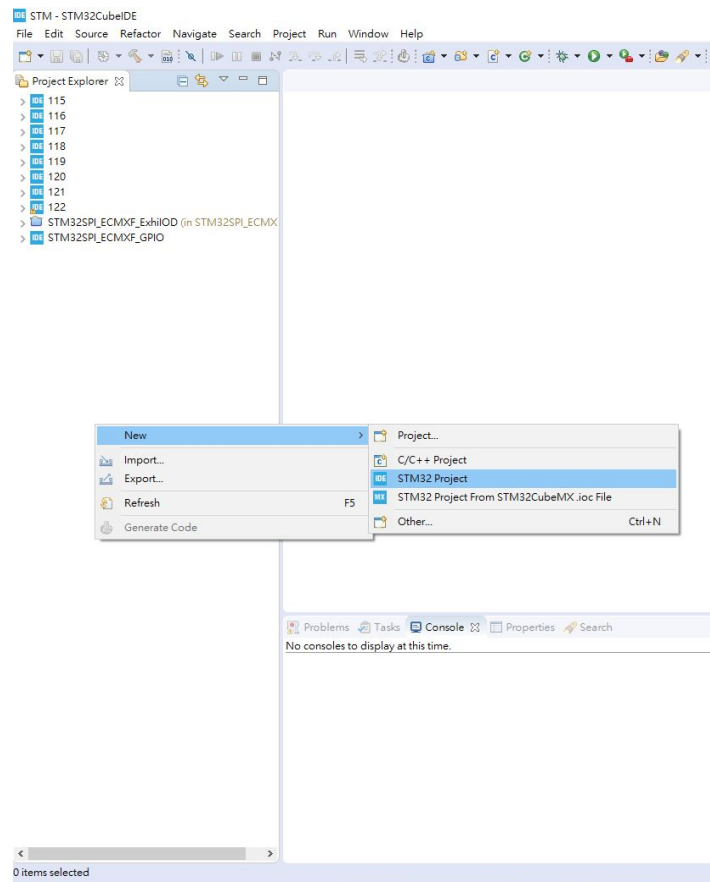
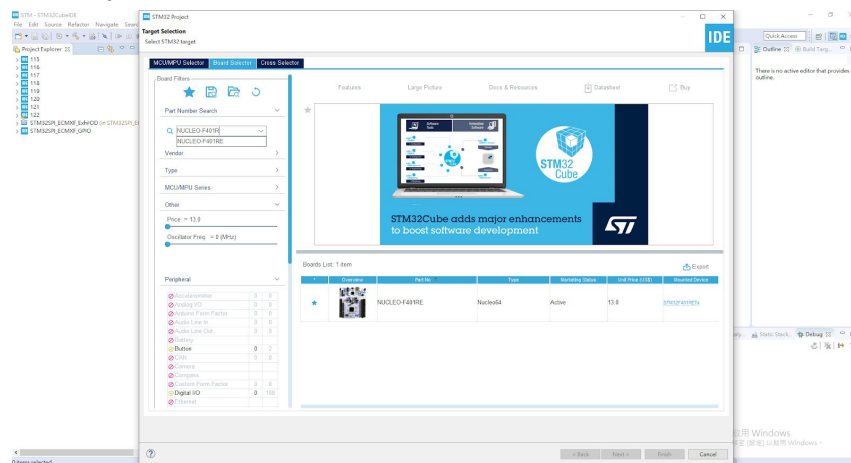


STM32 Setup Instruction (NUCLEO-F401RE)

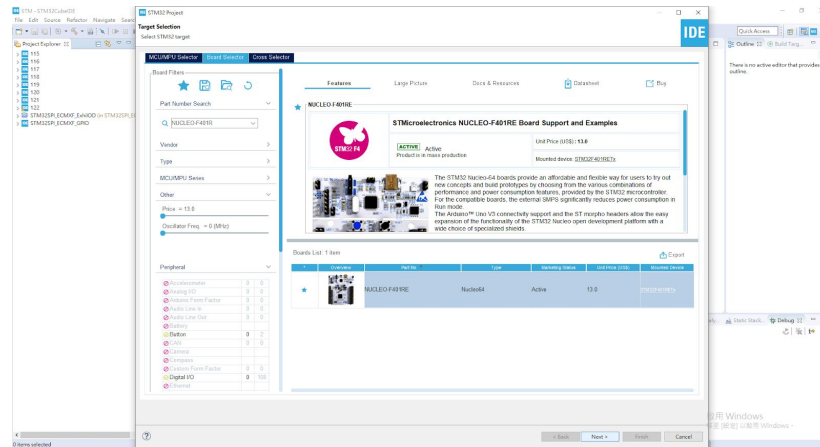
1. Open STM32Cube IDE.
2. Right click and select “STM32 Project”.



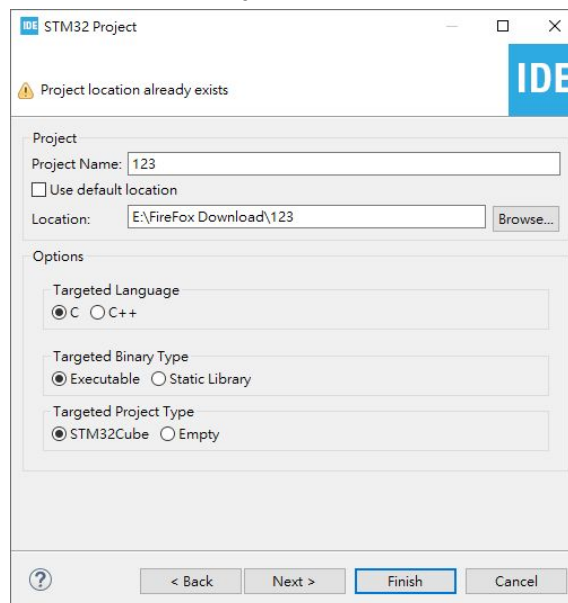
3. Select the board you have. For example, the selected board is NUCLEO-F401RE.



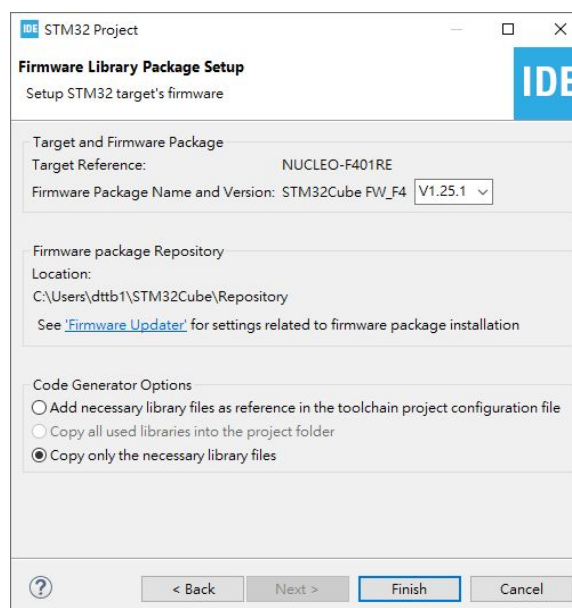
4. Then click “Next” to continue setup.



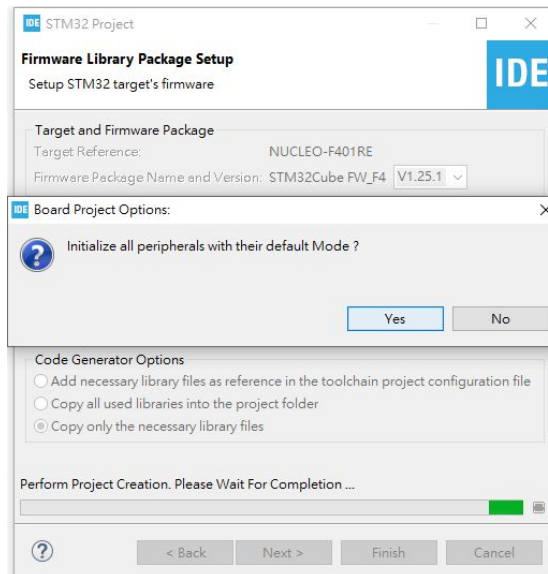
5. Insert the project name and decide project location.



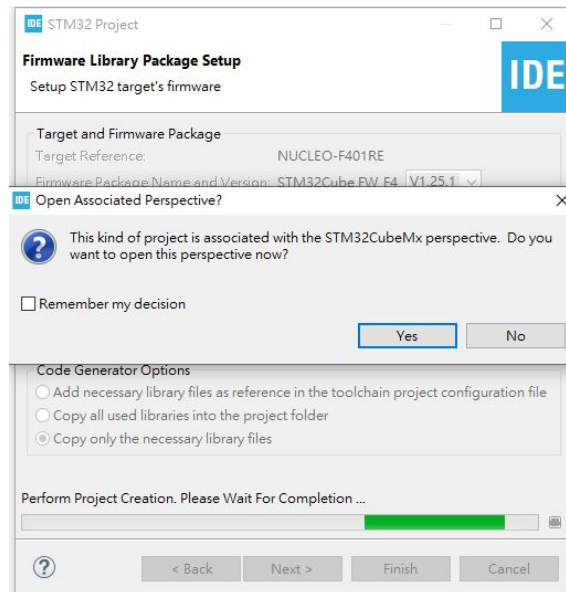
6. Choose firmware then click “Finish”. If you don’t have the firmware you are selected, the system will process to download the firmware.



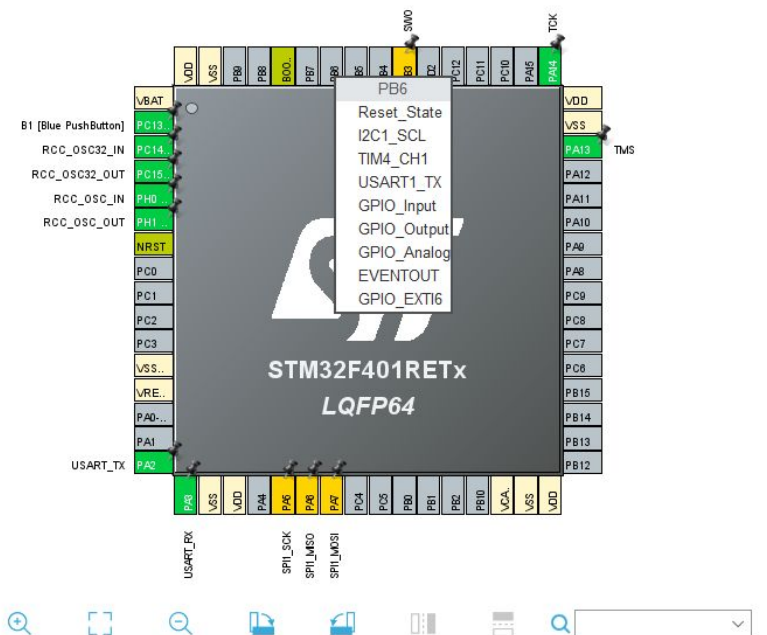
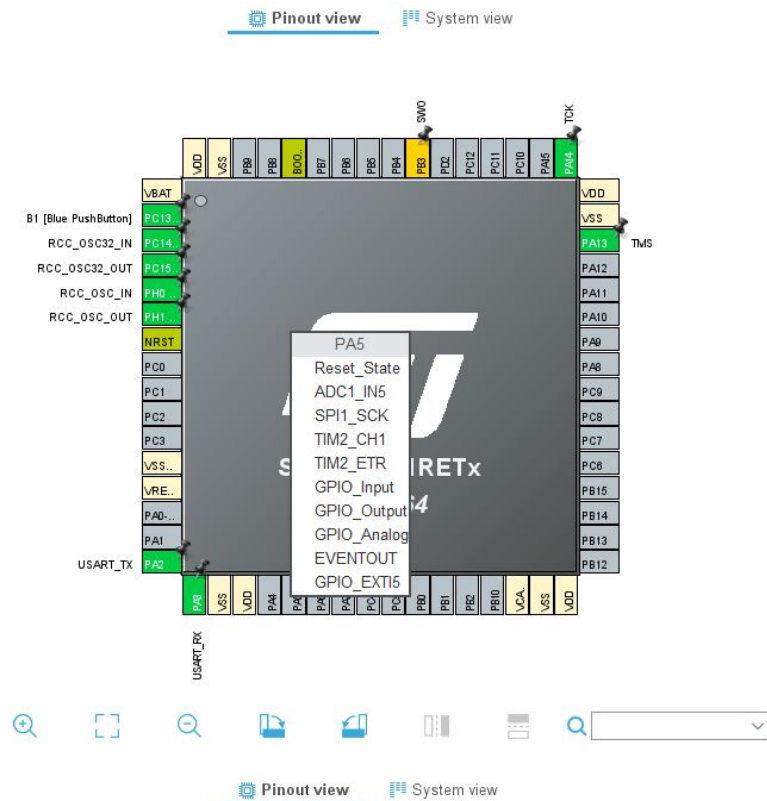
7. Initialize all peripherals with their default Mode? click “Yes”.



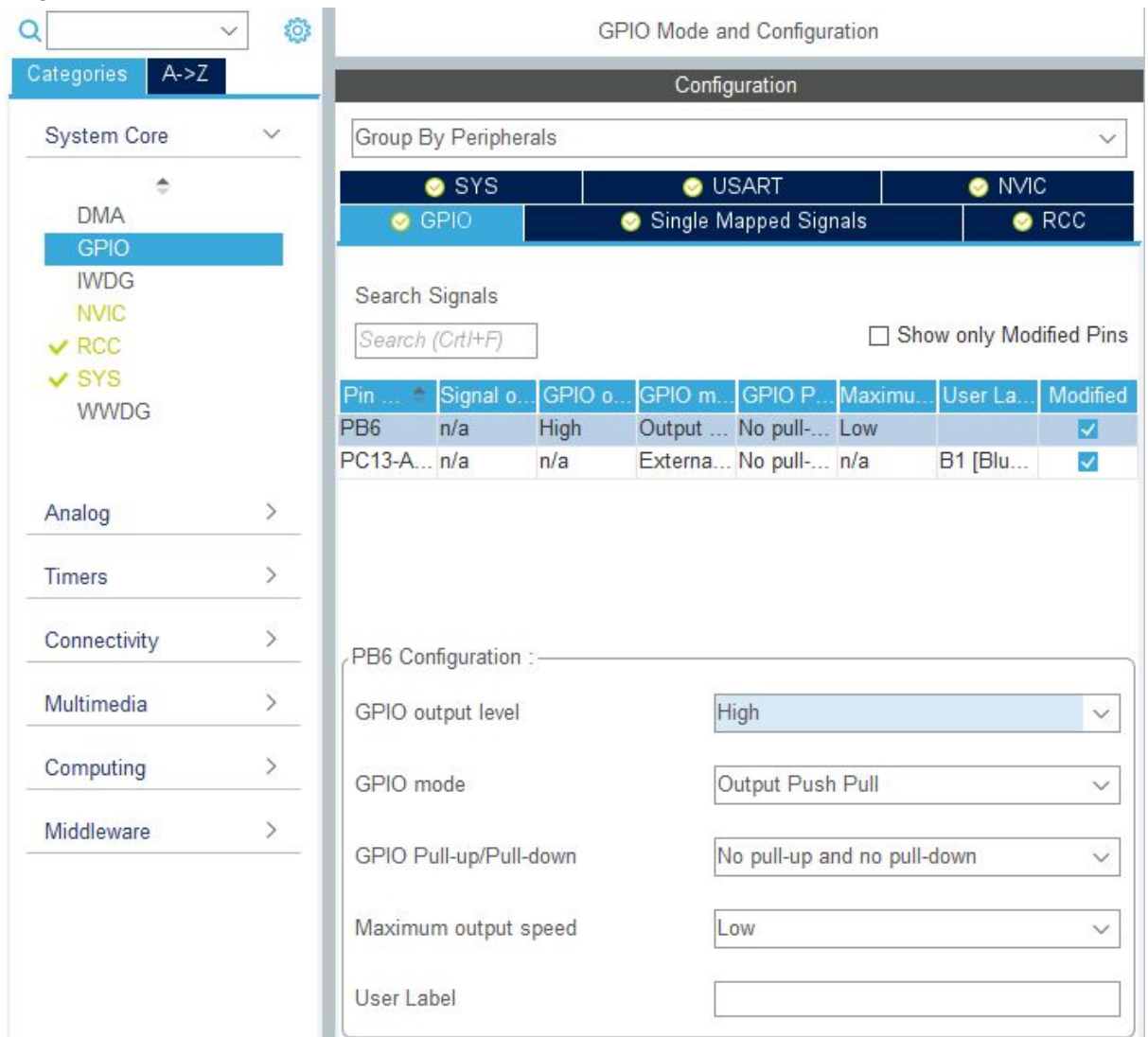
8. Open the STM32CubeMx for pinout define.



9. In pinout view, choose the SPI1_SCK at PA5, SPI1_MISO at PA6, SPI1_MOSI at PA7, GPIO_Output at PB6.



10. Then click “System Core” for further GPIO setting, change “GPIO output level” to “High”.



The screenshot shows the STM32CubeIDE interface for configuring GPIO pins. The left sidebar displays the 'System Core' category, with 'GPIO' selected. The main window shows the 'GPIO Mode and Configuration' dialog, specifically the 'Configuration' tab. A table lists the configured pins, and the 'PB6' pin is selected. Below the table, the 'PB6 Configuration' details are shown, including the 'GPIO output level' set to 'High'.

Pin	Signal	GPIO o...	GPIO m...	GPIO P...	Maximu...	User La...	Modified
PB6	n/a	High	Output ...	No pull-...	Low		<input checked="" type="checkbox"/>
PC13-A...	n/a	n/a	Externa...	No pull-...	n/a	B1 [Blu...	<input checked="" type="checkbox"/>

PB6 Configuration :

- GPIO output level: High
- GPIO mode: Output Push Pull
- GPIO Pull-up/Pull-down: No pull-up and no pull-down
- Maximum output speed: Low
- User Label:

11. Move to "TIM2" in "Timers". Define "Clock Source" as "Internal Clock" and input "0xFFFFFFFF" in "Counter Period".

The screenshot displays the STM32CubeMX configuration tool for the TIM2 timer. On the left, the 'Timers' category is expanded, and TIM2 is selected. The main panel is titled 'TIM2 Mode and Configuration' and is divided into two sections: 'Mode' and 'Configuration'.

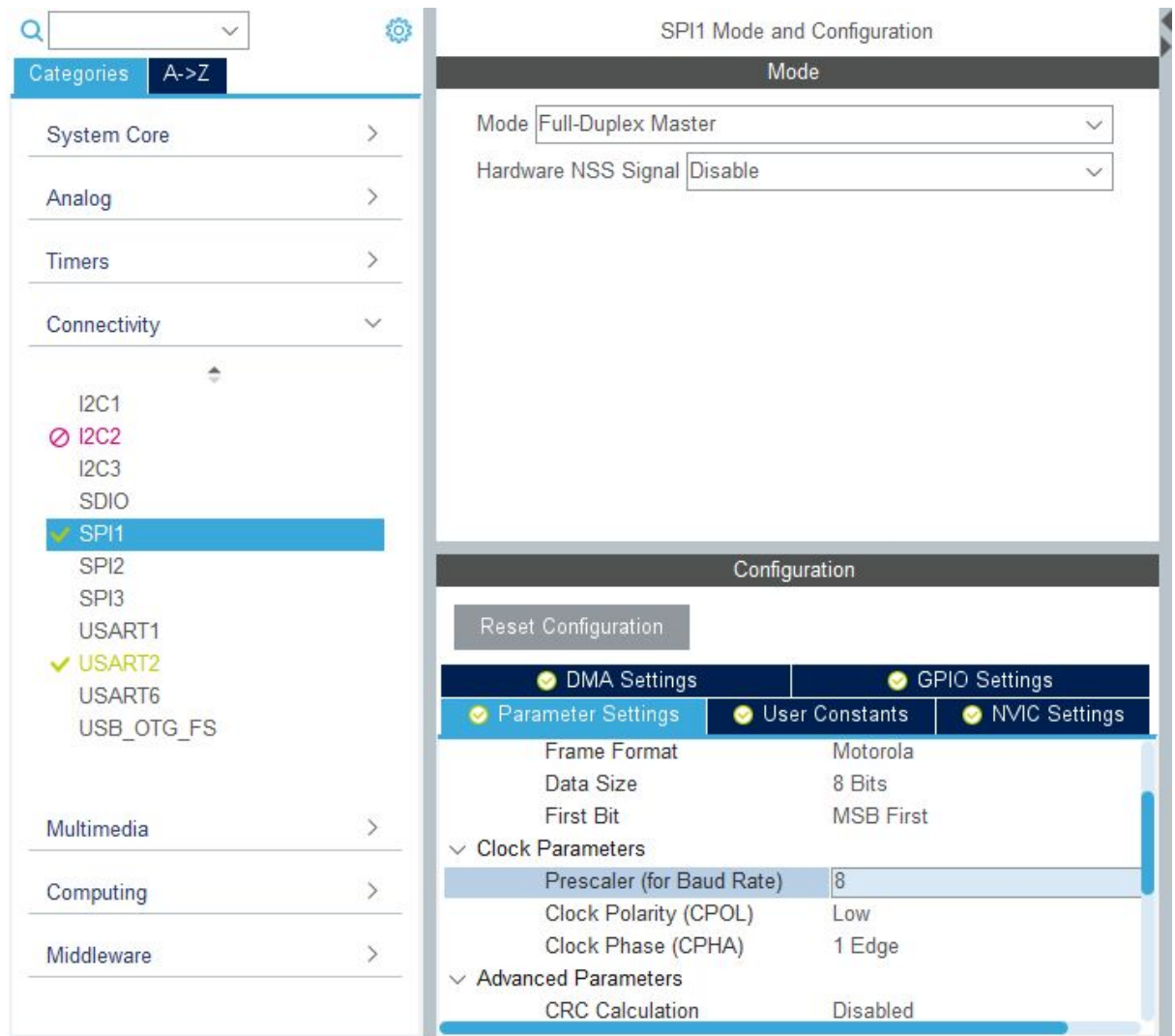
Mode Section:

- Slave Mode: Disable
- Trigger Source: Disable
- Clock Source: Internal Clock
- Channel1: Disable
- Channel2: Disable
- Channel3: Disable
- Channel4: Disable
- Combined Channels: Disable
- ☐ Use ETR as Clearing Source
- ☐ XOR activation

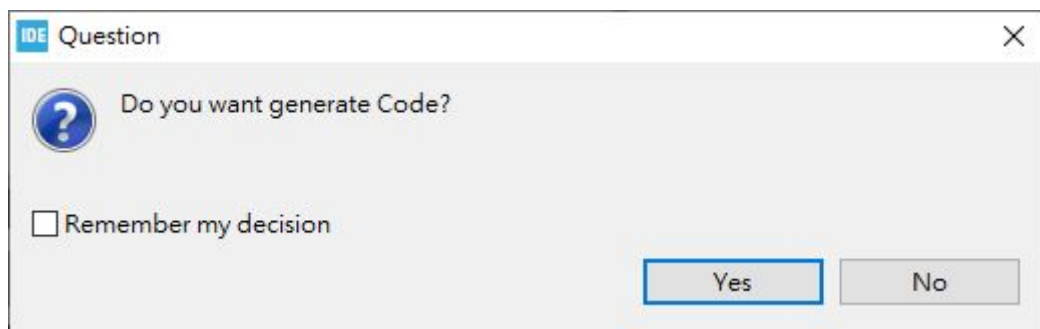
Configuration Section:

- Reset Configuration button
- Tabs: User Constants, NVIC Settings, DMA Settings, and Parameter Settings (selected).
- Configure the below parameters :
- Search (Ctrl+F) input field
- Counter Settings expanded:
 - Prescaler (PSC - 16 bits val...): 0
 - Counter Mode: Up
 - Counter Period (AutoReload...): 0xffffffff
 - Internal Clock Division (CKD): No Division
 - auto-reload preload: Disable

12. Switch to “SPI1” in “Connectivity”, choose mode into “Full-Duplex Master” and set “8” for “Prescaler (for Baud Rate)”.



13. After finishing all setting, press the button to save and generate the code.



14. If you find the following code setup by system in your Private variables means you got all the sample code function needed.

```
/* Private variables -----*/
SPI_HandleTypeDef hspi1;

TIM_HandleTypeDef htim2;

UART_HandleTypeDef huart2;
```

15. Open “main.h” and add sample code show as below:

```
/* USER CODE BEGIN Includes */
#include "platform.h"
#include "EcmUsrDriver.h"
/* USER CODE END Includes */

/* Exported macro -----*/
/* USER CODE BEGIN EM */
#ifndef PRINTF
#define PRINTF( str, ... ) \
do{ \
    int n; \
    n = sprintf( printbuf, (str), ##_VA_ARGS_ ); \
    HAL_UART_Transmit( &huart2, (uint8_t *)printbuf, n, 0xffffffff); \
}while(0)
#endif
#ifndef GETCHAR
#define GETCHAR userGetchar
#endif
/* USER CODE END EM */

/* USER CODE BEGIN EFP */
extern UART_HandleTypeDef huart2;
extern char printbuf[];
/* USER CODE END EFP */
```

16. Open “main.c” and add setting show as following pictures:

```
/* USER CODE BEGIN PFP */
char printbuf[128];
int main_ini(void);
/* USER CODE END PFP */

/* USER CODE BEGIN 2 */
main_ini();
/* USER CODE END 2 */

/* USER CODE BEGIN TIM2_Init 2 */
HAL_TIM_Base_Start(&htim2);
/* USER CODE END TIM2_Init 2 */
```

17. Copy the files: EcmDriver.h, EcmUsrDriver.h, PdoDefine.h, and platform.h into your corresponding “Inc” file.
18. Copy the files: crc32.c, EcmUsrDriver.c, main_ini.c, and platform.c into your corresponding “Src” file.
19. Ready to run sample code.