

Sprint 2 – Power Management & Remote Battery Monitoring

Project: Project Echo

Name: Hari

Sprint: 2

Task: Investigate and implement power management solution for IoT devices

1. Objective

The objective of this task is to design and implement a technical solution that enables:

- Remote monitoring of battery health statistics of IoT devices.
- Intelligent power cycling to reduce energy consumption.
- Wake-on-sound functionality to activate the device only when animal activity is detected.
- Safe handling of low-power and power-failure scenarios to preserve device functionality and data.

This solution operates at the **IoT device level** and integrates with the existing Project Echo backend infrastructure.

2. System Overview

The IoT device operates primarily in a **low-power sleep mode** to conserve energy. It periodically reports battery health information to the backend system and wakes only when environmental sound activity is detected.

Device Power States

- **Deep Sleep Mode:** Minimal power consumption.
- **Listening Mode:** Microphone active, waiting for sound interrupt.
- **Active Mode:** Audio recording and processing.
- **Transmit Mode:** Data and battery statistics sent to backend.
- **Ultra-Low Power Mode:** Triggered during critical battery levels.

3. Power Cycling and Wake-on-Sound Logic

The device uses an **interrupt-based microphone trigger** to wake from sleep when sound exceeds a predefined threshold.

Operational Flow

Device Start



Read Battery Voltage



Send Battery Health Data to Server



Enter Deep Sleep Mode



Sound Detected?

├─ No → Remain in Sleep

└─ Yes → Wake Device

→ Record Audio

→ Process Audio

→ Transmit Data

→ Return to Sleep

This approach ensures the device remains energy efficient while still responding to environmental activity.

4. Technical Implementation (Pseudo-Code)

The following pseudo-code represents the firmware-level implementation logic.

```
void setup() {  
    initMicrophoneInterrupt();
```

```
    initBatteryMonitoring();  
}  
  
void loop() {  
    float batteryVoltage = readBatteryADC();  
    int batteryPercent = calculateBatteryPercentage(batteryVoltage);  
  
    sendBatteryStats(batteryVoltage, batteryPercent);  
  
    if (batteryPercent < 15) {  
        enterUltraLowPowerMode();  
    }  
  
    enterDeepSleep();  
}  
  
void onSoundDetected() {  
    wakeDevice();  
    startAudioRecording();  
    processAudio();  
    sendAudioData();  
    enterDeepSleep();  
}
```

5. Remote Battery Health Monitoring

Battery health statistics are transmitted periodically to the backend system, allowing remote diagnostics and maintenance planning.

Monitored Parameters

- Battery voltage
- Battery percentage
- Charging status (solar / battery)
- Current power mode

Example Telemetry Payload

```
{  
  
  "device_id": "project_echo_01",  
  
  "battery_voltage": 3.68,  
  
  "battery_percentage": 72,  
  
  "charging_status": "solar",  
  
  "power_mode": "sleep"  
}
```

This data enables real-time monitoring and alerting when devices approach critical power levels.

6. Power Failure and Low-Battery Handling

To ensure reliability in field deployment, the following fail-safe mechanisms are implemented:

- When battery drops below a critical threshold:
 - Recording and processing are disabled.
 - A final status update is sent to the server.
 - Device enters ultra-low power mode.
- On power restoration:
 - Device resumes normal operation.
 - Unsent data can be retransmitted if applicable.

This approach minimizes data loss and prevents battery damage.

7. Integration with Existing System

The power management solution operates upstream of the existing Project Echo data flow. Battery and device status telemetry integrates with the Echo Engine and API for centralized monitoring, while audio data follows the existing processing pipeline.

8. Future Improvements

- MPPT-based solar charging for improved efficiency.
- Adaptive sleep intervals based on battery health.
- Predictive battery health analytics using historical data.
- Remote firmware updates for power management tuning.

9. Summary

This task delivers a complete **technical implementation design** for power management, including:

- Remote battery health monitoring
- Energy-efficient power cycling
- Wake-on-sound activation
- Safe power failure handling

The solution is implementation-ready and can be directly translated into production firmware in future sprints.