Eigene ZIGBEE-Geräte mit ESP32 entwickeln

Praxisüberblick mit Hands-on-Beispielen für
ESP32 Arduino Core, Home Assistant
und Zigbee2MQTT

Zigbee | Zigbee2MQTT | Home Assistant

Praktische Beispiele
Schritt für Schritt mitmachen

ESP32 Arduino Core
Bewährte Tools. Maximale Flexibilität

Nahtlos Integriert
Mit Home Assistant und Zigbee2MQTT

Markus Edenhauser

Eigene ZIGBEE-Geräte mit ESP32 entwickeln

Praxisüberblick mit Hands-on-Beispielen für ESP32 Arduino Core, Home Assistant und Zigbee2MQTT

Zigbee | Zigbee2MQTT | Home Assistant

Praktische Beispiele
Schritt für Schritt mitmachen

ESP32 Arduino Core
Bewährte Tools. Maximale Flexibilität

Nahtlos Integriert
Mit Home Assistant und Zigbee2MQTT

Markus Edenhauser

Eigene Zigbee-Geräte mit ESP32 entwickeln

Praxisüberblick mit Hands-on-Beispielen für ESP32 Arduino Core, Home Assistant und Zigbee2MQTT

Markus Edenhauser, MA MSc

2026-07-01

Eigene Zigbee-Geräte mit ESP32 entwickeln

Eigene Zigbee-Geräte mit ESP32 entwickeln

Juli 2026 1. Auflage
Selbstverlag

Markus Edenhauser, MA MSc
Völser Straße 41
6020 Innsbruck
Österreich
pixeledi.eu/impressum

Du bist mein schönstes `Zigbee`-Signal. Mit dir bleibt selbst das wildeste Mesh im Leben stabil.

Einführung

Für wen dieses Buch gedacht ist

Dieses Buch richtet sich an Entwickler, die mit Arduino bereits produktiv arbeiten und jetzt einen praktischen Schnelleinstieg in Zigbee suchen. Gemeint sind Leser, die nicht bei der ersten Blink-LED stehen, sondern bereits eigene kleine bis mittlere Projekte gebaut haben, seriellen Output sinnvoll lesen können und typische Build-, Flash- und Integrationsprobleme aus der Praxis kennen.

Du musst dafür kein Embedded-Spezialist sein, aber du solltest bereit sein, einen Schritt über reine Sketch-Routinen hinauszugehen. Der zentrale Wechsel in diesem Buch ist methodisch: weg von „läuft auf dem Tisch irgendwie“, hin zu reproduzierbaren, nachvollziehbaren Geräteprojekten mit klarer Struktur, sauberer Konfiguration und überprüfbarem Verhalten in Zigbee2MQTT und Home Assistant.

Nicht im Fokus steht ein kompletter Einstieg in Elektronikgrundlagen oder allgemeine Mikrocontroller-Basics. Das Buch setzt voraus, dass Begriffe wie GPIO, I2C, serieller Monitor und grundlegende Arduino-Projektstruktur nicht völlig neu sind. Der Mehrwert liegt stattdessen darin, diese Vorkenntnisse gezielt in den Zigbee-Kontext zu überführen und typische Stolperstellen im Übergang zu vermeiden.

Besonders passend ist das Buch für Leser, die sich genau diese Fragen stellen:

1. Warum joinen manche Geräte sofort sauber, andere aber nur sporadisch?

2. Warum erscheinen Exposes in Zigbee2MQTT nicht wie erwartet?
3. Warum ist ein Gerät im Labortest stabil, im Dauerbetrieb aber nicht?
4. Wie plant man ein Zigbee-Gerät so, dass Wartung und Erweiterung später nicht zum Chaos werden?

Wenn du aus Arduino IDE oder PlatformIO kommst und auf diese Fragen Antworten suchst, bist du exakt in der Zielgruppe. Dieses Buch schließt bewusst die Lücke zwischen funktionierendem Demo-Code und alltagstauglichem Zigbee-Gerätedesign.

Was du in diesem Buch lernst

Du lernst in diesem Buch nicht nur, wie ein Gerät grundsätzlich joined und Daten sendet. Der Fokus ist ein praxisnaher Schnelleinstieg mit klarer Schritt-für-Schritt-Umsetzung und einem sauberen Betriebsbezug. Ziel ist, dass du am Ende nicht nur Code kompilierst, sondern Geräte entwirfst, die im Alltag stabil laufen und sich später wartbar erweitern lassen.

Im praktischen Kern arbeitest du entlang einer klaren Projektlinie:

1. Einstieg über einfache Aktorik und reproduzierbare Join-Abläufe.
2. Übergang zu Sensorik mit nachvollziehbarer Messwertübertragung.
3. Ausbau zu Sleep- und Battery-End-Device-Mustern.
4. Event- und Sicherheitsnahe Geräte wie Button und Türkontakt.
5. Netzbetriebene Aktorik mit Relais und sicherem Integrationspfad.

Technisch lernst du mehrere Ebenen gleichzeitig sauber zu verknüpfen:

1. Gerätemodell: Rollen, Endpoints, Cluster, Attribute.
2. Übertragung: Reporting-Strategien statt blindem Intervall-Senden.
3. Betrieb: Zusammenspiel aus Zigbee2MQTT, Home Assistant und Logs.
4. Robustheit: Recovery-Verhalten, Re-Interview, Neu-Join und typische Fehlerpfade.
5. Energieverhalten: Sleep-Zyklen, Keep-Alive-Bezug und Auswirkungen auf Netzstabilität.

Ein zentraler Lernerfolg ist die Diagnosefähigkeit. Du lernst, Probleme nicht nur „wegzudebuggen“, sondern strukturiert einzugrenzen: Hardwarepfad, Netzwerkpfad, Gerätemodell und Automationspfad. Genau diese Trennung macht den Unterschied zwischen zufälligem Erfolg und reproduzierbarer Projektarbeit.

Am Ende sollst du eigene Zigbee-Geräte planen können, ohne bei jeder Designentscheidung raten zu müssen. Du weißt dann, welche Entscheidungen früh wichtig sind, welche später korrigierbar bleiben und wie du neue Geräte kontrolliert in ein bestehendes Smart-Home-Setup integrierst.

Verwendete Hardware

Eine vollständige und gepflegte Materialliste liegt im zugehörigen GitHub Repository. Dort findest du die jeweils aktuelle Zusammenstellung mit Boards, Sensoren, Zubehör und Variantenhinweisen für die einzelnen Projekte.

Auszugsweise folgt hier eine Auflistung, die ich im Verlauf ergänze.

Die Hardwarebasis bleibt bewusst flexibel, weil mehrere ESP32-Zigbee-Varianten sinnvoll einsetzbar sind. Für den praktischen Einstieg arbeiten wir zunächst mit sehr einfacher Peripherie wie LED und Taster, gehen danach zu SHT3x, SHT40 und BH1750 über und bauen später auf Battery-End-Device, Türkontakt und Relais-Aktorik auf.

Für die Beispiele eignen sich ESP32-H2, ESP32-C6 und ESP32-C5 sehr gut. Wichtig beim ESP32-H2: Er hat kein WiFi und ist für Zigbee und Thread ausgelegt.

Für die 230V-Relaisstrecke gilt ein klarer Rahmen. Der Fokus liegt auf Zigbee-Integration und sauberer Gerätestruktur, nicht auf elektrotechnischer Grundausbildung. Netzspannung erfordert geeignete Komponenten, sichere Verdrahtung und die Einhaltung lokaler Vorschriften.

Arduino IDE und PlatformIO

Als Entwicklungsumgebung kannst du sowohl die Arduino IDE als auch PlatformIO verwenden. Inhaltlich sind alle gezeigten Beispiele so aufgebaut, dass du sie in beiden Umgebungen nachbauen kannst.

Didaktisch ist der Weg im Buch bewusst zweistufig:

1. Zu Beginn zeige ich exemplarisch den Einstieg mit der Arduino IDE, damit Setup, Boardauswahl und erste Build-/Flash-Schritte schnell nachvollziehbar sind.
2. Danach wechseln wir im Hauptteil auf PlatformIO, weil bei vielen Projekten Konfigurationen, Bibliotheksstände und Build-Parameter deutlich reproduzierbarer dokumentiert bleiben.

Wichtig ist: Der Wechsel auf PlatformIO bedeutet nicht, dass du künftig ausschließlich PlatformIO verwenden musst. Alle Projekte aus dem Buch lassen sich weiterhin problemlos in der Arduino IDE umsetzen, wenn das dein bevorzugter Workflow ist. Du bist also nicht an eine bestimmte Entwicklungsumgebung gebunden und kannst jederzeit zwischen Arduino IDE und PlatformIO wechseln.

Der Grund für PlatformIO als Referenzpfad ist rein pragmatisch: Bei mehreren Geräten, Varianten und Iterationen spart ein versionierbares Projektlayout Zeit, reduziert Konfigurationsfehler und macht Debugging nachvollziehbarer. Zusätzlich gibt es gezielte Begleitvideos, damit die Einrichtung in der Praxis zuverlässig funktioniert.

Backend und Voraussetzungen

Als Backend verwenden wir `Home Assistant OS` mit `Zigbee2MQTT`. ZHA sollte grundsätzlich ebenfalls funktionieren, ist in diesem Buch aber nicht der getestete Referenzpfad.

Wichtige Voraussetzung: Die Grundinstallation von Home Assistant OS wird im Buch nicht im Detail gezeigt und sollte bereits vorhanden sein. Dazu gibt es bereits viele gute YouTube-Videos, auch wenn sie nicht von mir sind :).

Der Fokus liegt bewusst nicht auf Proxmox, der Installation oder dem Server-Setup, sondern auf der praktischen Arbeit mit Zigbee. Deshalb steigen wir direkt mit einem vorbereiteten Home Assistant OS ein und richten Zigbee2MQTT Schritt für Schritt ein. Zigbee2MQTT kann zwar auch ohne Home Assistant betrieben werden, im Buch nutzen wir Home Assistant jedoch als Hauptpfad, da sich diese Kombination in der Praxis besonders bewährt hat.

Aufbau des Buches

Das Buch ist als praktischer Schnelleinstieg entlang einer technischen Lernkurve aufgebaut, nicht entlang abstrakter Theorieblöcke. Du bekommst früh ein lauffähiges Gerät, damit die Toolchain und der Pairing-Prozess sitzen. Danach steigen wir schrittweise von einfacher Aktorik über interne Messwerte zu externer Sensorik auf.

Im Mittelteil wird aus dem reinen Funktionsnachweis ein belastbares Gerätedesign. Dort behandeln wir Reporting, Plausibilisierung, Batteriebetrieb und Sleep-Verhalten. Anschließend folgen Button- und Kontaktgeräte, bei denen Ereignisse, Reaktionszeit und Alltagstauglichkeit stärker in den Vordergrund rücken.

Später kommen Betriebsalltag, Fehlersuche und Sicherheitsaspekte in gebündelter Form. Der Ausblick zeigt dir dann, wie du aus den Bausteinen eigene Produkte und nicht nur Einzelprojekte entwickeln kannst.

Jedes Projektkapitel folgt dabei einer konsistenten technischen Logik und führt dich Schritt für Schritt durch typische Einsteigerbeispiele für Smart-Home-Sensoren und Aktoren.

Codebeispiele und GitHub Repository

Alle Codebeispiele gehören zu einem gemeinsamen Repository, damit du Stände nachvollziehen, vergleichen und reproduzieren kannst.

https://github.com/pixelEDI/esp32-zigbee-kurs

Im Repository findest du immer die aktuelle Version der Code-Beispiele. Du kannst das Projekt klonen oder als ZIP-

Datei herunterladen - ohne Anmeldung. Ohne Git: **<> Code → Download ZIP** anklicken, um die Dateien direkt im Browser zu laden.

> Den Download-Link für das PDF findest du am Ende des Buches im Kapitel **Downloadlinks**.

Im Repository findest du außerdem alle Grafiken und Diagramme separat. Das ist besonders praktisch, weil du sie dort je nach Anzeige und Endgerät in voller Größe und mit besserer Lesbarkeit ansehen kannst.

Nutze die Beispiele nicht als Copy-Paste-Bausteine, sondern als Referenz-implementierungen. Der Lerneffekt entsteht dann, wenn du Parameter bewusst veränderst und die Auswirkungen in Zigbee2MQTT, Home Assistant und seriellen Logs korrelierst.

Wenn du dieses Buch so durcharbeitest, hast du am Ende nicht nur einzelne funktionierende Sketche, sondern ein technisches Verständnis dafür, warum ein Zigbee-Gerät im Labor und im realen Einsatz unterschiedlich reagieren kann und wie du diese Unterschiede kontrollierst.

KI Transparenz Hinweis

Dieses Buch wurde inhaltlich von mir selbst konzipiert und geschrieben. KI Werkzeuge wurden unterstützend eingesetzt, ähnlich wie Dokumentationen oder Foren. Die Konzepte, Beispiele und Struktur stammen aus meiner eigenen praktischen Erfahrung mit ESP32 Projekten.

KI war ein Werkzeug zur Unterstützung bei Formulierungen und Strukturierung, nicht der Autor.

Die Verantwortung für den Inhalt liegt vollständig bei mir.

Lizenz und Haftung

Die Beispiele und Übungsprojekte dieses Buches dürfen für eigene Projekte frei verwendet werden. Du kannst sie anpassen, erweitern und in deinen eigenen Projekten einsetzen.

Ich übernehme keine Haftung für Schäden, Datenverluste oder Fehlfunktionen, die durch unsachgemäße Verwendung entstehen können. Arbeite sorgfältig und teste neue Konzepte idealerweise zunächst in kleinen Projekten.

Wichtiger Hinweis für Kapitel `1-Kanal-Relaismodul 230V mit Zigbee schalten`: Dort wird mit einem 230V-Relaismodul gearbeitet. An der Netzspannungsseite dürfen ausschließlich Elektrofachkräfte arbeiten, prüfen und in Betrieb nehmen. Die gezeigten Inhalte dienen nur der technischen Information und ersetzen keine elektrotechnische Ausbildung oder Abnahme.