**# Technical Documentation: ESP32 Based IoT Device for Inverter Control**

**1. Overview**

This document provides a complete technical guide for integrating an ESP32 with a SIM7600 LTE module and Modbus RTU communication. The ESP32 receives JSON-encoded control commands from an EMQX MQTT server, parses the JSON, converts the data to Modbus RTU format, and sends it to an inverter via RS485.

**2. Hardware Components**

* **ESP32 Dev Kit** (ESP32-WROOM-32 or ESP32-S3)
* **SIM7600 Module** (SIM7600G or SIM7600E LTE)
* **RS485 to TTL Converter** (MAX485 or equivalent)
* **Inverter with Modbus RTU Support**
* **Power Supply (5V for ESP32, SIM7600)**

**3. Software Tools Used**

* **Arduino IDE** (ESP32 Development)
* **TinyGSM Library** (For SIM7600 LTE Communication)
* **PubSubClient Library** (For MQTT Communication)
* **ArduinoJson Library** (For JSON Parsing)
* **ModbusMaster Library** (For Modbus RTU Communication)
* **EMQX MQTT Broker** (MQTT Server)
* **MQTTX / Mosquitto** (MQTT Testing Tools)

**4. Hardware Connections**

**ESP32 ↔ SIM7600 (UART1)**

| **ESP32 Pin** | **SIM7600 Pin** |
| --- | --- |
| 17 (TX1) | RX |
| 16 (RX1) | TX |
| GND | GND |
| 5V | VCC |

**ESP32 ↔ RS485 Converter (UART2)**

| **ESP32 Pin** | **RS485 Converter Pin** |
| --- | --- |
| 27 (RX2) | RO (Receiver Out) |
| 26 (TX2) | DI (Data In) |
| 25 | RE & DE (Enable Pins) |
| GND | GND |

**5. System Architecture**

1. **MQTT Control Messages:** The MQTT server sends JSON-encoded control messages.
2. **ESP32 Receives JSON:** Parses the command and extracts relevant data.
3. **Modbus RTU Conversion:** Converts JSON data into Modbus RTU format.
4. **RS485 Communication:** ESP32 sends Modbus RTU commands to the inverter.
5. **Response Handling:** The inverter sends responses back to ESP32 via RS485.
6. **MQTT Feedback:** ESP32 publishes inverter status back to the MQTT server.

**6. Code Implementation**

**Attached the Arduino code file.**

**7. How to Run the System**

**1) Set Up EMQX MQTT Broker**

* Install EMQX on a server or use **EMQX Cloud**.
* Create a test topic (inverter/control).
* Configure **authentication** if required.

**2) Upload Arduino Code**

* Connect **ESP32 to PC** via USB.
* Open **Arduino IDE** → Select **ESP32 Dev Module**.
* Install necessary libraries (TinyGSM, ModbusMaster, PubSubClient, ArduinoJson).
* Upload the **code to ESP32**.

**3) Power Up Hardware**

* Connect **ESP32, SIM7600, and RS485 module**.
* Insert a **working SIM card** into SIM7600.
* Open **Serial Monitor (115200 baud)**.

**4️) Send Control Commands (MQTTX or Mosquitto)**

**Ex) :**

To send a **command to turn ON the inverter**, publish:

{

"register": 80,

"value": 1

}

**8. Troubleshooting**

| **Issue** | **Possible Cause** | **Solution** |
| --- | --- | --- |
| No MQTT Connection | Incorrect EMQX IP | Verify broker settings |
| No Response from Inverter | Wrong RS485 wiring | Check connections |
| JSON Parsing Error | Incorrect JSON format | Validate JSON data |

**9. Conclusion**

This project successfully integrates **ESP32, SIM7600, Modbus RTU, and MQTT** to remotely control an inverter via RS485. This setup can be used for **smart energy systems, remote monitoring, and IoT applications.**

**Future Improvements:**

* Add **Modbus read functionality** to retrieve inverter status.
* Implement **TLS encryption for MQTT**.
* Create a **web dashboard** for remote control.