|  |
| --- |
| Starting with  STM32CubeIDE |
| [Dmytro Lyzun]  [GitHub: <https://github.com/Dima1381>] |

[Starting work in STM32CubeIDE, writing a first program with target: STM32F103C6T6]

[Last Updated: 27.02.2025]

This tutorial shows how to use the STM32CubeMX tool to initialize the peripherals, build and generate your starting projects with the initialization C code using HAL libraries.

Here I will go through the all process from launching the IDE to building the program on target. In this section I am going to develop a program called "A flashing led code". The idea is that the led on the target will display some code by blinking a certain number of times at intervals.

You can find the result in the same directory on github (file header)

Prerequisites:

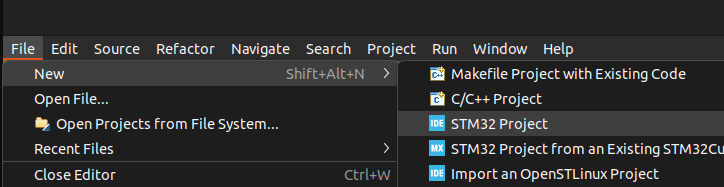
* You should already have installed STM32CubeIDE.

Hardware:

* STM32F103x (in my case STM32F103C6T6);
* Jumper wires: M-M;
* ST-LINK for STM32 (in my case ST LINK V2);

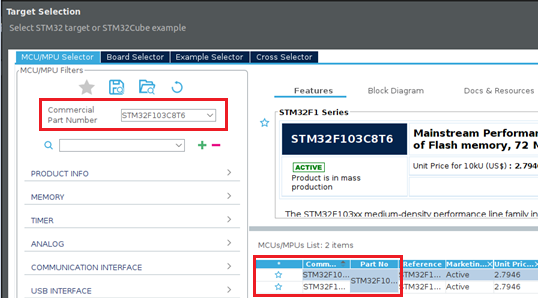
**Create new STM32Project in CubeIDE**

Click File->New->STM32Project



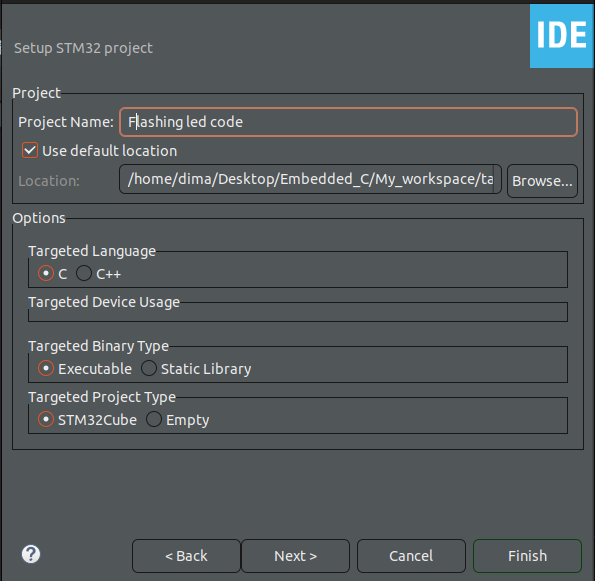
Picture 1. Create new Project.

Find and select your target:



Picture 2. Selected target.

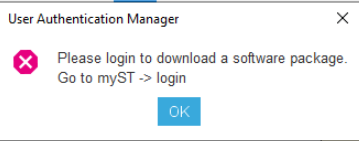
Write name project and click “Finish”:



Picture 3. Setup Project.

**Attention:** If you get the following error, just log in.

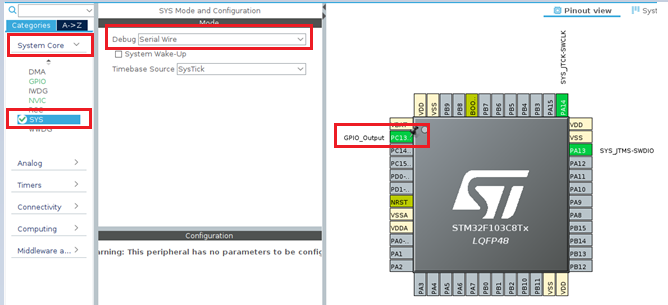
Click Help -> STM32Cube updates -> Connection to myST:



Picture 4. Error with login

After that, let's move on to some settings:

1. Click System Core -> SYS -> Debug: Serial Wire.
2. Setup Pinout, for out led blink we will use Pin 13. Click on PC13 -> GPIO\_OutPut.
3. Click “Save” and after STM32Cude create project and generate some code.



Picture 5. Setting project.

We will working in next file :

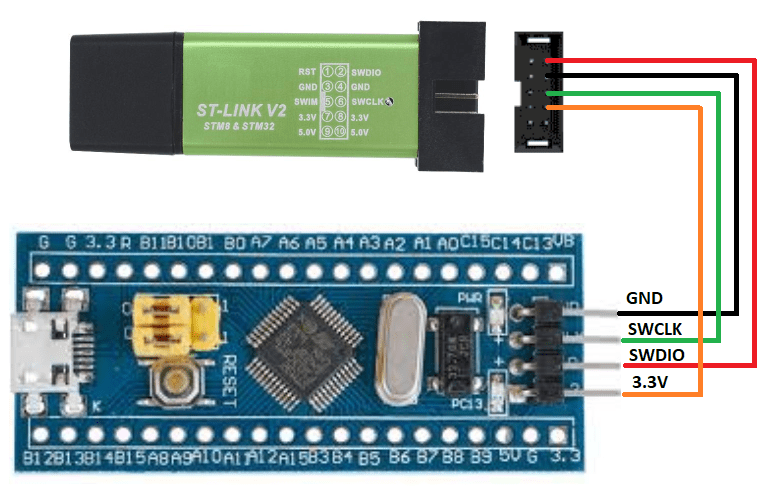
* Core-> Src -> main.c.

There you can find an infinite while loop – this is our work area.

Before write code – we should setup hardware.

**Connection Hardware**

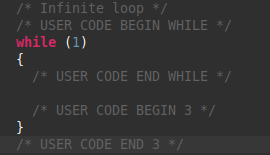
To connect the microcontroller and the programmer i used this circuit scheme:



Picture 6. Connect the microcontroller and the programmer.

After that, we connect ST-LINK to our host (PC).

We return to our workspace (while loop) in STM32Cube.

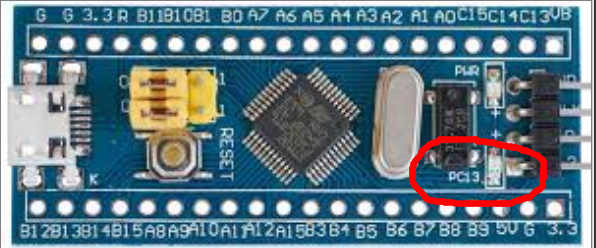


Picture 7. Main work area.

**Writing code**

[Idea]:

There is a free LED on the board, under the port number PC13 (photo below). The idea is that this LED should reproduce certain signals. The program will generate an int array of 5 values ​​in the range (0-10). Please note that the code can only contain numbers from 0 to 9, so the **number 10 will be read as 0**. Led PC13 - will blink each value from the array. That is, the array will store a certain code, for example: [1, 3, 10, 7, 1]. Then Led PC13 will blink first 1 time, then 3, 10, 7, 1. We will pause between each value (1-2 seconds).



Picture 8. The LED (PC13) we are working with.

Let's create a function that will take an array and a size, then loop through the array and reproduce each value (Reproduce means turning the LED on and off as many times as there are values ​​in the array). You need to declare the function, and place the function itself after the main function:

|  |
| --- |
| **void** **blink\_code**(**int** \*array, **int** size) {  // Let's turn off the LED for 4 seconds to simulate operation.  **HAL\_GPIO\_WritePin**(GPIOC, GPIO\_PIN\_13, *GPIO\_PIN\_SET*);  **HAL\_Delay**(4000); // Delay function in milliseconds    **for**(**int** i = 0; i < size; i++) {  **for**(**int** j = 0; j < array[i]; j++) {  // HAL\_GPIO\_WritePin(<TypeOfPort>, <Number\_of\_Port>, value (0-1));  **HAL\_GPIO\_WritePin**(GPIOC, GPIO\_PIN\_13, *GPIO\_PIN\_RESET*);// set 0 - Turn off the LED  **HAL\_Delay**(500);  **HAL\_GPIO\_WritePin**(GPIOC, GPIO\_PIN\_13, *GPIO\_PIN\_SET*);// set 1 - Turn on the LED  **HAL\_Delay**(500);  }  **HAL\_Delay**(1500); // Deley after each value in array  }  } |

Next, we move to the infinite loop and implement the main logic there.

|  |
| --- |
| **srand**(**time**(NULL));  **while** (1)  {  **int** array[SIZE];  **for**(**int** i = 0; i < SIZE; i++) { // Filling an array with random values  **int** rvalue = **rand**() % 11; // rvalue in the range 0 to 10  array[i] = rvalue;  }  **blink\_code**(array, SIZE);  } |

Declarations before the main function:

|  |
| --- |
| **#define** **SIZE** 5  **void** **blink\_code**(**int**\*, **int**); |

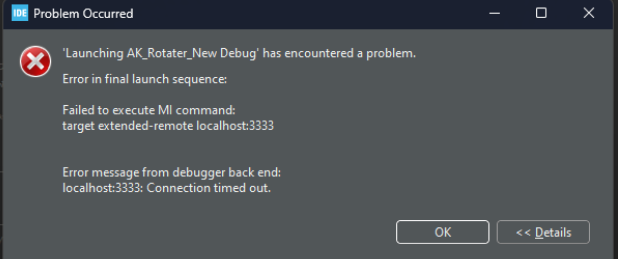
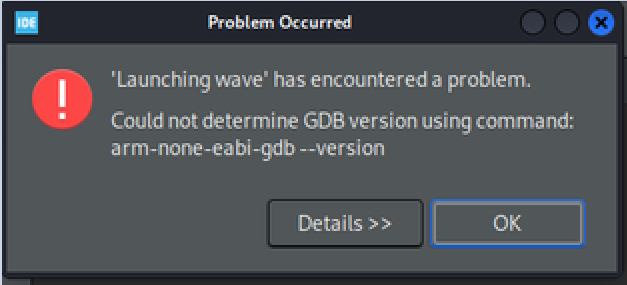
Include next library:

|  |
| --- |
| **#include** <stdlib.h>  **#include** <time.h> |

**Build Project**

Attantion:

If this is your first time creating a project in STM32CubeIDE.

When trying to compile the project, you may get the following errors:

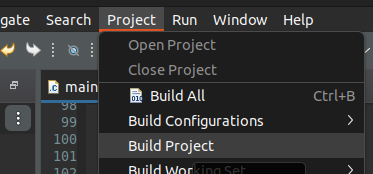
Picture 9. Some errors you may get.

Working with ARM controllers, which uses arm-none-eabi-gdb. This is dependent on libncurses.so.5

To resolve this issue, run the following command in the terminal:

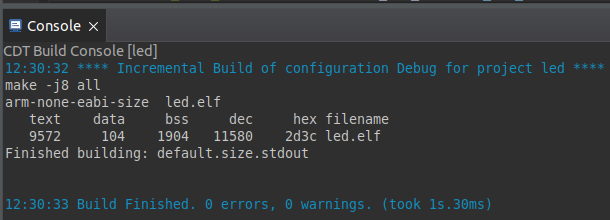
* sudo apt-get install libncurses5

Next, we will compile the project, click Project -> Build Project.



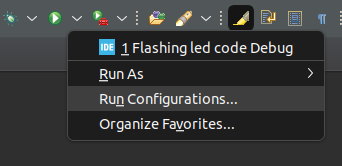
Picture 10. Building Project.

Make sure, that there no one error after building:



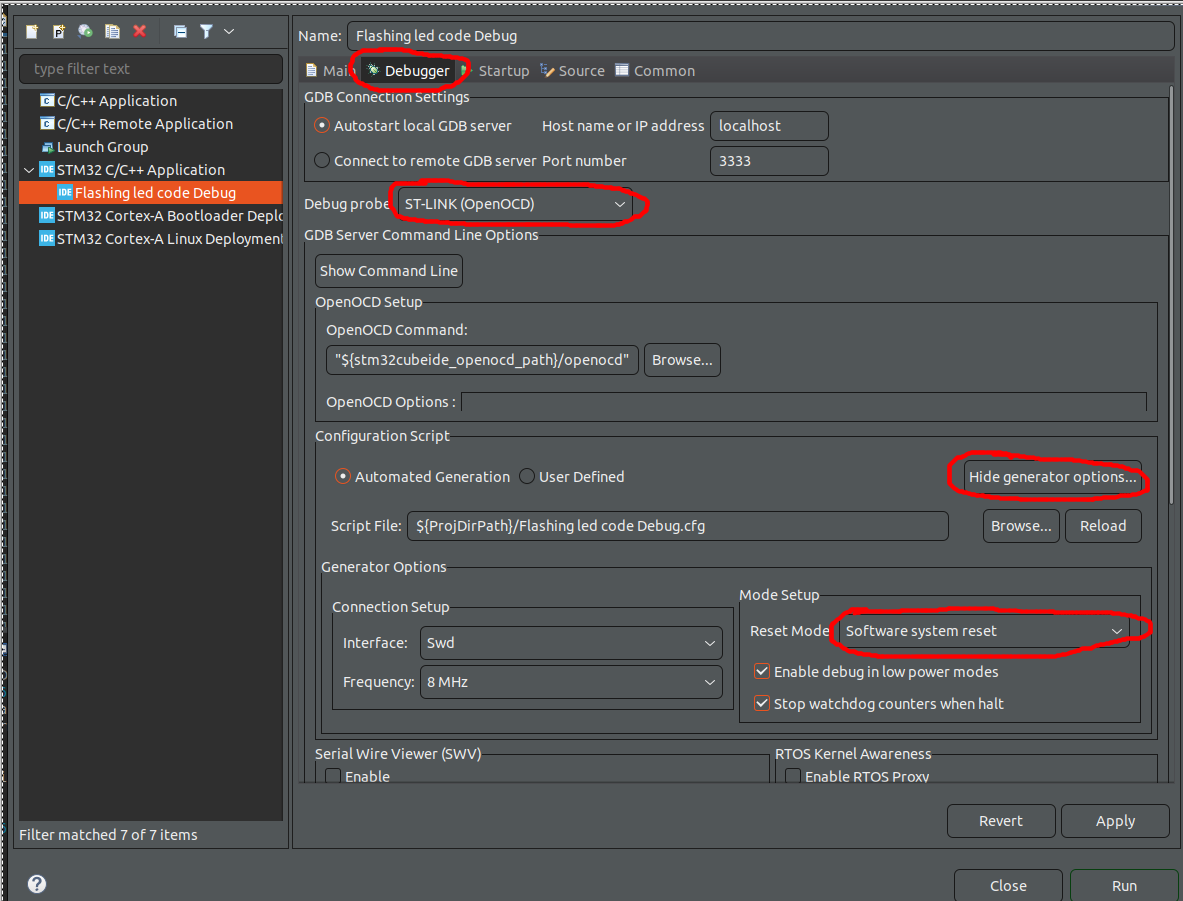
Picture 11. Successful project compilation.

After building, click Run -> Run Configuration…



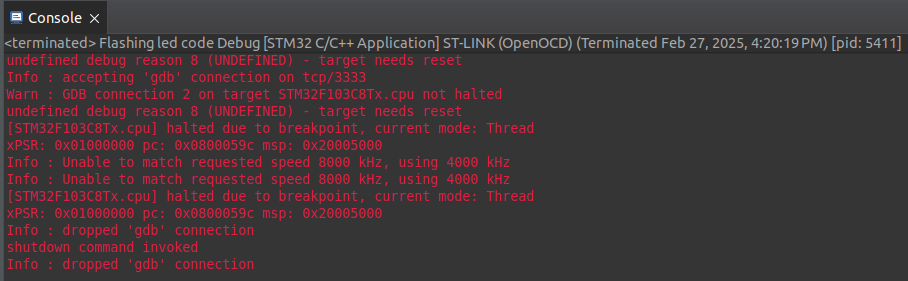
Picture 12. Run Configuration.

Set the following settings and click “Run”:



Picture 13. Create, manage, and run configuration.

Execution process:



Picture 14. Successful Execution process.

After that, the program is compiled and sent to the microcontroller.

You can find the result and the program code on GitHub