ESPRESSIF

Zigbee com Arduino

Conectando seus dispositivos ao Home Assistant

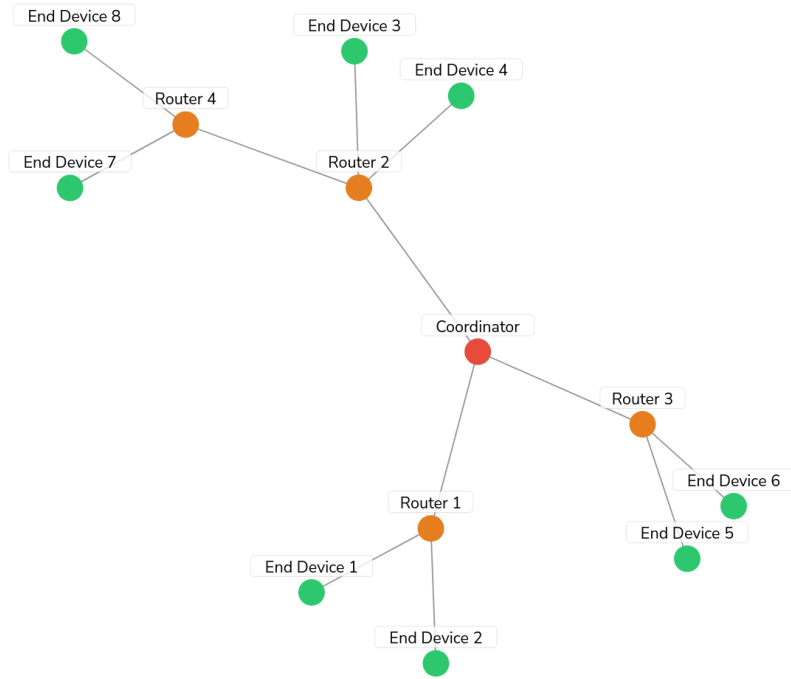
Lucas Saavedra Vaz

2025-08-05

Protocolo Zigbee

Protocolo de rede mesh sem fio de baixo consumo

- Frequência **2.4 GHz**
- Padrão **IEEE 802.15.4**
- **Rede mesh** - Dispositivos retransmitem dados
- **Baixa latência** - Resposta rápida
- **Interoperável** - Certificado pela Zigbee Alliance



Tipos de Nós Zigbee

Coordinator | Router | End Device

Vantagens do Zigbee

Por que escolher Zigbee para IoT?

- **Baixo consumo de energia** - Baterias duram meses/anos
- **Rede mesh auto-reparável** - Alcance estendido e confiabilidade
- **Sem dependência de Wi-Fi** - Funciona independentemente
- **Controle local** - Nenhuma nuvem necessária
- **Baseado em padrões** - Interoperabilidade entre fabricantes

Benefícios Principais

- Vida útil da bateria: 1-2 anos
- Alcance: 10-100m por salto
- Tamanho da rede: centenas de dispositivos
- Latência alcançável: < 100ms

Funções dos Dispositivos Zigbee

Três tipos de dispositivos em uma rede Zigbee

Coordinator

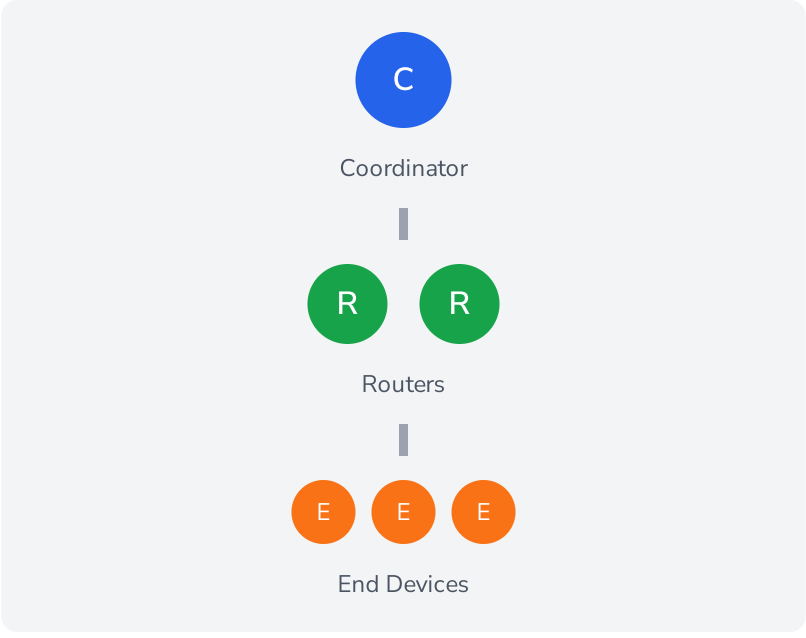
- **Gerenciador da rede** - Forma a rede
- **Sempre ligado** - Conectado à rede elétrica
- **Único na rede** - Ponto de entrada
- **Armazena informações da rede** - Chaves de segurança

Router

- **Retransmissor de dados** - Estende o alcance da rede
- **Sempre escutando** - Alimentado pela rede elétrica
- **Múltiplos permitidos** - Peça central da rede mesh
- **Pode conectar dispositivos** - Função de pai

End Device

- **Alimentado por bateria** - Modo de sleep
- **Alcance limitado** - Conexão direta
- **Operação simples** - Sensores/atuadores
- **Dependente do pai** - Router/coordinator



Suporte Zigbee da Espressif

Selecione os chips ESP32 com rádio IEEE 802.15.4

SoCs com suporte a Zigbee

ESP32-C6
Single-core
160MHz
✓ Zigbee
Wi-Fi 6 + BT5

ESP32-C5
Single-core
160MHz
✓ Zigbee
Wi-Fi 6 Dual-band + BT5

ESP32-H2
Single-core
96MHz
✓ Zigbee
BT5 apenas (sem Wi-Fi)

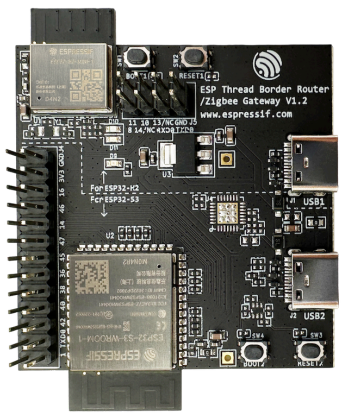
Co-processador
Conexão UART
Qualquer ESP32

C6, C5, H2 têm suporte nativo IEEE 802.15.4 • Outros usam co-processador por UART

Limitações do Zigbee da Espressif

Considerações importantes para desenvolvimento

- **Alternância de rádio** - Necessário gerenciar WiFi/Zigbee. WiFi e Zigbee não podem funcionar ao mesmo tempo.
- **Gerenciamento de memória** - RAM limitada para redes grandes.
- **Otimização de energia** - Maior consumo de energia que chips dedicados a Zigbee. Requer uso eficaz dos modos de sleep.
- **Tamanho da rede** - Limite prático de ~50 dispositivos por coordinator.



ESP Thread Border Router / Zigbee Gateway V1.2

Biblioteca Zigbee Arduino

Integrada ao Arduino Core da Espressif

- **Baseada no SDK oficial da Espressif** - Pronta para produção
- **Compatível com Zigbee 3.0** - Padrão mais recente
- **Testada com Home Assistant** - Compatibilidade comprovada
- **API Arduino** - Programação familiar

Características Principais

- **Nenhuma instalação necessária** - Incluída no Arduino Core da Espressif desde 3.0.6
- **Todas as funções de dispositivo** - Coordinator, Router, End Device
- **Variedade de tipos de endpoint** - 20+ classes de dispositivo
- **Suporte a múltiplos endpoints** - Até 240 por dispositivo
- **Gerenciamento de rede** - Escaneamento, comissionamento, binding, etc.
- **Suporte de segurança** - Criptografia AES
- **Atualizações** - Atualizações de firmware over-the-air

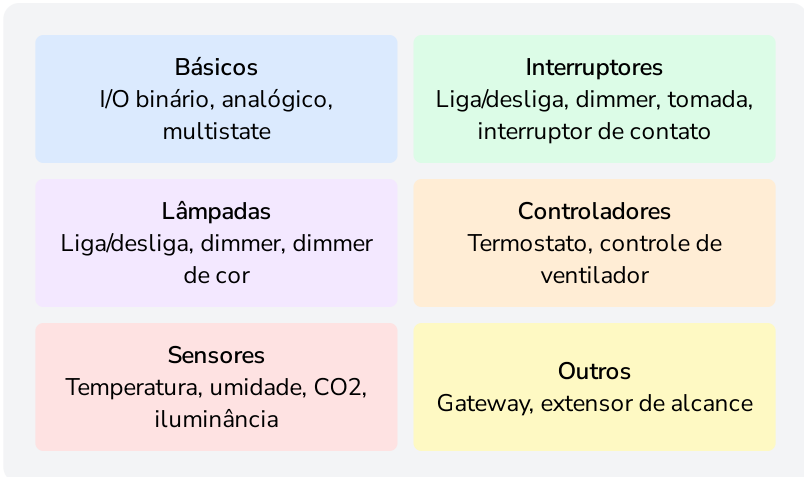
Endpoints Zigbee

Blocos de construção para funcionalidade do dispositivo

Classes de Endpoints

- **Interfaces virtuais** - Até 240 por dispositivo
- **Funções independentes** - Cada uma serve um propósito específico
- **Grupos de clusters** - Funcionalidades relacionadas agrupadas
- **Identificação do dispositivo** - Perfil e IDs de dispositivo

Exemplos de Endpoints



- Para mais informações e outros endpoints, consulte a documentação.

Exemplo de Luz On/Off (Router)

Implementação simples de luz Zigbee com liga/desliga - Zigbee_On_Off_Light

Configuração do Sketch

- **Zigbee Mode** - Selecione se seu dispositivo é um Coordinator/Router ou End Device
- **Partition Scheme** - Selecione o esquema de partição correspondente ao Zigbee Mode

```
void loop() {
  if (digitalRead(button) == LOW) {  // Botão pressionado
    delay(100);
    int startTime = millis();
    while (digitalRead(button) == LOW) {
      delay(50);
      if ((millis() - startTime) > 3000) {
        Serial.println("Resetando Zigbee para configuração de fábrica e reiniciando em 1s.");
        delay(1000);
        Zigbee.factoryReset();
      }
    }

    zbLight.setLight(!zbLight.getLightState());
  }
  delay(100);
}
```

Sensor de Temp./Umidade com Sleep (ED)

Sensor de ambiente alimentado por bateria - Zigbee_Temp_Hum_Sensor_Sleepy

```
void loop() {
  // ... Mesmo que exemplo anterior ...

  // Chama a função para medir temperatura e colocar o dispositivo em sleep
  measureAndSleep();
}
```

Dispositivo Multi-Endpoint

Múltiplos sensores/switches em um dispositivo - Zigbee_Pressure_Flow_Sensor

```
ZigbeeBinary zbBinary = ZigbeeBinary(1);                          // Switch (Saída)
ZigbeeFlowSensor zbFlowSensor = ZigbeeFlowSensor(2);              // Sensor de Fluxo (Entrada)
ZigbeePressureSensor zbPressureSensor = ZigbeePressureSensor(3); // Sensor de Pressão (Entrada)

void setup() {
  // ... resto do código ...

  Zigbee.addEndpoint(&zbBinary);
  Zigbee.addEndpoint(&zbFlowSensor);
  Zigbee.addEndpoint(&zbPressureSensor);

  // ... resto do código ...
}

void loop() {
  // ... resto do código ...

  zbFlowSensor.report();
  zbPressureSensor.report();

  // ... resto do código ...
}
```

Integração Home Assistant

Integração perfeita de dispositivos Zigbee

Descoberta Automática

- **ZHA** - Integração Zigbee integrada do Home Assistant
- **Zigbee2MQTT** - Integração alternativa com broker MQTT
- **Perfil HA padrão** - Biblioteca Arduino usa perfil Home Automation
- **Configuração personalizada** - Necessária para atualizações OTA ou clusters personalizados

Reconhecimento de Dispositivo

- **Clusters padrões** - Reconhecidos automaticamente (On/Off, Level, etc.)
- **Perfis de dispositivo** - Criação adequada de entidades baseada nos tipos de endpoint
- **Reportagem automática** - Intervalos configurados para envio de atualizações
- **Suporte a comandos** - Controle bidirecional (HA - dispositivo)
- **Monitoramento de bateria** - Tensão e porcentagem para dispositivos com sleep

Descoberta e Configuração de Dispositivos

Colocando dispositivos no Home Assistant

Processo de Emparelhamento

1. Habilite o emparelhamento no ZHA, clique em Adicionar dispositivo Zigbee
2. Ligue o ESP32, deve estar nas configurações de fábrica
3. Dispositivo se conecta automaticamente à rede
4. Entidades criadas no HA baseadas nos endpoints
5. Pronto para automação

Demonstração ao Vivo

Implementações Zigbee do mundo real

Dispositivos de Demonstração

- **Cortina** - ESP32C6 + M5Stack Glass unit para simular a janela
- **Detector de vibração** - ESP32-C6 + SW-420
- **Sensor de CO2, temperatura e umidade** - ESP32-C6 + SCD41

Cenários de Demonstração

- **Status da rede ao vivo** - Todos os dispositivos conectados via dongle Zigbee
- **Dashboard Home Assistant** - Exibição de dados de sensores em tempo real
- **Gatilhos de automação** - Detecção de vibração controla LEDs de outras demos
- **Controle de cortinas** - Cortinas podem ser controladas do dashboard ou botões no display
- **Monitoramento de ambiente** - Rastreamento de temperatura, umidade, CO2

Obrigado!

Perguntas e Discussão

Vamos construir dispositivos Zigbee juntos!

ESP32 + Arduino + Zigbee + Home Assistant

Recursos e Links

📚 Documentação Zigbee 🔧 Repositório arduino-esp32 👨‍💻 @lucasssvaz no GitHub