This is the board I have. I Have a question about it. B-L475E-IOT01A (STM32L4)

**ChatGPT said:**

ChatGPT

Memory updated

Sure! What's your question about the B-L475E-IOT01A board?

4o

**You said:**

Is there a way that I could program it wirelessly either through bluetooth or wifi? I'd like to write new code on my computer, and then wirelessly reprogram the board

**ChatGPT said:**

ChatGPT

Yes, you can program the **B-L475E-IOT01A** wirelessly using either **Wi-Fi or Bluetooth**, but it requires some setup. Here are a few approaches:

**1. Wi-Fi: OTA (Over-the-Air) Updates**

* **Using STM32CubeProgrammer with Wi-Fi**
  + This board has the **ISM43362-M3G-L44** Wi-Fi module.
  + You can implement a bootloader that downloads and flashes new firmware over Wi-Fi.
  + STM32CubeIDE and STM32CubeProgrammer can be configured for remote updates.
* **Using Mbed OS OTA**
  + If you're using **Mbed OS**, you can implement **Mbed OTA updates** via Wi-Fi.

**2. Bluetooth (BLE) DFU (Device Firmware Update)**

* The board has a **SPBTLE-RF** BLE module.
* You can implement a **Bluetooth DFU (Device Firmware Update)** system.
* This requires setting up a bootloader that can receive firmware via BLE.
* **STM32CubeFWL4** has examples for DFU over BLE.

**3. Serial over Wi-Fi or BLE**

* You could use a **Wi-Fi-to-serial** or **BLE-to-serial** bridge to communicate with the board.
* Then, use a bootloader like **STM32 UART bootloader** to flash new firmware.

**Easiest Approach?**

If you’re looking for something quick and relatively easy:

* **Mbed OS OTA (Wi-Fi)** is a good option.
* **STM32 BLE DFU** can work if you want Bluetooth-based updates.

Would you like more details or example code for any specific approach?

4o

**You said:**

yes, more details on this option please Mbed OS OTA (Wi-Fi) is a good option

**ChatGPT said:**

ChatGPT

**Mbed OS Over-the-Air (OTA) Updates via Wi-Fi on the B-L475E-IOT01A**

Mbed OS provides a framework for **OTA updates** using Wi-Fi, allowing you to **remotely flash new firmware** to your B-L475E-IOT01A board. This is particularly useful for IoT applications where physical access to the board is limited.

**Steps to Implement Mbed OS OTA Over Wi-Fi**

**1. Set Up Mbed OS on Your Computer**

1. **Install Mbed CLI** (if you haven’t already):

CopyEdit

pip install mbed-cli

1. Clone the Mbed OS repository:

perl

CopyEdit

mbed new my-mbed-project

cd my-mbed-project

**2. Enable Wi-Fi on the Board**

The B-L475E-IOT01A board has the **ISM43362 Wi-Fi module**, which Mbed OS supports.  
In your mbed\_app.json, configure Wi-Fi credentials:

json

CopyEdit

{

"config": {

"wifi-ssid": {

"value": "\"YOUR\_WIFI\_SSID\""

},

"wifi-password": {

"value": "\"YOUR\_WIFI\_PASSWORD\""

}

},

"target\_overrides": {

"\*": {

"nsapi.default-wifi-security": "WPA\_WPA2",

"nsapi.default-wifi-ssid": "YOUR\_WIFI\_SSID",

"nsapi.default-wifi-password": "YOUR\_WIFI\_PASSWORD"

}

}

}

**3. Add OTA Update Support**

Mbed OS has **Pelion Device Management** for OTA updates. Follow these steps:

1. Add the **Firmware Update Client** library to your project:

bash

CopyEdit

mbed add https://github.com/ARMmbed/mbed-cloud-client-example

1. **Enable the bootloader:** Create a file mbed\_app.json (if not already there) and add:

json

CopyEdit

{

"target\_overrides": {

"B\_L475E\_IOT01A": {

"target.features\_add": ["BOOTLOADER"],

"platform.stdio-baud-rate": 115200,

"mbed-http": 1,

"update-client.application-details": "0x08020000"

}

}

}

1. **Compile and Flash the Bootloader**:

arduino

CopyEdit

mbed compile -t GCC\_ARM -m B\_L475E\_IOT01A --profile release

**4. Deploy the OTA Update System**

Now that the board has **Wi-Fi access and a bootloader**, you can:

* **Upload new firmware to a remote server.**
* **Use Mbed’s cloud or your own server to push OTA updates.**

For a **local OTA server**, you can:

* Set up an **HTTP(S) server** on your computer.
* Write a script on the board to check for updates and download them.

**Summary**

* ✅ **Wi-Fi enabled** via the ISM43362 module
* ✅ **Bootloader added** for OTA updates
* ✅ **Firmware updates** can be pushed over the internet
* ✅ **Uses Mbed OS and Pelion Device Management**

Would you like an example of a script that **checks for updates** and downloads new firmware?

4o

**You said:**

so I need to have a server on my computer?

**ChatGPT said:**

ChatGPT

Yes, if you want to **wirelessly program** your **B-L475E-IOT01A** using Wi-Fi (OTA updates), you’ll need a way to **host the new firmware** somewhere the board can access. There are two main options:

**1️⃣ Local OTA Server (On Your Computer)**

If your board and computer are on the same network, you can **host the firmware file** on your PC using a simple HTTP server.

**Steps to Set Up a Local OTA Server**

1️⃣ **Compile your new firmware**:

arduino

CopyEdit

mbed compile -t GCC\_ARM -m B\_L475E\_IOT01A --profile release

This will generate a .bin file in BUILD/B\_L475E\_IOT01A/GCC\_ARM/.

2️⃣ **Start a Simple HTTP Server** on your PC:  
If you’re using **Python 3**, run:

yaml

CopyEdit

python -m http.server 8000

This will start a basic web server on **port 8000**, serving files from the current directory.

3️⃣ **Modify the Board’s Code to Download Updates**  
On your board, write a script that **connects to Wi-Fi, checks the server for updates, and downloads new firmware**. Example in Mbed OS:

cpp

CopyEdit

#include "mbed.h"

#include "WiFiInterface.h"

#include "HTTPClient.h"

WiFiInterface \*wifi;

HTTPClient http;

char firmware\_url[] = "http://192.168.1.100:8000/firmware.bin"; // Replace with your PC’s IP

void check\_for\_update() {

printf("Checking for updates...\n");

// Connect to Wi-Fi

wifi = WiFiInterface::get\_default\_instance();

if (!wifi || wifi->connect("YOUR\_SSID", "YOUR\_PASSWORD", NSAPI\_SECURITY\_WPA\_WPA2) != 0) {

printf("Failed to connect to Wi-Fi\n");

return;

}

// Download firmware

printf("Downloading new firmware...\n");

FILE \*file = fopen("/fs/firmware.bin", "wb");

if (file) {

http.get(firmware\_url, file);

fclose(file);

printf("Firmware downloaded!\n");

// Implement flashing mechanism here...

} else {

printf("Failed to open file for writing\n");

}

}

int main() {

check\_for\_update();

}

* This script **downloads new firmware from your PC**.
* You need to implement a way to **flash the new firmware** after downloading.

**2️⃣ Cloud-Based OTA Server**

If you want updates to work **even when your PC is off**, you can upload firmware to a **cloud server** like:

* **AWS S3**
* **Google Cloud Storage**
* **A simple HTTP server like GitHub Pages or an Apache server**

In this case, replace firmware\_url with the public **firmware URL**, and the board will fetch it from the internet.

**Which Method Should You Use?**

✔ **Local OTA Server (on your PC)**

* Faster for development & debugging
* Requires PC and board on same Wi-Fi network
* Great for testing **before deploying to the cloud**

✔ **Cloud-Based OTA Server**

* Board can update from anywhere
* No need to keep your PC running
* Best for **real-world IoT deployments**

Would you like help with **flashing the new firmware automatically** after downloading it?

4o

**You said:**

why would I be flashing firmware? Isn't the firmware already on the board? I thought I would be flashing the software or program?

**ChatGPT said:**

ChatGPT

Good question! Let's clarify the difference between **flashing firmware** and **updating software on the board** in your case.

**1️⃣ What is Firmware vs. Software in Embedded Systems?**

* **Firmware**: This is the core program running on the microcontroller, including the bootloader and the main application.
* **Software/Application**: This refers to the code you write that runs on top of the firmware.

Since your **B-L475E-IOT01A** is a microcontroller board, **your application code is actually part of the firmware**. Unlike a computer where you install software separately from the OS, in embedded systems, updating your code usually means flashing new firmware.

**2️⃣ Why Would You Need to Flash the Firmware for Wireless Updates?**

In a microcontroller:

* Your application **is compiled into a single binary file (.bin or .hex)**.
* To update your code, you **replace the old binary with the new one**.
* This requires **writing the new binary into flash memory** (hence, "flashing" the board).

If you want to update your board **wirelessly**, you need a way to:

1. **Download the new binary file** over Wi-Fi.
2. **Write it to flash memory** to replace the existing program.
3. **Restart the board** to run the updated code.

**3️⃣ How Can You Flash Software Wirelessly?**

Since you don't want to manually connect the board every time to reprogram it, you need a **bootloader** that can:

1. **Check for new firmware updates over Wi-Fi**.
2. **Download the new program** to a temporary location.
3. **Write the new program into flash memory**.
4. **Reboot the board to run the updated code**.

**4️⃣ How Can You Do This on Your Board?**

Your **B-L475E-IOT01A** supports **Mbed OS OTA updates**, which means:

* A small **bootloader** remains on the board.
* Your main program runs separately in memory.
* When an update is detected, the **bootloader erases the old program and writes the new one**.

**Steps to Enable Wireless Updates**

1. **Ensure the board has a bootloader**:
   * Mbed OS has built-in bootloader support for STM32.
   * You may need to enable it by modifying mbed\_app.json.
2. **Modify your program to check for updates**:
   * Download the new .bin file over Wi-Fi.
   * Store it in a reserved flash memory area.
3. **Trigger the update**:
   * The bootloader swaps the new firmware into place.
   * The board restarts and runs the new program.

Would you like help setting up a **bootloader** to make this process automated?