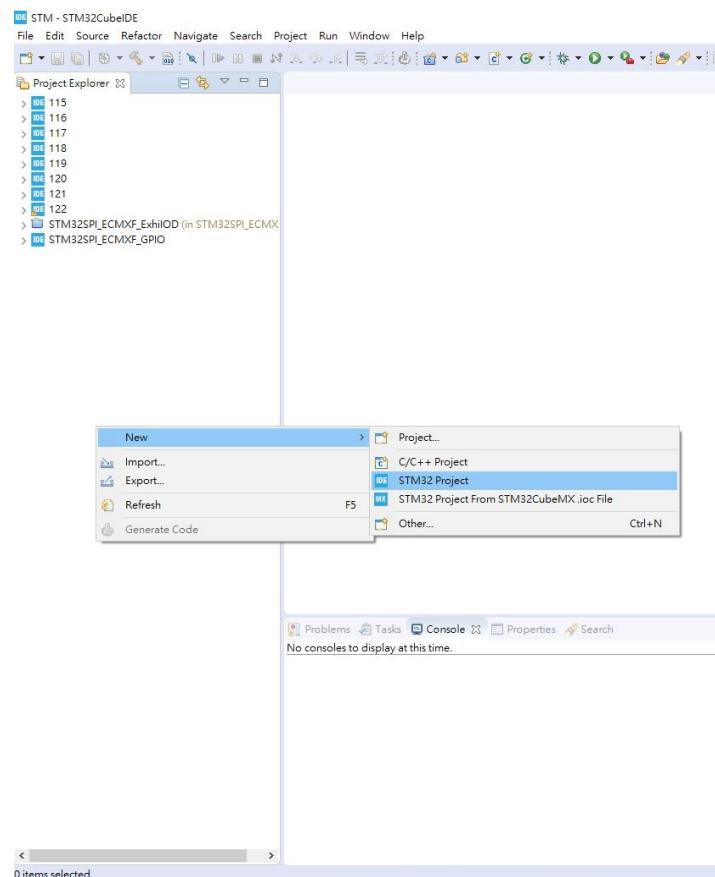
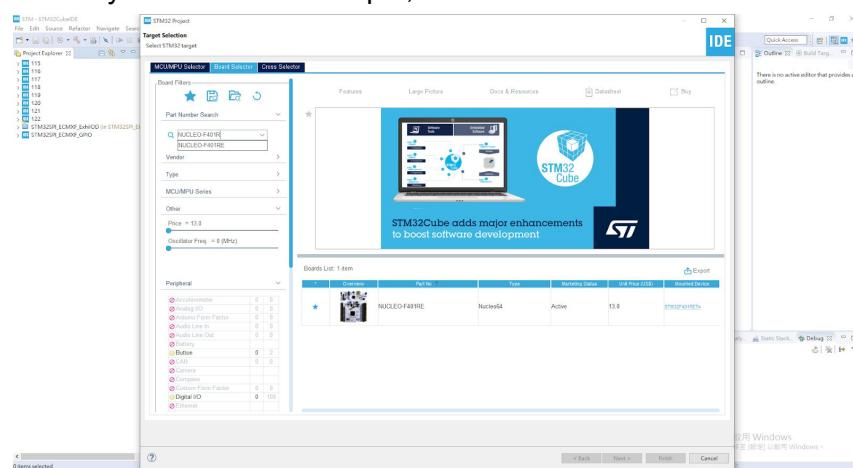


STM32 Setup Instruction (NUCLEO-F401RE)

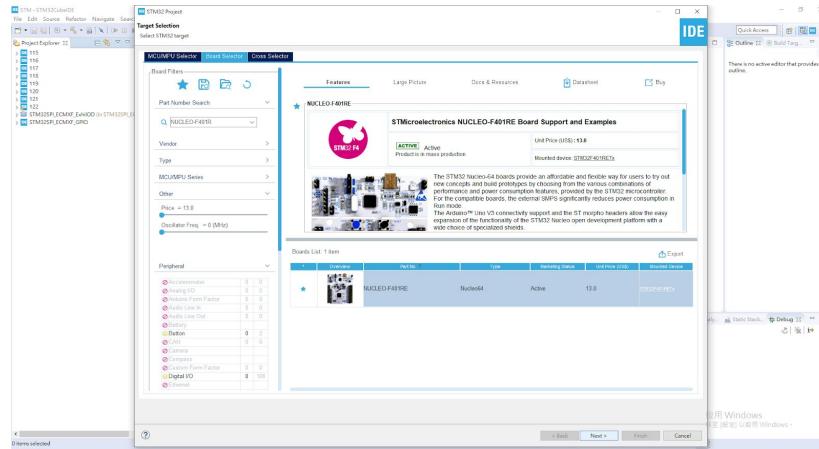
1. Open STM32Cube IDE.
2. Right click and select “STM32 Project”.



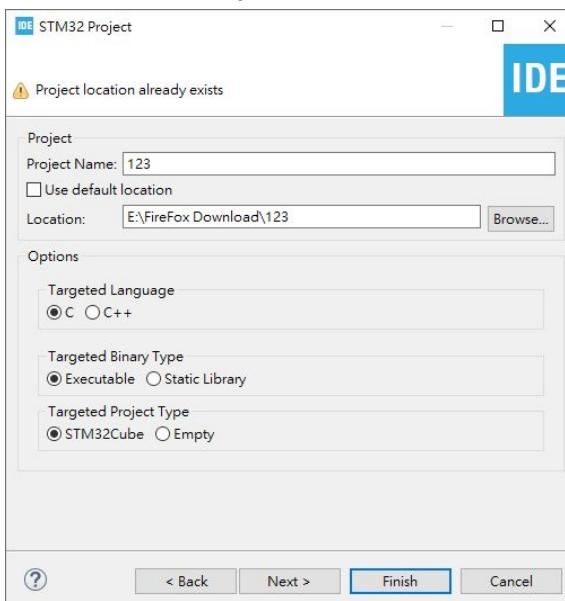
3. Select the board you have. For example, the selected board is NUCLEO-F401RE.



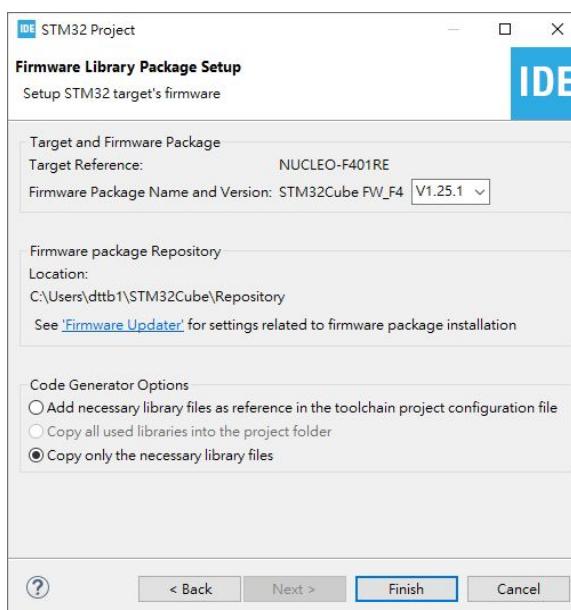
4. Then click “Next” to continue setup.



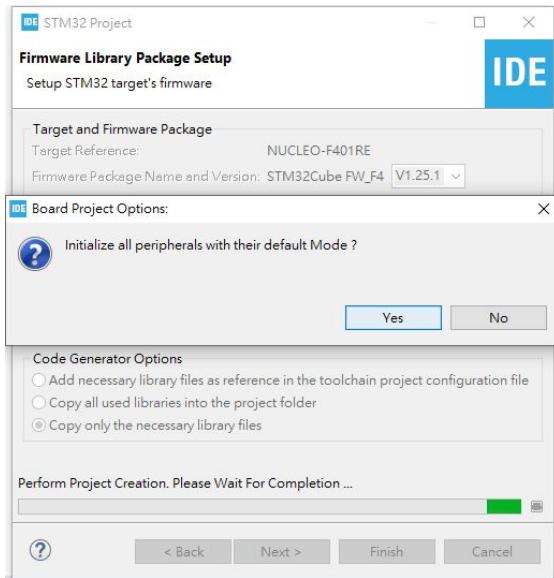
5. Insert the project name and decide project location.



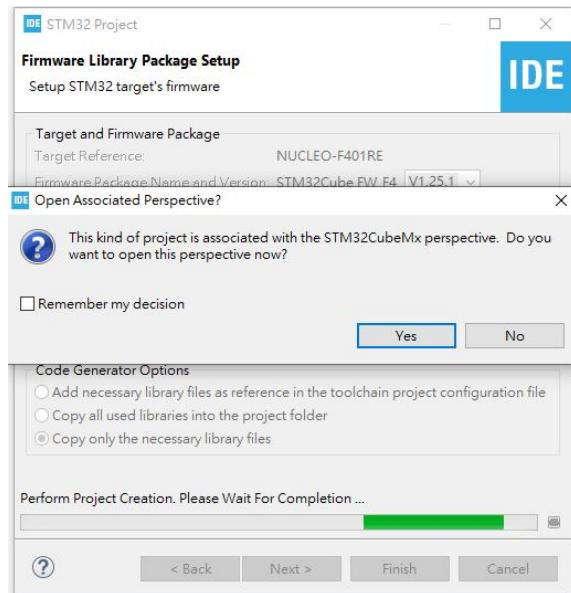
6. Choose firmware then click “Finish”. If you don’t have the firmware you are selected, the system will process to download the firmware.



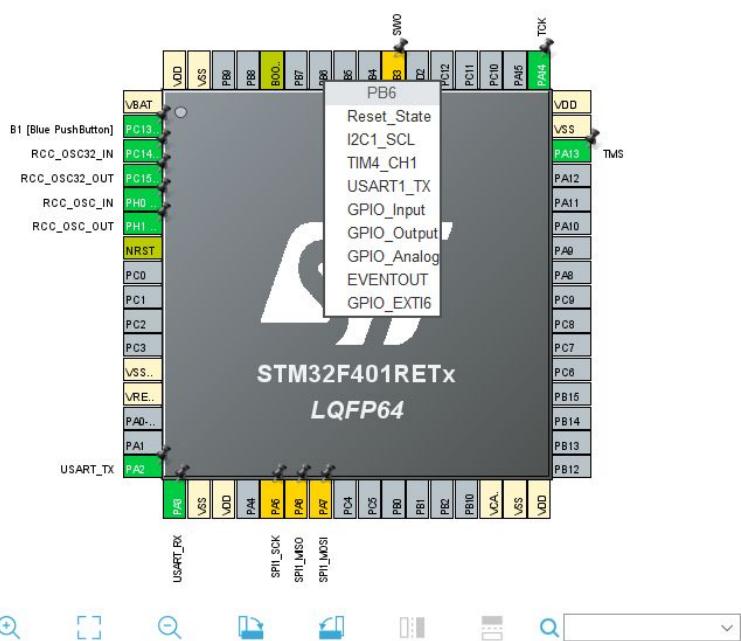
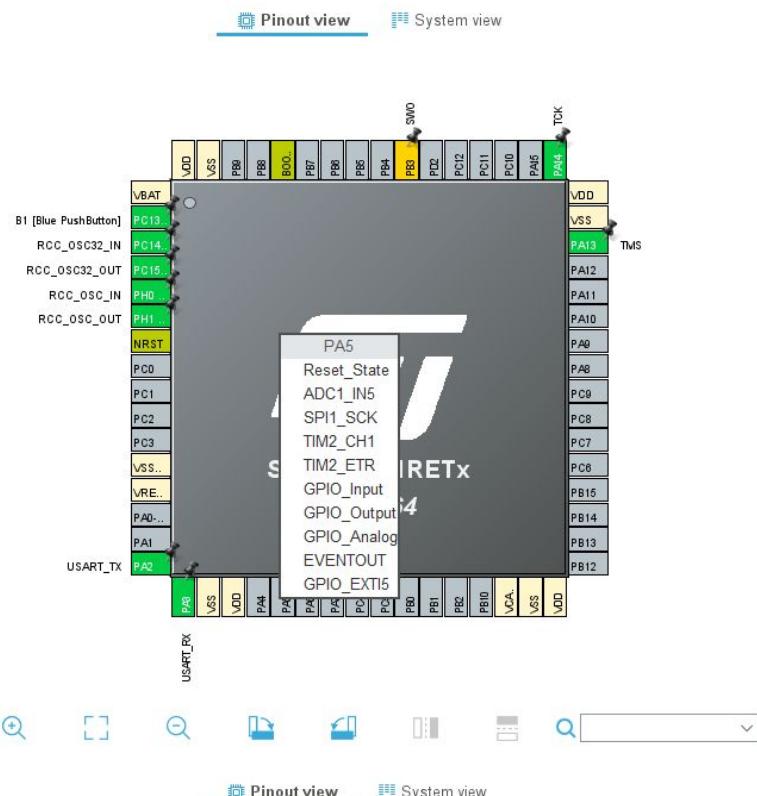
7. Initialize all peripherals with their default Mode? click "Yes".



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 STM32CubeMX software interface. On the left, a sidebar lists categories: System Core, DMA, GPIO (selected), IWDG, NVIC, RCC, SYS, and WWDG. The main area is titled "GPIO Mode and Configuration" and "Configuration". It includes a "Group By Peripherals" dropdown and checkboxes for SYS, USART, NVIC, GPIO, Single Mapped Signals, and RCC. A "Search Signals" input field and a "Show only Modified Pins" checkbox are also present. A table lists pins PB6 and PC13-A... with their current configurations. Below this is a "PB6 Configuration" section with five dropdown menus: "GPIO output level" (set to "High"), "GPIO mode" (set to "Output Push Pull"), "GPIO Pull-up/Pull-down" (set to "No pull-up and no pull-down"), "Maximum output speed" (set to "Low"), and "User Label" (empty). The "GPIO output level" dropdown is highlighted.

Pin ...	Signal o...	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	Extern...	No pull-...	n/a	B1 [Blu...	<input checked="" type="checkbox"/>

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

TIM2 Mode and Configuration

Mode

- 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

Reset Configuration

User Constants NVIC Settings DMA Settings

Parameter Settings

Configure the below parameters :

Search (Ctrl+F) ⏪ ⏩ ⓘ

Counter Settings

- 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).

SPI1 Mode and Configuration

Mode: Full-Duplex Master

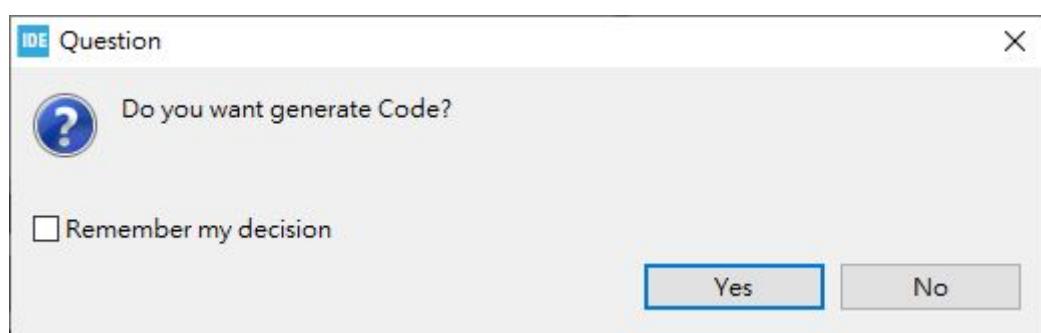
Hardware NSS Signal: Disable

Configuration

Prescaler (for Baud Rate): 8

Frame Format	Motorola
Data Size	8 Bits
First Bit	MSB First
Clock Parameters	
Prescaler (for Baud Rate)	8
Clock Polarity (CPOL)	Low
Clock Phase (CPHA)	1 Edge
Advanced Parameters	
CRC Calculation	Disabled

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.