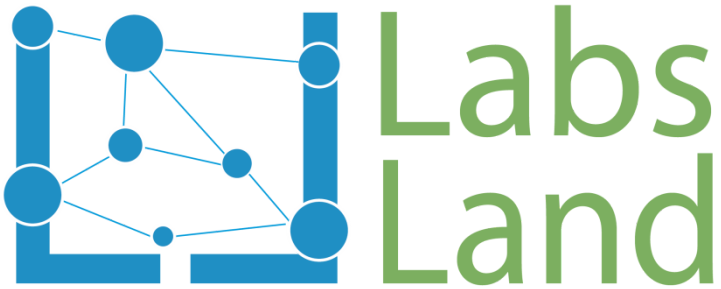


# GENERAL GUIDE FOR USING THE STM32 NUCLEO LABORATORY (WITHOUT ONLINE IDE)

## Table of contents

# 1. Introduction

The STM32 Nucleo remote lab is intended to allow experimentation with the Nucleo WB55RG development system from STMicroelectronics. The development board supported by the lab is ideal for the realization of Internet of Things (IoT) and Low Energy Consumption (CLEC) experiments.

The lab version referred to in this guide is the version without an online IDE. In this lab version the user can create a program for the STM32 Nucleo using any of the standard tools available (in general offline tools like Keil IDE, or STM32CubeIDE, or a GCC-based toolchain). You can then compile that program and upload the resulting binary file (.bin / .hex / .elf / .axf) directly to the remote lab. This approach is very flexible as it allows you to use virtually any tool.

Note that there is an alternative version of the lab (which is not the one referred to in this guide) that provides a simplified online IDE.

# 2. The laboratory

## How it works

LabsLand's STM32 Nucleo NO-IDE lab is relatively simple to use.


*Figure 1. Initial laboratory interface.*

An overview of the process the user follows to use the laboratory is as follows:

1. Configure the microcontroller and write the code, using any standard tool (e.g., STM32CubeMX, Keil, STM32CubeIDE...).
2. Compile the code in any of these tools, resulting in a binary file. Supported binary files are .bin, .hex, .elf or .axf. Depending on the tool they will be generated in one directory or another; but normally all tools will generate one or more of these files.
3. Access the LabsLand remote lab and select the binary file to be tested (.bin / .hex / .elf / .axf). Figure 1 shows the initial screen of the lab. Figure 2 shows the file selection dialog.
4. Once programmed on the board, interact with and display the board remotely as normal. Figure 3 shows the lab running the user code and allowing interaction with the board.

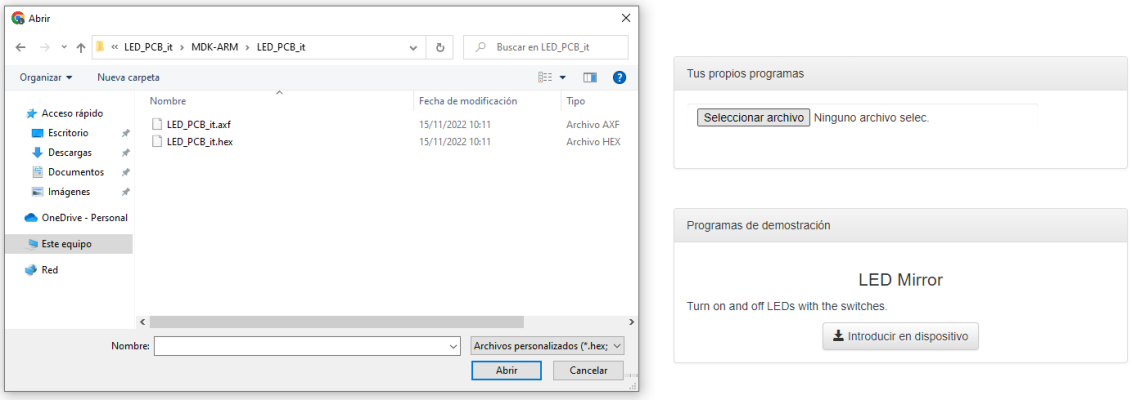


*Figure 2. Binary file selection dialog.*

Finally, all that remains is to test that the program works as expected by interacting with the laboratory through the user's web interface.


*Figure 3. Main interface of the laboratory, interacting with the board.*

# Hardware configuration

Before starting with the programming of the development board, it is necessary to set-up the microcontroller hardware. This set-up is as or almost more important than the programming itself. It defines the functionality of each pin of the microcontroller, enables various functionalities offered by each development system and can modify the internal configurations of the microprocessor.

## Fixed hardware

The remote laboratory naturally has a specific hardware configuration. That is, the lab consists not only of the processor as such, but also of the Nucleo-WB55RG development board, plus a set of peripherals connected in a specific way.

The lab user will be able to configure various features (e.g. the counters) and program it freely; but of course he/she will not be able to change the physical peripherals connected or rearrange the cables.

In order to be able to work comfortably with the lab a schematic diagram and a Fritzing style diagram are provided, available in the user web interface (see later section with link to these diagrams). By means of these diagrams the user can find out which peripherals are available, to which pins they are connected, which inputs and outputs must be respected, etc.

Note that, in particular, the inputs and outputs must always keep their direction (input or output). You can choose whether to use them or not, but you cannot change an input pin to an output pin or vice versa. Another aspect of the hardware design that also cannot be changed is the serial communication transmission parameters (115200 Baud and 8N1).

## Schematic and Fritzing diagrams

To find the schematic diagram click here.

To find the Fritzing diagram click here.

## Pin description

As a summary, all pins to be configured as input or output are listed in the following table.

| Pin | Function | | Pin | Function |
|---|---|---|---|---|
| PA0 | Input | | PB11 | Input |
| PA1 | Input | | PB12 | Input |
| PA2 | Input | | PB13 | Output |
| PA3 | Input | | PB14 | Output |
| PA4 | Output | | PB15 | Output |
| PA5 | Output | | PC0 | Input |
| PA6 | Input | | PC1 | Input |
| PA7 | Output | | PC2 | Input |
| PA8 | Output | | PC3 | Input |
| PA9 | Output | | PC4 | Output |
| PA10 | Output | | PC5 | Input |
| PA15 | Output | | PC6 | Input |
| PB0 | Output | | PC10 | Input |
| PB1 | Output | | PC12 | Input |
| PB2 | Input | | PC13 | Input |
| PB4 | Input | | PD0 | Output |
| PB5 | Output | | PS1 | Output |
| PB8 | Input | | PE4 | Output |
| PB9 | Input | | | |

*Table 1. GPIO general description.*

# 3. Creation of the binary

## Introduction

The objective at the beginning will be to create a program and generate a binary file by compiling that program. In the case of this version of the lab (no online IDE) the user will use any toolchain he wants, which can be the same as the ones he would use with a traditional "on-site" physical board.

While this happens outside the LabsLand lab, typically the first program a user will use is STM32CubeMX to generate an empty project with a given initial configuration.

If starting an STM32CubeMX project from scratch it will be critical to take into account the Fritzing and schematic diagrams provided, since for any project to work properly it will need to take into account the hardware connections and configuration.

While starting from scratch is an option, we also provide a template and a base configuration of STM32CubeMX that is directly compatible with the lab. The goal is that it can be used as a starting point. You can download it here.

Beyond the initial project generation stage, typically a user, just as they would if working with a "live" board, will use a particular toolchain. There are a variety of toolchains available, including development tools such as development environments (IDEs), different compilers, etc.

Regardless of the toolchain used, the result after compiling the code should be a binary file ending in .bin, .hex, .elf or .axf. Although they are different formats, they all generally contain the same thing: the compiled version of the program. Depending on the toolchain used, one or another will be generated by default. Several toolchains, in fact, generate more than one. Normally, even though they generate two different formats, they will be the same program, and will therefore be redundant.

This binary file with the compiled program will be the input received by the remote lab. The remote lab will assign a board to the user, program this binary on that board, and allow the user to interact with the board to test it.

## Main code development tools for the creation of lab upload files

As mentioned more briefly in the previous section, for the creation of the files to be uploaded to the remote laboratory (*.bin*, *.hex*, *.elf* or *.axf)*, there are several toolchains that can be used. The important thing is to respect the configured hardware, that ideally the code does not contain compilation errors (and ideally no other errors either), and that the result of this compilation is at least one of the 4 files mentioned above.

There are different possible toolchains, but the following are 4 popular combinations of the many possible combinations for obtaining *.bin*, *.hex*, *.elf* or *.axf* files.

- STM32CubeMX + Keil uVision.
    - First, the microcontroller is configured using the STM32CubeMX. Once it is defined, the code is generated and edited in the Keil uVision. The code can be programmed and compiled in Keil. Subsequently, the binary file generated by the program must be collected and uploaded to the online laboratory.
- STM32CubeIDE.
    - This program is practically identical to the previous toolchain but concentrating everything in one. First the microcontroller is configured by means of the STM32CubeIDE, in the same way as in the STM32CubeMX. Then, for programming and compilation, it is not necessary to change application, the STM32CubeIDE itself has that technology. Afterwards, the binary file generated by the program must be collected and uploaded to the online laboratory.
- Keil Studio Online.
    - This tool allows the programming and compilation of the code online, without the need to download or install an application. Subsequently, the binary file generated by the program must be collected and uploaded to the online laboratory. The configuration of the microcontroller is not as visual as in the previous ones, but it also allows to define its functionalities to the user's taste.

This version of the lab is based on the user simply providing a binary file to be programmed on the remote board, so that the user is free to use the platform with which he/she feels most comfortable, without any particular limitation in this respect.

Of course, even then this file will be programmed in a specific hardware configuration, so for it to work correctly the user will have to take this configuration into account in the configuration of his project and in the development of his program.