Getting started with STM32

Hello and welcome to this presentation and steps of how to get started with stm32. It is mainly for beginners which covers software, packages and driver installation.

STM32 have wide range of boards and development kit available which allows you to develop firmware as per your application.

There are various toolchains which allows you to develop your firmware

- IAR Embedded Workbench® for ARM® (EWARM) by IAR systems®
- Microcontroller development kit for ARM® (MDK-ARM) by Keil®
- TrueSTUDIO® by Atollic®
- System Workbench for STM32 (SW4STM32) by AC6

Most widely used toolchains are KEIL (32kB code limitation) and System Workbench (freeware with no code size limitation).

1) Hardware Requirements
- Choose any STM32 board which you would like to explore and start with.
- Programmer ST-LINK V2 for STM8 and STM32.

2) Software Requirements
- Download the preferred Integrated Development Environment (IDE).
- Download STM32CubeMX which allows you to configure pins as per your need.
- Complete video tutorial for downloading KEIL and SW4STM32 have been made for you.

You can watch the video here:

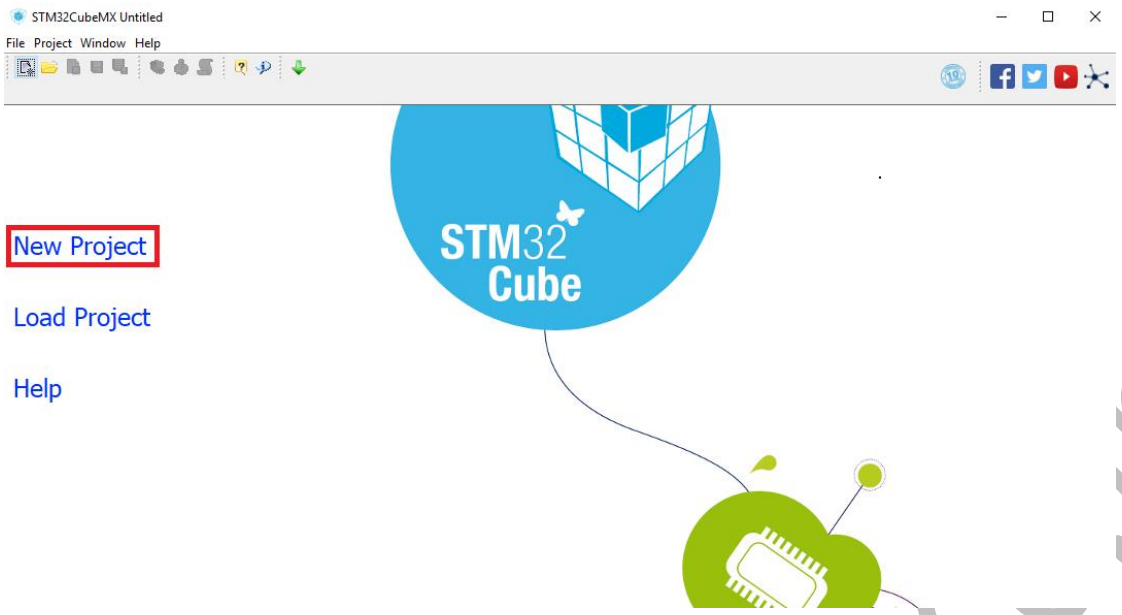
Since we are working with small snippets of code, we can use KEIL IDE because it provide some amazing debugging features into IDE itself.

The board we are using is STM32F103C8T6 and the packages of it are installed as shown in the installation video.

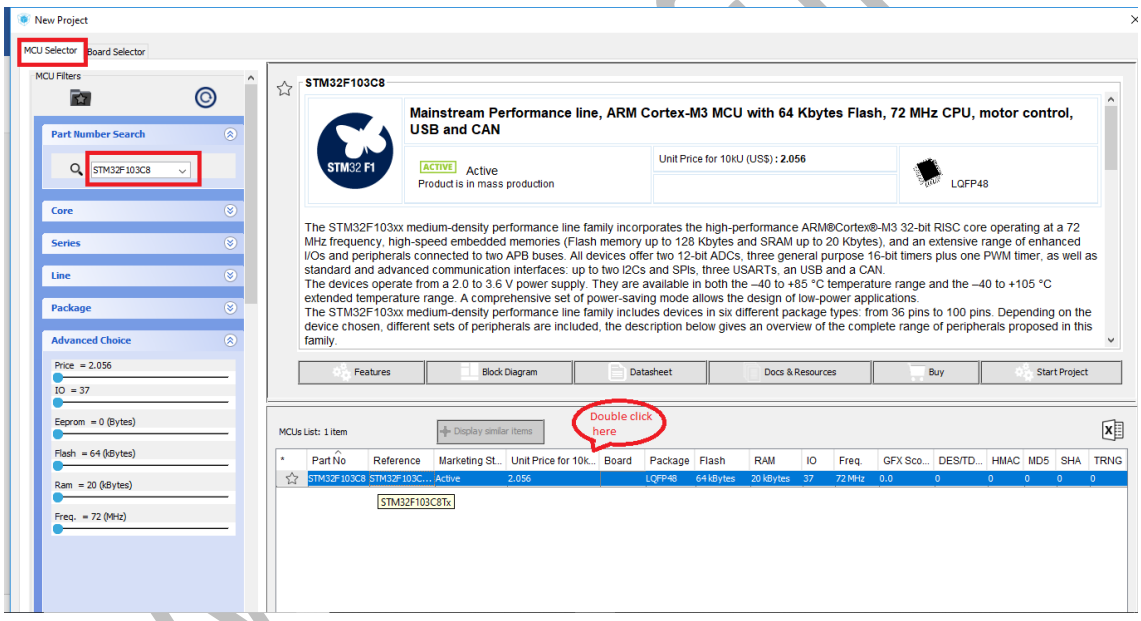
Now we will see how we can start programming the board using our IDE and blink a LED

Steps are as follows >>

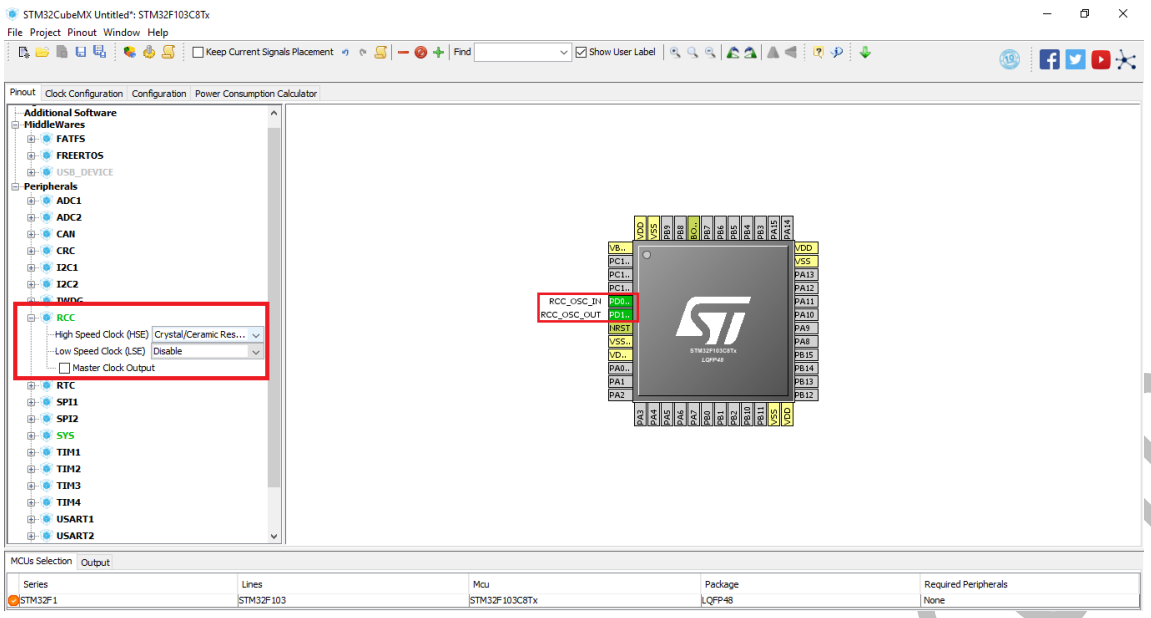
- Start with STM32CubeMX and Click on New Project.



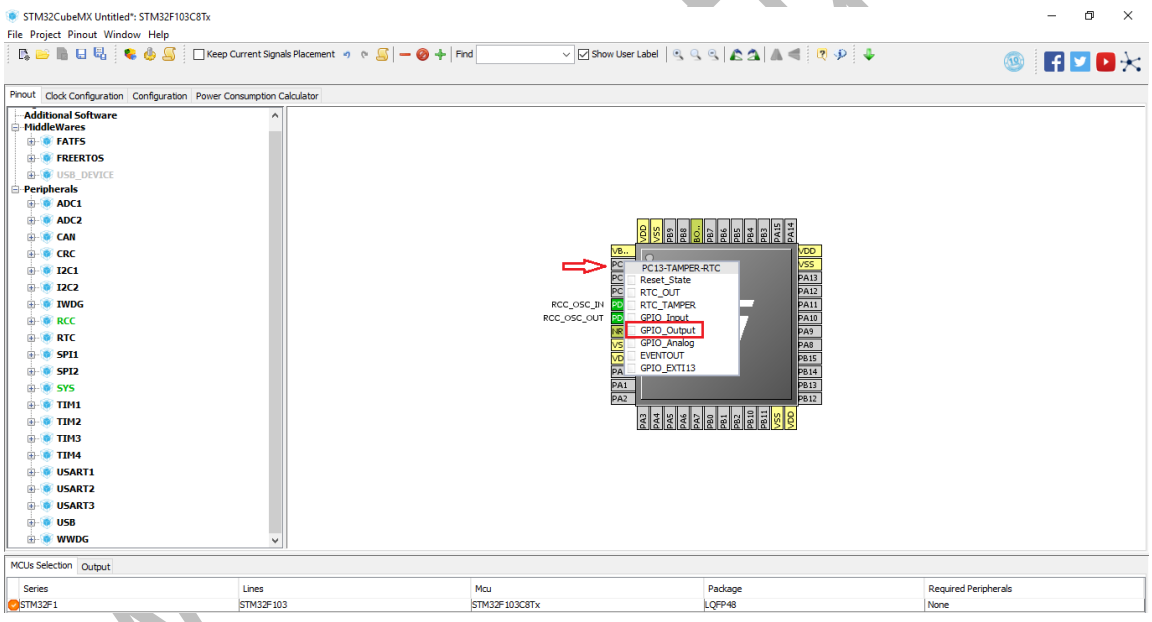
- Choose the board or development kit which you are using and double click on the board and the software will take you to pin configuration.

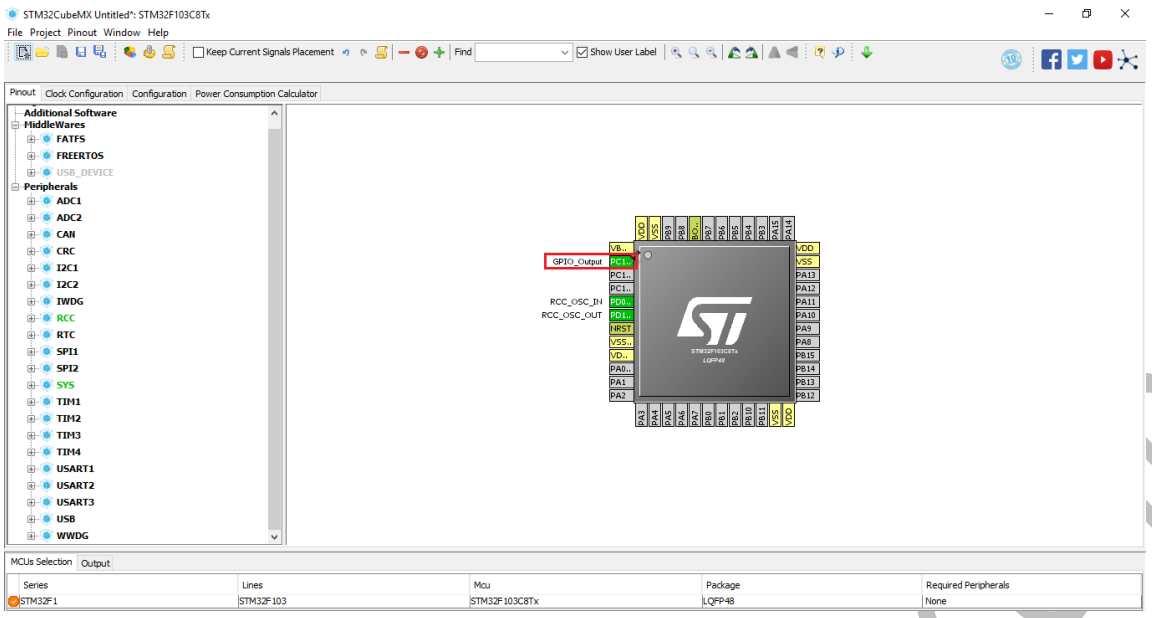


- Initialize clock for your system by clicking on RCC > High Speed Clock (HSE) > Crystal/ Ceramic Resonator. It should highlight the corresponding pins when you initialize the clock.

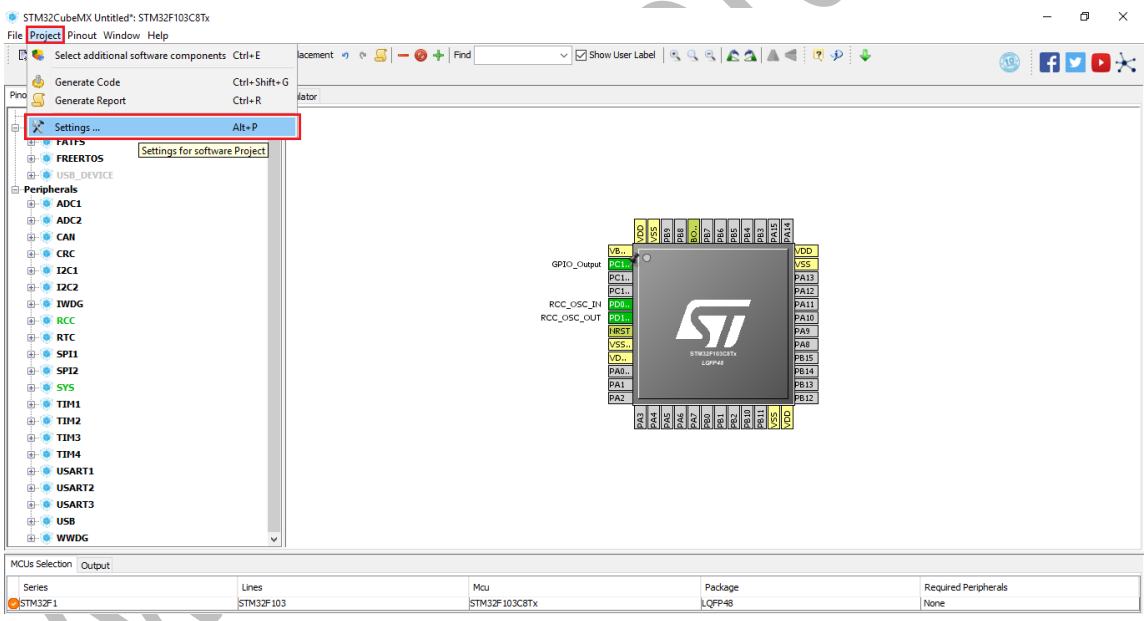


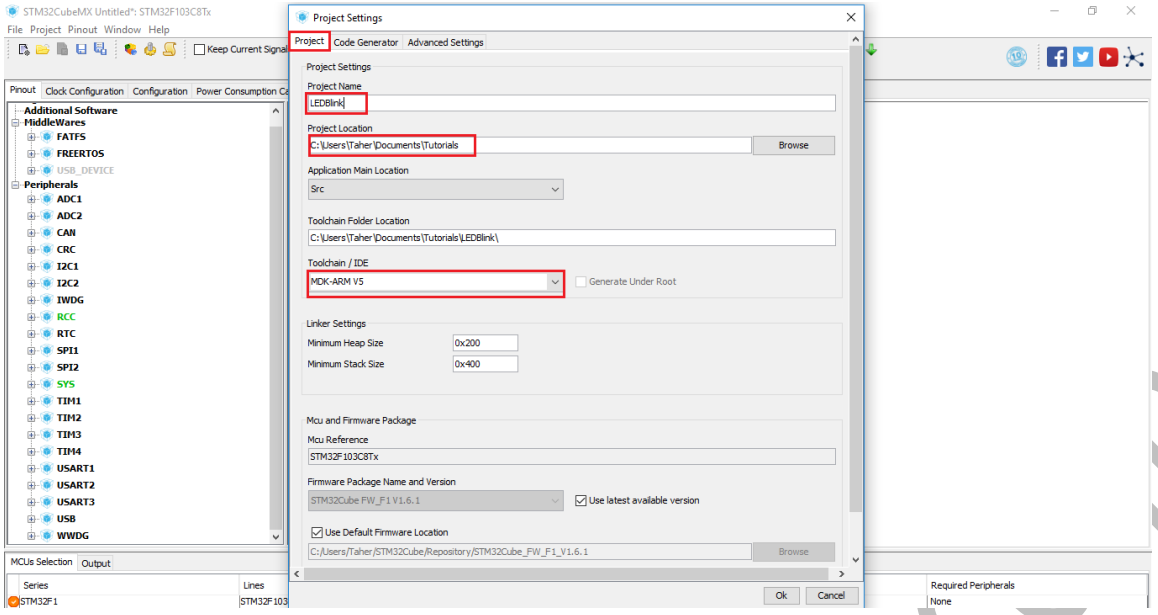
- STM32F103C8T6 (Blue Pill Board) have inbuilt LED on PC13. If you are using any other board or kit, you can connect an external LED followed by a current limiting resister for Led protection. We initialize the PC13 pin as GPIO output. Pin initialization is completed.

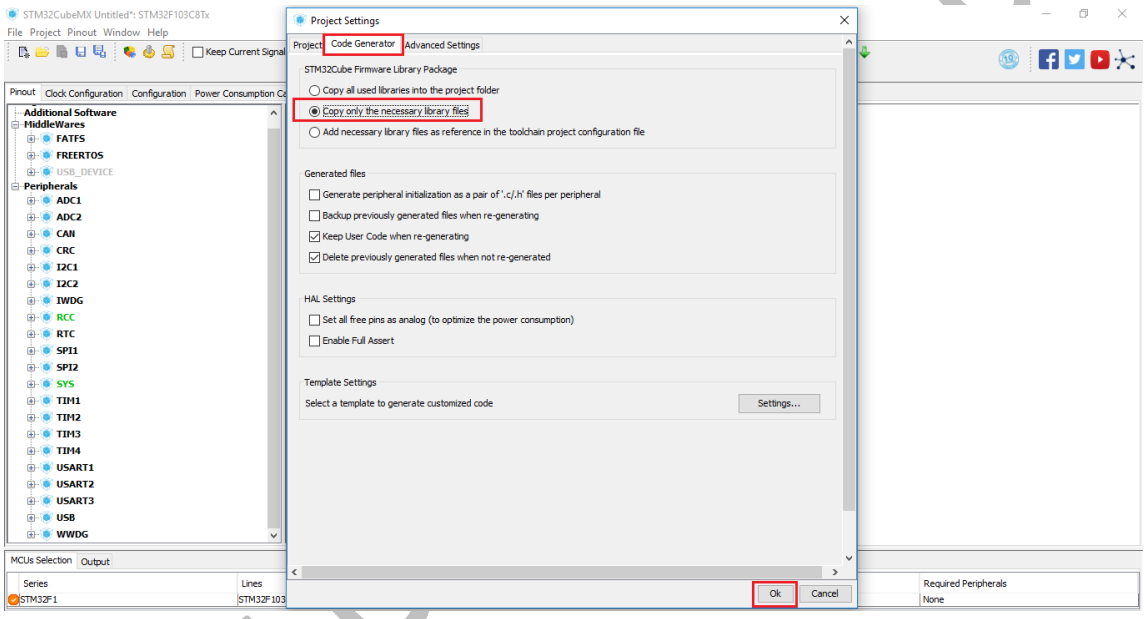




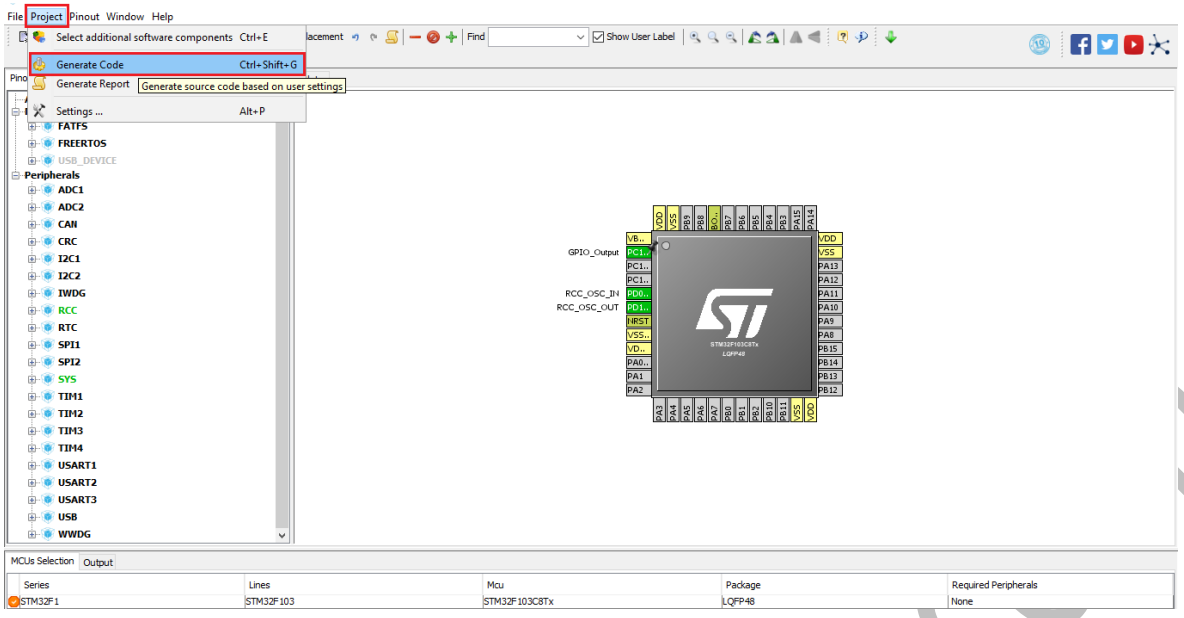
- Now we select Project > Settings > Give project name > dedicate a folder for it > Select the toolchain option > select MDK-ARM V5 > Go to Code Generate > select Copy only the necessary library files > click ok



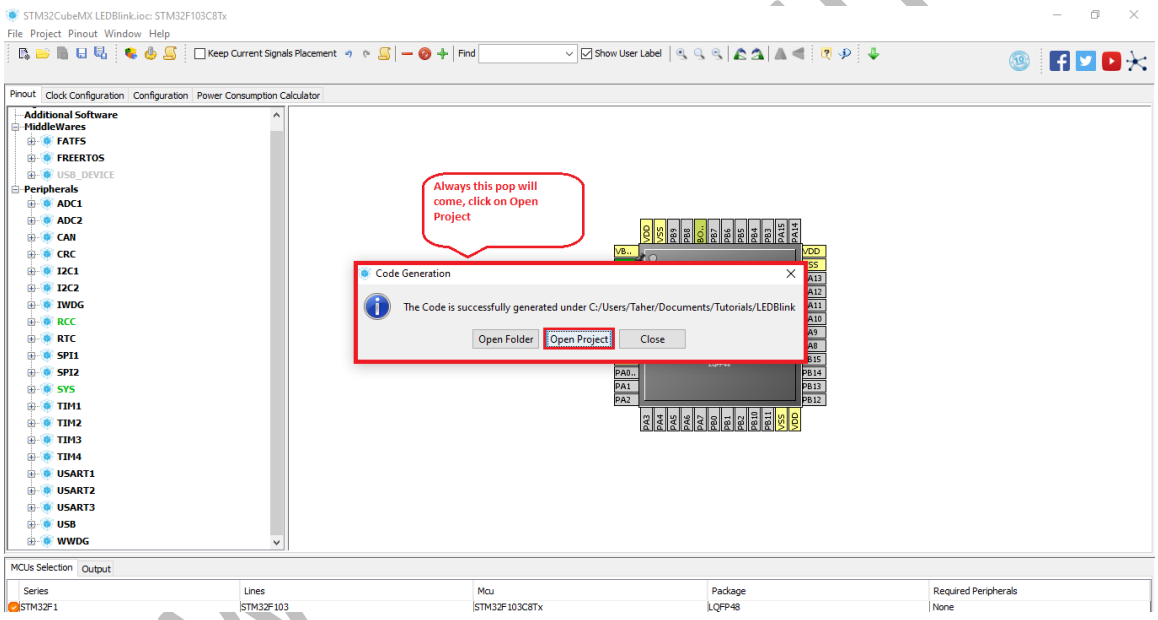




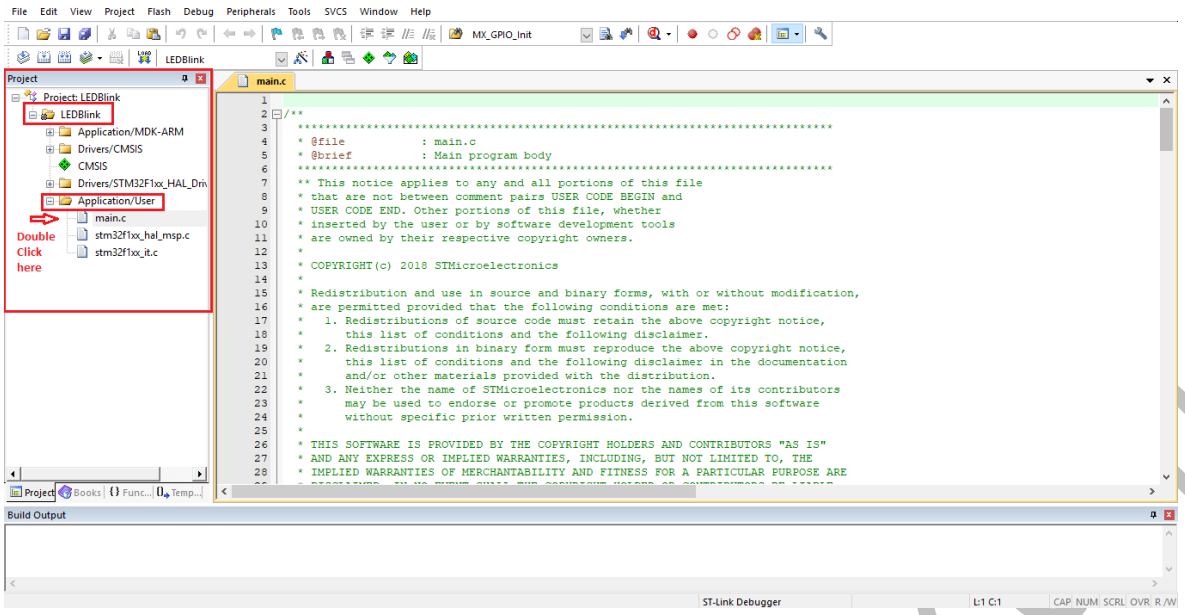
- Now select Project > click on Generate Code.



- After code generation is done, it will pop Code Generation click > Open Project.



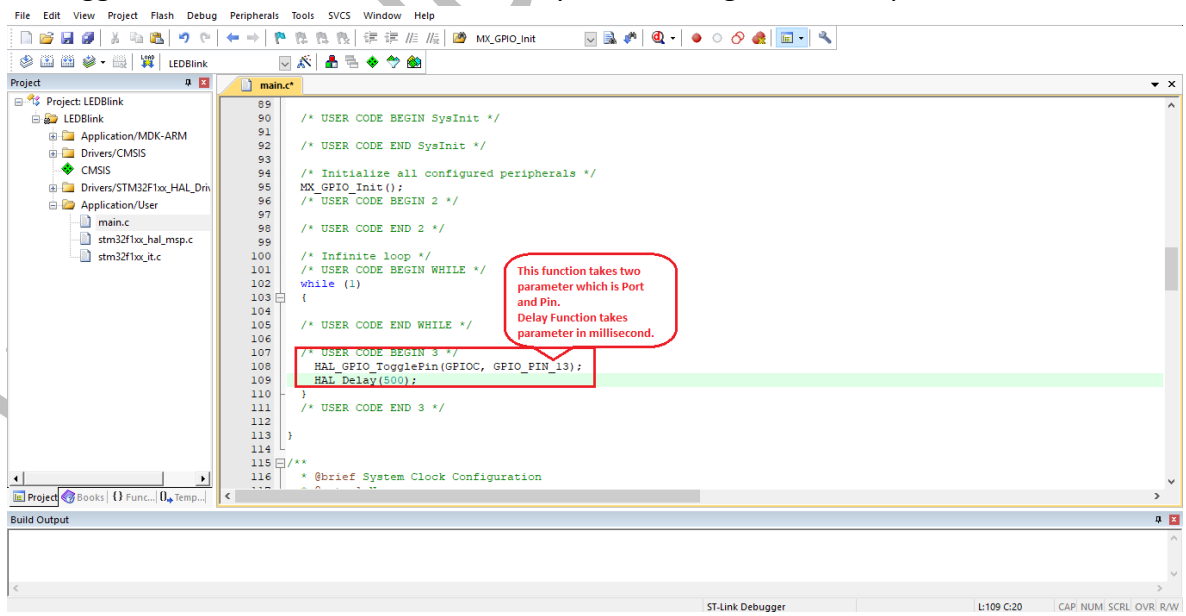
- This tab will take you to your IDE or toolchain which you have chosen. On this project section > click on + sign > Application/User > main.c. This main.c is the file which have all the firmware you code or develop. Just double click it and the file will open.



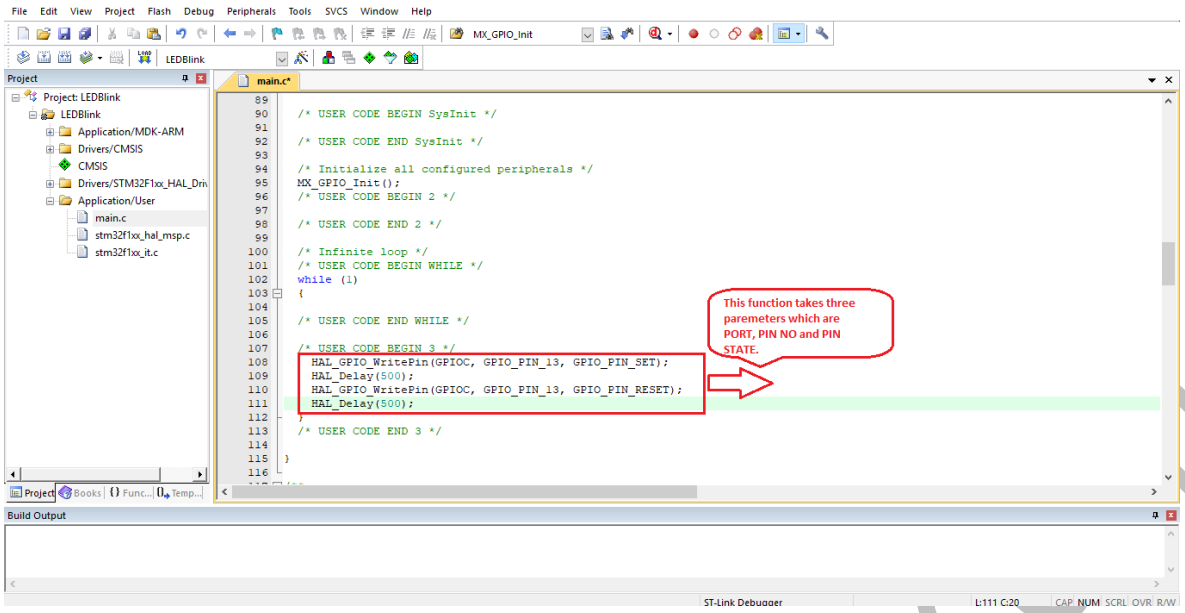
- Scroll you main.c file and you will see header includes, function definition, int main loop (where GPIO, CLOCK and other peripherals initialization are done), while loop (where you code logic which you want to run infinity times), at the end of loop the HAL(Hardware Abstract Layer) initialization are done.

- Now we need to toggle our on-board led which is on PC13.

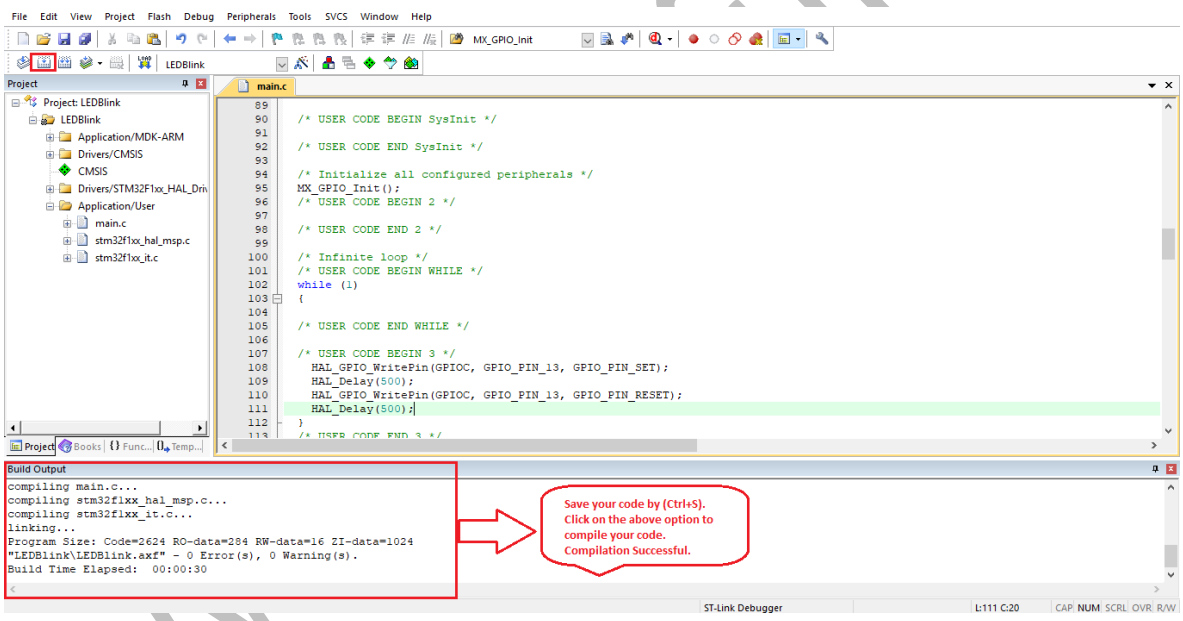
- This function takes two parameter which is Port and Pin. Delay takes parameter in milliseconds. So we toggle our led with 0.5 seconds of delay between high and low operation.



- Other way of toggling a led is using write function which takes three parameters that are Port, Pin and current state.



- Once you have done with your code, save it and compile it. If your code have any error it will be displayed in output space. If your code doesn't have any error, it will be compiled successfully.



- Now connect ST-LINK V2 with your board in the configuration mentioned in the videos and flash you code and check out the output.