some sensor modules have different pin spacings or unaligned layouts that made it difficult to insert directly into the board.
- The ESP32's wide footprint required soldering it via male headers to prevent damage from excess heat.
- Tracing short circuits and cold joints required patient rechecking using a multimeter and microscope.
- Ensuring that high-signal integrity was maintained on I2C lines (for MPU6050 and GPS) was tricky on a hand-soldered board.

The successful soldering and mounting of the circuit on the FR4 zero board ensured a stable and field-ready configuration for our IoT miner safety helmet. This phase was crucial in transitioning from the prototyping stage to a deployable hardware solution.



(a) Soldering in progress



(b) Final soldered circuit

Figure: Soldering process and final circuit assembly

8 Enclosure and Helmet Mounting

Ensuring a practical and rugged enclosure for our IoT-based miner helmet was crucial for real-world deployment. Given the harsh underground mining conditions—such as high humidity, temperature, and narrow passageways—the entire circuit had to be securely housed on a standard miner's helmet without adding excessive bulk or discomfort.

The following objectives guided our enclosure design:

- **Compactness:** The circuitry and battery should be mounted in a minimal footprint to avoid interfering with the miner's movement.
- **Stability:** The board and wires should remain fixed during continuous motion and shocks.
- **Accessibility:** The ESP32 and charging pins should remain accessible for programming, debugging, and battery recharge.
- **Visibility and Safety:** The design should not obstruct the miner's vision or balance.

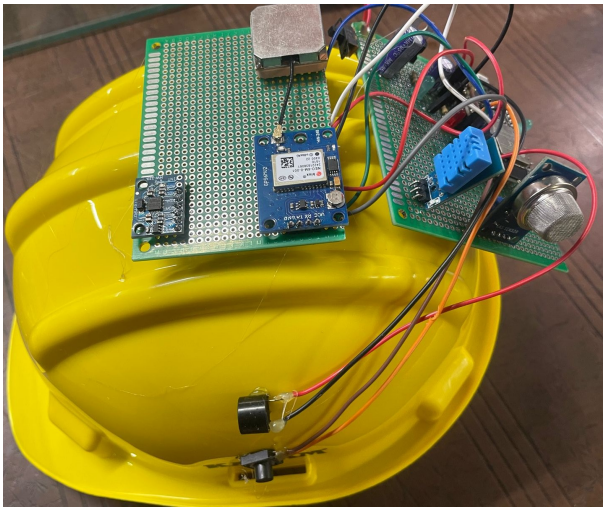
8.1 Mounting Process

The assembled circuit on an FR4 zero PCB board was mounted on the top rear section of the helmet. After testing multiple adhesives and mounting strategies, we finalized the use of a **hot glue gun** for the following reasons:

- It provides strong adhesion to plastic surfaces commonly used in helmets.
- It allows easy removal or repositioning if necessary.
- It absorbs minor vibrations and mechanical stress.

Wires from the sensors were routed along the helmet contour and fixed in place using dabs of hot glue to avoid entanglement during field use.

(a) Standard miner helmet used for circuit mounting



(b) Helmet with mounted circuitry and sensors

8.2 Challenges Faced

Mounting the circuit presented several real-world challenges, especially considering the need for field durability:

- **Surface unevenness:** The curved surface of the helmet made it difficult to get the board to lie flush. We overcame this by layering hot glue strategically under the PCB for contact uniformity.
- **Glue cooling time:** Hot glue begins to solidify within seconds. Proper positioning had to be achieved very quickly to avoid asymmetry.
- **Thermal sensitivity:** Prolonged application of hot glue near sensor wiring risked damaging heat-sensitive components, so we used quick, low-volume bursts of glue.
- **Weight balancing:** Mounting components on one side of the helmet made it tilt initially. We had to reposition the battery and ESP32 module to maintain center of gravity.

Durability Testing

Once mounted, the helmet was subjected to:

- **Shake tests** to simulate miner movement

- **Drop tests** from a height of 1 meter to evaluate mount resilience
- **Mock field runs** with volunteers wearing the helmet in dim-lit, humid tunnels

The hot glue-based mounting mechanism withstood the stress tests without dislodgement, validating the ruggedness of our enclosure design.

9 System Workflow

The complete workflow of our IoT-based safety helmet for mica miners is designed to ensure real-time environmental monitoring, miner localization, and emergency alert generation. The process integrates multiple embedded sensors, a microcontroller, wireless communication, and a cloud-based dashboard for seamless data visualization and remote control.

Working:

- The ESP32 microcontroller serves as the central unit, constantly reading data from all connected sensors including gas (MQ135), temperature and humidity (DHT11), motion (MPU6050), and location (Neo 6M GPS).
- The collected data is preprocessed and transmitted via WiFi to the **Blynk Cloud**, where it is logged and displayed on a mobile/web dashboard through widgets like value displays, notification triggers, and maps.
- If the gas concentration exceeds a safe threshold (e.g., due to methane, ammonia, or benzene), an automated alert is sent to the monitoring personnel via Blynk notifications. The onboard LED tied to gas readings also increases its brightness to visually warn the miner locally.
- Simultaneously, temperature and humidity readings are tracked, and another LED changes brightness based on temperature levels—allowing the miner to gauge ambient heat visually without checking a display.
- The MPU6050 sensor detects falls or abnormal movements by analyzing sudden changes in acceleration and orientation. If a fall is detected (based on threshold conditions in X, Y, Z axes), an emergency alert is triggered.
- The GPS module continuously sends coordinates to Blynk's map widget, ensuring real-time tracking of the miner's position inside or near the mining site. This is crucial during emergencies like cave-ins or when a miner goes missing.

- A virtual button is available on the Blynk dashboard, which when pressed by the monitoring authority, activates the onboard **active buzzer** mounted on the helmet. This acts as an audible SOS signal for the miner in danger, alerting them and people nearby.
- All sensor data is not only visualized in real time but also logged in the cloud for post-incident analysis, helping in improving safety standards and audit trails for mining operations.
- The system operates autonomously once powered, requiring no user intervention from the miner, thereby suiting its use in rugged, real-world underground environments.

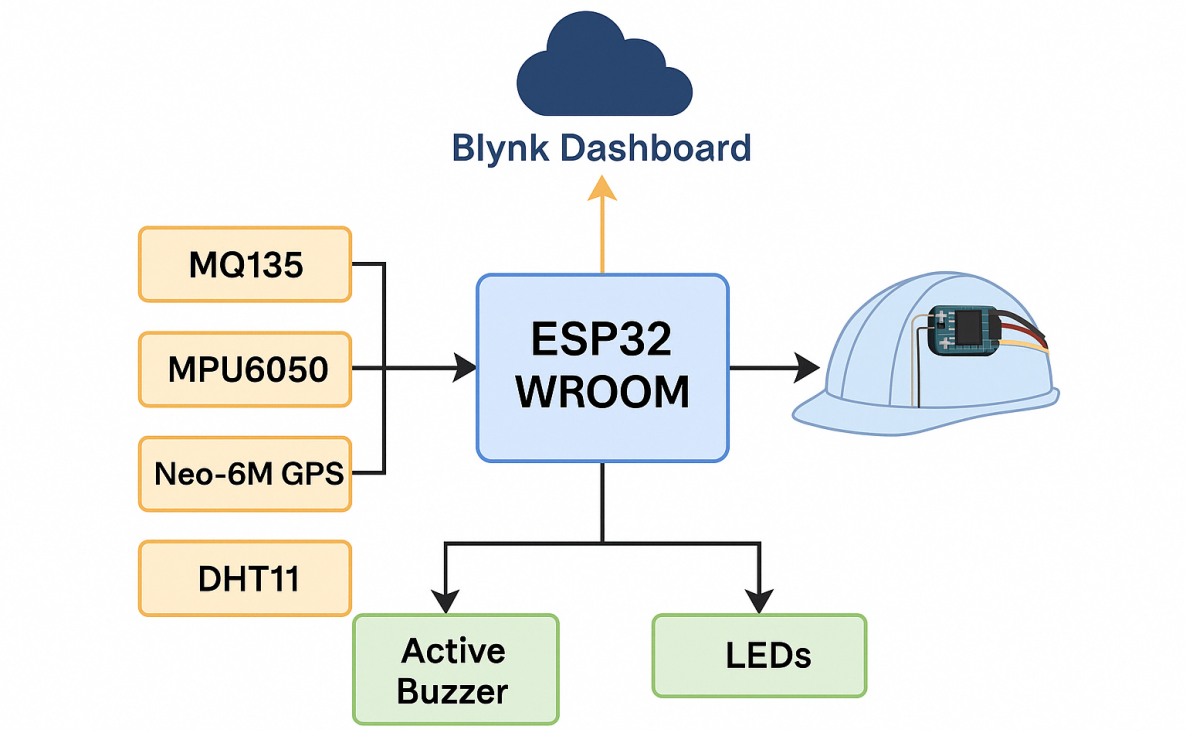


Figure: System workflow diagram for the IoT-based helmet

10 Limitations and Future Scope

While our IoT-based smart helmet for mica miners provides a practical and low-cost solution for improving underground mining safety, it is important to acknowledge that this implementation is currently in the **prototype stage**. As with any prototype, there are several limitations in terms of durability, accuracy, scalability, and real-world deployment. Understanding these constraints is essential for refining and advancing the project.

10.1 Limitations

- **Prototype-level integration:** The entire circuit has been manually assembled on an FR4 zero PCB and mounted using a hot glue gun. This method is not industrial-grade and may not withstand long-term exposure to the harsh mining environment including high humidity, vibrations, or impact forces.
- **Limited power management:** The current version lacks an integrated power management system or battery backup, which is crucial for reliable field operation. Power failure or disconnection leads to complete system inactivity.
- **WiFi dependency:** The system currently relies solely on WiFi connectivity for communication with the Blynk cloud. However, underground mining environments usually lack stable WiFi coverage. In such cases, GSM-based modules or LoRa communication would be better suited.
- **GPS accuracy in closed environments:** The Neo 6M GPS module may not provide reliable signals deep inside mines due to lack of satellite visibility, limiting its effectiveness in real-time tracking when miners are underground.
- **Sensor calibration and accuracy:** All sensors used are low-cost modules intended for academic purposes. Gas concentration thresholds and fall detection sensitivity may not be calibrated precisely, potentially resulting in false positives or missed alerts.
- **No onboard display or voice feedback:** The current prototype does not include a local display or audio guidance system, which could help the miner receive critical information without relying on visual indicators like LEDs alone.

10.2 Future Scope

- **Industrial-grade hardware design:** The next version could include a custom-designed PCB housed inside a weatherproof, rugged enclosure that complies with IP65+ standards for dust and water resistance.
- **Battery and solar power integration:** Introducing a rechargeable battery along with solar cell charging can make the device truly portable and independent of external power supply.
- **Alternative communication protocols:** To counter the WiFi limitations, we can implement GSM (SIM800L), LoRa, or NB-IoT for data transmission in remote or underground areas where conventional networks are unavailable.

- **Edge processing and ML integration:** Deploying lightweight machine learning models on the ESP32 for anomaly detection (e.g., unusual gas patterns or miner behavior) can add a layer of intelligent decision-making to the system.
- **Wearability and ergonomic design:** The helmet can be redesigned to reduce weight and improve balance by embedding all electronics inside compact enclosures. Detachable modules can also be explored.
- **Expansion to multi-miner networks:** Future iterations could include mesh networking capabilities for tracking and monitoring multiple miners simultaneously using a centralized base station and distributed nodes.
- **Emergency response coordination:** Integration with government safety protocols and local emergency services (through APIs or automated calling systems) can enhance post-incident response time and reduce casualties.
- **Data analytics dashboard:** A web-based analytics interface can be built to analyze historical sensor data, detect hazardous trends over time, and improve preventive maintenance strategies in mines.

This prototype demonstrates the **proof-of-concept** for an affordable miner safety device. With the right engineering enhancements and field testing, it can be developed into a **deployable safety standard** that could drastically improve lives in unregulated mining sectors.

11 References and Bibliography

11.1 Tutorials and Resources

- **YouTube Demonstration:** https://www.youtube.com/watch?v=jVSgC2HRkKg
- **ESP32 with Neo-6M GPS Module Tutorial:** https://randomnerdtutorials.com/esp32-neo-6m-gps-module-arduino/
- **ESP32 with MPU6050 Accelerometer and Gyroscope Tutorial:** https://randomnerdtutorials.com/esp32-mpu-6050-accelerometer-gyroscope-arduino/

11.2 Datasheets

- **ESP32-WROOM-32:** https://www.espressif.com/sites/default/files/documentation/esp32-wroom-32_datasheet_en.pdf
- **MQ135 Gas Sensor:** https://www.olimex.com/Products/Components/Sensors/Gas/SNS-MQ135/resources/SNS-MQ135.pdf
- **MPU6050 Accelerometer and Gyroscope:** https://invensense.tdk.com/wp-content/uploads/2015/02/MPU-6000-Datasheet1.pdf
- **Neo 6M GPS Module:** https://content.u-blox.com/sites/default/files/products/documents/NEO-6_DataSheet_%28GPS.G6-HW-09005%29.pdf
- **DHT11 Temperature and Humidity Sensor:** https://www.mouser.com/datasheet/2/758/DHT11-Technical-Data-Sheet-Translated-Version-1143054.pdf
- **Active Buzzer (LTE12 Series):** https://www.electronicoscaldas.com/datasheet/LTE12-Series.pdf
- **Standard LED:** https://www.farnell.com/datasheets/1498852.pdf