



# EspnowMqttPeer2Peer

## ESP-NOW/MQTT Hybrid Communication Protocol

\*Technical Documentation – v2.1.0\*

Author: Eng. Marcelo Pimentel

✉ [marcelo-pimentel@hotmail.com](mailto:marcelo-pimentel@hotmail.com)



---

## Table of Contents

1. Introduction
2. API Reference
3. Code Examples
4. Project Structure
5. Platform Compatibility
6. Key Event Sequence
7. Troubleshooting
8. Revision History

---

# 1. Introduction

## 1.1 Protocol Overview

### Inspiration and Core Concept

This system was designed to **unify MQTT and ESP-NOW paradigms**, creating a transparent communication layer where:

**ESP-NOW acts as "Wireless MQTT"**: Messages are routed using topic patterns (source/destination/action), simulating MQTT's publish/subscribe model without requiring a central broker.

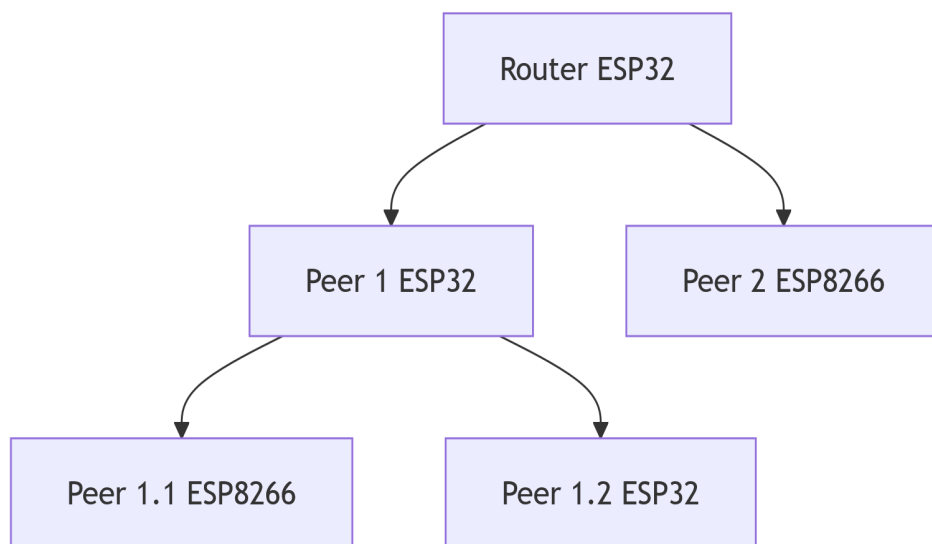
**Full Abstraction**: Developers interact with a single API while the library automatically chooses between:

**ESP-NOW**: For local peer-to-peer communication (ESP32/ESP8266)

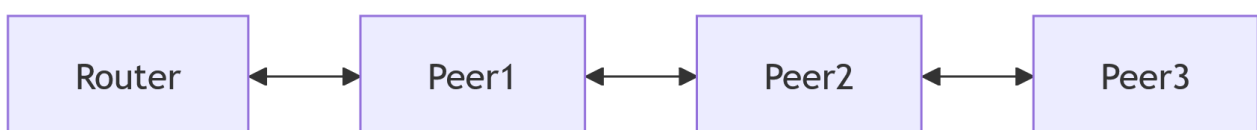
**MQTT**: For cloud/remote connectivity (ESP32 Router only)

A hybrid communication system combining:

- ESP-NOW for direct device-to-device messaging
- MQTT for cloud/remote communication
- Unified API for seamless protocol switching
- **Hierarchical Tree**:



- **Linear Chain**:



## 1.2 Key Features

Feature	Description
Dual-Mode Operation	Automatic ESP-NOW/MQTT selection
Event-Driven Architecture	Critical network event callbacks
Multi-Hop Routing	Message forwarding through peers
Cross-Platform Support	ESP32 (Router/Peer), ESP8266 (Peer)

### 1.3 Use Cases

- Industrial sensor networks
- Smart home automation
- Agricultural monitoring systems

---

## 2. API Reference

### 2.1 Core Classes

#### MqttEspNowRouter (ESP32 Only)

cpp

```
class MqttEspNowRouter {
public:
    void begin(uint8_t wifiChan, uint8_t espnowChan, const char* name,
               const uint8_t* mac, const char* mqttName,
               std::vector<DeviceInfo>& peers, const char* mqttSrv,
               uint16_t port, const char* user = "", const char* pwd = "");

    void subscribe(const String& src, const String& dest,
                  const String& action, LocalHandler h, RouteType t);

    void publishMqtt(const String& src, const String& dest,
                    const String& action, const String& msg);
};
```

#### EspNowPeer

cpp

```
class EspNowPeer {
public:
    void begin(uint8_t channel, const char* name,
               std::vector<DeviceInfo>& routers,
               std::vector<DeviceInfo>& peers);

    void subscribe(const String& src, const String& dest,
                  const String& action, LocalHandler h);
};
```

### 2.2 Data Structures

cpp

```

struct DeviceInfo {
    String name;
    uint8_t mac[6]; // MAC address in byte array format
    bool online = false;
    unsigned long lastPing = 0;
};

```

---

### 3. Code Examples

#### 3.1 Router Initialization

cpp

/\* File: examples/Router/RouterBasic.ino \*/

#include <MqttEspNowRouter.h>

```

// Network configuration
const uint8_t routerMac[] = {0x12,0x34,0x56,0x78,0x9A,0xBC};
std::vector<DeviceInfo> peers = {{ "Sensor1", {0xAA,0xBB,0xCC,0xDD,0xEE,0xFF}}};

MqttEspNowRouter router;
wifiConnManager wifi;

void setup() {
    wifi.onWifiReady([]() {
        wifi.onEspNowReady([]() {
            router.begin(6, 6, "MainRouter", routerMac,
                        "CloudBroker", peers, "mqtt.server.com", 1883);

            router.subscribe("CloudBroker", "Sensor1", "LED",
                            [] (String msg) { /* Handler */ }, ROUTE_MQTT);
        });
    });
    wifi.begin(/* ... parameters ... */);
}

```

#### 3.2 Peer Implementation

cpp

/\* File: examples/Peer/PeerBasic.ino \*/

#include <EspNowPeer.h>

```

EspNowPeer peer;
std::vector<DeviceInfo> routers = {{ "MainRouter",
{0x12,0x34,0x56,0x78,0x9A,0xBC}}};

void setup() {
    peer.begin(6, "TempSensor1", routers, {});

    peer.subscribe("MainRouter", "TempSensor1", "REPORT",
                  [] (String msg) { /* Handle command */ });
}

void loop() {
    peer.publishENow("TempSensor1", "MainRouter", "TEMP", readTemp());
    delay(10000);
}

```

---

## 4. Project Structure

### Directory Layout

```
ESPNow-MQTT-Hybrid/
├── src/                # Core library
│   ├── EspNowPeer.cpp  # Peer implementation
│   └── MqttEspNowRouter.h # Router class
├── examples/          # Sample implementations
│   ├── Router/         # ESP32 router examples
│   └── Peer/           # ESP32/ESP8266 peer examples
├── docs/              # Documentation
│   ├── ESPNowMqttProtocol.pdf
│   └── wiring_diagrams/ # Hardware schematics
└── library.json        # PlatformIO metadata
```

---

## 5. Platform Compatibility

### Hardware Support

Feature	ESP32 Router	ESP32 Peer	ESP8266 Peer
MQTT Client	✓	✗	✗
ESP-NOW Transmitter	✓	✓	✓
Dual Protocol Routing	✓	✓	Limited

### Software Requirements

- PlatformIO Core 6.1+
  - Arduino Framework 3.0+
  - ESP32 Arduino Core 2.0.9+
- 

## 6. Key Event Sequence

### Initialization Flow

1. WiFi Connection Establishment
2. ESP-NOW Protocol Initialization
3. MQTT Broker Connection (Router Only)
4. Peer/Router Registration
5. Message Handler Setup

### Event Timeline

[0ms] WiFi.begin()  
[1200ms] onWifiReady()  
[1500ms] esp\_now\_init()  
[1600ms] onEspNowReady()  
[1700ms] MQTT.connect()  
[2000ms] Ready for Operation

---

## 7. Troubleshooting

### Common Issues

Error Code	Description	Solution
0x3001	ESP-NOW Not Initialized	Check WiFi channel
0x102	MQTT Connection Failed	Verify broker credentials
N/A	Message Loss	Verify MAC addresses

### Debugging Tips

```
cpp
// Enable verbose logging
#define COMM_DEBUG 1 // 0-Disable, 1-Basic, 2-Verbose

// In setup():
Serial.setDebugOutput(true);
```

---

## 8. Revision History

Version	Date	Changes
2.1.0	2024-03-15	Added event API
2.0.2	2024-02-28	ESP8266 fixes
1.4.1	2023-12-10	Initial release

---

\*Documentation generated on 2024-03-20 - MIT License\*