Experiment 0: Introduction

In this experiment, student will learn how to use the board and how to program STM32F103C8T6 on the Blue-pill development board and KU lab board.

1. STM32F103C8T6 Blue-pill development board

Blue-pill is the barebone development board consists of STM32F103C8T6 IC as the main microcontroller. The controller uses LQFP48 surface mount package which has 48 pins as shown in Figure 1. Basically, the Blue-pill is the extension of the controller pins as shown in Figure 2 with the addition of basic requirement circuit such as Mode selection (1), Programmer interface pins (3), onboard LED, power system and crystals. The Blue-pill schematic is shown in Figure 3.

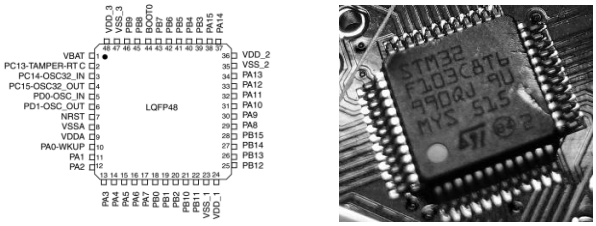


Figure 1. STM32F103C8T6

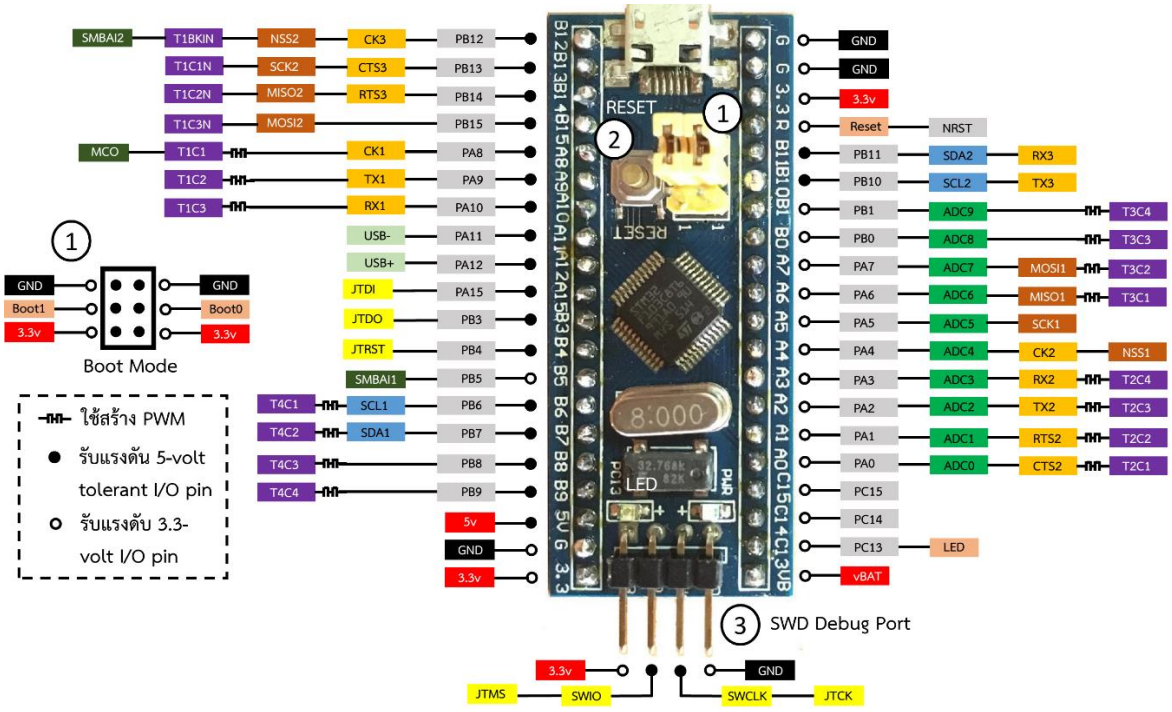


Figure 2. Blue-pill pins out.

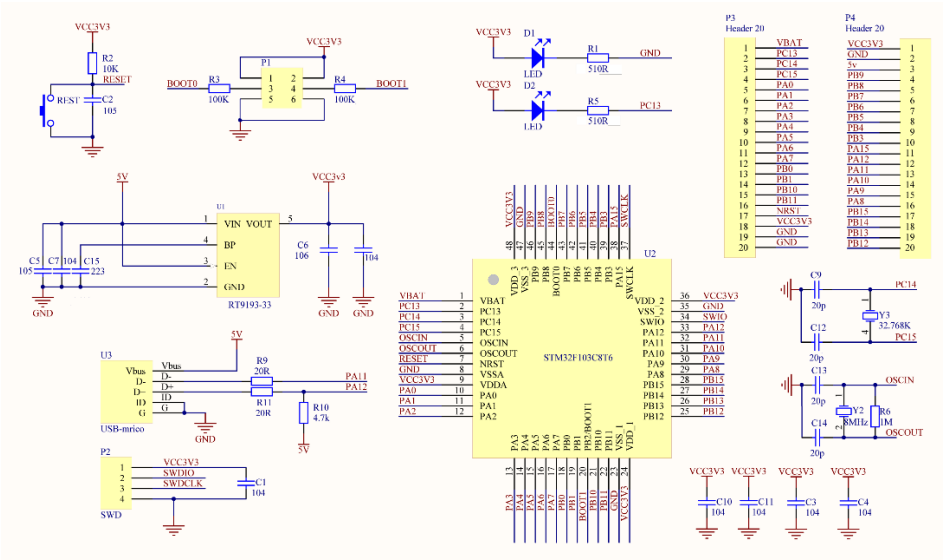


Figure 3. Blue-pill schematic.

In order to conduct the experiment in the microprocessor lab class, the extension board of KU lab board is connected to the Blue-pill board. The KU lab board consist of on-board LCD 16 characters 2 lines module, 2 variable resistor, FTDI USB to serial port connector, PCF8591 8-bits ADCs with DAC IC, PCF 8574 I/Os expansion IC, 4 push button switches and on-board LEDs. This board also has other connector which connect to the rest of the Blue-pill board pins. The print circuit board or PCB of this lab board is shown in Figure 4 while the schematic is shown in Figure 5.

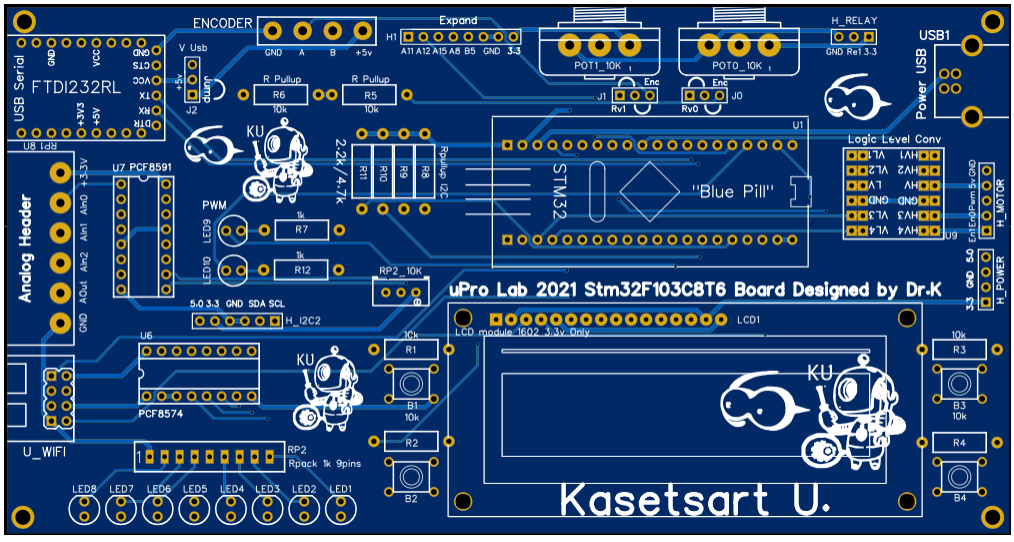


Figure 4. KU Lab print circuit board.

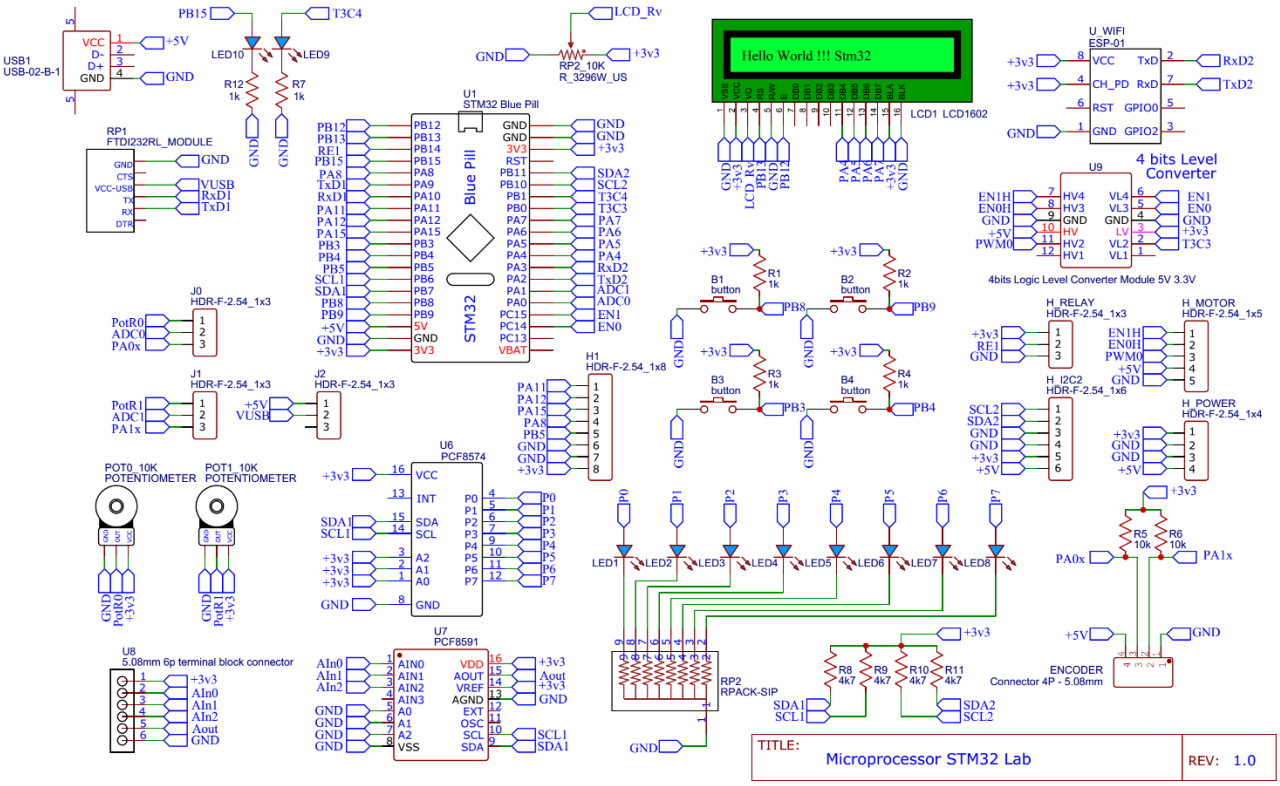


Figure 5. KU Lab board schematic

2. Code Editor

All of the experiments in this class use the TrueSTUDIO for STM32 IDE as their primary code editor. TrueSTUDIO is an Integrated Development Environment (IDE) developed by Atollic specifically for STmicroelectronics' ARM microprocessors. The base editor of this IDE is Eclipse, while the compiler tools are GNU Compiler Collection (gcc). Because the IDE is designed for ST's microcontrollers and processors, it includes STmicroelectronics' Standard Peripherals Library (SPL), which means it doesn't require any additional libraries to function properly. Atollic TrueSTUDIO for STM32 9.3.0 is used in this class. Since Atollic was purchased by STmicroelectronics and utilized in ST's IDE dubbed STM32Cube IDE, this is the last standalone version. Although, TrueSTUDIO is older than STM32Cube IDE, but, it uses SPL library as the main library from development for writing embed code in C language unlike the STM32Cube IDE which

based on HAL library. SPL is a straight-forward library for understanding ARM architecture because it can be understood immediately from the IC datasheet.

Figure 6. TrueSTUDIO.

To use the IDE, after installation process, one must setup the workspace in order to save the project and collect all of the files. The default workspace is **C:\Users\"User Name"\Atollic\TrueSTUDIO\STM32_workspace_9.3** as shown in Figure 7.

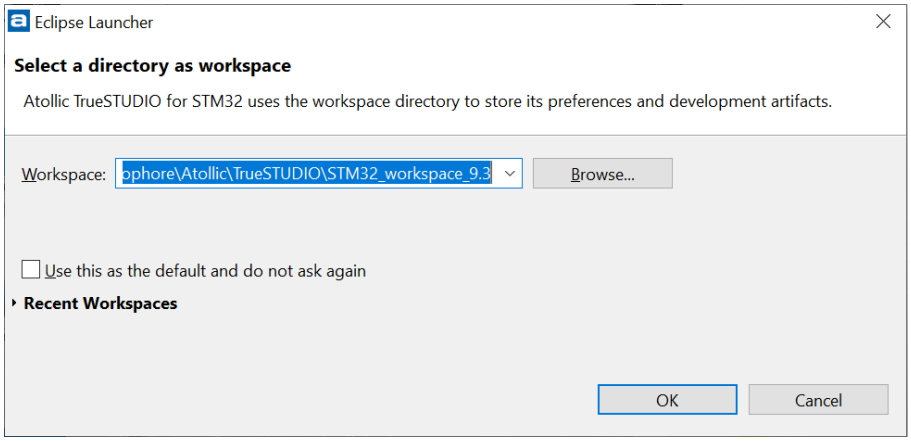


Figure 7. TrueSTUDIO workspace.

Then, the new project can be created by selecting **File -> New -> C Project**. Assign project name and select embedded project. Then, choose STM32F1 family and STM32F103C8 is the series microcontroller (MCUs).

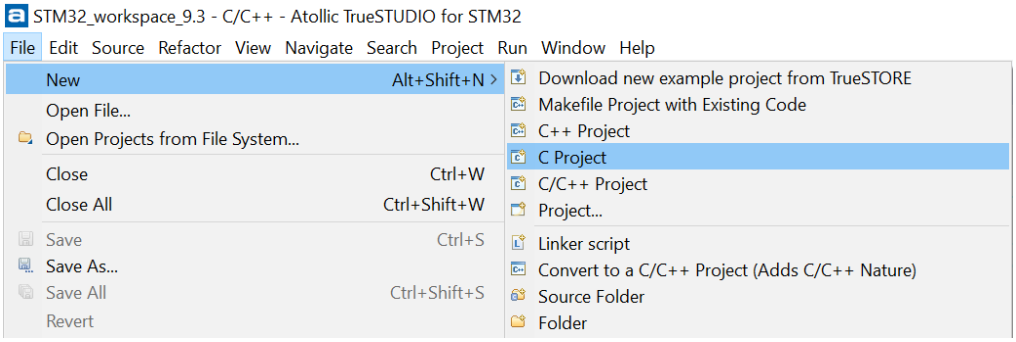


Figure 8. New project.

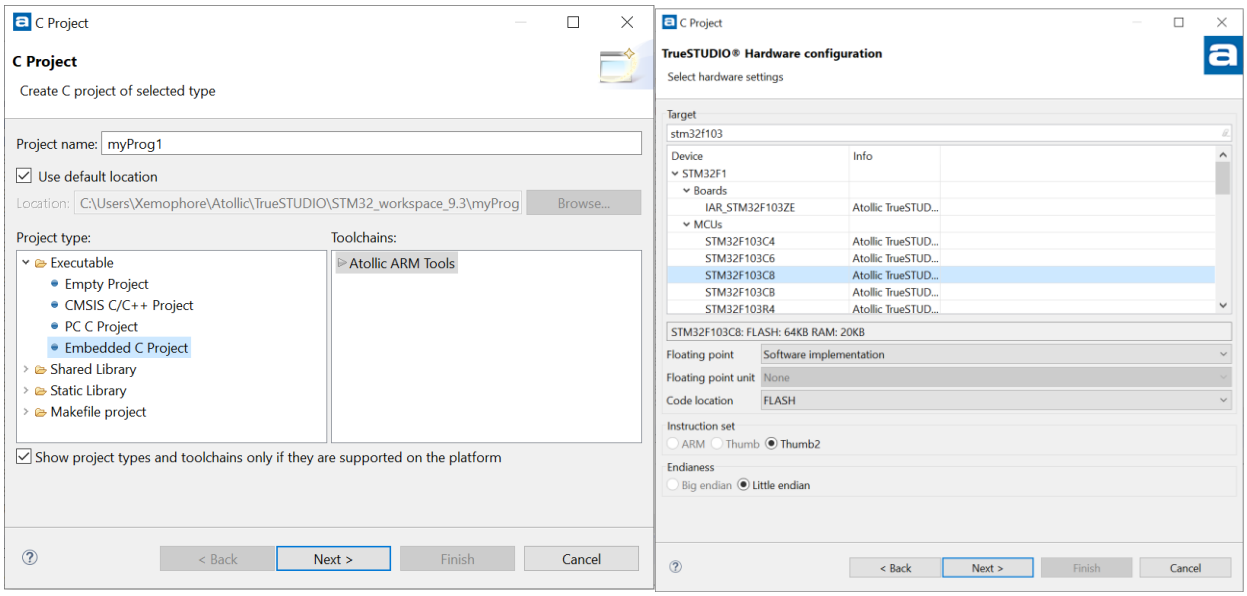


Figure 9. C project creation and MCUs specification

The option to use the “tiny printf” function is available.This choice allows the programmer to utilize less c code in the project than if they used the standard "printf/sprintf/fprintf" function, which requires more binary output (footprint). Finally, the programmer can choose the debugger probe, which is ST-Link by default. Because the serial programmer via bootloader is selected in this class, it doesn't matter which debugger you choose.

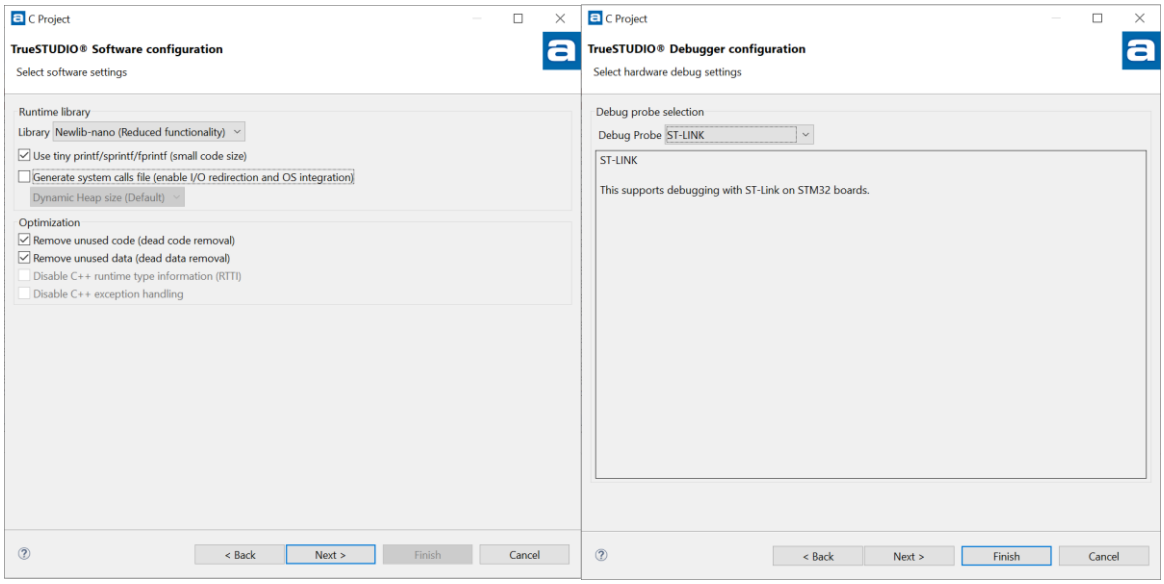
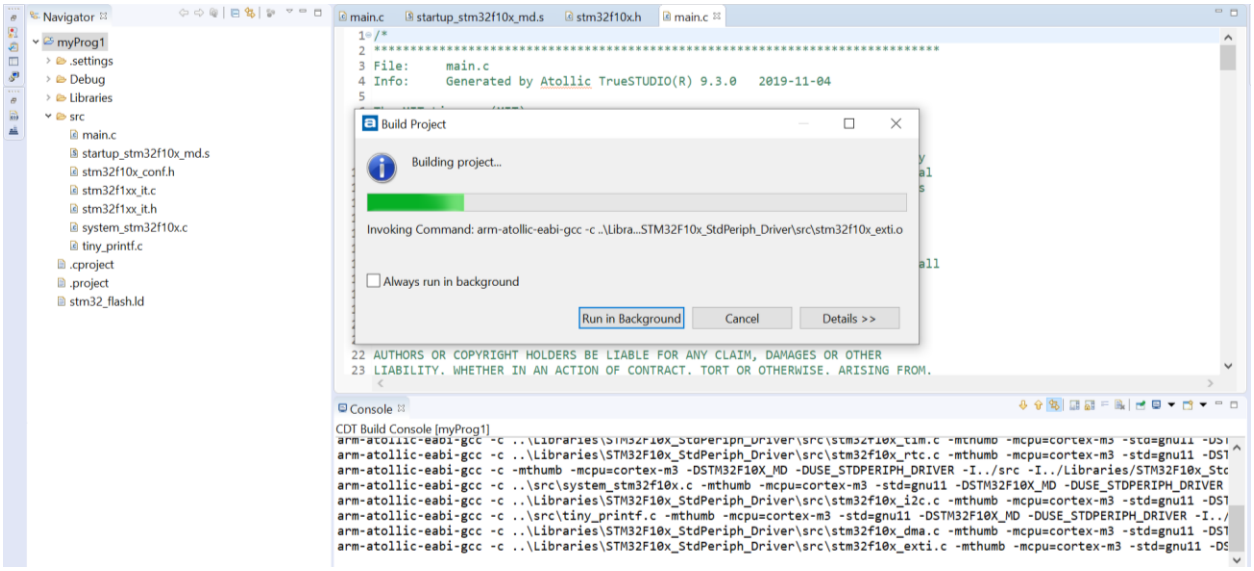


Figure 10. Tiny printf and debugger options.

After project is created, it requires time to compile the default source code files as shown in Figure 11.



Fgiure 11. First time project's code complied.

There is the source code files structure that are generated by default after create the project as shown in Figure 12. We will write our code into the **main.c** is this project.

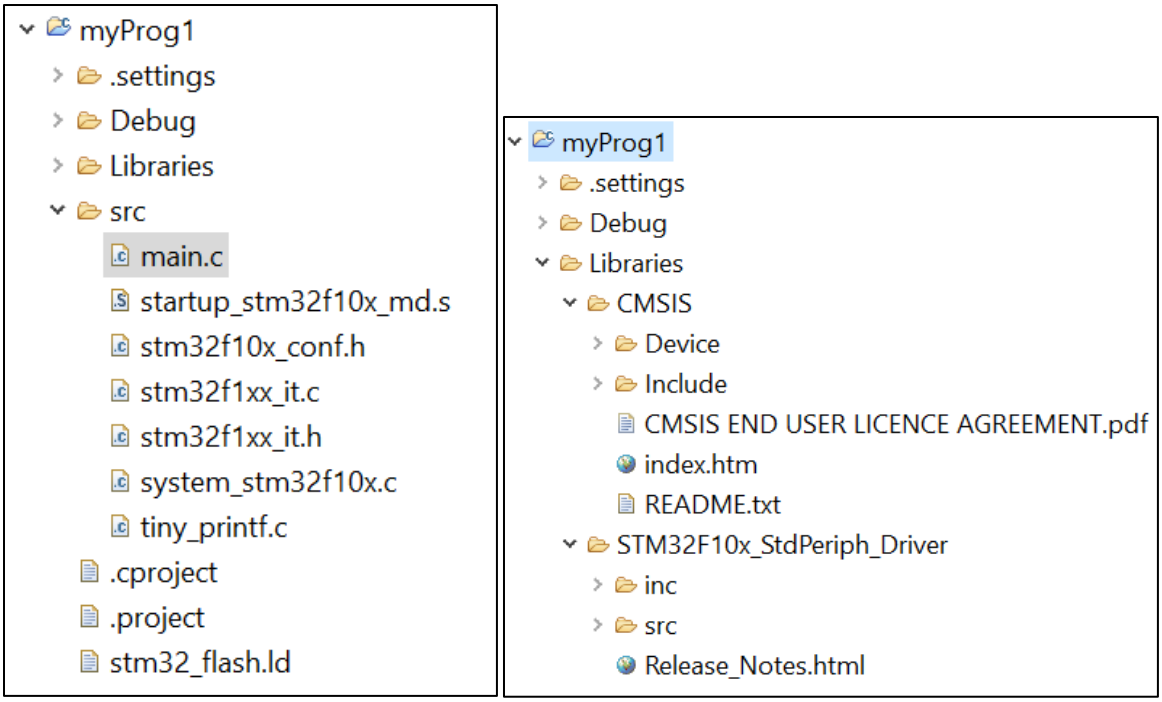


Figure 12. Project's source code files structure.

In order to get the binary file for the microcontroller, the output properties must be set by selecting **Project -> Properties** and selecting **C/C++ Build -> Settings -> Other -> Output format -> Convert build output -> Intel Hex** as shown in Figure 13.

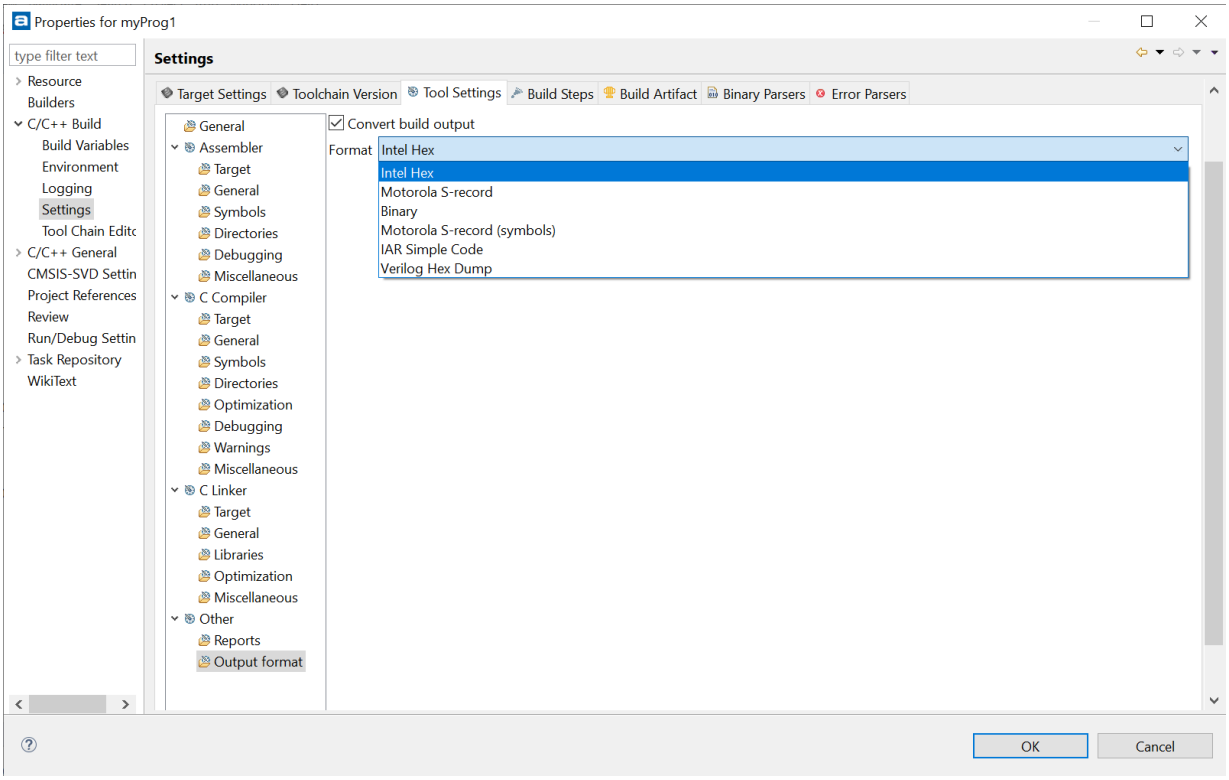


Figure 13. Output properties selection.

In this step, the developer can start the program through **main.c** and can compile the project via **Project -> Build All** as shown in the Figure 14.

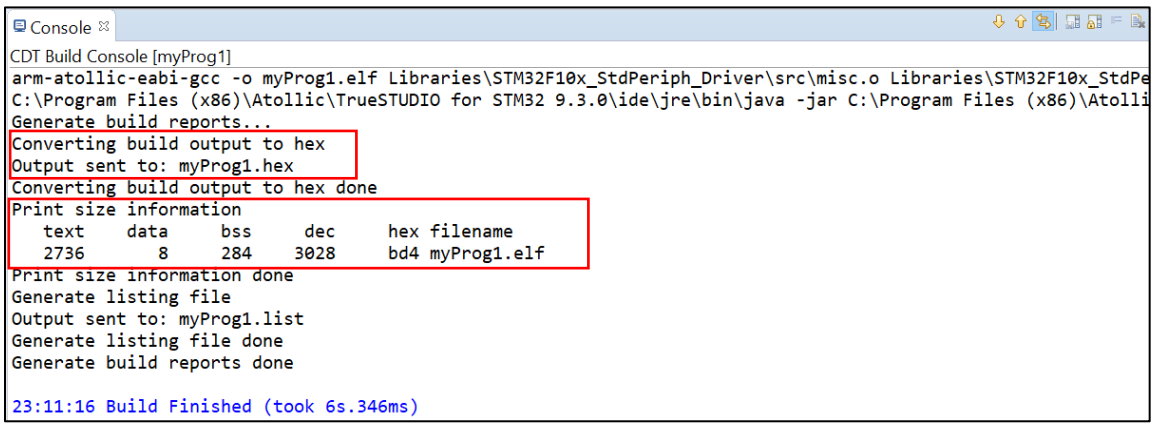


Figure 14. Result from compiling the project.

3. Programing STM32F103 via bootloader

The Binary file is loaded onto the chip using STM32 Flash loader demonstrator (FLASHER-STM32) through USART1 via RS-232 standard to connect to the computer via

the Serial port, however the Serial port has been issued for newer PCs or laptops. Therefore, programming the STM32F103C8T6 on Bluepill requires an additional electronic circuit to convert the USB signal into a serial signal consisting of TX and RX signals. An off-the-shelf USB to serial converter module is utilized to make the experiment less complicated. FTDI FT232RL is the IC that used to convert USB signal into serial signal. Bluepill requires only GND, Vcc, TX and RX signals. This module provides a selectable power supply voltage or Vcc at 3.3v or 5.0v. In this class, only 5v selection is used. After connecting the USB to Serial module to KU lab board and PC or notebook, the Windows OS will search for the FT232RL chip and install the driver for connecting to the circuit automatically in case of Windows 10. If the driver installed successfully when open Device Manager, computer will show the topic Port and when you press it, you will see the Port name as shown in Figure 15. This port name (number) will be used in the next step.

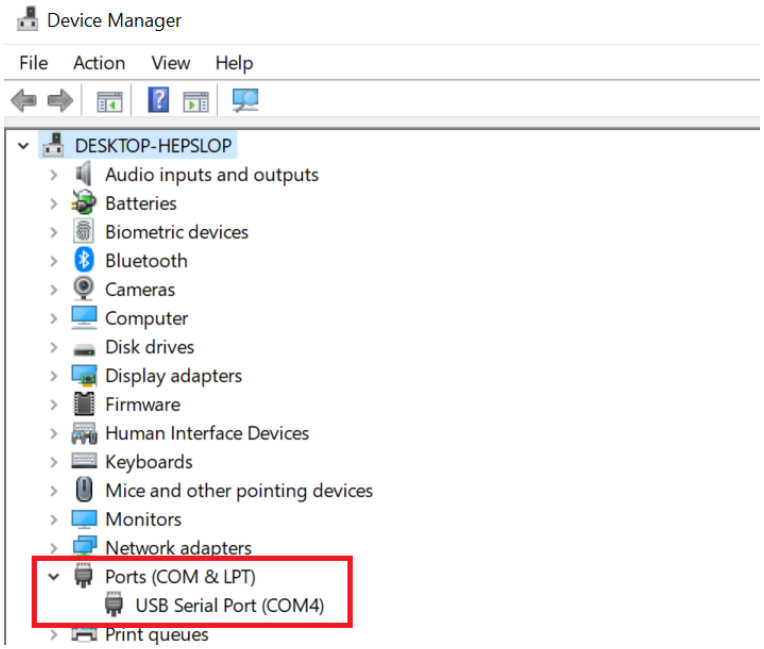


Figure 15. USB to Serial Port in Windows' Device Manager

Before using the programmer software, make sure the Blue-pill is in the bootloader programming mode by selecting the jumper as below.

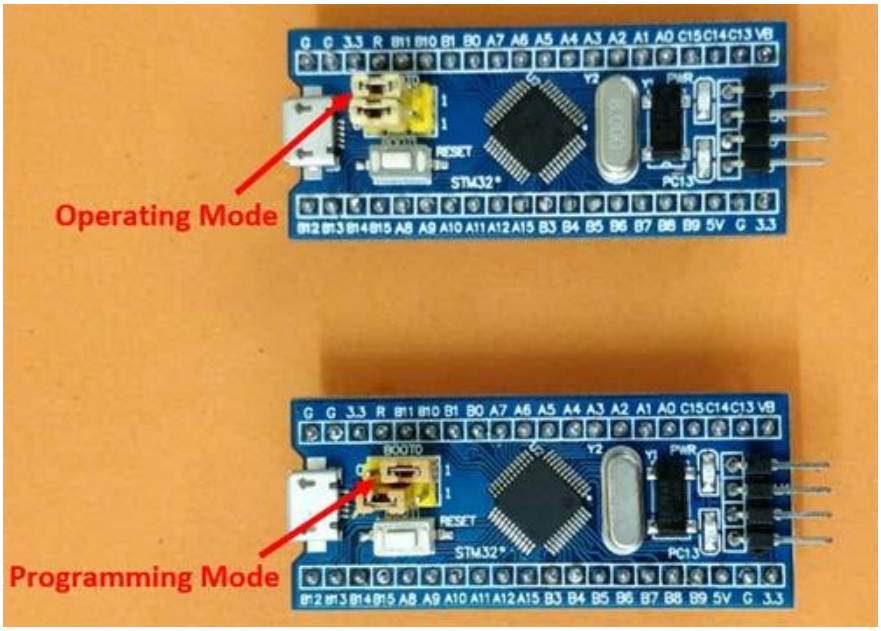


Figure 16. Blue-pill Modes/

Open the program STM32 Flash loader demonstrator (FLASHER-STM32) and select UART Port Name as the port name found in Device Manager and press Next. After that, Flash Loader will search for STM32 via serial port automatically. If the programmer finds the microcontroller, it will show massage Target is readable as the result. This window also presents the size of Flash memory in the microcontroller as shown in Figure 15. After pressing Next, the programmer will show the family of microprocessor (STM32F1_Med-density_64K or STM32F1_Med-density_128K). Then, press Next again. For programming into the STM32F103C8T6 on Bluepill, press the circle in front of the title **Download to device** and press the button **...** to browse to your binary file ***.hex** or ***.bin** depends on your compiler as shown in Figure 18. After select binary file then press OK and press Next to program the binary file to your microcontroller. If the programing successes, the result will show in Figure 19.

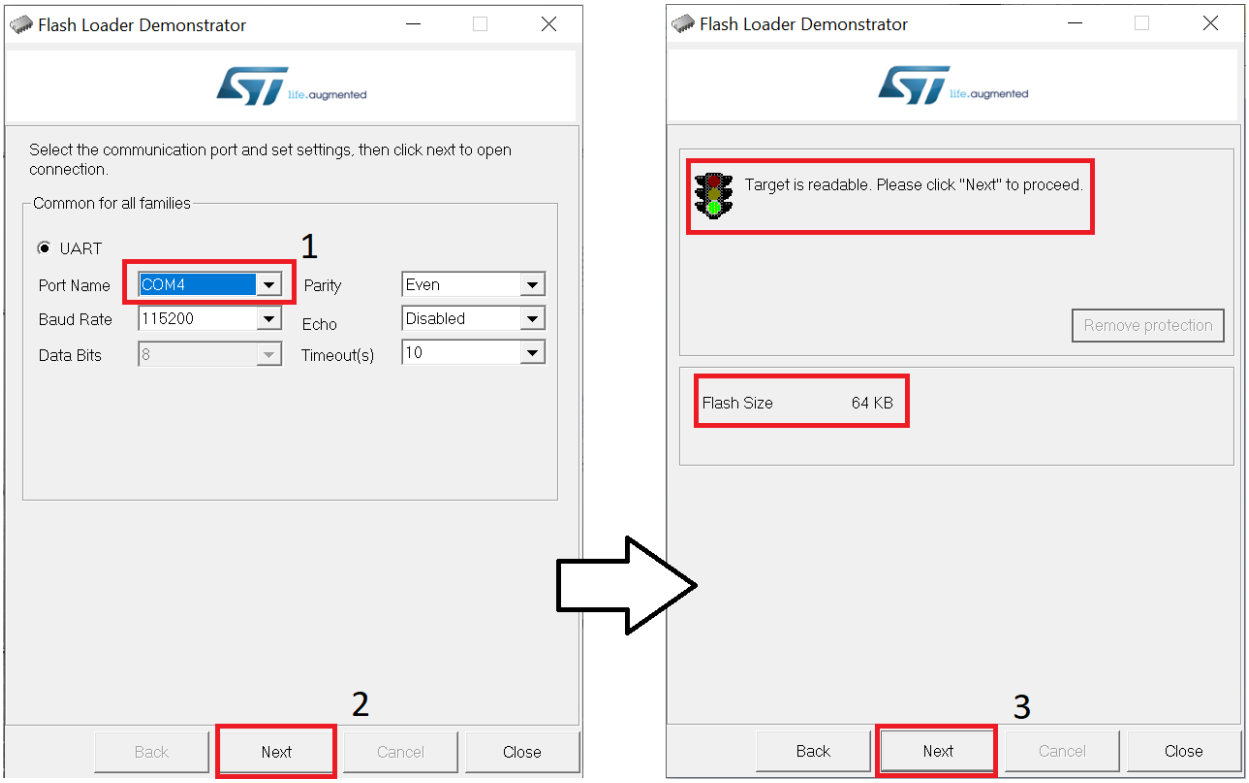


Figure 17. Serial Port Selection and Target microcontroller IC information display.

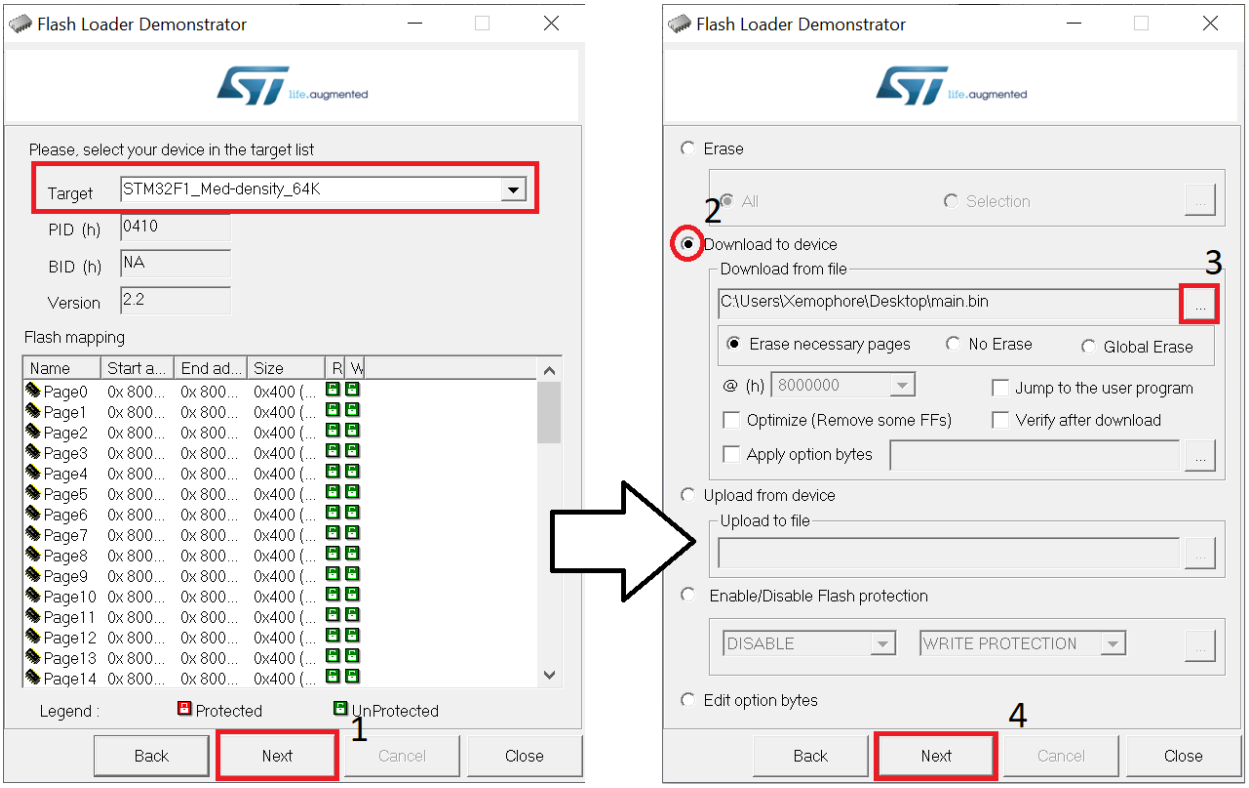


Figure 18. Binary file selection.

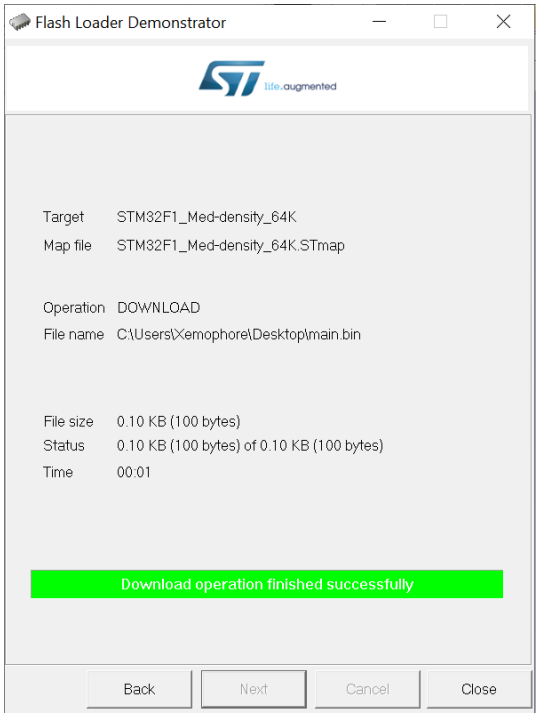


Figure 19. Successfully programing binary file to STM32F103C8T6

Most of the time, the binary file location is the issue. Therefore, in order to get correct binary file, first open the workspace of the TrueSTUDIO and find the project name and double click to access the project folder. Then, go to folder name **Debug** and the correct binary file will be in this Debug and has the same name as the project name.

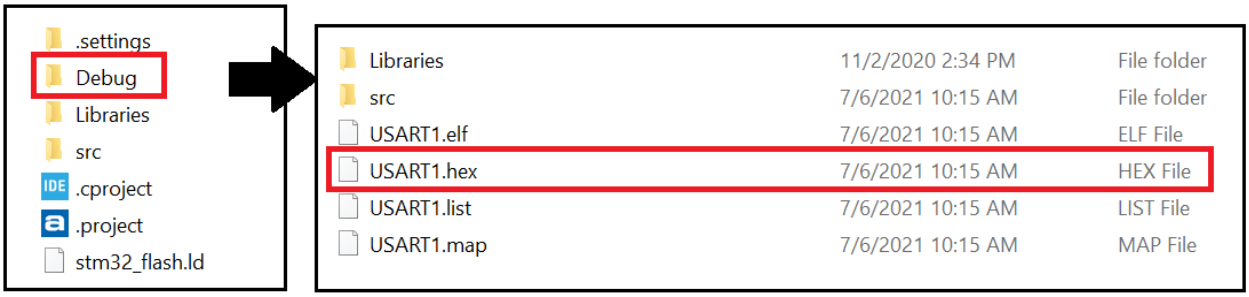


Figure 20. The correct binary file.

4. Testing the KU Lab board

Starting by create the new project in TrueSTUDIO and named the project as **TestKUBoard**. Make sure that the is no other project is opened. The add new include and function **my_delay( )** to the main.c before **main( )** function. Then delete all codes in the **main ( )** function and write down the new code as follow.

```c
/* Includes */
#include <stddef.h>
#include "stm32f10x.h"
#include "stm32f10x_conf.h"
#include "stm32f1xx_it.h"

void my_delay(unsigned int count)
 {
    while(count>0)
    {    count--;  }
}

int main(void)
{
  unsigned char state =0;

  RCC_APB2PeriphClockCmd(RCC_APB2Periph_GPIOC, ENABLE);
  GPIO_InitStruct.GPIO_Pin = GPIO_Pin_13;
  GPIO_InitStruct.GPIO_Speed = GPIO_Speed_50MHz;
  GPIO_InitStruct.GPIO_Mode = GPIO_Mode_Out_PP;
  GPIO_Init(GPIOC, &GPIO_InitStruct);
  GPIO_WriteBit(GPIOC,GPIO_Pin_13,1);

  while(1)
    {
      //do nothing //
      my_delay(400000);
      if (state == 0)
          {
              state = 1;
              GPIO_WriteBit(GPIOC,GPIO_Pin_13,0);
          }
          else {
              state = 0;
              GPIO_WriteBit(GPIOC,GPIO_Pin_13,1);
          }
     }
}
```

Compile that project and make sure that the **TestKUBoard.hex** is created. Then download it to the KU lab board via USB-to-serial convertor module using STM32 Flash loader demonstrator (FLASHER-STM32). Don't forget to change Blue-pill board to programming mode. After finish programming, don't forget to change Blue-pill board mode back to operating mode. If everything is corrected, the LED on Blue-pill board will be flashing.