ACTIVATE SECURE MEMORY

# Link

<https://www.youtube.com/watch?v=qCuVBD2dmTA&list=PLnMKNibPkDnFzux3PHKUEi14ftDn9Cbm7&index=13>

# Description

In this paperwork, we will configure the secure an application loaded in the memory, and then secure portion of code.

Contents

[Link 1](#_Toc89967869)

[Description 1](#_Toc89967870)

[Prerequisites 2](#_Toc89967871)

[STM32 Board 2](#_Toc89967872)

[ST-Link cable 2](#_Toc89967873)

[STM32CubeProgrammer 2](#_Toc89967874)

[STM32CubeMX 2](#_Toc89967875)

[STM32CubeIDE 2](#_Toc89967876)

[Walkthrough 3](#_Toc89967877)

[Step 1 : Launch STM32CubeMX and generate the code 3](#_Toc89967878)

[Step 2 : Create A blinky code 4](#_Toc89967879)

[Step 3 : Separate the memory 5](#_Toc89967880)

[Step 4 : Now create a bootloader code. 6](#_Toc89967881)

[Step 5 : Secure the memory 7](#_Toc89967882)

# Prerequisites

A picture containing chart

Description automatically generated

## STM32 Board

## ST-Link cable

## STM32CubeProgrammer

## STM32CubeMX

## STM32CubeIDE

# Walkthrough

## Step 1 : Launch STM32CubeMX and generate the code

Launch STM32CubeMX and select the right board depending on the one you are using. In my case I use the G071RB Nucleo board. Then you can generate the code of your project.  
Don’t forget to select the correct IDE (in my case STM32CubeIDE).  
Please make sure that the PC13 is set as a GPIO\_input because we want to use it as a push button.  
Generate to exact same projects.  
Graphical user interface, application

Description automatically generated

Step 2 : Create A blinky code  
To do this blinky code, we will set the memory vector in our code. (It is not the same as in the video because I guess that now those functions take more memory and can’t compile).  
This will put our code in this section of memory.  
Graphical user interface, text, application, email

Description automatically generated

Step 3 : Separate the memory  
Now in the FLASH.id you have to create the new section of memory. This one will permit us to put the unsecure code only in FLASH\_unsecure memory.

Text

Description automatically generated

When you have finished you can compile the code and execute the debugger.  
You will see that the light is blinking.

Step 4 : Now create a bootloader code.  
The following code will load the code located at 0x800E00 (our blinky application) when the user button is pressed. Graphical user interface, application

Description automatically generated  
  
Graphical user interface, text, application, email

Description automatically generated  
When you have finished you can compile the code and execute the debugger. When you will push the button, the led will start to blink.

Step 5 : Secure the memory  
To secure the memory, open CubeProgrammer, go to OB and in the FLASH Security set the address of the memory location of your blinky application as the unsecure part.  
You couldn’t read at memory location 8x8000000 but you could see the code at 0x800E000.  
Graphical user interface, application

Description automatically generated