

# **Zigbee Tracker System User Guide**

# Table of Contents

<b>1</b>	<b>About This Document</b>	<b>3</b>
<b>2</b>	<b>Revision History</b>	<b>3</b>
<b>3</b>	<b>Overview</b>	<b>3</b>
3.1	Basic Information	3
3.2	Basic Requirements	4
<b>4</b>	<b>Configuration</b>	<b>5</b>
4.1	Configuring Zigbee Tracker v2.0	5
4.1.1	Flashing the STM32 Bootloader	5
4.1.2	Configuring the Zigbee Module	6
4.1.3	Uploading STM32 Code (For End Devices)	7
4.2	Configuring Zigbee Coordinator v2.0	8
4.2.1	Configuring the Zigbee Module	8
4.2.2	Uploading ESP32 Code	8
4.3	Configuring Zigbee Tracker System	9
4.3.1	Placement of Coordinator and Routers	9
4.3.2	Charging and Activating the End Devices	10
4.3.3	Powering the Coordinator and Routers	10
4.3.4	Verifying Functionality	10
<b>5</b>	<b>Troubleshooting</b>	<b>11</b>
5.1	Common Problems	11
<b>6</b>	<b>Related Documentation</b>	<b>11</b>

# 1 About This Document

This document describes the set-up procedures required to operate a Zigbee mesh network for tracking/environment monitoring based on the Zigbee Tracker v2.0 and Zigbee Coordinator v2.0. *This document assumes the User has rudimentary knowledge of Zigbee mesh networks, Arduino IDE, and industry standard terminology (i.e. USB, MAC Address).*

## 2 Revision History

## 3 Overview

### 3.1 Basic Information

This User Guide is provided with a folder containing all the necessary code.

It is recommended to refer to the Zigbee Tracker v2.0 and Zigbee Coordinator v2.0 datasheets for additional information.

This guide will configure the Zigbee Tracker v2.0 as both an End Device and Router, and the Zigbee Coordinator v2.0 as a Coordinator as part of the Zigbee Tracker System.

The End Devices run a tracking and environment monitoring algorithm that is activated by significant motion. The Routers route data packets from the End Device, and the Coordinator sets up the Zigbee network and posts data to Firebase.

The following steps (in order) are needed to set-up the Zigbee Tracker System, which will be covered in [Section 4](#):

#### *Configuring Zigbee Tracker v2.0*

- Flashing the STM32 Bootloader
- Configuring the Zigbee Module
- Uploading STM32 Code

#### *Configuring Zigbee Coordinator v2.0*

- Configuring the Zigbee Module
- Uploading ESP32 Code

#### *Configuring Zigbee Tracking System*

- Positioning the Coordinator and Routers
- Charging and Activating the End Devices
- Powering the Coordinator and Routers
- Verifying Functionality

## 3.2 Basic Requirements

In order to set up a basic Zigbee Tracker System, the following hardware is required:

- Zigbee Coordinator v2.0
- At least 3 Zigbee Tracker v2.0 (1 as End Device, 2 as Router)
- Micro USB to USB A cables
- Li-ion battery for End Device
- 5 or 12V DC Supply (optional, can be replaced with the above-mentioned USB cables)
- Zigbee Debugger
- FFC 10p 0.5mm

Additionally, a computer is required for the initial set up, with the minimum requirements:

- x86-64 architecture, Windows 7 or later
- User accessible USB 2.0 ports (or newer)
- Internet access

The following files are provided with this user guide to be used in the set up:

- Arduino libraries
- PC13 binary bootloader file
- Zigbee Tracker code
- Zigbee Coordinator code
- Serial passthrough code

The following software/files must be installed on the computer:

- Arduino IDE
- ST Flash Loader Demonstrator
- STM32 Virtual COM Port Driver

## 4 Configuration

### 4.1 Configuring Zigbee Tracker v2.0

The Zigbee Tracker v2.0 is a compact, battery-powered device. This section will describe the steps needed to set up the Zigbee Tracker v2.0 as an End Device or Router.

*It is recommended to refer to the Zigbee Tracker v2.0 Datasheet.*

#### 4.1.1 Flashing the STM32 Bootloader

The Zigbee Tracker v2.0 has an onboard STM32 microcontroller that is used to issue commands via UART to the Zigbee Module. The STM32 microcontroller should be flashed with the bootloader, allowing it to be programmed via the USB port.

The following steps are to flash the STM32 bootloader using the Zigbee Debugger:

1. Download the ST Flash Loader Demonstrator [here](#), if not already done.
2. Download the ST COM Port Driver [here](#), if not already done.
3. Connect the Zigbee Tracker v2.0 to the Zigbee Debugger with the FFC and plug the USB connector of the Zigbee Debugger into a PC, as shown in Figure 4.1.1a.
4. Open the ST Flash Loader Demonstrator, leave all options as default, and click “Next”.
5. After a few seconds, it should indicate “Target is readable”. Click “Next”. If it hangs, check the connections and redo the process.
6. Ensure all flash pages are green (unprotected). Click “Next”. If not all the pages are green, the STM32 was hard locked by another user.
7. Select “Download to device” and choose the binary file provided with this User Guide (Zigbee Tracker System Files > Software > Binaries > generic\_boot20\_pc13.bin). Select “Verify after download”. Then click “Next”.
8. Wait for the download to complete, and exit the program.

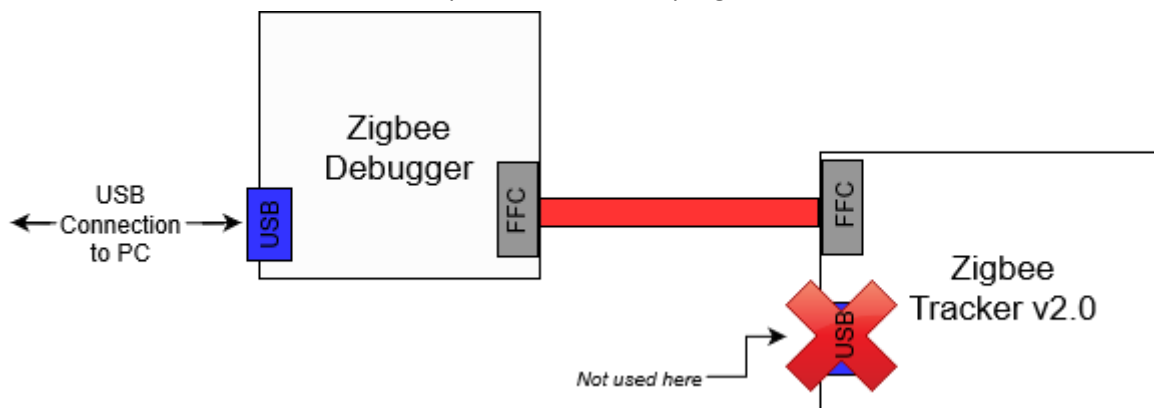


Figure 4.1.1a: Zigbee Tracker v2.0 Bootloader Connections

Note: This Subsection can be skipped if the User has no intention of USB programming or insufficient flash space to accommodate a bootloader.

### 4.1.2 Configuring the Zigbee Module

The Zigbee Tracker v2.0 has an onboard Zigbee Module, which is connected to the STM32 microcontroller via UART and the FFC connector. The Zigbee Module contains factory firmware, allowing it to be controlled by AT commands via UART; using the CC Debugger connections overwrites the factory firmware.

As shown in Figure 4.1.2a, there are no direct connections between the USB port and the Zigbee Module's UART pins, the STM32 microcontroller is programmed with serial passthrough code to act as a "bridge", in order to configure the Zigbee Module directly from the PC.

*Configuring the Zigbee Tracker v2.0 as an End Device or Router node differs in that the End Device requires an additional sleep mode command.*

The following steps are to configure the Zigbee Module from the PC:

1. Download Serial Port Utility [here](#), if not already done.
2. Download the Arduino Core for STM32 [here](#). Instructions on installation can be found [here](#).
3. Open the serial passthrough code in Arduino IDE. It can be found in the folder provided with this User Guide (Zigbee Tracker System Files > Software > Arduino > Serial.ino).
4. Go to "Tools" and select the board "Generic STM32F1 Series".
5. Configure the board settings as shown in Figure 4.1.2b, and upload the code.
6. After successfully programming the STM32, open Serial Port Utility and issue the necessary commands. The command set can be found [here](#) or in the folder provided with this User Guide (Zigbee Tracker System Files > Documentation > Zigbee Command Set). The command set allows for both AT commands (i.e. AT+DEV=E) and HEX commands (i.e. FD 02 01 02 FF). In order to select the AT/HEX command mode, modify the state of the STM32's pin PB11 (HIGH = AT, LOW = HEX). It is recommended to use the HEX commands, as the message function is only available with HEX commands. This User Guide and provided code will use HEX commands.
7. Select "Hex" in Serial Port Utility for both send and receive settings, and open the port.
8. To configure the Zigbee Module as an **End Device**, send FD 02 01 02 FF. To configure as **Router**, send FD 02 01 01 FF. Additionally for **End Devices**, send FD 02 0D 78 FF to set sleep. After that, send FD 01 12 FF to apply the changes. Additional commands may be optionally issued to control different parameters. For the purpose of this User Guide, no additional commands are needed.
9. Close Serial Port Utility.

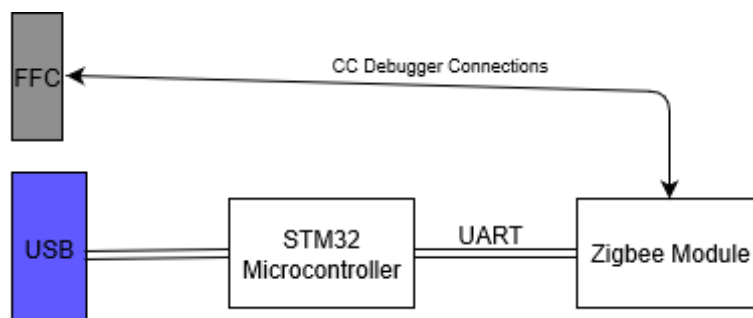


Figure 4.1.1a: Zigbee Module Programming Connections

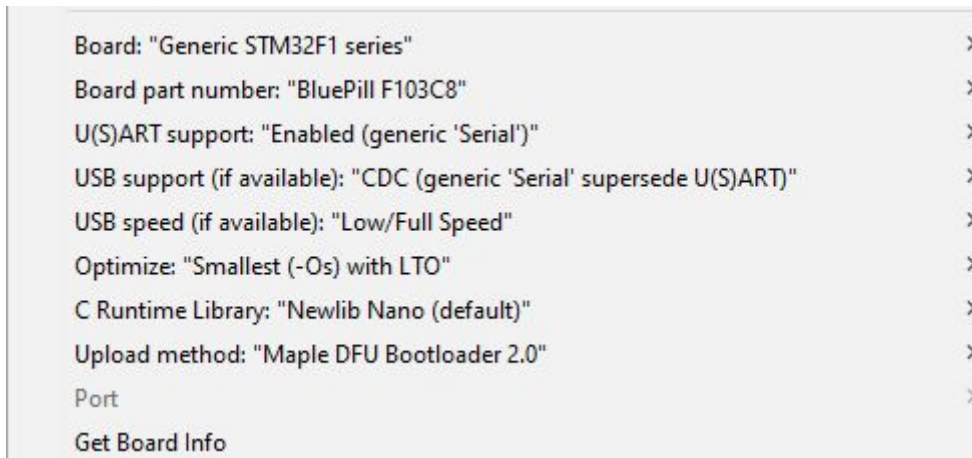


Figure 4.1.2b: Zigbee Tracker v2.0 Board Settings in Arduino IDE

### 4.1.3 Uploading STM32 Code (For End Devices)

**IMPORTANT:** If configured as a Router, this Subsection can be skipped.

The Zigbee Tracker v2.0 has an onboard STM32 microcontroller that is used to issue commands via UART to the Zigbee Module. This step is similar to [Subsection 4.1.1](#), except that the uploaded code is for tracking and environment monitoring, since the Zigbee Module has already been configured in [Subsection 4.1.2](#). *Configuring the Zigbee Tracker v2.0 as an End Device or Router node differs in that the Router configuration does not utilize the STM32 to run any functional code.*

The following steps are used to upload the code to the STM32:

1. Copy the Arduino libraries provided with this User Guide (Zigbee Tracker System Files > Software > Arduino > Libraries) into the default Arduino library folder (depending on your configuration, normally Program Files (x86) > Arduino > libraries).
2. Open the Zigbee Tracker code in Arduino IDE. It can be found in the folder provided with this User Guide (Zigbee Tracker System Files > Software > Arduino > Zigbee Tracker v2.ino).
3. Go to "Tools" and select the board "Generic STM32F1 Series".
4. Configure the board settings as shown in Figure 4.1.2b, and upload the code.

## 4.2 Configuring Zigbee Coordinator v2.0

The Zigbee Coordinator v2.0 is a standalone WiFi connected device. This section will describe the steps needed to set up the Zigbee Coordinator v2.0 as a Coordinator.

*It is recommended to refer to the Zigbee Coordinator v2.0 Datasheet.*

### 4.2.1 Configuring the Zigbee Module

The Zigbee Coordinator v2.0 has an onboard Zigbee Module, which is connected to the ESP32 microcontroller via UART. Unlike the Zigbee Tracker v2.0, no bootloader is needed and code can be directly uploaded onto the ESP32.

Zigbee Modules are set as Coordinators by default, so no additional configuration is needed for the purpose of this User Guide.

If additional configuration is required, use the serial passthrough code (for ESP32) provided with this User Guide. The steps for uploading code to the ESP32 are provided in [Subsection 4.2.2](#).

### 4.2.2 Uploading ESP32 Code

The Zigbee Tracker v2.0 has an onboard ESP32 module that is used to issue commands via UART to the Zigbee Module. The ESP32 requires a specific reset and boot signal, which is handled by the Zigbee Coordinator v2.0's onboard circuitry - no user intervention is required.

The following steps are used to upload the code to the ESP32:

1. Download Serial Port Utility [here](#), if not already done.
2. Download the Arduino Core for ESP32 [here](#). Instructions on installation can be found [here](#).
3. Open the Zigbee Coordinator code in Arduino IDE. It can be found in the folder provided with this User Guide (Zigbee Tracker System Files > Software > Arduino > Zigbee Coordinator v2.ino).
4. Create a Firebase Realtime Database [here](#), if not already done. Instructions can be found [here](#). Take note of the database secret on the Firebase website (*gear icon* > Project settings > Service accounts > Database secrets).
5. Modify Zigbee Coordinator v2.ino lines 4-7 as shown in Figure 4.2.2a. FIREBASE\_HOST is the URL of the Firebase Realtime Database, and FIREBASE\_AUTH is the Database secret.
6. Go to "Tools" and select the board "ESP32 Dev Module".
7. Leave board settings as default, and upload the code.

```
4 #define FIREBASE_HOST "" //Do not include https:// in FIREBASE_HOST
5 #define FIREBASE_AUTH ""
6 #define WIFI_SSID ""
7 #define WIFI_PASSWORD ""
```

Figure 4.2.2: Firebase and WiFi parameters



## 4.3 Configuring Zigbee Tracker System

Configuring the Zigbee Tracker System should only be done after all of the devices (Coordinator, Routers and End Devices) have been configured in accordance with [Section 4.1](#) and [Section 4.2](#).

### 4.3.1 Placement of Coordinator and Routers

The Zigbee Tracking System works on the principle that RSSI (Received Signal Strength Index), which is an indicator of signal strength, decreases between any two nodes as the distance increases (nonlinearly). Zigbee Tracking on an End Device measures the RSSI of all nearby Routers, and connects to the one with the highest RSSI (i.e. nearest).

In practice, RF energy can be easily distorted or attenuated by stray objects (especially conductors), so the highest RSSI may not be the nearest.

This can be resolved by placing a single Router (or Coordinator) in an enclosed room as in Figure 4.3.1a, thereby using the walls of the room as RF attenuators (i.e. reduce RSSI). In this way, it can be determined with relatively high confidence which room the connected End Device is in, which is sufficient for most indoor tracking applications.

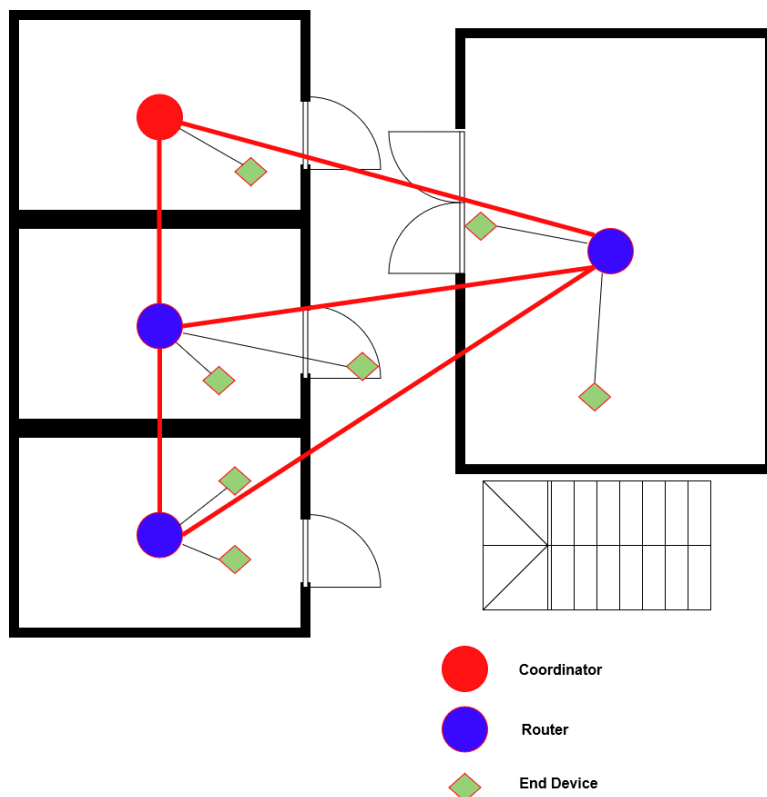


Figure 4.3.1a: Ideal placement of fixed Zigbee Devices

### 4.3.2 Charging and Activating the End Devices

The Zigbee Tracker v2.0 (End Devices) can be powered from single cell, 3.7V lithium-ion batteries. The battery can be charged through the USB port of the Zigbee Tracker v2.0. Automatic charge termination/safety monitoring mitigates the need for user intervention. Information on Charge LED Indicator status can be found in the Zigbee Tracker v2.0 Datasheet.

### 4.3.3 Powering the Coordinator and Routers

The Zigbee Tracker v2.0 (Routers) should be powered via the USB port instead of from a battery.

The Zigbee Coordinator v2.0 (Coordinator) can be powered from either its USB port or DC jack. Additional information can be found in the Zigbee Coordinator v2.0 Datasheet.

### 4.3.4 Verifying Functionality

The Zigbee Tracker v2.0 (End Devices) runs its tracking and environment monitoring algorithm only on significant motion. There is a 5 second cooldown to prevent spam in the event of constant motion of the End Device.

To test the Zigbee Tracker System, verify that all devices are on and introduce significant motion to any End Device (i.e. shaking). The data should be posted on Firebase in the format shown in Figure 4.3.4a. Repeating this action on the same device after the cooldown will update this data entry.

The Zigbee Tracker v2 code (for End Devices) will blink the NWK LED if it detects a fault with onboard sensors. The location of the NWK LED is indicated in the Zigbee Tracker v2.0 Datasheet.

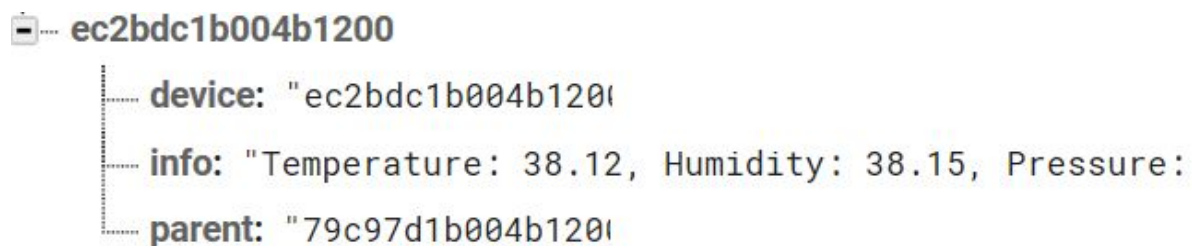
A screenshot of a Firebase Realtime Database entry. At the top, there is a device ID 'ec2bdc1b004b1200' next to a small icon of a battery. Below this, a vertical list of data fields is shown, each preceded by a right-facing curly bracket. The fields are: 'device: "ec2bdc1b004b1200"', 'info: "Temperature: 38.12, Humidity: 38.15, Pressure:', and 'parent: "79c97d1b004b1200'.

Figure 4.3.4a: Typical data entry on Firebase

## 5 Troubleshooting

### 5.1 Common Problems

Problem	Suggested Solution
Zigbee Tracker v2.0 USB code upload failed	STM32 sleep mode preventing USB module from responding. Reconnect Zigbee Tracker v2.0 USB port while Arduino IDE attempts to upload the code to force power-on reset.
Zigbee Coordinator v2.0 USB code upload failed	Serial port access timed out. Reconnect Zigbee Coordinator v2.0 USB port.
ST Flash Loader Demonstrator hangs after opening COM port	Target device not found. Force close the program and check connections.
Zigbee Tracker v2.0 configured as End Device, NWK indicator blinking continuously.	Hardware fault. One or more sensors unresponsive.
ST Flash Loader Demonstrator indicates that flash is write protected.	Undo the write protection if possible.

## 6 Related Documentation

### Documents related to the Zigbee Tracker System User Guide

Zigbee Coordinator v2.0 Datasheet

*Technical documentation of the Zigbee Coordinator v2.0 device.*

Zigbee Tracker v2.0 Datasheet

*Technical documentation of the Zigbee Tracker v2.0 device.*