

LoRa Sensor Network - Complete Setup Guide

Overview

This system consists of:

- **Master Node:** Collects data, logs to SD card, provides web dashboard, uploads to cloud
 - **Slave Nodes (1-3):** Collect sensor data (temp, pressure, altitude, GPS, battery) and transmit via LoRa
-

Hardware Requirements

Master Node

- ESP32 Development Board
- SX127x LoRa Module (433MHz)
- SD Card Module
- Micro SD Card (4GB+ recommended)
- Active Buzzer
- GPS Module (NEO-6M or similar)

Slave Nodes (each)

- ESP32 Development Board
- SX127x LoRa Module (433MHz)
- BMP280 or BMP085 Pressure/Temperature Sensor
- LED
- Push Button

- GPS Module (NEO-6M or similar)
 - Battery Monitor Circuit (150k Ω + 100k Ω voltage divider)
 - 2S Li-ion Battery (7.4V nominal)
-

Required Libraries

Install these via Arduino Library Manager:

- LoRa by Sandeep Mistry
- ArduinoJson by Benoit Blanchon
- Adafruit BMP280 Library
- Adafruit BMP085 Library
- Adafruit Unified Sensor
- TinyGPSPlus by Mikal Hart

Configuration Steps

Step 1: Configure Node IDs

For each slave node, edit the Node ID:

```
cpp
#define NODE_ID "NODE1" // Change to NODE1, NODE2, or NODE3
```

Step 2: WiFi Configuration

Master Node - Choose WiFi mode:

Option A: ESP32 Creates Hotspot (Default)

cpp

```
#define WIFI_MODE_AP true
const char* AP_SSID = "LoRa_Network";
const char* AP_PASS = "lora12345";
```

Option B: Connect to Laptop Hotspot

cpp

```
#define WIFI_MODE_AP false
const char* STA_SSID = "YourLaptopHotspot";
const char* STA_PASS = "yourpassword";
```

Step 3: Ubidots Cloud Setup

1. Create free account at ubidots.com
2. Get your API token from Account → API Credentials
3. Update master code:

cpp

```
#define UBIDOTS_TOKEN "YOUR_TOKEN_HERE"
```

Step 4: Alert Thresholds

Customize in master code:

cpp

```
#define FALL_THRESHOLD 10.0 // Altitude drop in meters
#define BATTERY_LOW 15 // Battery warning %
#define MOVEMENT_THRESHOLD 100.0 // Movement in meters
#define OFFLINE_THRESHOLD 900000 // 15 minutes
```

🔌 Pin Connections

Master Node Connections

LoRa SX127x:

- NSS → GPIO 5
- RST → GPIO 14
- DIO0 → GPIO 26
- SCK → GPIO 18
- MISO → GPIO 19
- MOSI → GPIO 23

SD Card Module:

- CS → GPIO 15
- SCK → GPIO 18 (shared with LoRa)
- MISO → GPIO 19 (shared with LoRa)
- MOSI → GPIO 23 (shared with LoRa)

GPS Module:

- RX → GPIO 16
- TX → GPIO 17

Buzzer:

- Positive → GPIO 27
- Negative → GND

Slave Node Connections

LoRa SX127x: (Same as master)

BMP280/085 (I2C):

- SDA → GPIO 21
- SCL → GPIO 22

GPS Module:

- RX → GPIO 16
- TX → GPIO 17

LED:

- Anode → GPIO 25 (through 220Ω resistor)
- Cathode → GND

Button:

- One pin → GPIO 4
- Other pin → GND

Battery Monitor:

- Battery + → $150k\Omega$ → Junction → $100k\Omega$ → GND
 - Junction → GPIO 34 (ADC)
-

Usage Instructions

1. Access Web Dashboard

If ESP32 is AP Mode:

1. Connect laptop WiFi to "LoRa_Network" (password: lora12345)
2. Open browser to: `http://192.168.4.1`

If ESP32 is Station Mode:

1. Check Serial Monitor for IP address
2. Open browser to that IP

2. Dashboard Features

- **Real-time Node Data:** Temperature, pressure, altitude, battery, GPS
- **Interactive Map:** Shows node locations
- **Alert History:** View all triggered alerts
- **Download CSV:** Export all logged data
- **Clear SD Card:** Remove old data

3. Alert Types

The system automatically detects:

Fall Detection

- Triggers when altitude drops >10m suddenly
- Use case: Person/equipment falling

Low Battery

- Warns when battery <15%
- Prevents unexpected shutdowns

Movement Detection

- Alerts on >100m GPS displacement
- Detects tampering or theft

Node Offline

- Notifies if no data for 15+ minutes
- Identifies connectivity issues

Manual Alert

- Press button on slave node
- Emergency notification

4. Cloud Integration

When internet is available:

- Data uploads to Ubidots every 60 seconds
- View on Ubidots dashboard with charts
- Set up email/SMS alerts via Ubidots Events

Data Management

SD Card Files

data.txt: Main sensor log

```
Timestamp,NodeID,Temp,Pressure,Altitude,Battery,Latitude,Longitude,Alert,RSSI,SNR,JSON
```

alerts.txt: Alert history

```
Timestamp,NodeID,AlertType,Message
```

Auto Cleanup

Currently stores all data. To enable auto-cleanup, add this to master loop:

```
cpp
// Clean up data older than 7 days
if (millis() % 3600000 == 0) { // Check every hour
    cleanupOldData();
}
```

Troubleshooting

LoRa Not Working

- Check antenna is connected
- Verify frequency (433MHz, 868MHz, 915MHz match)
- Ensure SPI pins are correct
- Check power supply (3.3V for SX127x)

SD Card Not Detected

- Format as FAT32
- Check CS pin connection

- Verify shared SPI with LoRa working
- Try different SD card

GPS No Fix

- Takes 30-60 seconds for first fix outdoors
- Must have clear sky view
- Check baud rate (9600)
- System uses Bangalore fallback if no GPS

WiFi Issues

- Check SSID/password
- Verify ESP32 WiFi antenna
- Try AP mode if station mode fails
- Check serial monitor for IP address

Battery Reading Wrong

- Verify voltage divider resistors
- Check VREF_CALIBRATION value
- Measure actual battery voltage
- Adjust BATT_FULL/BATT_EMPTY constants

Advanced Features

1. Add More Nodes

Simply flash more slave nodes with unique IDs:

cpp

// In master code

```
const String nodes[] = {"NODE1", "NODE2", "NODE3", "NODE4"};
```

2. Custom Alert Logic

Add to `checkSmartAlerts()`:

cpp

// High temperature alert

```
if (nodeData[idx].temp > 40.0) {  
    logAlert(nodeId, "High temperature warning");  
}
```

// Rapid pressure change (weather)

```
float presDelta = abs(nodeData[idx].pres - nodeData[idx].prevPres);  
if (presDelta > 10.0) {  
    logAlert(nodeId, "Rapid pressure change");  
}
```

3. Change Poll Interval

cpp

```
#define POLL_INTERVAL 5000 // Poll every 5 seconds instead of 3
```

4. Email Alerts via Ubidots

1. Go to Ubidots → Events
2. Create event: "If battery < 15%, send email"
3. Configure email recipients

4. System auto-triggers on threshold

Mobile Access

The web dashboard is responsive and works on:

- Laptop browsers
- Tablets
- Smartphones

Simply connect to WiFi and access the IP address.

Security Notes

- Change default WiFi passwords
 - Keep Ubidots token secure
 - Use HTTPS for production (requires certificate)
 - Implement authentication for public deployments
-

Performance Tips

1. **Battery Life**

- Increase poll interval
- Use deep sleep between transmissions
- Reduce LoRa TX power if nodes are close

2. **Range**

- Use external antennas

- Position nodes with clear line of sight
- Increase LoRa spreading factor

3. Reliability

- Add more retries
- Implement data buffering
- Use watchdog timers

Support

For issues:

1. Check Serial Monitor for error messages
2. Verify all connections
3. Test each component individually
4. Review pin definitions match your hardware

License & Credits

- Built on Arduino framework
- Uses Sandeep Mistry's LoRa library
- Adafruit sensor libraries
- TinyGPS++ by Mikal Hart

System Version: 2.0 Enhanced **Last Updated:** 2025 **Features:** GPS tracking, WiFi dashboard, Cloud sync, Smart alerts

