

NEEEICUM

núcleo de estudantes de engenharia  
eletrônica industrial e computadores  
da universidade do minho

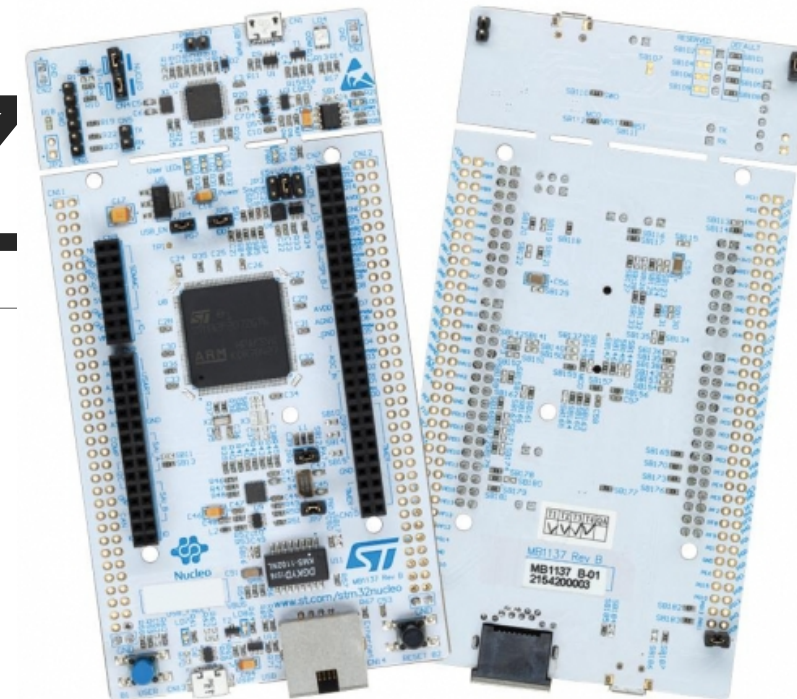
ESRG

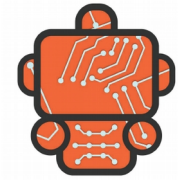
EMBEDDED SYSTEMS  
RESEARCH  
GROUP

# STM 32 F767Z

TOOLS SET UP AND BASIC CONCEPTS

ÁLVARO LEITE  
NOV 2019





# Requirements

---

Computer with Windows OS

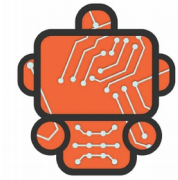
Internet connection

15 Gb of free space on disk

Micro-USB cable

STM32 development board

Java Runtime Environment (JRE)



# Agenda

---

Skill level: Beginner

Why STM32?

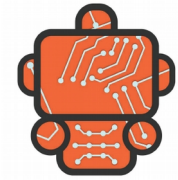
Choosing the IDE

What tools are needed?

ST-LINK/V2

Keil MDK-ARM

STM32CubeMX



# Agenda

---

Skill level: Beginner

Create the first Project

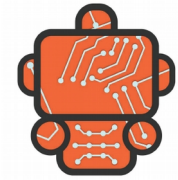
Compile the program and Flash the board

Terminal

File Tree

Resources

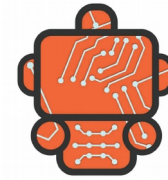
Conclusion



# Why STM32?

---

- 32 bits professional development board
- Cheap to buy and develop with
- Same toolchain and API through all ranges of products
- Cross compatibility
- Lots of documentation
- And much much more...



# Choosing the IDE

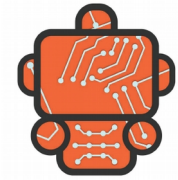
---

During the course we will use the Keil ARM IDE as our main IDE. However there are other alternatives that you may consider using in the future.

The decision to use the Keil ARM IDE was made based in your past experiences, and in the well known performance achieved by those IDE.

A recent alternative for non Windows OS, is the STM32CubeIDE, if you want use that IDE, you must assume all responsibility about that. You can find a installation guide at this link: <https://bit.ly/330GSHS>

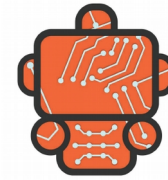
If you want to install the Keil continue, if you want to install STM32CubeIDE, go to slide 27.



# What tools are needed?

---

- ST-LINK/V2: It's a ST tool that provides capabilities to load code to the board and on-chip debug functionalities.
- STM32CubeMX: It has a graphical software configuration tool that allows the generation of C initialization code using graphical wizards. And it makes developers' lives easier by reducing development effort, time and cost.
- Keil ARM: It's an IDE that combines project management, run-time environment, build facilities, source code editing and program debugging in a single powerful environment.



# ST-LINK/V2

---

The ST-LINK/V2 is an in-circuit debugger and programmer for the STM8 and STM32 microcontroller families. The single wire interface module (SWIM) and JTAG/serial wire debugging (SWD) interfaces are used to communicate with any STM8 or STM32 microcontroller located on an application board.

<https://bit.ly/2MCx4zl>

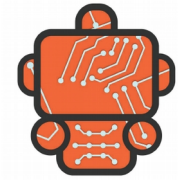


# ST-LINK/V2

---

## Get Software

Part Number	Supplier	Download
+ STSW-LINK009	ST	Get Software



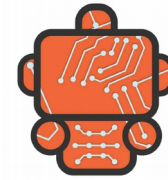
# ST-LINK/V2

---

Install it after download

If your computer has a 32 bits system use “dpint\_x86.exe”

If your computer has a 64 bits system use “dpint\_amd64.exe”

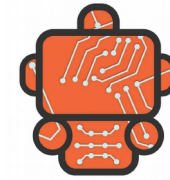


# Keil MDK-ARM

---

The MDK-ARM is a complete software development environment for Cortex™-M, Cortex-R4, ARM7™ and ARM9™ processor-based devices. MDK-ARM is specifically designed for microcontroller applications, it is easy to learn and use, yet powerful enough to deal with demanding embedded applications.

<https://www.keil.com/download/product/>



# Keil MDK-ARM



## Download Products

Select a product from the list below to download the latest version.



### MDK-Arm

Version 5.24a (July 2017)

Development environment for Cortex and Arm devices.



### C51

Version 9.57 (November 2017)

Development tools for all 8051 devices.



### C251

Version 5.59 (October 2016)

Development tools for all 80251 devices.

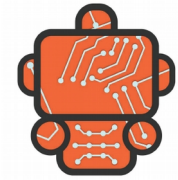


### C166

Version 7.56 (October 2016)

Development tools for C166, XC166, & XC2000 MCUs.

Keil products use a [License Management](#) system - without a current license the product runs as a Lite/Evaluation edition with a few [Limitations](#).



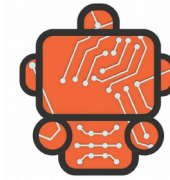
# Keil MDK-ARM

---

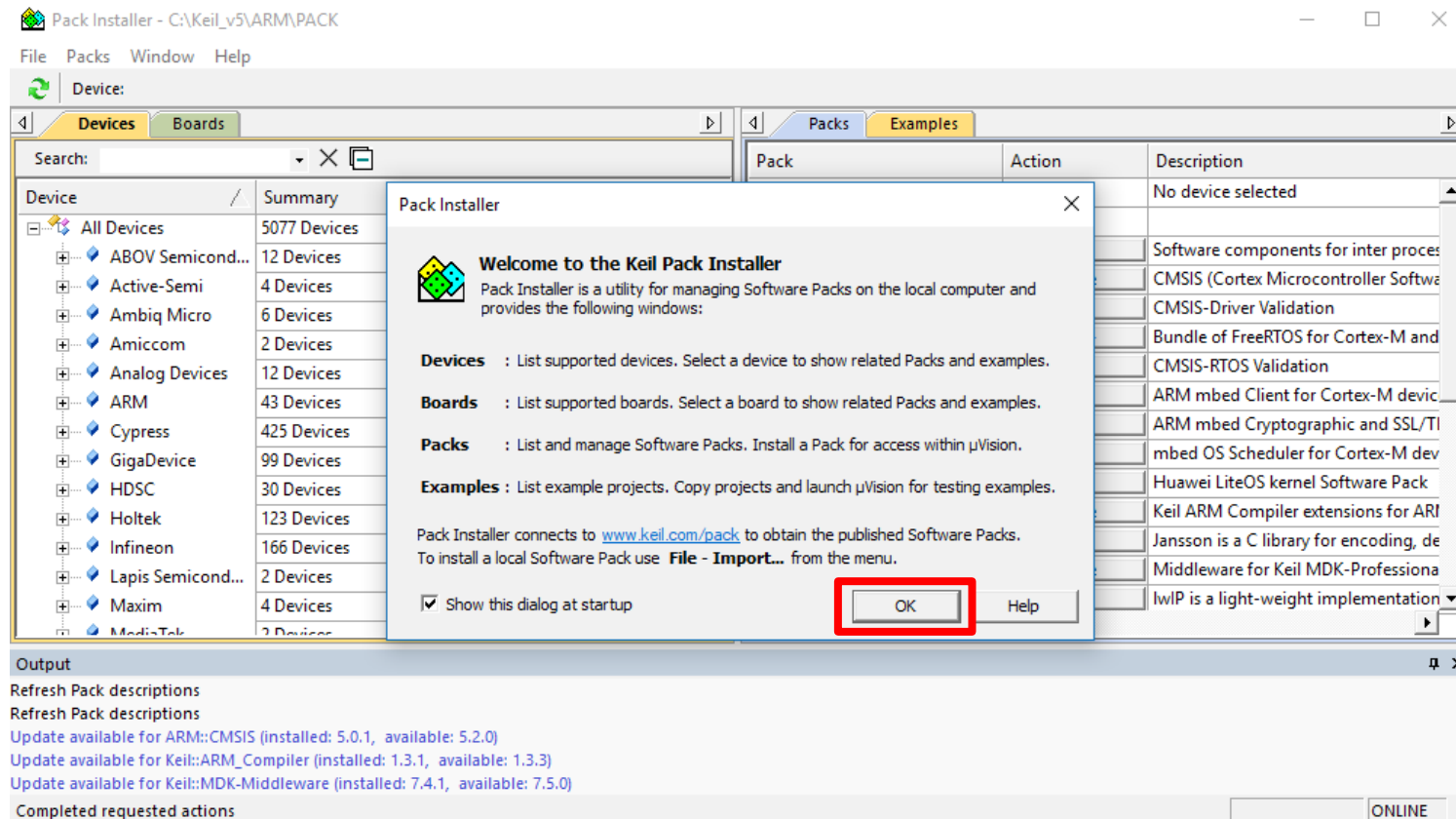
After download install it

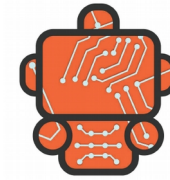
And open Keil for the first time





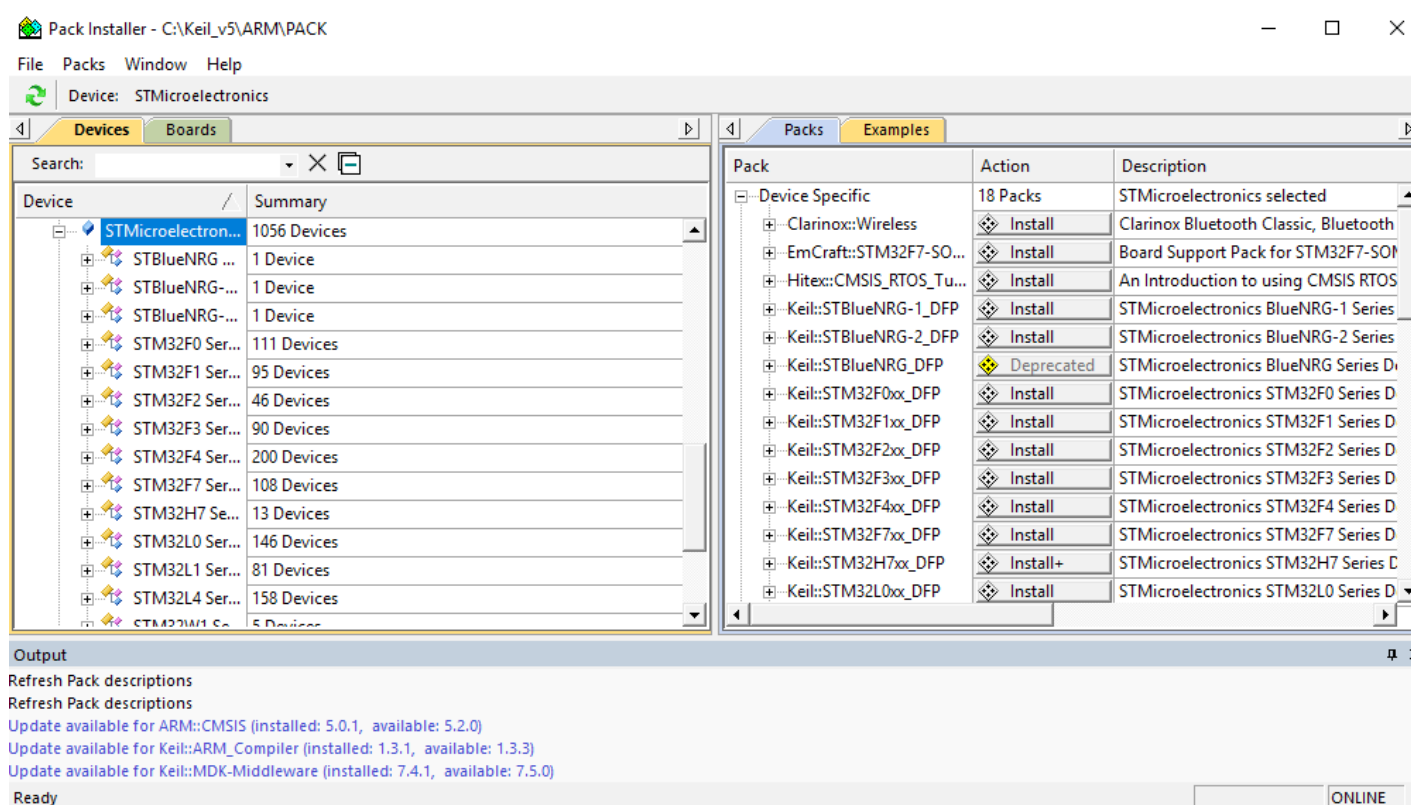
# Keil MDK-ARM

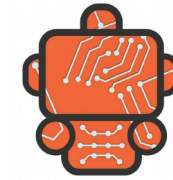




# Keil MDK-ARM

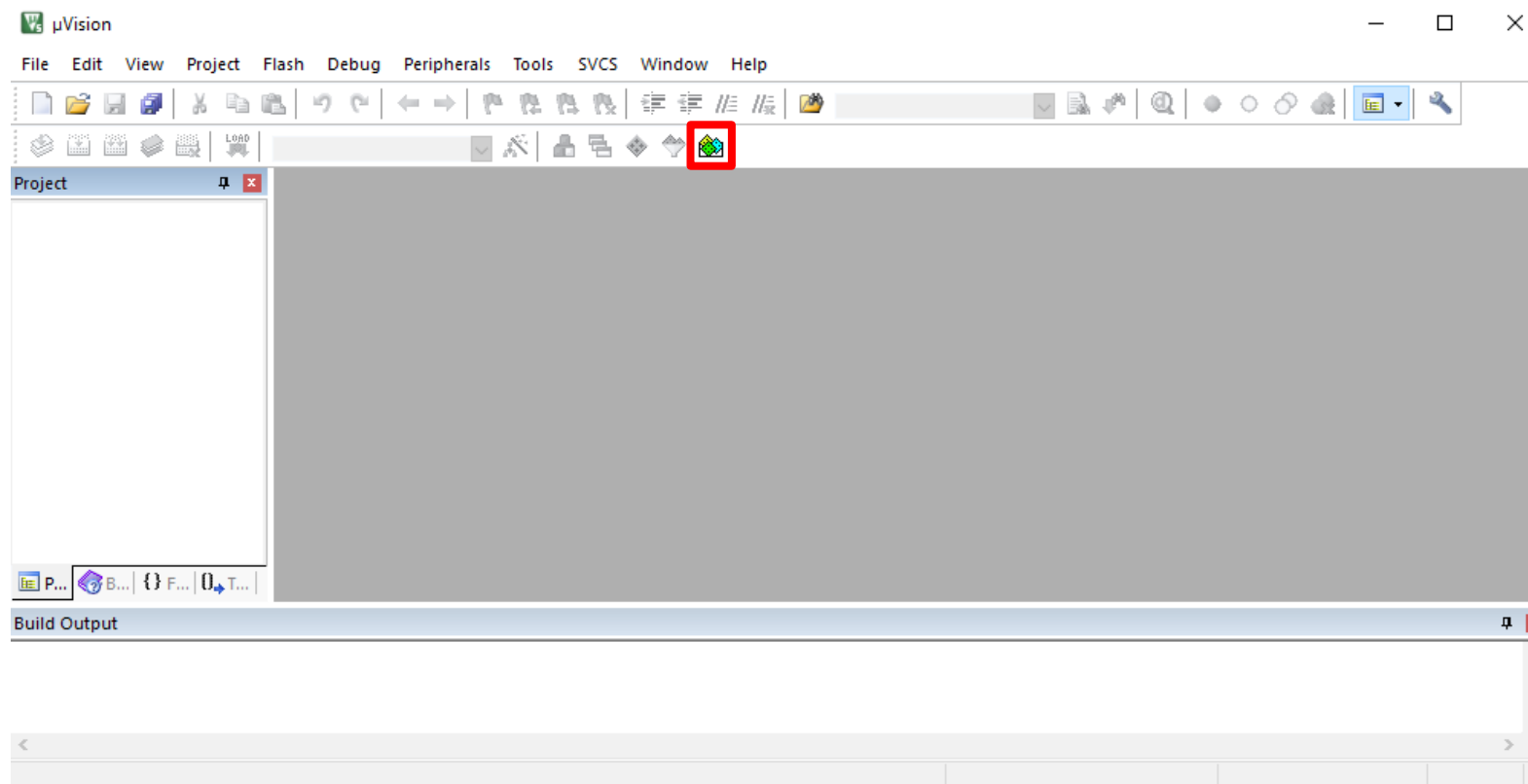
If you don't see this menu...



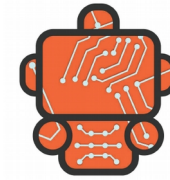


# Keil MDK-ARM

Click here...







# Keil MDK-ARM

In “Device” tab select: STMicroelectronics -> STM32F7 Series -> STM32F767 -> STM32F767ZI

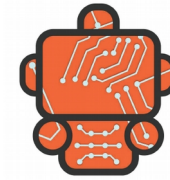
If you still don't see this menu wait until the update finishes

The screenshot shows the Keil MDK-ARM interface. On the left, the 'Device' tab is active, displaying a tree view of STM32F7 Series devices. The 'STM32F767ZI' device is selected, showing its specifications: ARM Cortex-M7, 216 MHz, 512 kB RAM, 4 MB ROM. On the right, the 'Packs' window is open, showing a list of 22 packs. The 'ARM::CMSIS' pack is highlighted, and its details are shown in the right pane. The 'Output' window at the bottom shows the status 'Ready' and 'ONLINE'.

Device	Count
STM32F4 Series	200 Devices
STM32F7 Series	108 Devices
STM32F722	10 Devices
STM32F723	9 Devices
STM32F732	5 Devices
STM32F733	6 Devices
STM32F745	10 Devices
STM32F746	16 Devices
STM32F756	8 Devices
STM32F765	12 Devices
STM32F767	12 Devices
STM32F767BG	1 Device
STM32F767BI	1 Device
STM32F767IG	2 Devices
STM32F767II	2 Devices
STM32F767NG	1 Device
STM32F767NI	1 Device
STM32F767VG	1 Device
STM32F767VI	1 Device
STM32F767ZG	1 Device
STM32F767ZI	1 Device
STM32F768	1 Device
STM32F769	8 Devices

Pack	Action	Description
ARM::AMP	Install	Software components for inter processor communication (Asymmetric Multi Processing ARM)
ARM::CMSIS	Update	CMSIS (Cortex Microcontroller Software Interface Standard)
ARM::CMSIS-Driver_Val...	Install	CMSIS-Driver Validation
ARM::CMSIS-FreeRTOS	Install+	Bundle of FreeRTOS for Cortex-M and Cortex-A
ARM::CMSIS-RTOS_Val...	Install	CMSIS-RTOS Validation
ARM::mbedClient	Install	ARM mbed Client for Cortex-M devices
ARM::mbedTLS	Install	ARM mbed Cryptographic and SSL/TLS library for Cortex-M devices
ARM::minar	Install	mbed OS Scheduler for Cortex-M devices
Huawei::LiteOS	Install	Huawei LiteOS kernel Software Pack
Keil::ARM_Compiler	Update	Keil ARM Compiler extensions for ARM Compiler 5 and ARM Compiler 6
Keil::Jansson	Install	Jansson is a C library for encoding, decoding and manipulating JSON data
Keil::MDK-Middleware	Update	Middleware for Keil MDK-Professional and MDK-Plus
lwIP::lwIP	Install	lwIP is a light-weight implementation of the TCP/IP protocol suite
Micrium::RTOS	Install	Micrium software components
Oryx-Embedded::Midd...	Install	Middleware Package (CycloneTCP, CycloneSSL and CycloneCrypto)
RealTimeLogic::SharkS...	Install	Version 1.0.0 (latest on website): all and super fast pre-compiled SharkSSL TLS library for Cortex-M
RealTimeLogic::SMQ	Install	Simple Message Queues (SMQ) is an easy to use IoT publish subscribe connectivity protocol
redlogix::redBlocks-Si...	Install	C Target Library for the redBlocks WYSIWYG SiL Simulator (supports flexible hardware simu
rt-thread::rtthread	Install	rt-thread software components
wolfSSL::wolfSSL	Install	Light weight SSL/TLS and Crypt Library for Embedded Systems
YOGITECH::fRSTL_AR...	Deprecated	!!! DEPRECATED Product !!! YOGITECH fRSTL Functional Safety EVAL Software Pack for ARM
YOGITECH::fRSTL_ST...	Deprecated	!!! DEPRECATED Product !!! YOGITECH fRSTL Functional Safety EVAL Software Pack for STM3

Output  
Refresh Pack descriptions  
Refresh Pack descriptions  
Ready  
ONLINE



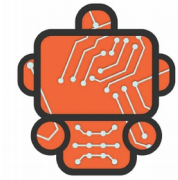
# Keil MDK-ARM

And install the following 5 packs

The screenshot shows the Keil MDK-ARM interface with the 'Packs' window open. The 'Packs' window displays a list of available software packs, with five specific packs highlighted by red rectangles:

- Keil::STM32F7xx\_DFP** (Up to date): STM32F7677I1Tx selected. Description: STMicroelectronics STM32F7 Series Device Support, Drivers and Examples.
- ARM::CMSIS** (Up to date): CMSIS (Cortex Microcontroller Software Interface Standard).
- ARM::CMSIS-Driver\_Val...** (Up to date): CMSIS-Driver Validation.
- Keil::ARM\_Compiler** (Up to date): Keil ARM Compiler extensions for ARM Compiler 5 and ARM Compiler 6.
- Keil::MDK-Middleware** (Up to date): Middleware for Keil MDK-Professional and MDK-Plus.

The 'Devices' window on the left shows a list of STM32F7 series devices, with STM32F7677I1Tx selected.



# STM32CubeMX

---

STM32Cube includes STM32CubeMX, a graphical software configuration tool that allows the generation of C initialization code, for STM32 microcontrollers very easily, using graphical wizards.

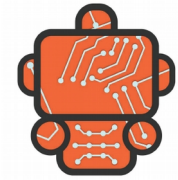
<http://www.st.com/en/development-tools/stm32cubemx.html>



# STM32CubeMX

## Get Software

Part Number ▲	General Description	Software Version ◆	Download	Previous versions ◆
+ STM32CubeMX	STM32Cube initialization code generator	5.3.0	<a href="#">Get Software</a>	<a href="#">Select version ▼</a>



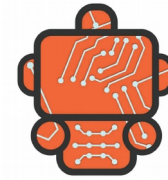
# STM32CubeMX

---

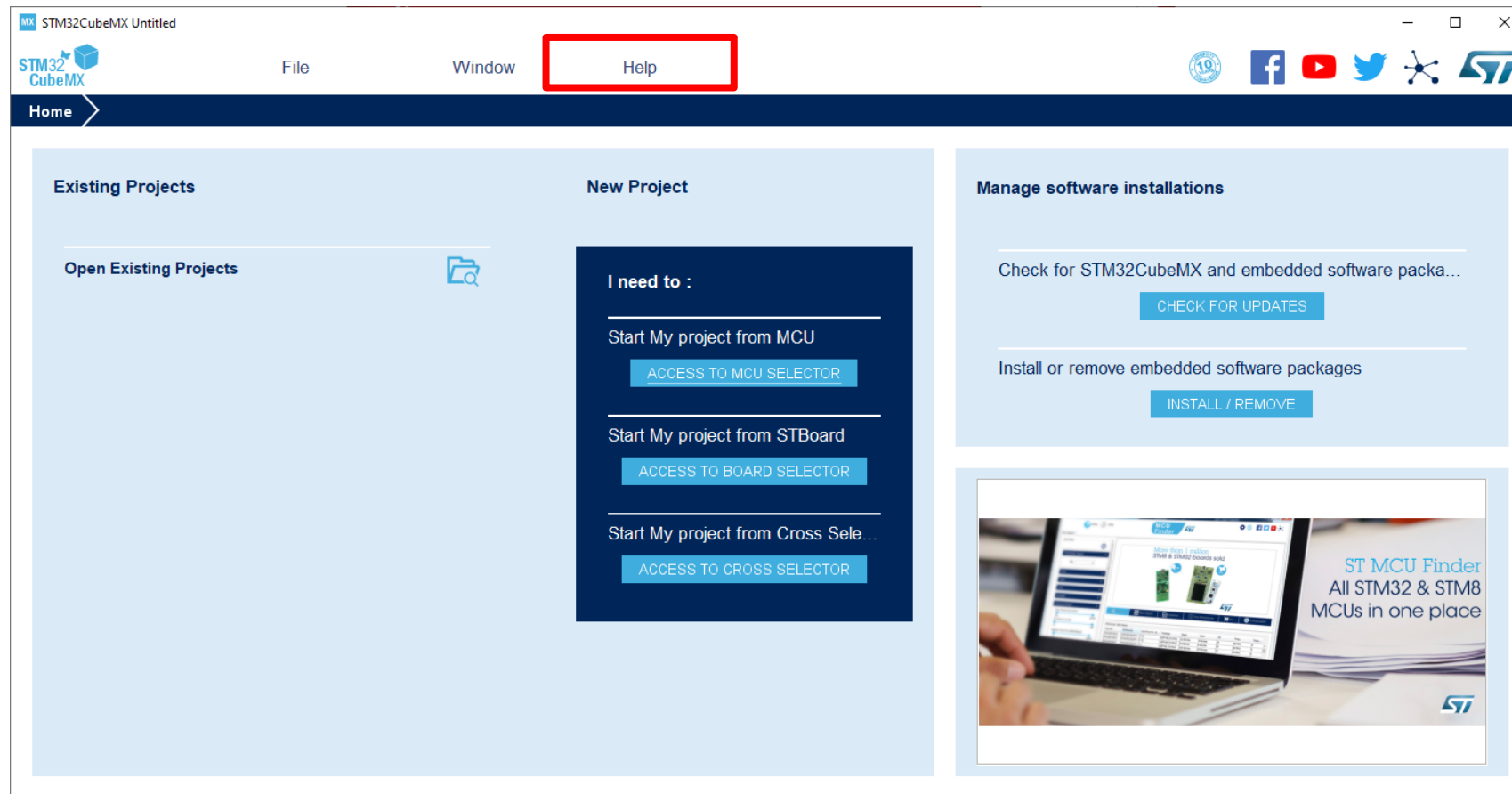
Install Cube after download

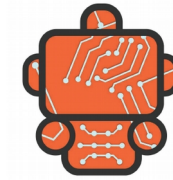
It requires Java Runtime Environment

Open STM32CubeMX when installation is finished

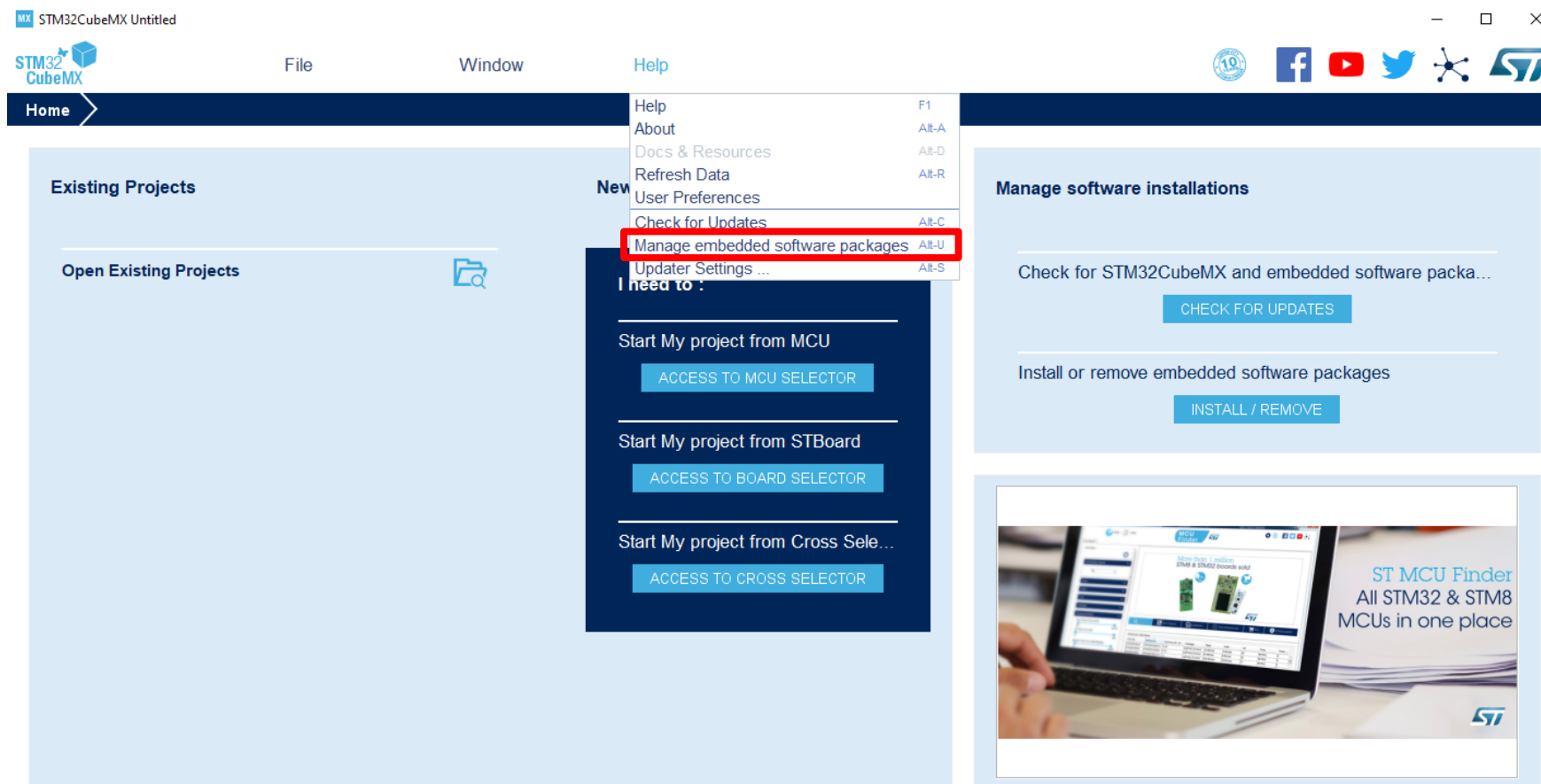


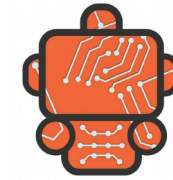
# STM32CubeMX





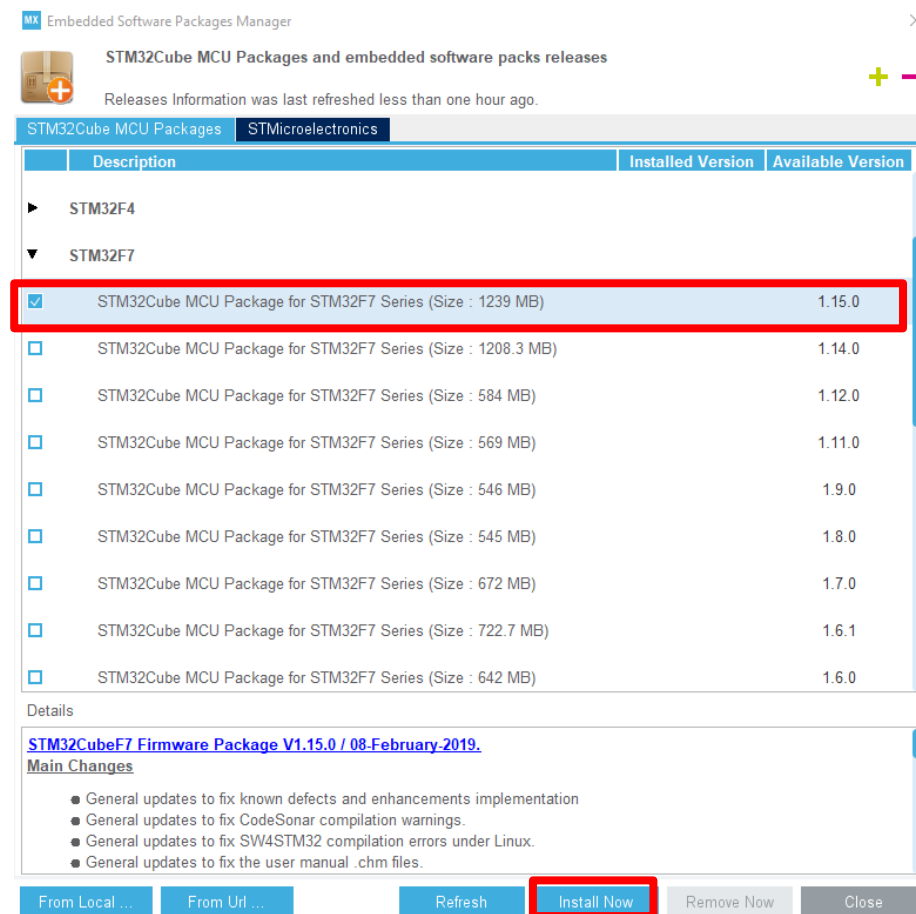
# STM32CubeMX





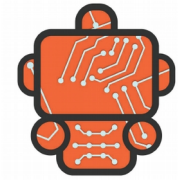
# STM32CubeMX

Select these:



Click “Install”





# STM32CubeMX

---

And... that's it. Installation complete.

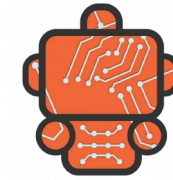
If you want to gain some space on disk, deleting some installation's wastes, continue, otherwise jump to page 27.

# Cleaning

---

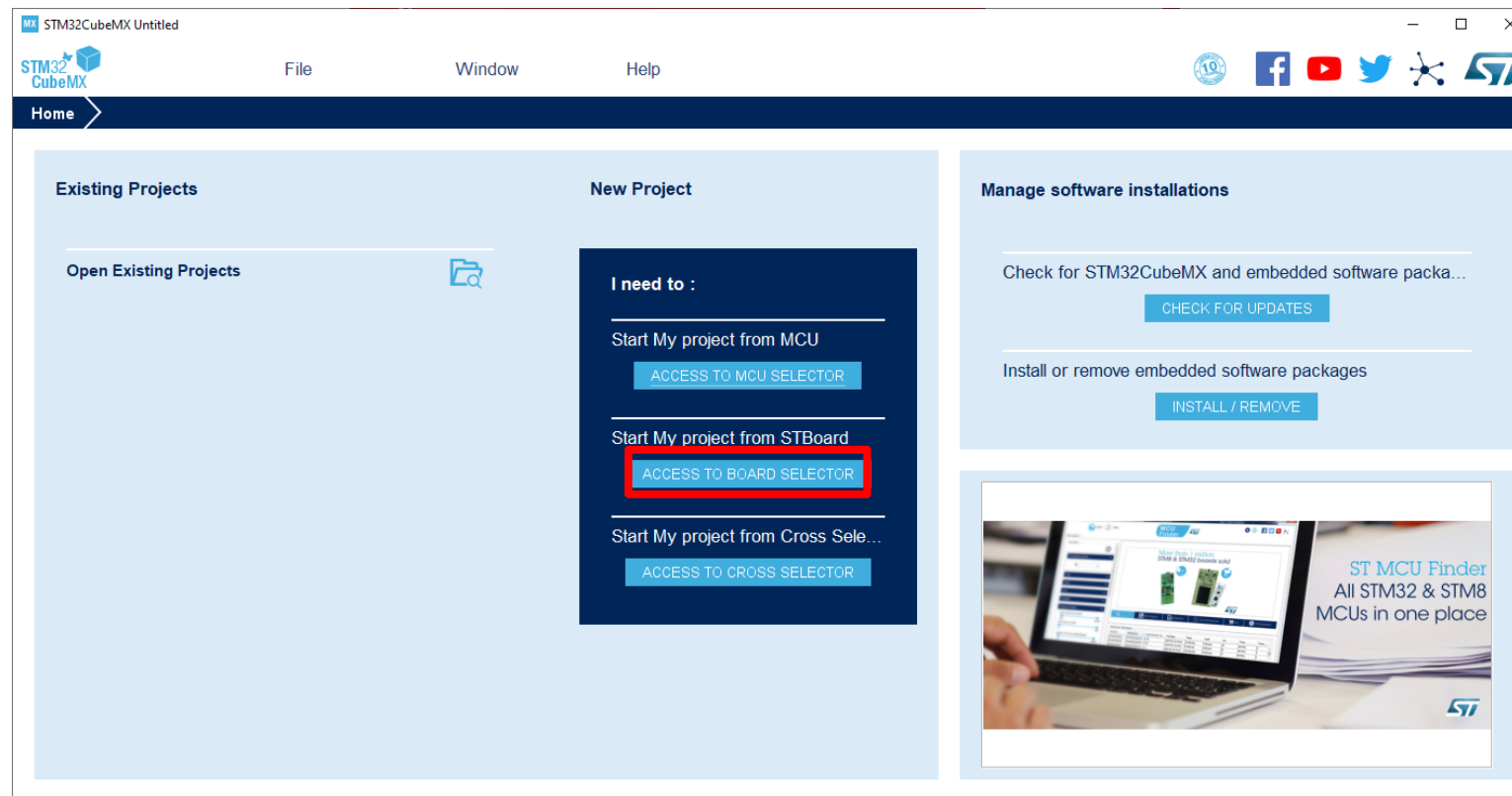
In “C:\Keil\_v5\ARM\PACK\.Download” delete all files

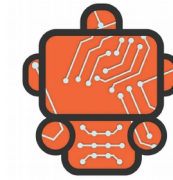
In “C:\Users\[username]\STM32Cube\Repository” delete all the **.zip** files



# Create the first Project

Open STM32CubeMX. Select “Access to Board Selector”.





# Create the first Project

Search for “767”, and double click in “Nucleo-F767ZI”.

MX New Project from a Board

MCU/MPU Selector | Board Selector | Cross Selector

Board Filters

Part Number Search

Vendor

Type

MCU/MPU Series

Other

Price = 23.0

Oscillator Freq. = 0 (MHz)

Peripheral

- Accelerometer
- Analog I/O
- Arduino Form Factor
- Audio Line In
- Audio Line Out
- Battery
- Button
- CAN
- Camera
- Compass
- Custom Form Factor
- Digital I/O
- Ethernet
- Gyroscope
- I2C
- Joystick
- LCD Display (Graphics)

New multicore STM32MP1 Series  
for Industrial and IoT applications

STM32MP1

OpenSTLinux Distribution

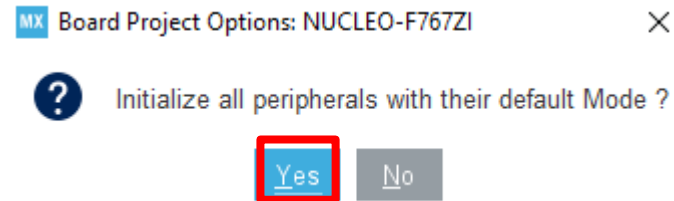
Boards List: 1 item

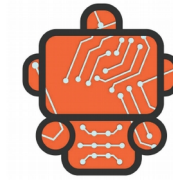
	Overview	Part No	Type	Marketing Status	Unit Price (US\$)	Mounted Device
☆		NUCLEO-F767ZI	Nucleo144	Active	23.0	<a href="#">STM32F767ZITx</a>

# Create the first Project

---

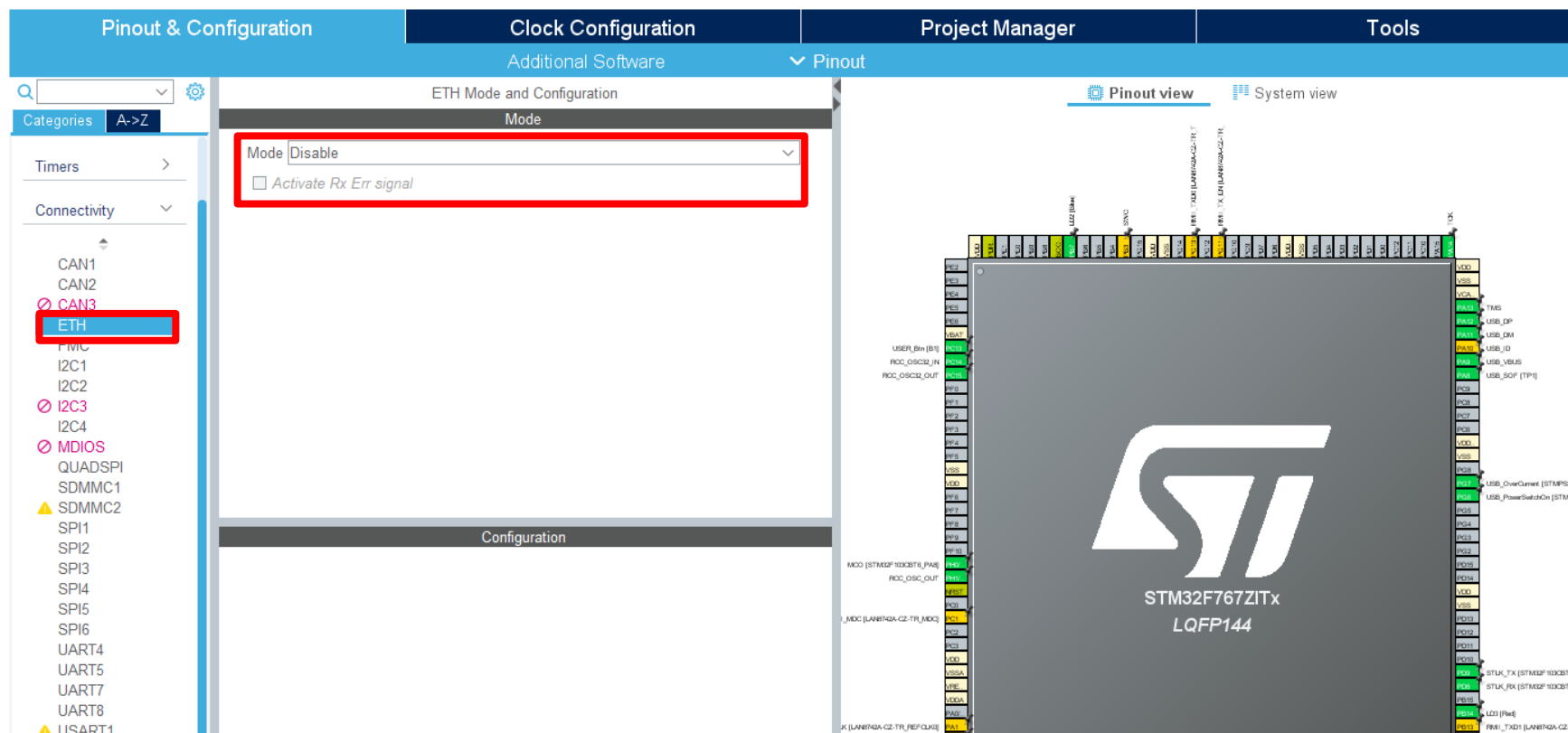
Initialize all peripherals in default mode.

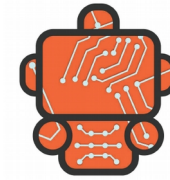




# Create the first Project

Disable the Ethernet peripheral.

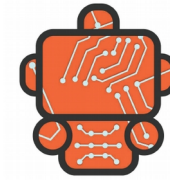




# Create the first Project

Disable the USB peripheral.

The screenshot displays the STM32CubeMX software interface. On the left, the 'Connectivity' tree lists various peripherals, with 'USB\_OTG\_FS' selected and highlighted by a red rectangle. The main window, titled 'USB\_OTG\_FS Mode and Configuration', shows the 'Mode' dropdown menu set to 'Disable', also highlighted by a red rectangle. Below the mode selection, there are checkboxes for 'Activate\_SOF' and 'Activate\_VBUS', both of which are currently unchecked. The right side of the interface shows the 'Pinout view' of the STM32F767ZITx LQFP144 microcontroller, with various pins and their functions labeled.



# Create the first Project

Go to “Project Manager”, give a name to the project, choose the location of the project, set the structure to advanced, and set the Toolchain to “MDK-ARM V5”.

Home > STM32F767ZITx - NUCLEO-F767ZI > **Untitled - Project Manager** > GENERATE CODE

Pinout & Configuration | Clock Configuration | **Project Manager** | Tools

**Project**

Project Settings

Project Name  
Blinking\_LED

Project Location  
C:\Users\Alvaro\Documents\STM32\Blinking\_Led Browse

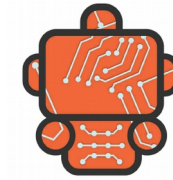
Application Structure  
Advanced ☐ Do not generate the main()

**Code Generator**

Toolchain Folder Location  
C:\Users\Alvaro\Documents\STM32\Blinking\_Led\Blinking\_LED\

Toolchain / IDE  
MDK-ARM V5 ☐ Generate Under Root

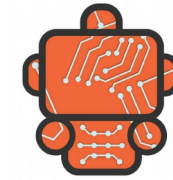




# Create the first Project

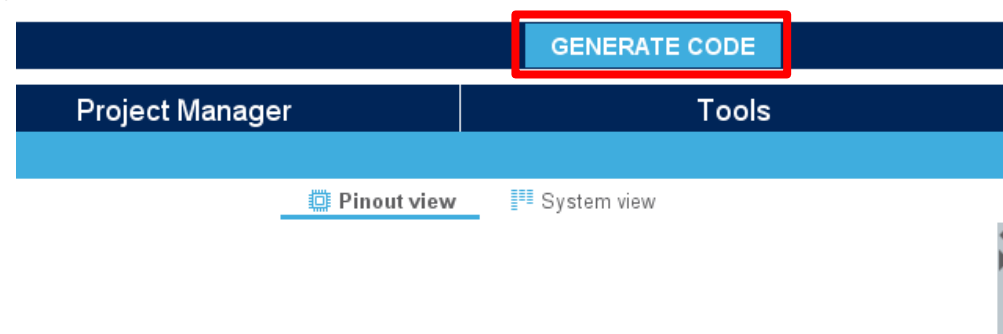
Under “Code Generator” select “Copy only the necessary library files”, and “Generate peripheral initialization as a pair of ‘.c/.h’ files per peripheral”.

Pinout & Configuration	Clock Configuration	Project Manager	Tools
Project	<p>STM32Cube MCU packages and embedded software packs</p> <ul style="list-style-type: none"><li><input type="radio"/> Copy all used libraries into the project folder</li><li><input checked="" type="radio"/> Copy only the necessary library files</li><li><input type="radio"/> Add necessary library files as reference in the toolchain project configuration file</li></ul>		
Code Generator	<p>Generated files</p> <ul style="list-style-type: none"><li><input checked="" type="checkbox"/> Generate peripheral initialization as a pair of '.c/.h' files per peripheral</li><li><input type="checkbox"/> Backup previously generated files when re-generating</li><li><input checked="" type="checkbox"/> Keep User Code when re-generating</li><li><input checked="" type="checkbox"/> Delete previously generated files when not re-generated</li></ul>		
	<p>HAL Settings</p> <ul style="list-style-type: none"><li><input type="checkbox"/> Set all free pins as analog (to optimize the power consumption)</li><li><input type="checkbox"/> Enable Full Assert</li></ul>		

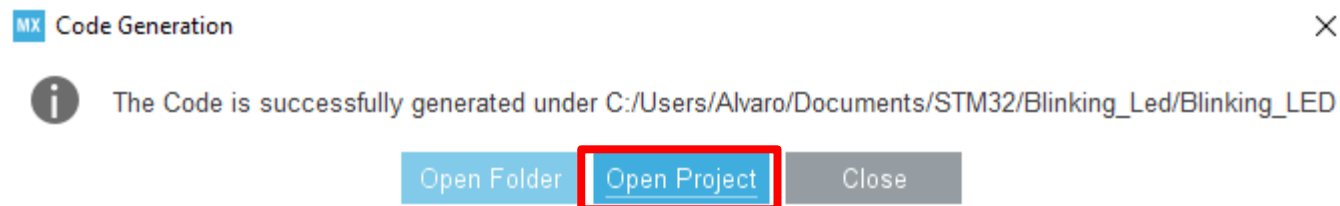


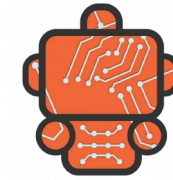
# Create the first Project

Generate the code.



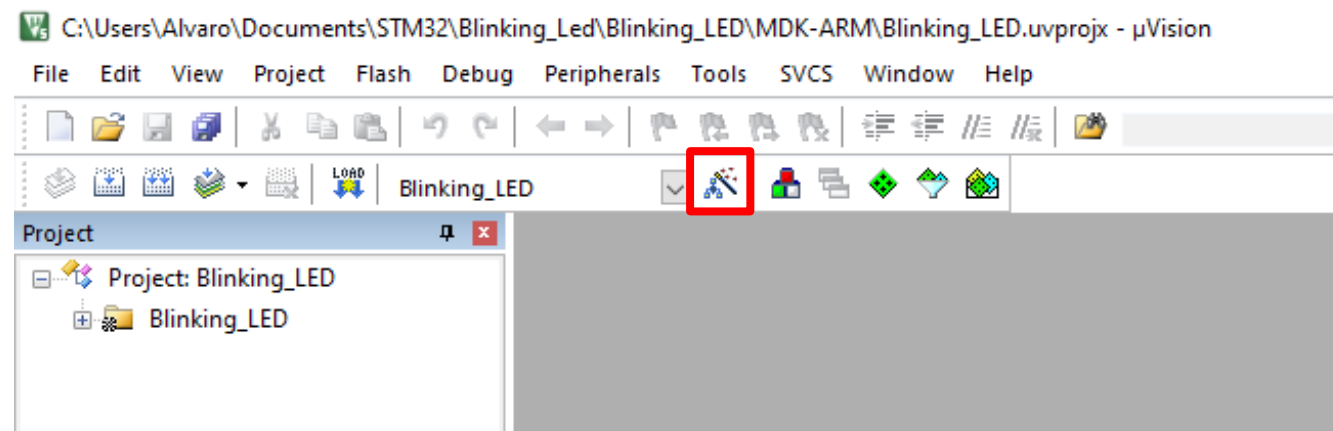
And Open the Project

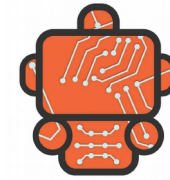




# Create the first Project

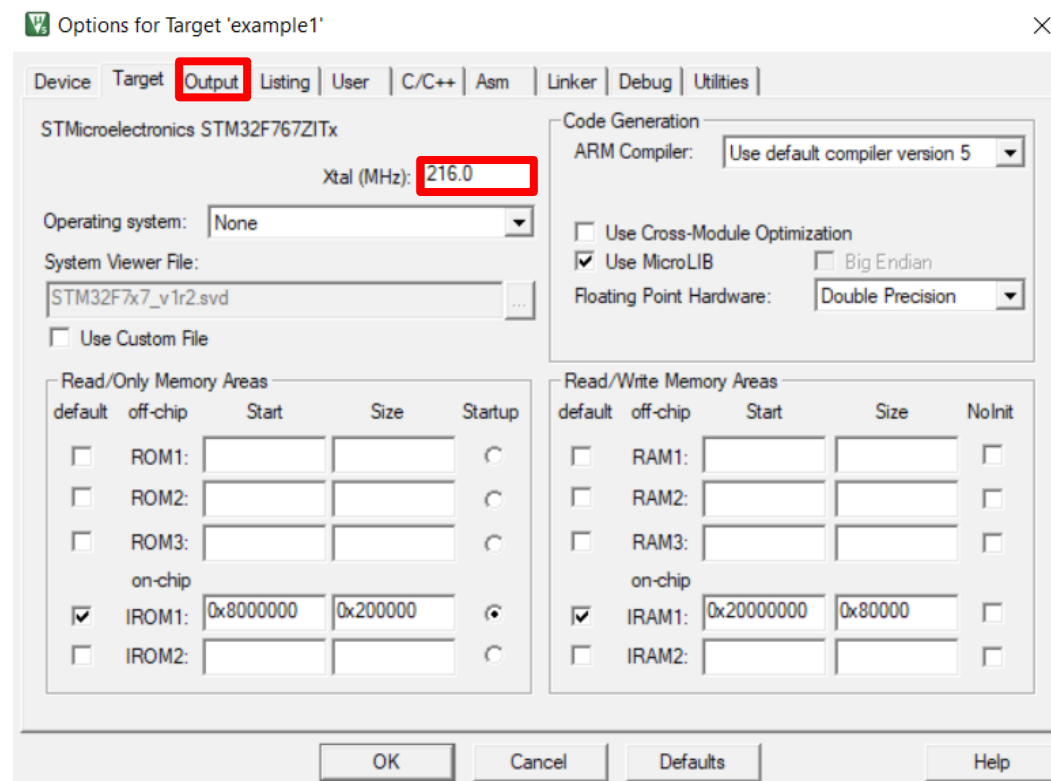
Open the settings menu.

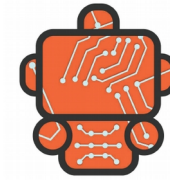




# Create the first Project

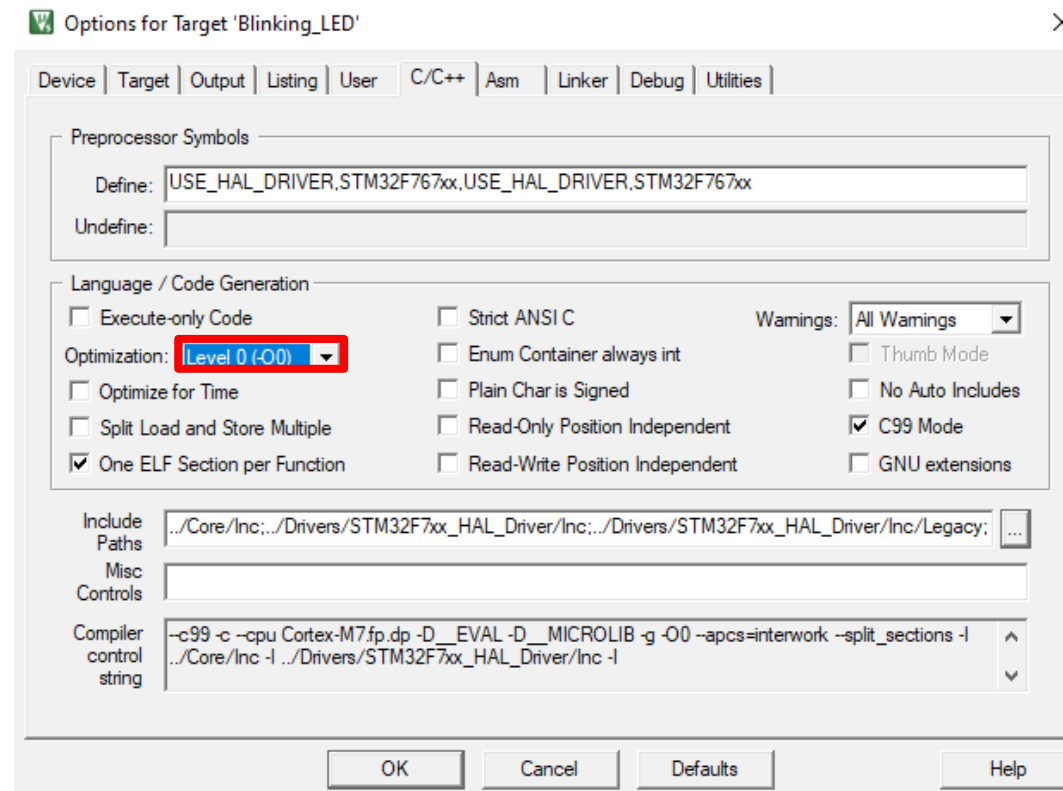
Under Outup tab, set Xtal to 216.

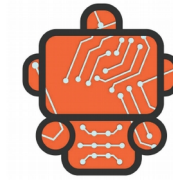




# Create the first Project

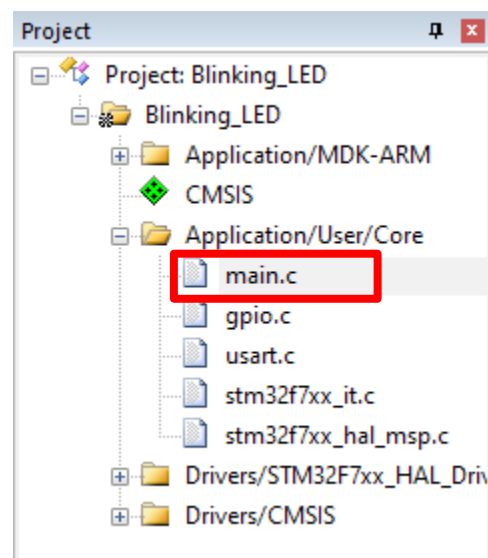
Under C/C++ tab, set the optimization level to 0.

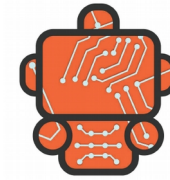




# Create the first Project

Open the main.c file

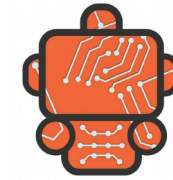




# Create the first Project

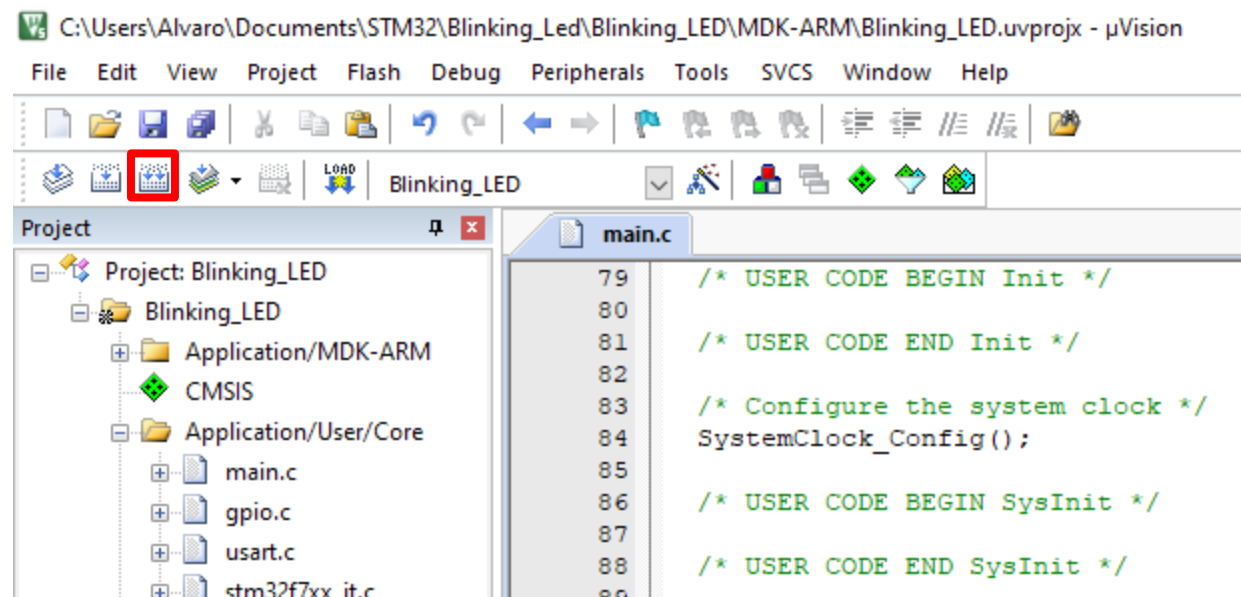
Add those two lines of code at lines 101 and 102.

```
97      /* Infinite loop */
98      /* USER CODE BEGIN WHILE */
99      while (1)
100     {
101         HAL_GPIO_TogglePin(GPIOB, GPIO_PIN_7);
102         HAL_Delay(1000);
103     /* USER CODE END WHILE */
```

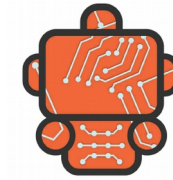


# Compile a program

To compile all target files (this is going to take a while)

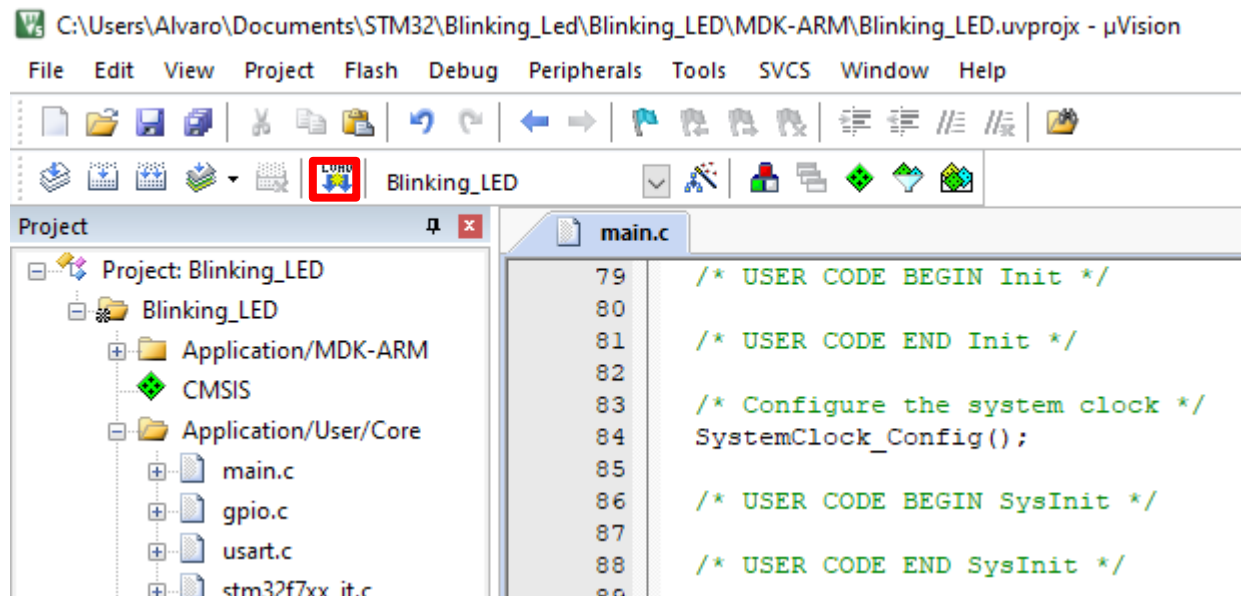






# Flash the board

Connect the development board to your pc through a micro USB cable (don't use the connection next to the ethernet port)

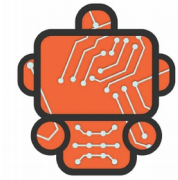


# Flash the board

---

When the flash is finished press the reset button on the board (black button)

The blue LED should blink at a frequency of 0,5Hz.

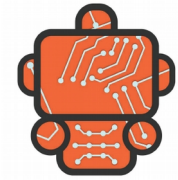


# Terminal

---

A good terminal is very convenient if you want work with serial port.  
You can use termite or this terminal:

<https://sites.google.com/site/terminalbpp/>



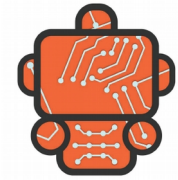
# File tree

---

Inside the project Directory, there are many sub directories and files, some of the most important are:

- **Drives:** where the CubeMX puts the drivers, you don't need to touch here
- **Inc:** where the user header files are, and where you should put all the header files you will create

(continues on the next page)



# File tree

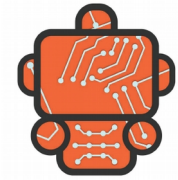
---

- **MDK-ARM:** where all the files for the Keil configurations and the output files of the compilation are, at this moment you only should touch the file with the extension **.uvprojx** (this file open the keil project)
- **Src:** where the user source files are, and where you should put all the source files you will create
- **.ioc:** STM32CubeMX file, contains all the Cube info for the project
- **.gitignore:** to specify intentionally untracked files to be ignore by git

# Resources

---

The following slides have some documentation that you should have with you every time you are working with the development board, in order to consult some information in case of doubts.



# Resources

---

STM32F76xxx and STM32F77xxx advanced Arm<sup>®</sup>-based

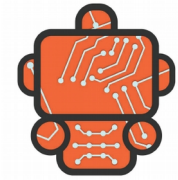
<https://bit.ly/2SrZbQS>

Description of STM32F7 HAL and Low-layer drivers

<https://bit.ly/2JsqPcE>

STM32CubeMX for STM32 configuration and initialization C code generation

<https://bit.ly/2OXkEmM>



# Resources

---

STM32F7 Series and STM32H7 Series Cortex® -M7 processor  
programming manual

<https://bit.ly/2qiOYJC>

STM32 Nucleo-144 boards

<https://bit.ly/2Df5i6S>

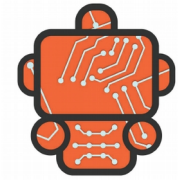
Getting started with STM32F7 Series MCU hardware development

<https://bit.ly/2StPQIj>

General-purpose timer cookbook

<https://bit.ly/2ITtSKB>





# Conclusion

---

Now you should have all tools installed and configured.

You should know how to create a project, compile and flash the board

Good work