ACTIVATE SECURE MEMORY

Link

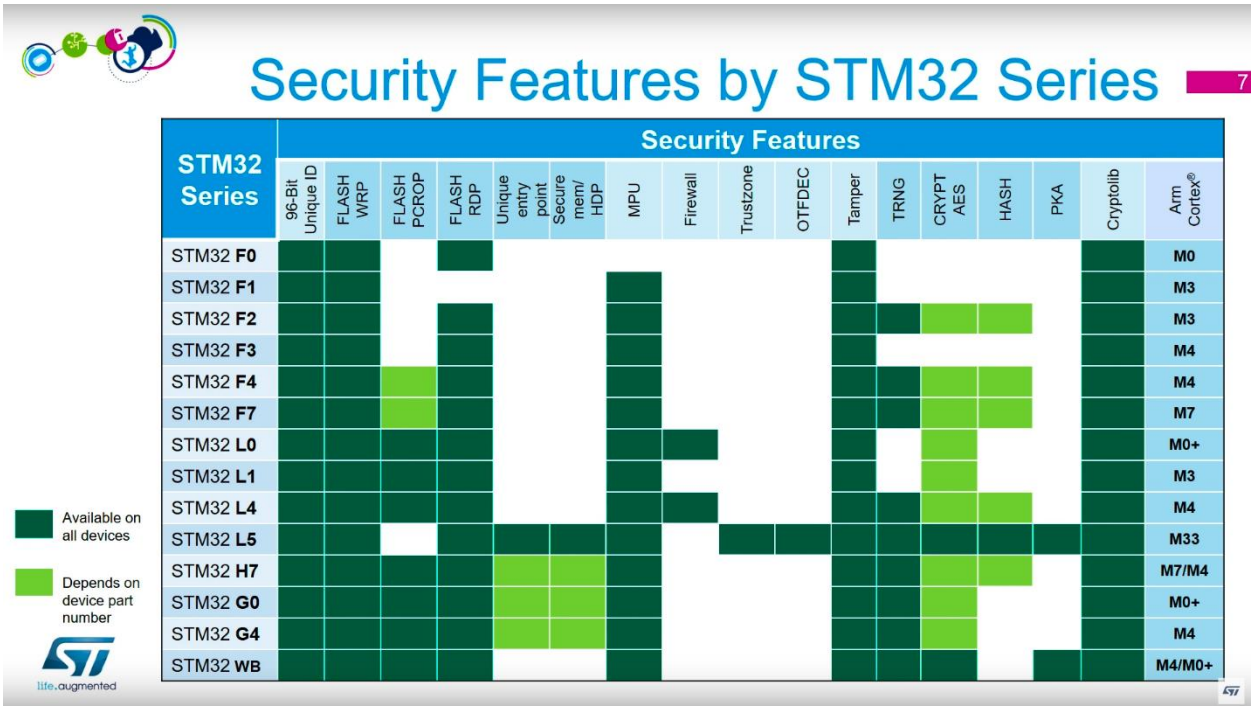
https://www.youtube.com/watch?v=qCuVBD2dmTA&list=PLnMKNibPkDnFzux3PHKUEi14ftDn9Cbm7&index=12

Description

In this paperwork, we will configure the secure memory, and then activate it to see how to protect some portions of code.

Contents

Prerequisites



STM32 Board

ST-Link cable

STM32CubeProgrammer

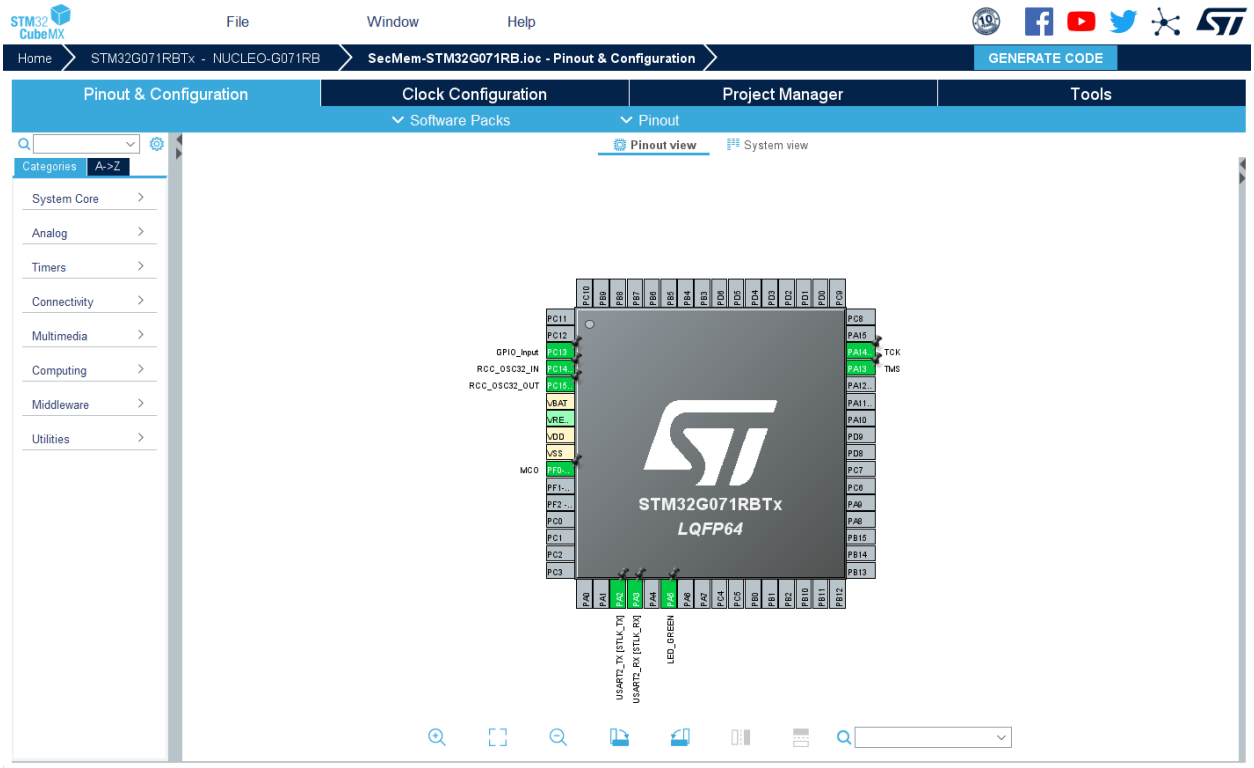
STM32CubeMX

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.



Step 2 : Create The test code

Basically we call the unsecure_code function in the main.

This function permits to call the secure_toggle_led function and to enter in the secure memory protection if the button is pressed.

The section attribute will link our part of memory to our function.

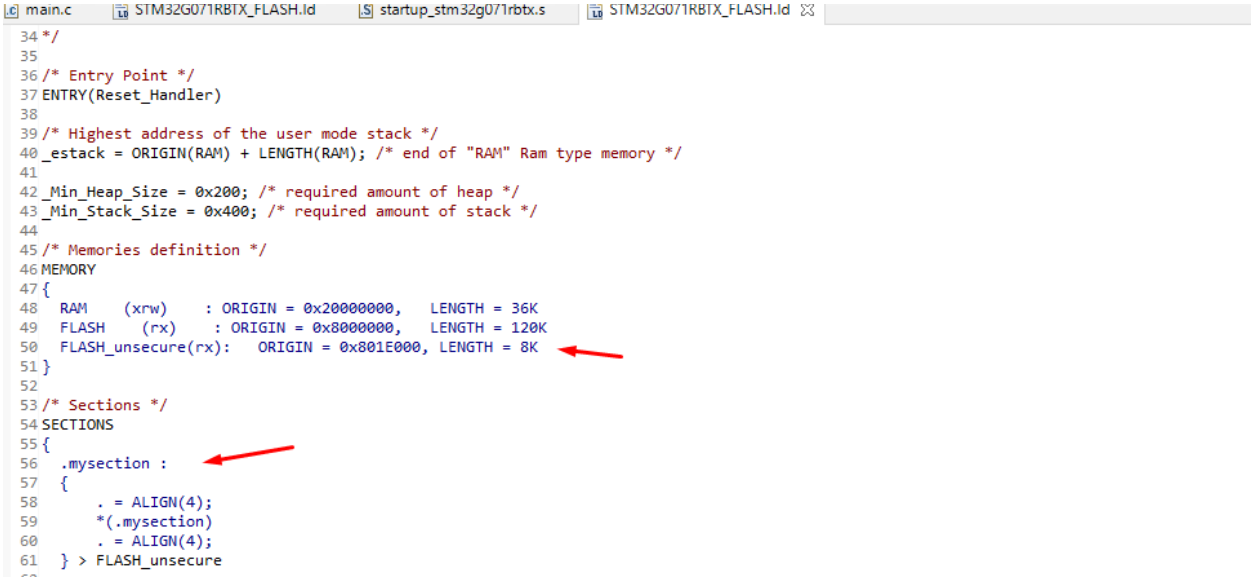
main.c | STM32G071RBTX_FLASH.ld | startup_stm32g071rbtx.s

```
/* Private user code ---------------------------------------------------------*/
/* USER CODE BEGIN 0 */
void secure_toggle_led(void)
{
    HAL_GPIO_TogglePin(GPIOA, GPIO_PIN_5);
}
void __attribute__ (( __section__ (".mysection"))) unsecure_code(void)
{
    while(1)
    {
        secure_toggle_led();
        for(uint32_t i=0;i<1000000;i++);
        if (HAL_GPIO_ReadPin(GPIOC, GPIO_PIN_13) == 0)
        {
            HAL_FLASH_Unlock();
            HAL_FLASHEx_EnableSecMemProtection(1);
            HAL_FLASH_Lock();
        }
    }
}
/* USER CODE END 0 */

/**
  * @brief  The application entry point.
  * @retval int
  */
int main(void)
{
  /* USER CODE BEGIN 1 */

  /* USER CODE END 1 */

  /* MCU Configuration--------------------------------------------------------*/

  /* Reset of all peripherals, Initializes the Flash interface and the Systick. */
  HAL_Init();

  /* USER CODE BEGIN Init */

  /* USER CODE END Init */

  /* Configure the system clock */
  SystemClock_Config();

  /* USER CODE BEGIN SysInit */

  /* USER CODE END SysInit */

  /* Initialize all configured peripherals */
  MX_GPIO_Init();
  MX_USART2_UART_Init();
  /* USER CODE BEGIN 2 */
  unsecure_code();
  /* USER CODE END 2 */

```

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.

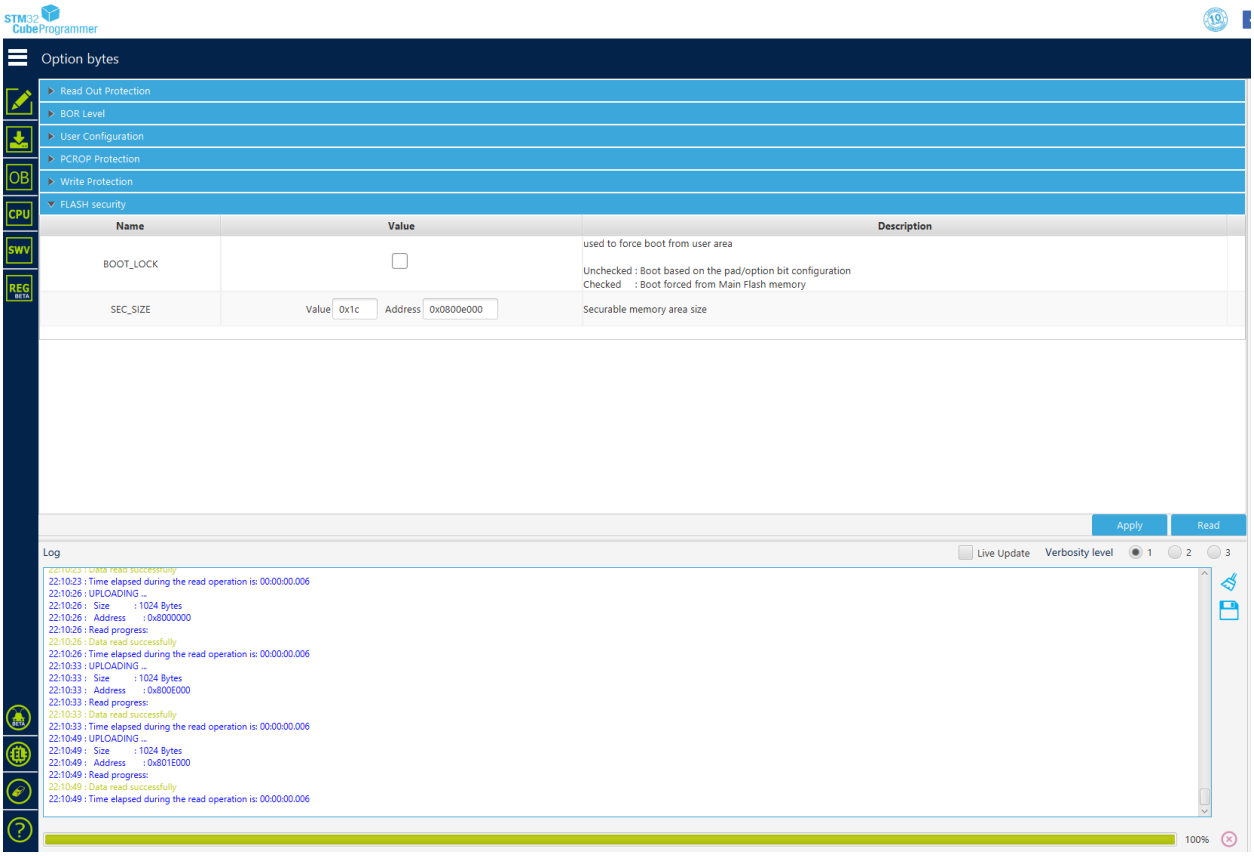
To do so we add a section .mysection that corresponds to the FLASH_unsecure we created.



When you have finished you can compile the code and execute the debugger.

Step 4 : Activate the secure memory

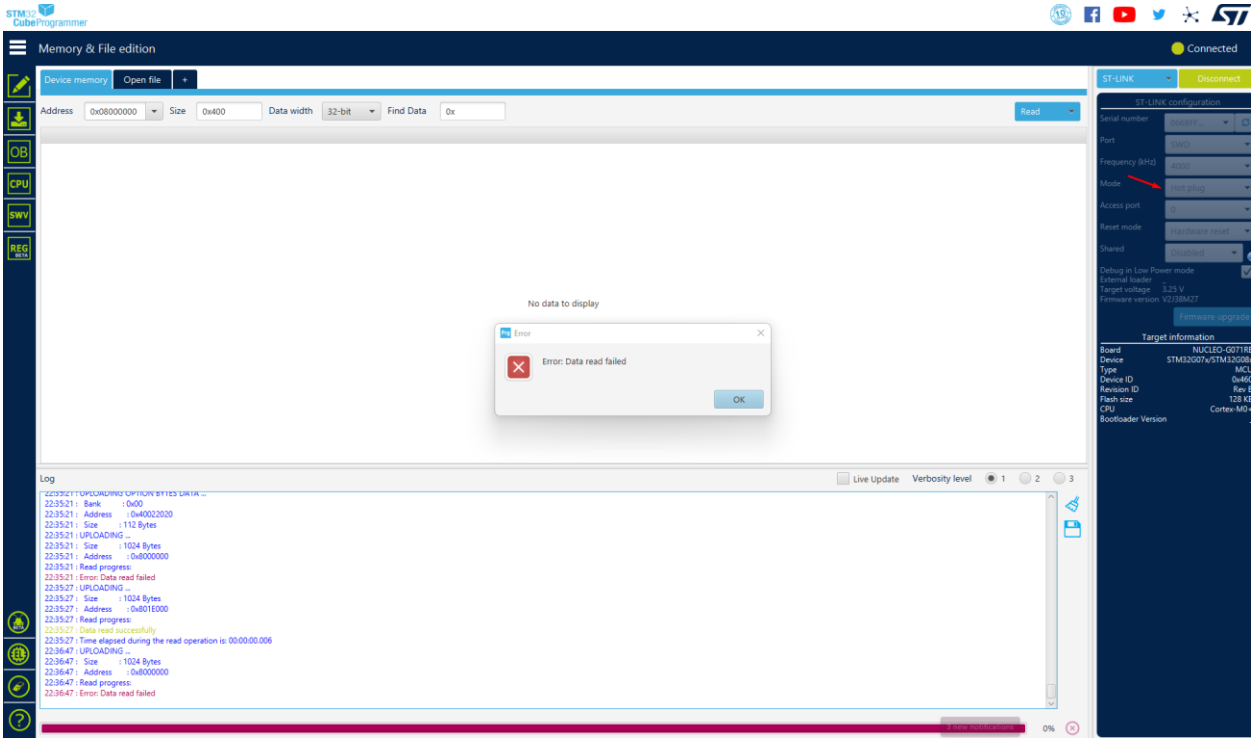
Launch the STM32Programmer and go to OB and then FLASH security.
In the SEC_SIZE field, put as a value 0x1c.



Apply the modifications and reset the board.

Step 5 : Check the secure memory

During this step, you have to disconnect and reconnect but with HOTPLUG mode.
Once it is realized you should not be able to see 0x08000000.



But at the same time you should be able to see the 0x0801E000 memory section that corresponds to the unsecured function.