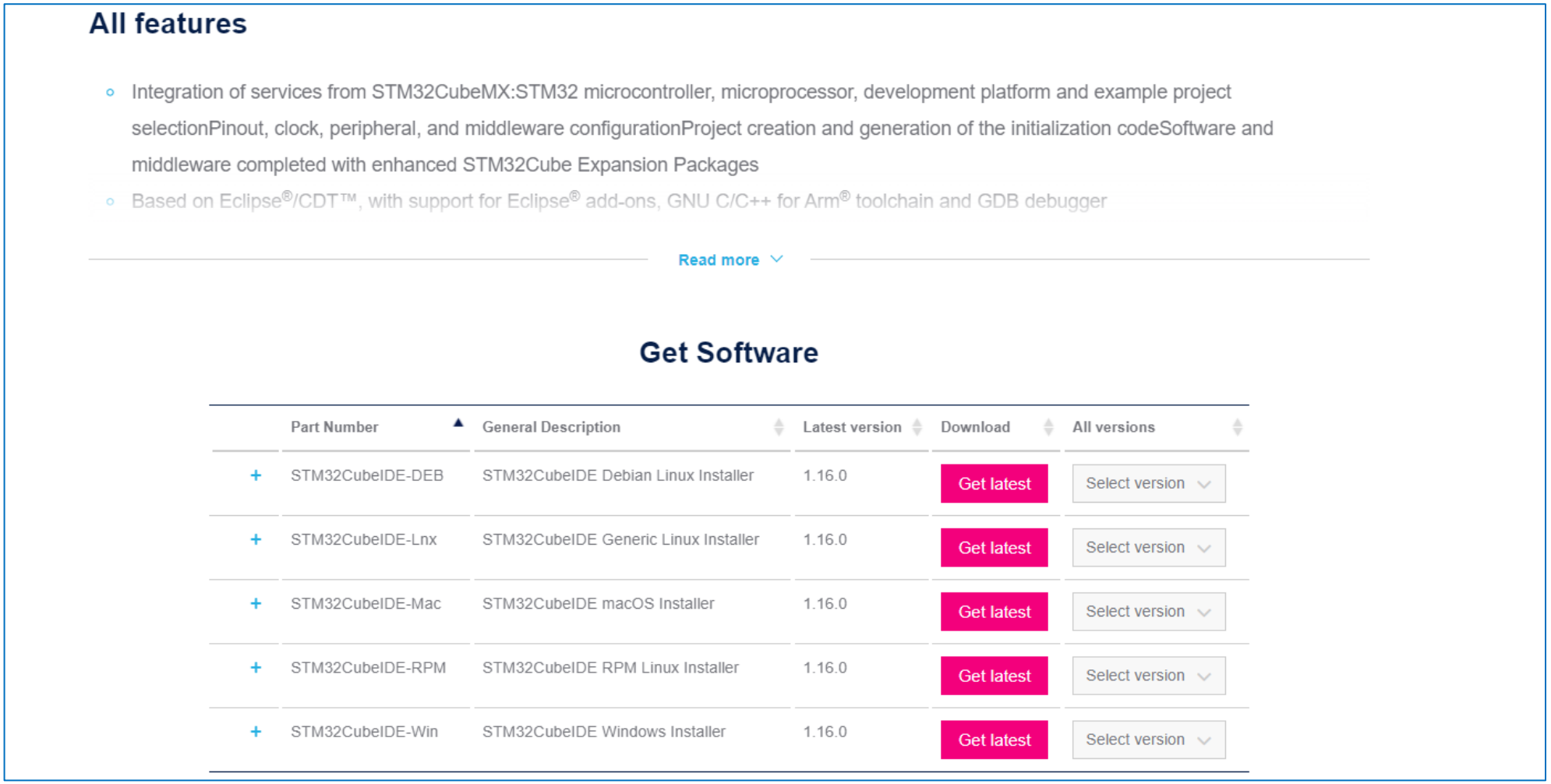
# Getting started with 7Semi DV-10111 (STM32G030F6P6)

In this guide we will be using STM32 cube IDE to program this board with couple of example codes.

Our first step will be to download the STM32 cube IDE software. Please follow the steps provided below.

1. Click on the link given to download the software:- STM32 cube IDE software
2. Scroll down the page until you see as the image shown below



## All features

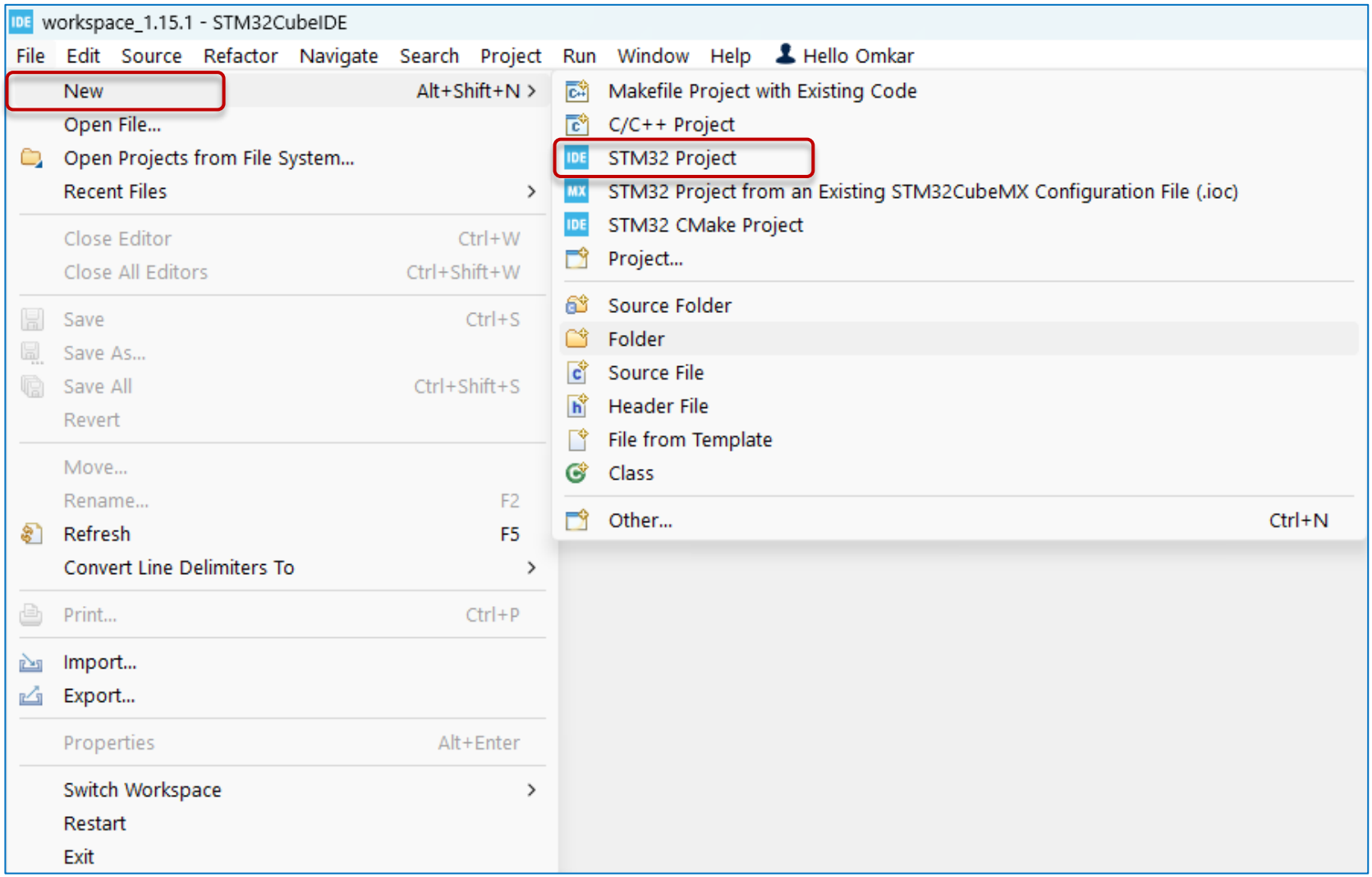
- Integration of services from STM32CubeMX:STM32 microcontroller, microprocessor, development platform and example project selectionPinout, clock, peripheral, and middleware configurationProject creation and generation of the initialization codeSoftware and middleware completed with enhanced STM32Cube Expansion Packages
- Based on Eclipse®/CDT™, with support for Eclipse® add-ons, GNU C/C++ for Arm® toolchain and GDB debugger

Read more

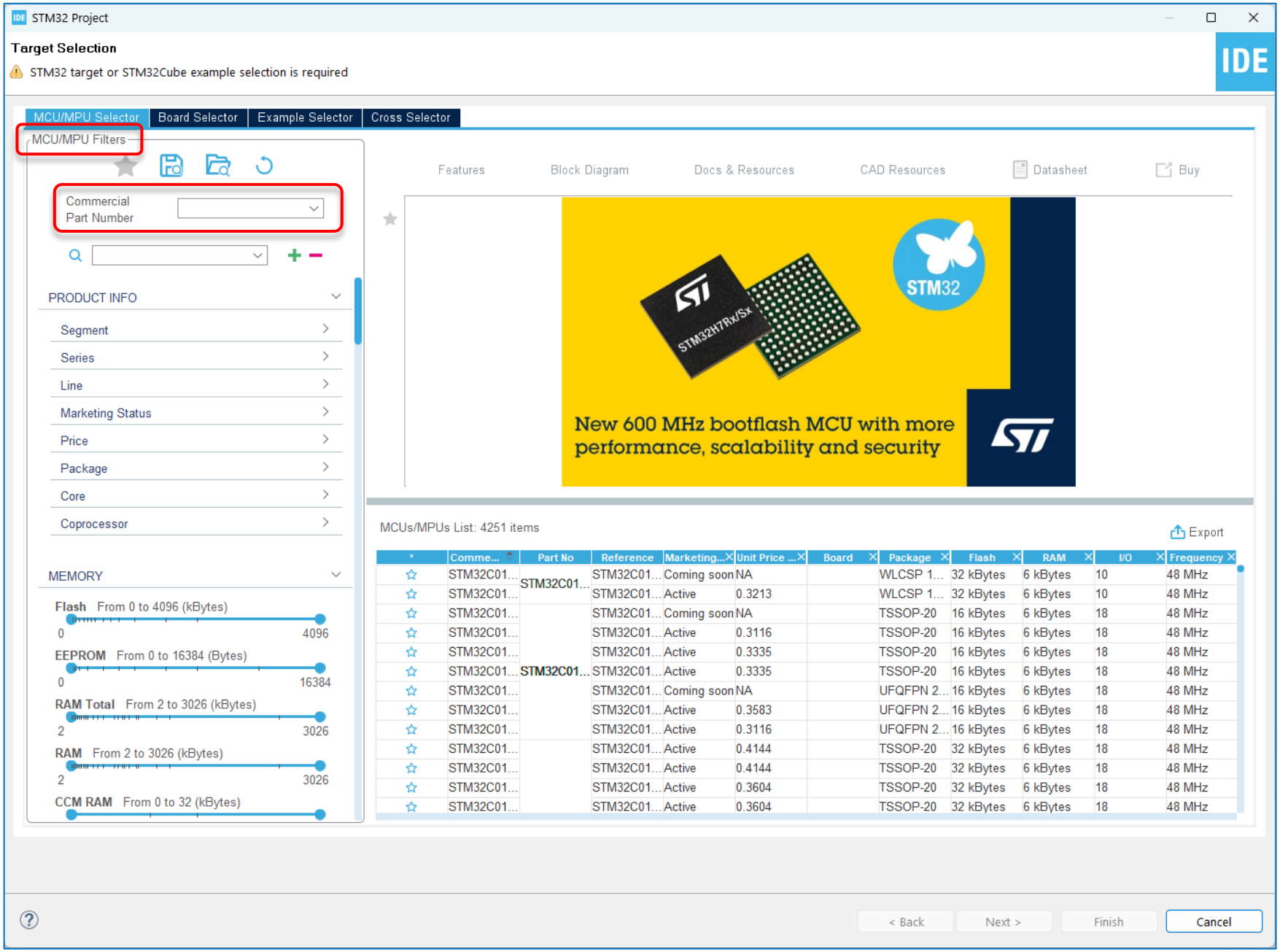
## Get Software

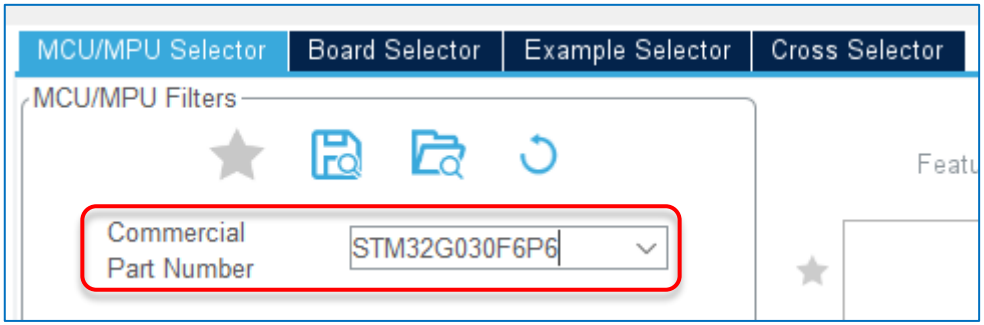
| | Part Number | General Description | Latest version | Download | All versions |
|---|---|---|---|---|---|
| + | STM32CubeIDE-DEB | STM32CubeIDE Debian Linux Installer | 1.16.0 | Get latest | Select version |
| + | STM32CubeIDE-Lnx | STM32CubeIDE Generic Linux Installer | 1.16.0 | Get latest | Select version |
| + | STM32CubeIDE-Mac | STM32CubeIDE macOS Installer | 1.16.0 | Get latest | Select version |
| + | STM32CubeIDE-RPM | STM32CubeIDE RPM Linux Installer | 1.16.0 | Get latest | Select version |
| + | STM32CubeIDE-Win | STM32CubeIDE Windows Installer | 1.16.0 | Get latest | Select version |

3. It will be good if you make your account on STMicroelectronics website in case there is any requirement further.
4. Select your operating system and click on **“Get latest”** to start downloading the software.
5. The download will begin and this will take some time to complete.
6. While downloading the software, keep the default settings and finish the process.
7. Now to upload the first program we well create a new project.
8. Go to **File** >> **New** >> **STM32 Project**

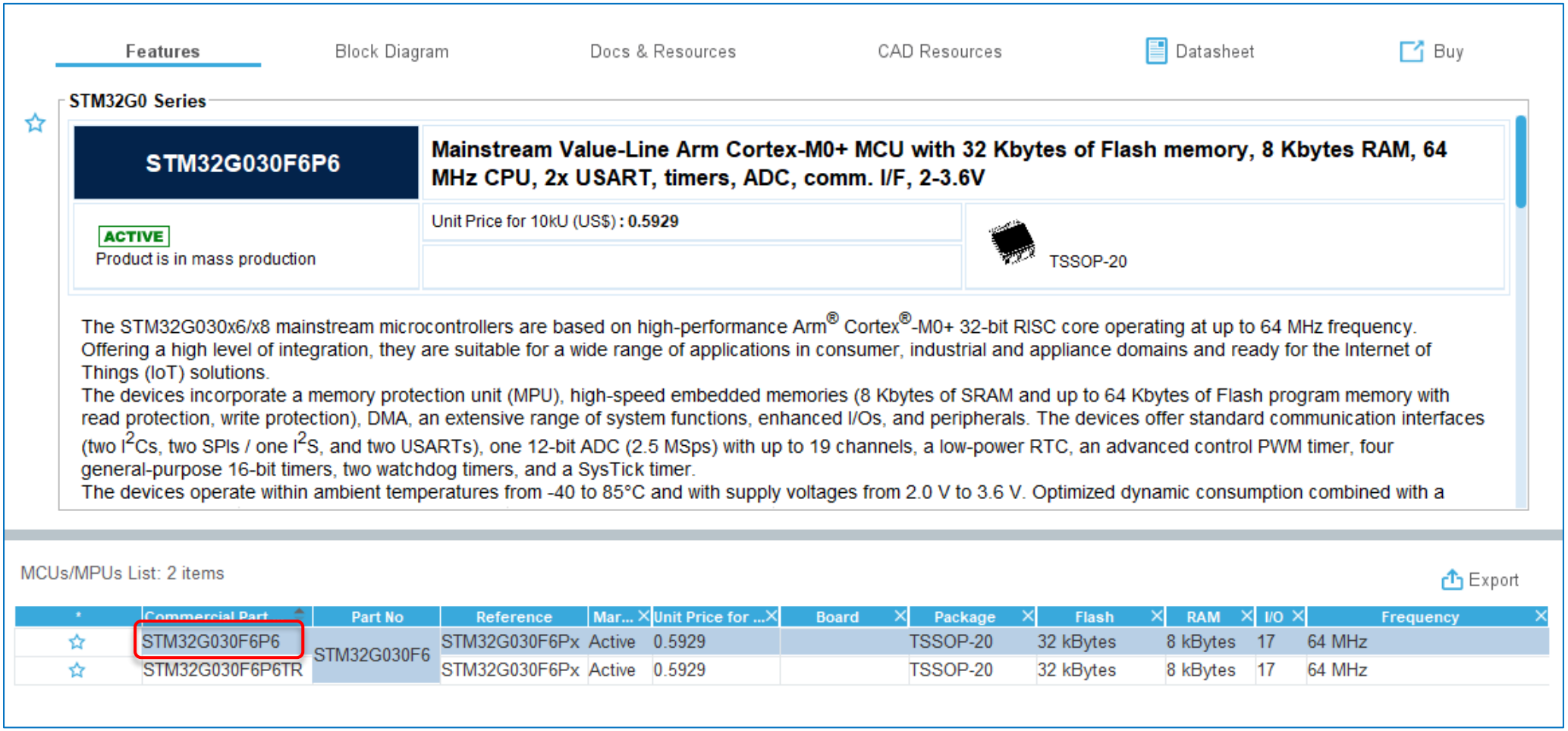


9. Now you will get to see a new tab to select our microcontroller.
Under **MCU/MPU Filters** menu enter the part number of the microcontroller used in **Commercial Part number** option, in our case **STM32G030F6P6**



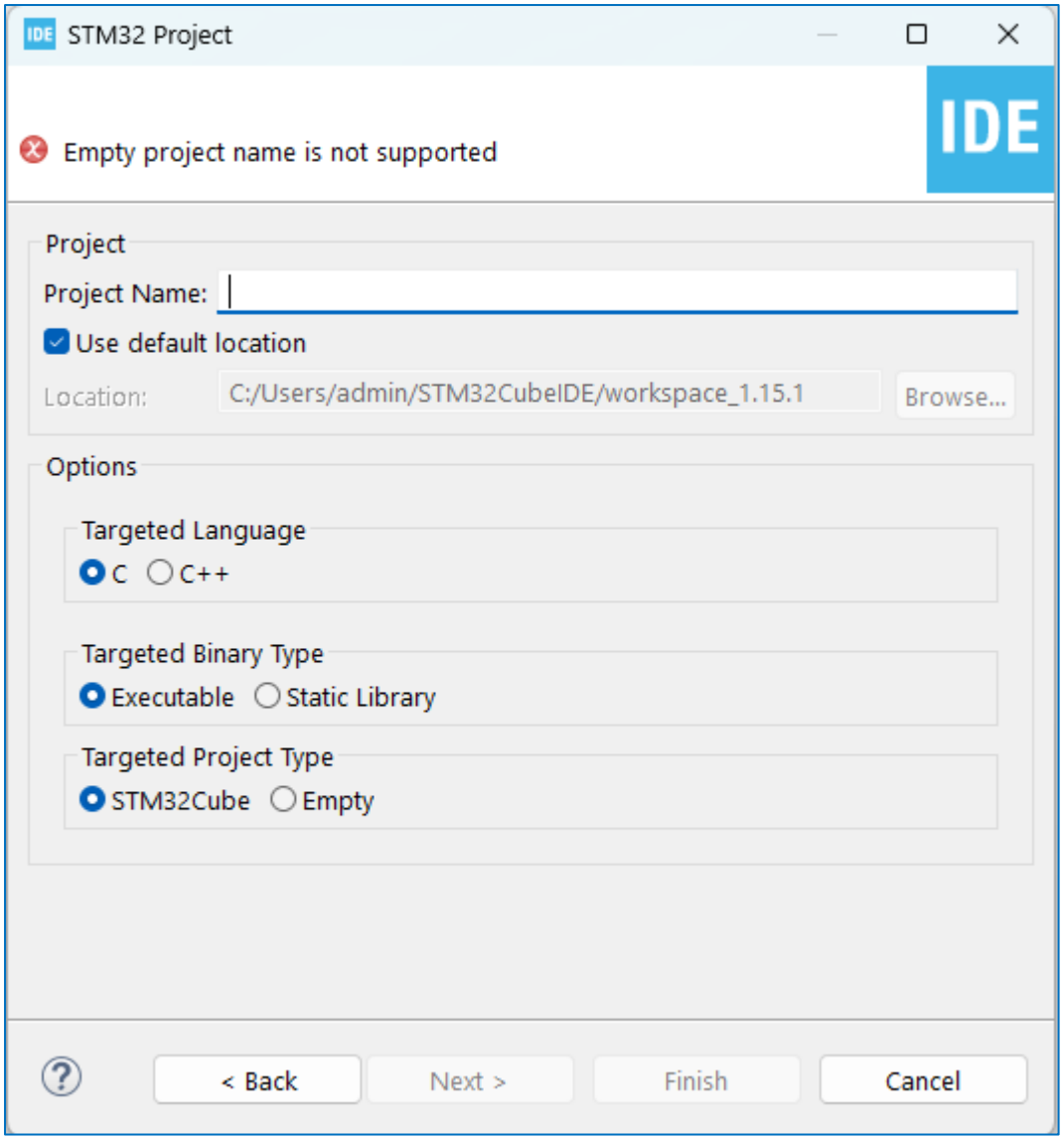



10. Now select the first option as shown below and click on **Next**:-

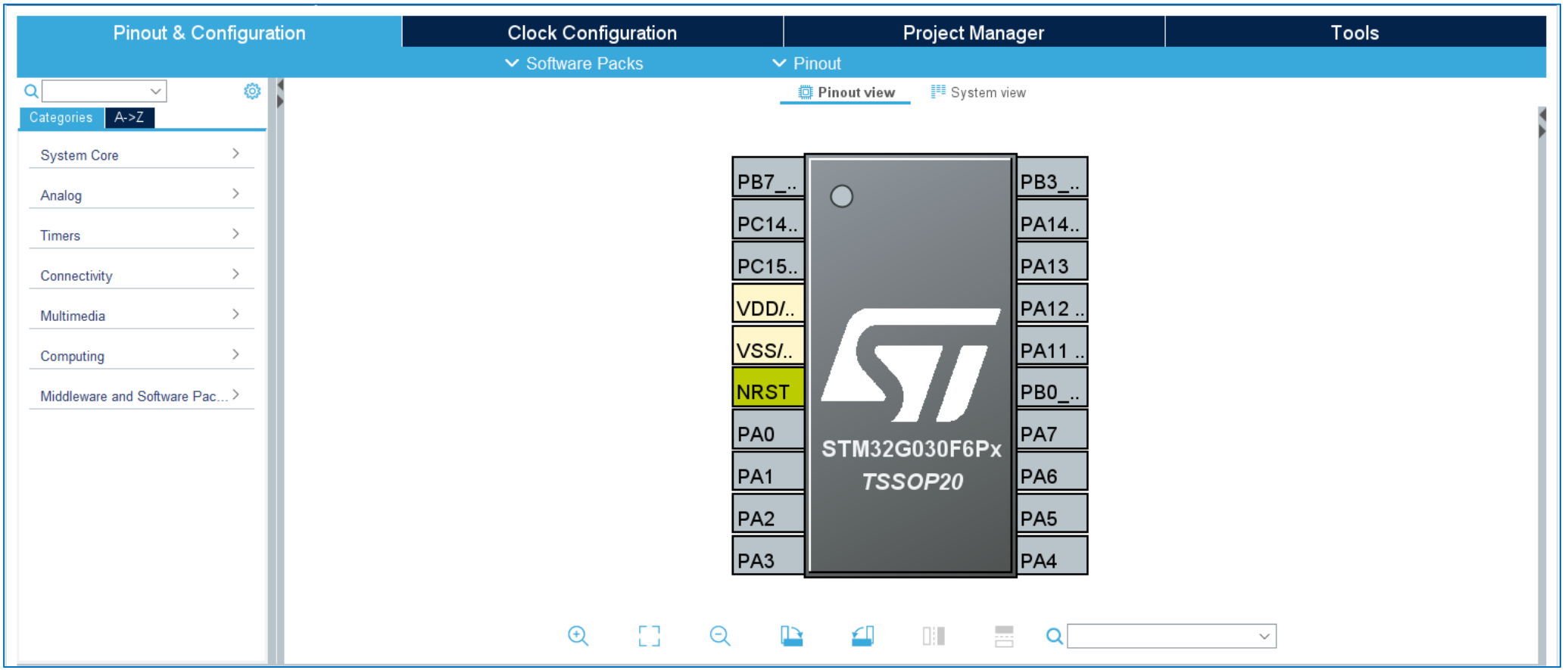



| * | Commercial Part | Part No | Reference | Mar... | Unit Price for ... | Board | Package | Flash | RAM | I/O | Frequency |
|---|---|---|---|---|---|---|---|---|---|---|---|
| ☆ | STM32G030F6P6 | STM32G030F6 | STM32G030F6Px | Active | 0.5929 | | TSSOP-20 | 32 kBytes | 8 kBytes | 17 | 64 MHz |
| ☆ | STM32G030F6P6TR | | STM32G030F6Px | Active | 0.5929 | | TSSOP-20 | 32 kBytes | 8 kBytes | 17 | 64 MHz |

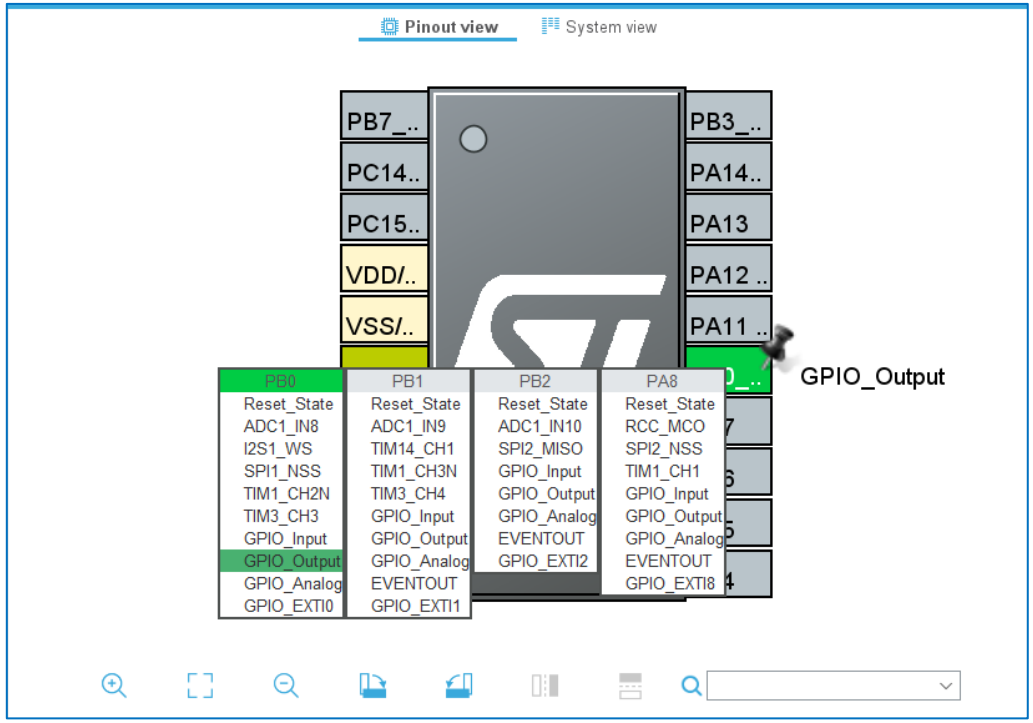
11. Give your project any name and click on **“Finish”**



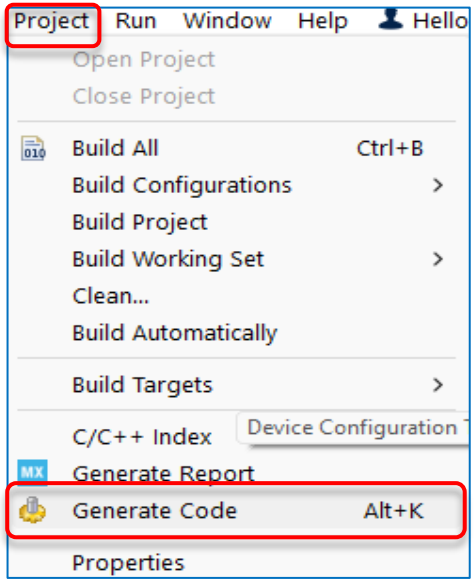
12. A new tab will open as shown in the below image, (open **.ioc file** from the left corner if not opened)



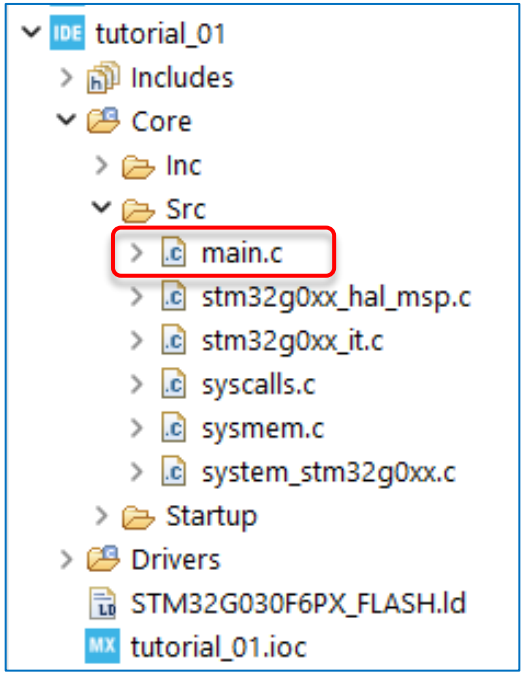
13. Configure the **“PB0”** as **“GPIO output”** pin as shown in the image below




14. Go to Project >> Generate code



15. To open the generated code, follow the image below.




This option is under **Project workspace** somewhere in the left side
Go to your project name (tutorial_01 in my case) >> **Core** >> **Src** >> **main.c**

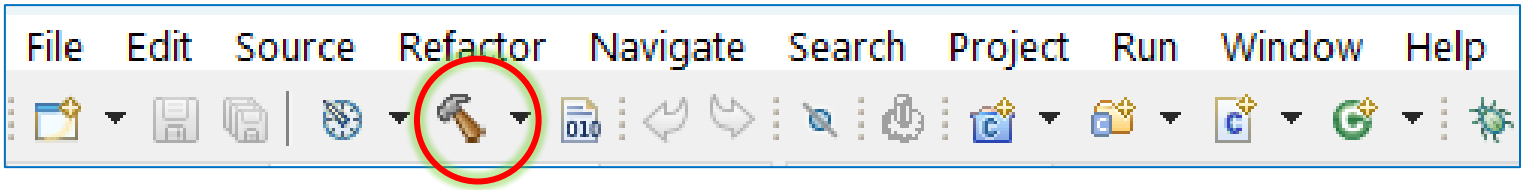
inside **main.c** you will find your generated code

Inside STM32 cube IDE, when we generate code then the necessary functions/ initialization related to the peripherals we want to use gets defined in the code so we don't need to write it again.

16. Open your **main.c** file to edit the code for blinking an LED connected to GPIO_PIN_0.
Enter the additional part of code inside **while (1)** loop. Don't change any other part of the code, keep it as it is.

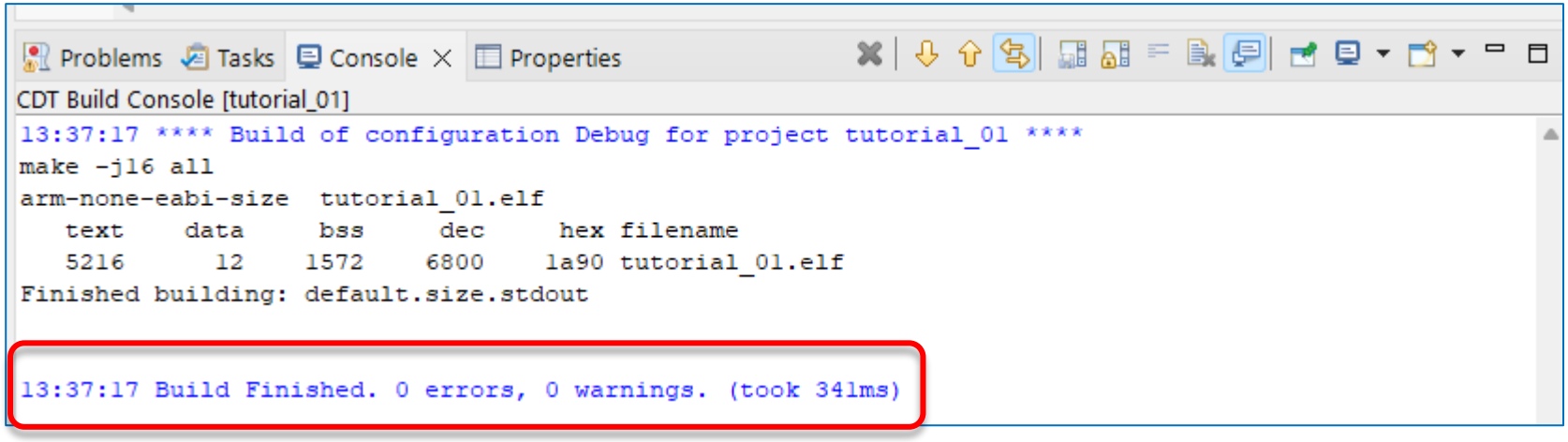
```
95   while (1)
96   {
97     /* USER CODE END WHILE */
98
99     /* USER CODE BEGIN 3 */
100    HAL_GPIO_WritePin(GPIOB, GPIO_PIN_0, 1);
101    HAL_Delay(100);
102    HAL_GPIO_WritePin(GPIOB, GPIO_PIN_0, 0);
103    HAL_Delay(100);
104  }
105  /* USER CODE END 3 */
106 }
```

17. Save your project and now we will try to **build the project** to check for any errors.

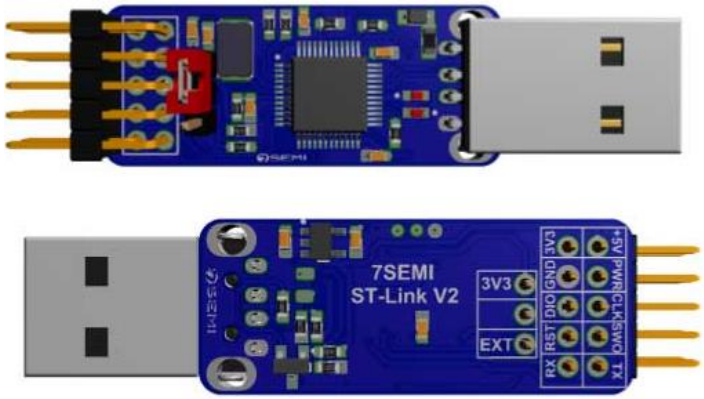



**Click on this icon to build your project.**

You will get this type of message if your build is successful.




18. After building the project its now time to upload this code to STM32 board. To upload the code you will need a ST-Link programmer

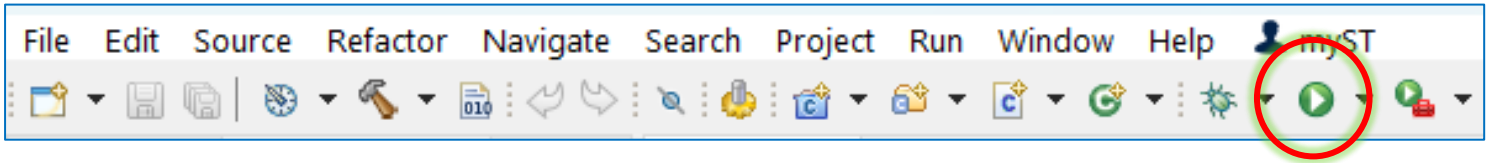


Connection of ST-Link with STM32G030F6P6 board

| ST-Link V2 | STM32 Board |
|---|---|
| 3V3 | 3V3 |
| DIO | DIO |
| CLK | CLK |
| RST | RST |
| GND | GND |

You have to just match the labels given on both the boards.

19. To flash the code into your STM32 board you have to click the highlighted icon as shown below.




Initially you will get the message as shown in the image below.

```
Waiting for debugger connection...
Debugger connected
Waiting for debugger connection...
Debugger connected
Waiting for debugger connection...
```

After debugger is connected successfully it will start flashing the code and you will get a message as shown in the image below.

```
File download complete
Time elapsed during download operation: 00:00:00.249


Verifying ...



Download verified successfully


Shutting down...
Exit.
```

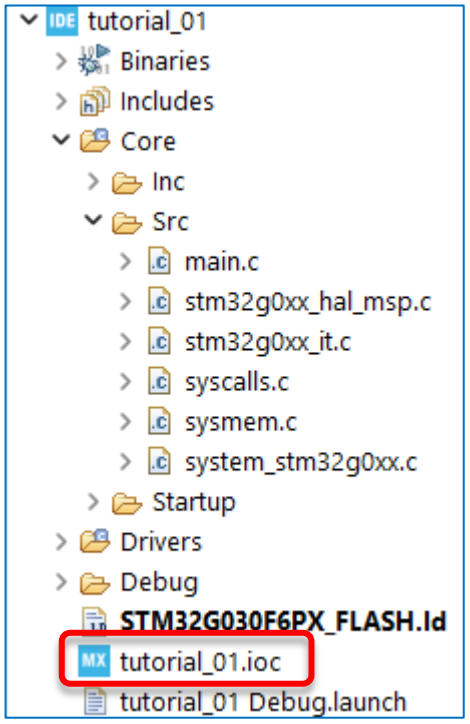
This means that the code is uploaded successfully and the LED connected to **PB0** should start blinking.

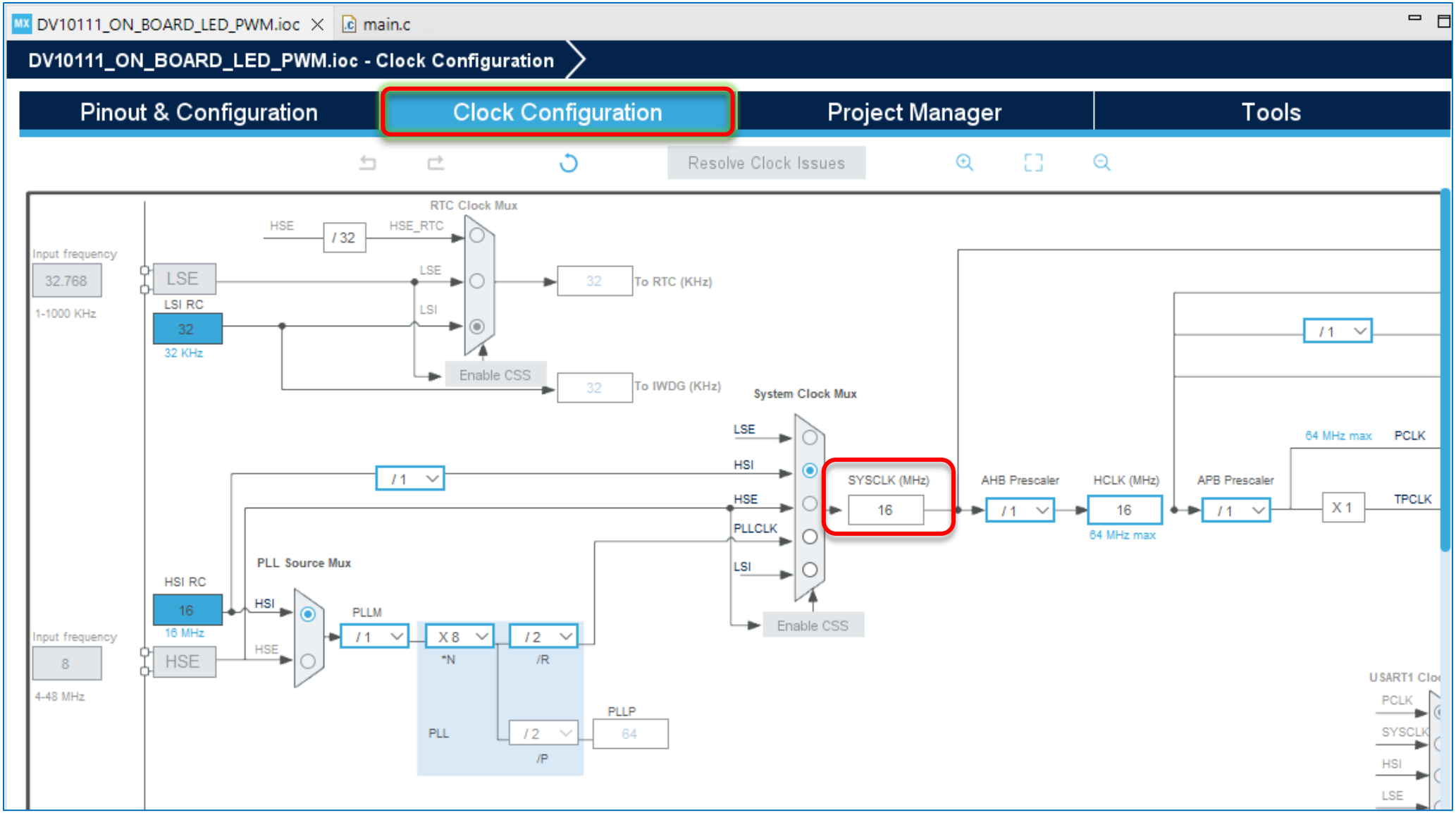
**Connect a 330 Ω resistor in series with the LED.**

20. For the 2nd sample code we will try to adjust the brightness of the LED using PWM. For this example we will need to configure the TIMER peripheral of the STM32 board and there are some basic calculations to set the PWM frequency.

First we want to know the System Clock frequency in order to calculate the PWM frequency.
After looking at the STM32G030F6P6 board you will come to know that there is not any **external crystal/ oscillator** connected with the chip. So, in this case we'll be using the internal System Clock for the timing purpose.

21. This information you will get to know in the **Clock configuration menu**.
→ First open the **.ioc file** as shown in the image below.





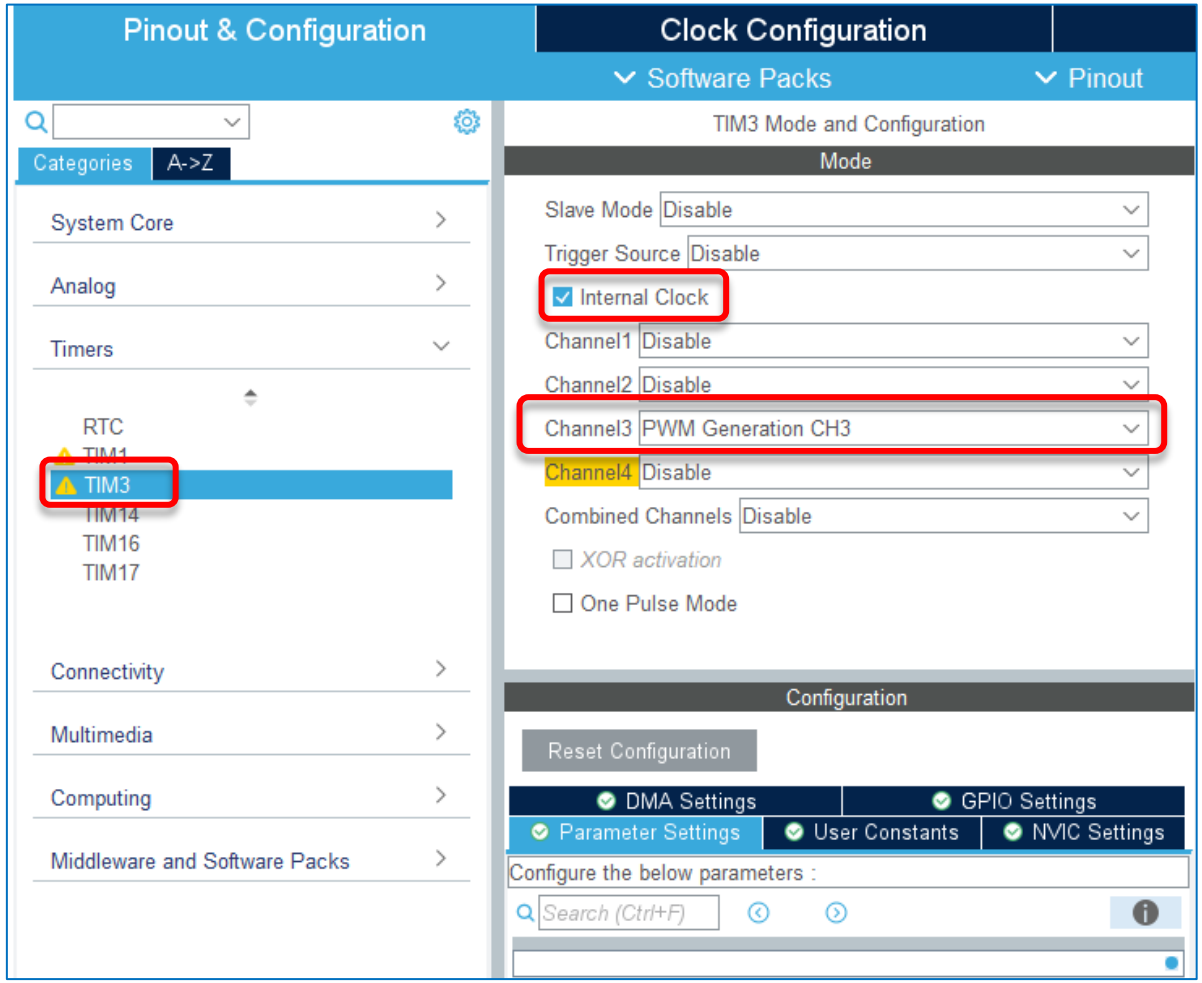

The SYSCLK is set as 16MHz.

$$\textbf{PWM frequency} = \frac{\textbf{Input frequency to timer unit}}{\textbf{(1+Prescalar) x counter period}}$$

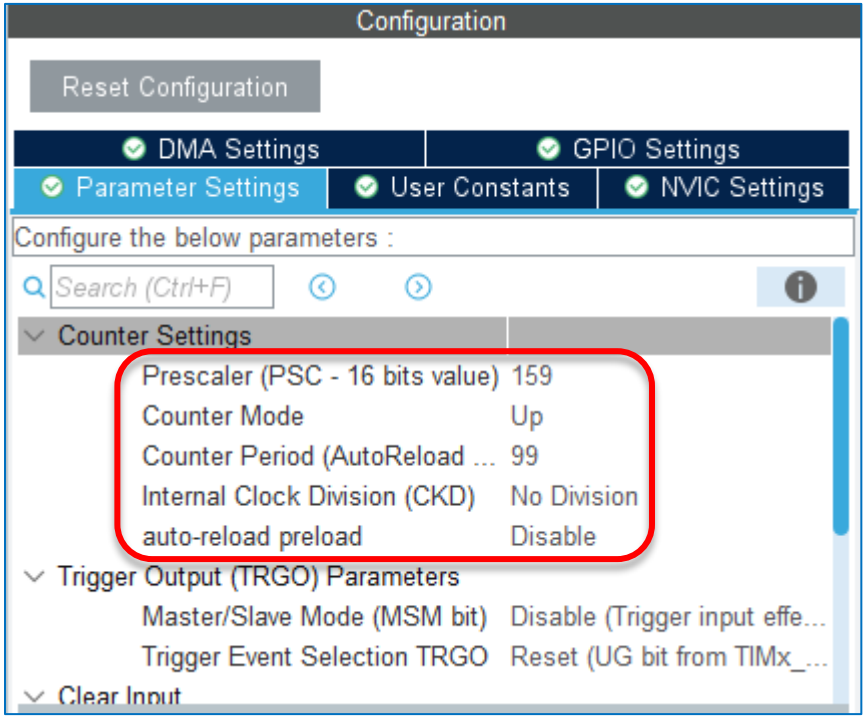
Input frequency to timer unit = 16MHz
Prescalar = 159
Counter period = 99

After substituting these values we get a PWM frequency of **1KHz**.

22. To set these values in the STM32 cube IDE refer the below image.
**Pinout & configuration** >> **Timers** >> **TIM3** >> **Mode** >> **Enable internal clock** >> **Set Channel 3 for PWM**



Now do the following settings as shown in the image below.




23. After doing the above settings generate a new code by following the **Step 14**.
Now you can replace the code previously written in the while loop and add the code given at the right side.

Upload the code by following the above steps and check whether the brightness of the LED is varying.

```
88
89    /* Initialize all configured peripherals */
90    MX_GPIO_Init();
91    MX_TIM3_Init();
92    /* USER CODE BEGIN 2 */
93    HAL_TIM_PWM_Start(&htim3, TIM_CHANNEL_3);
94    /* USER CODE END 2 */
95
96    /* Infinite loop */
97    /* USER CODE BEGIN WHILE */
98    while (1)
99    {
100     /* USER CODE END WHILE */
101
102   /* USER CODE BEGIN 3 */
103     int x;
104             for(x=0; x<100; x=x+1)
105             {
106                 __HAL_TIM_SET_COMPARE(&htim3, TIM_CHANNEL_3, x);
107                 HAL_Delay(100);
108             }
109             for(x=100; x>0; x=x-1)
110             {
111                 __HAL_TIM_SET_COMPARE(&htim3, TIM_CHANNEL_3, x);
112                 HAL_Delay(100);
113             }
114   }
115   /* USER CODE END 3 */
116 }
```