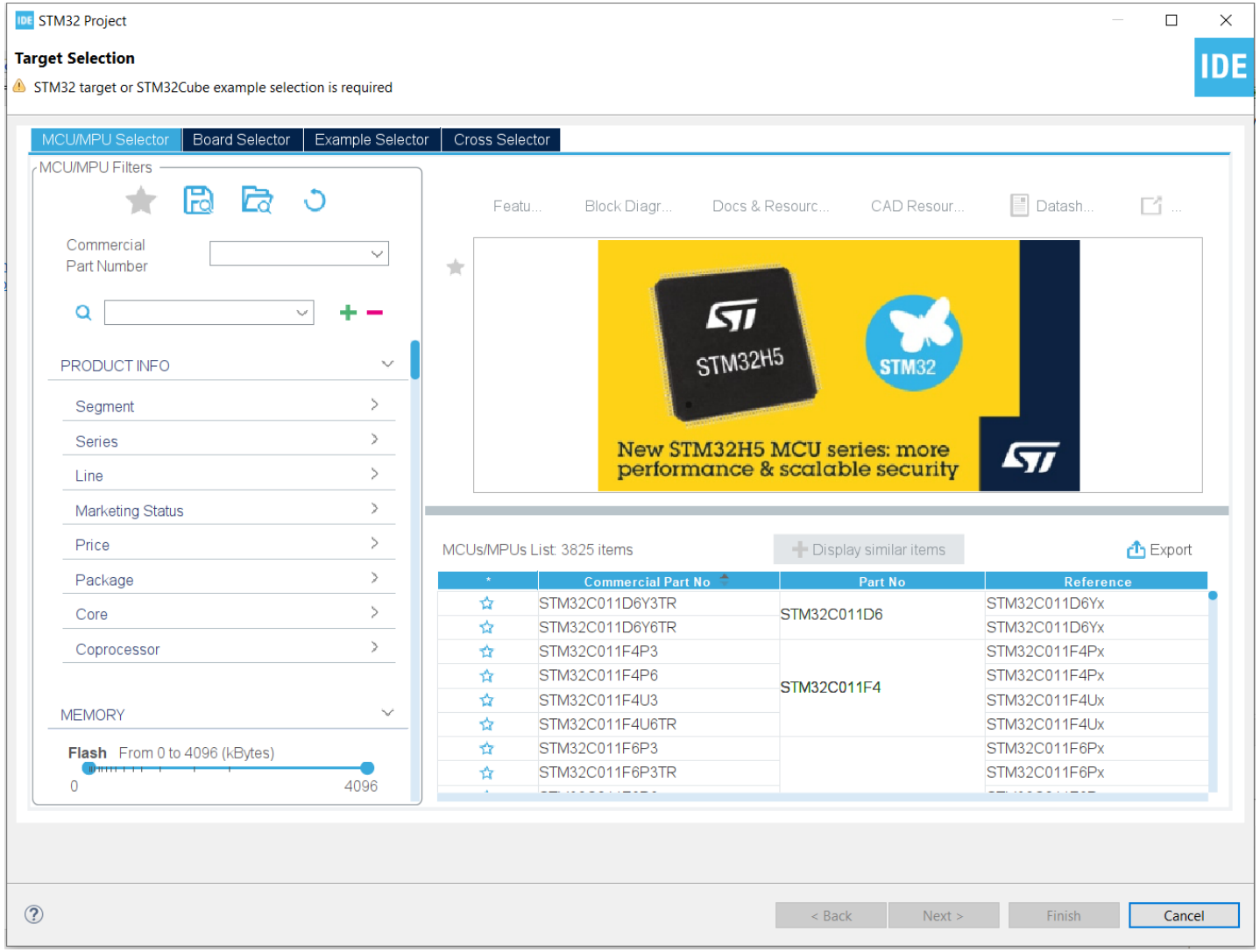
UCSD Embedded C Assignment 5

By

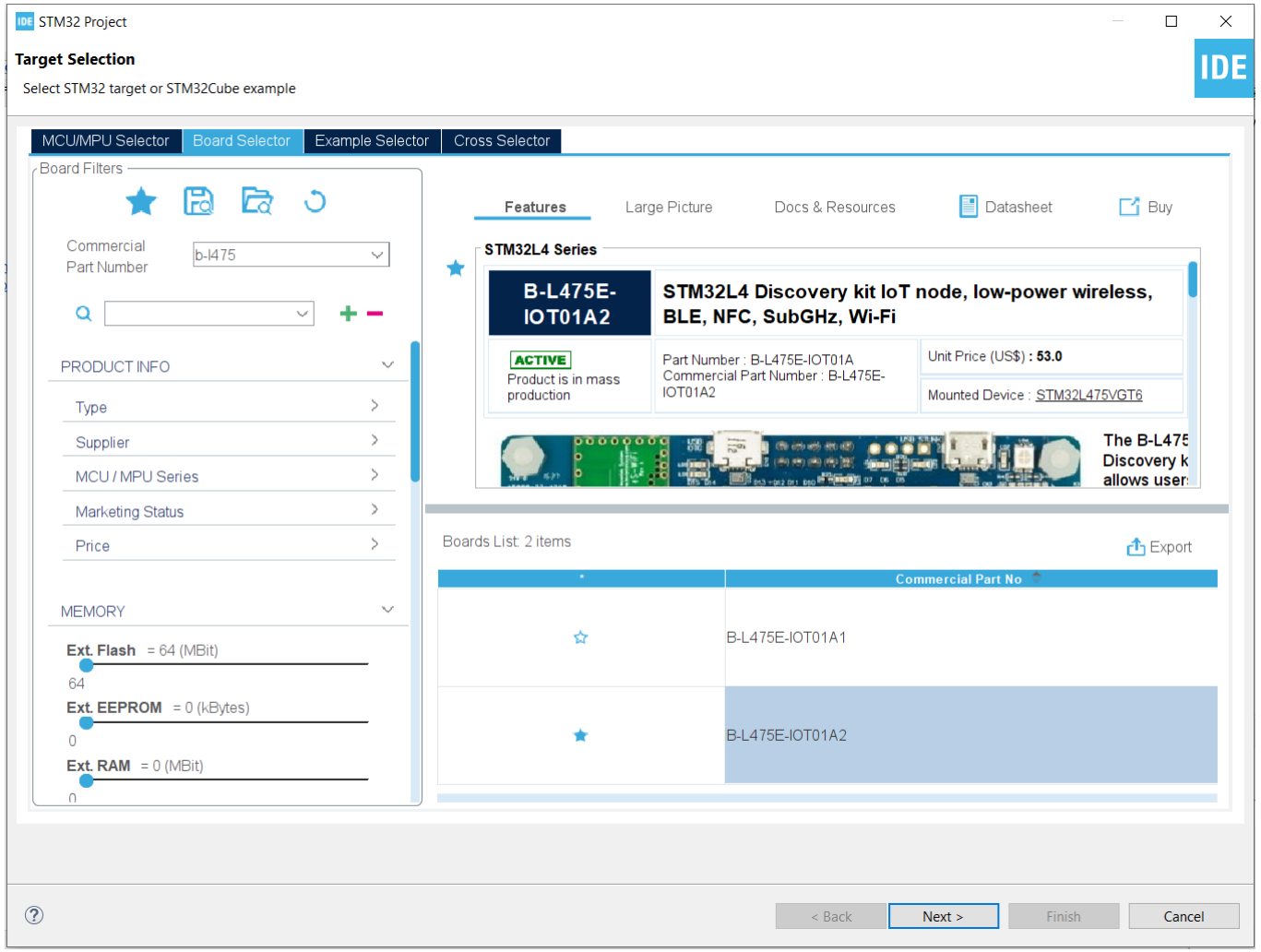
Hsuankai Chang

hsuankac@umich.edu

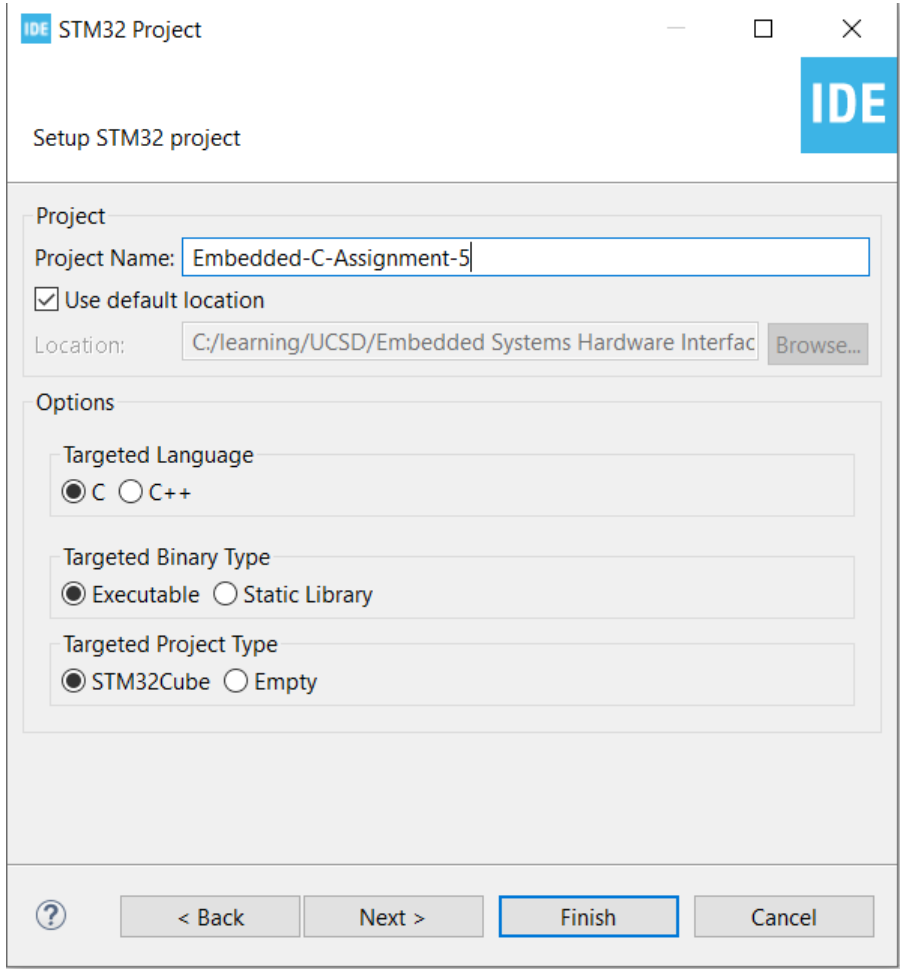
Step 1. Startup STM32CubeIDE and create new STM32 project



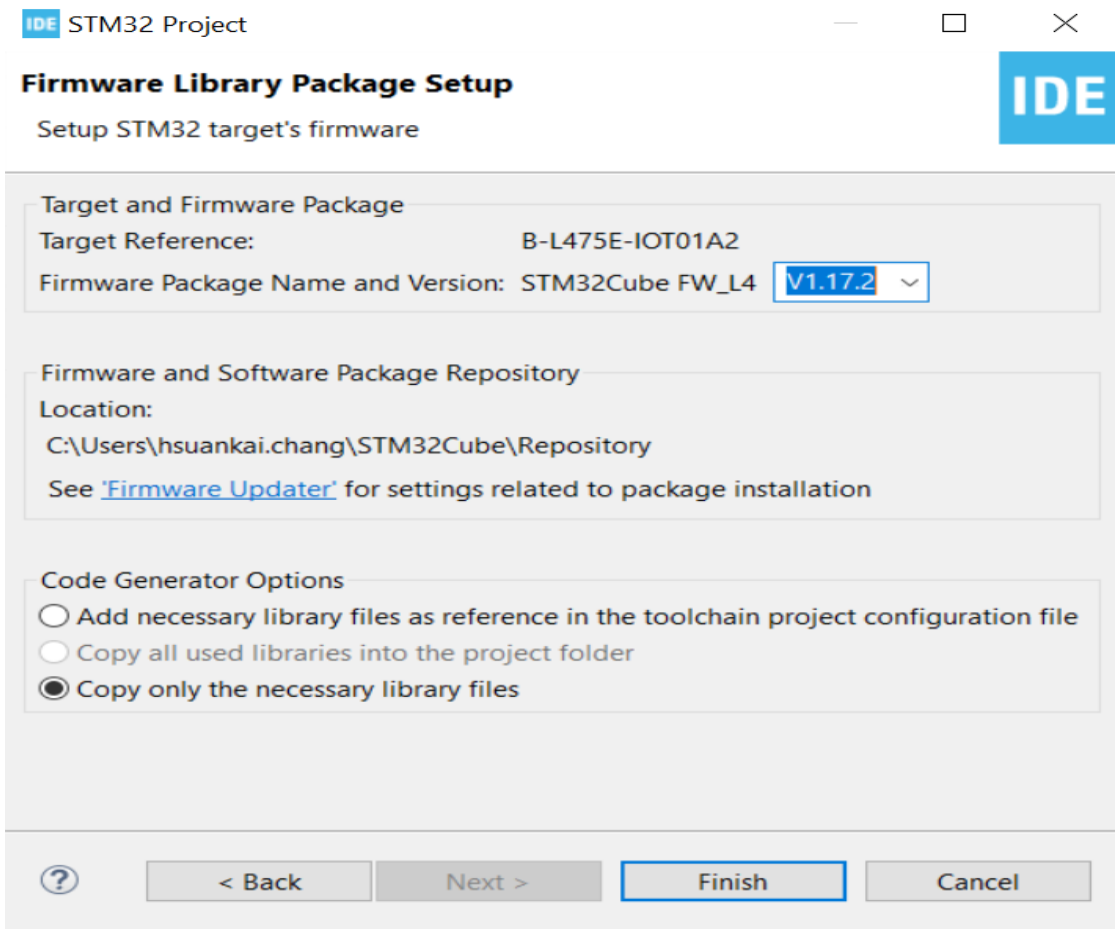
Step 2. Access board selector and type in the board you use, click Next



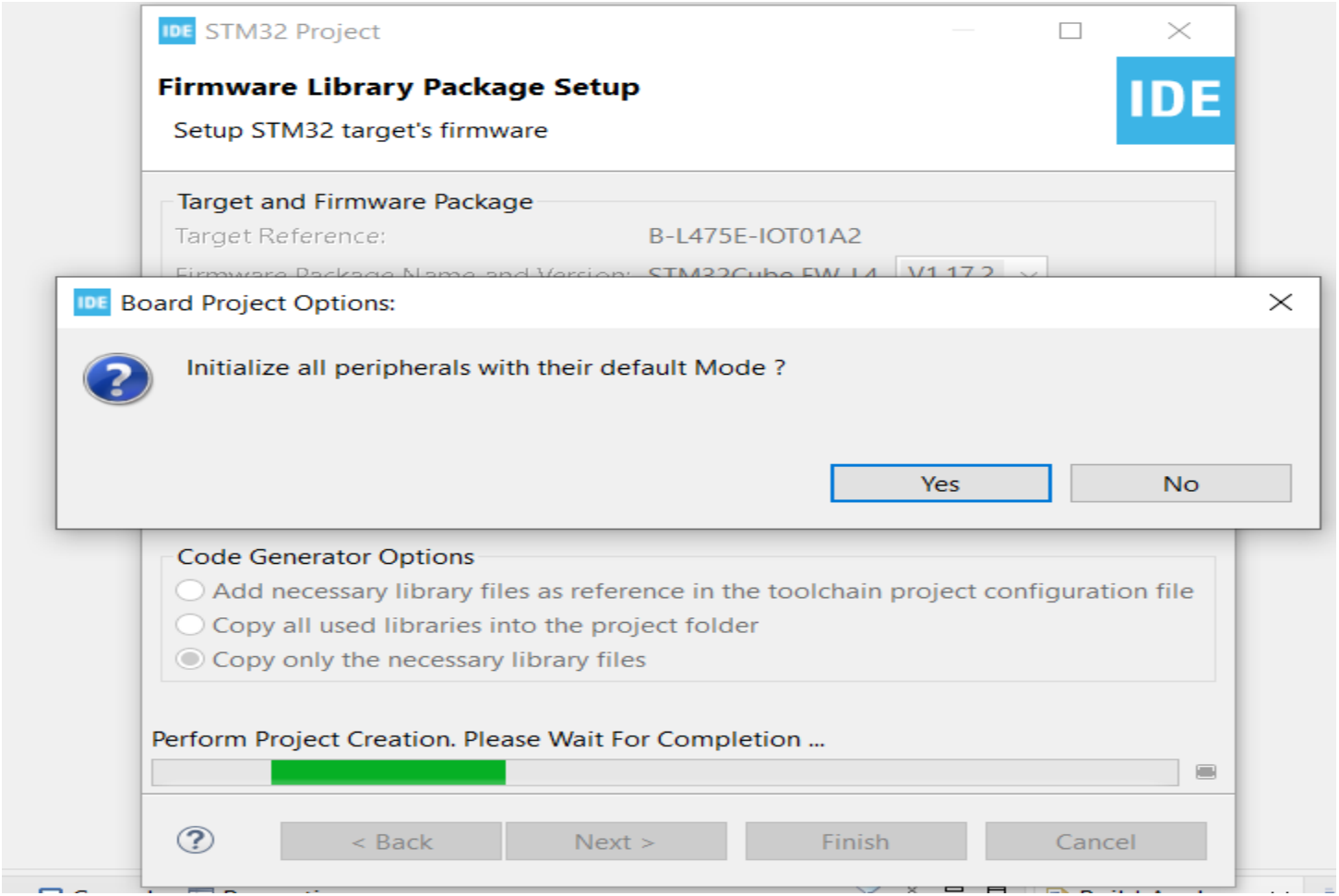
Step 3. Enter the project name then click Next



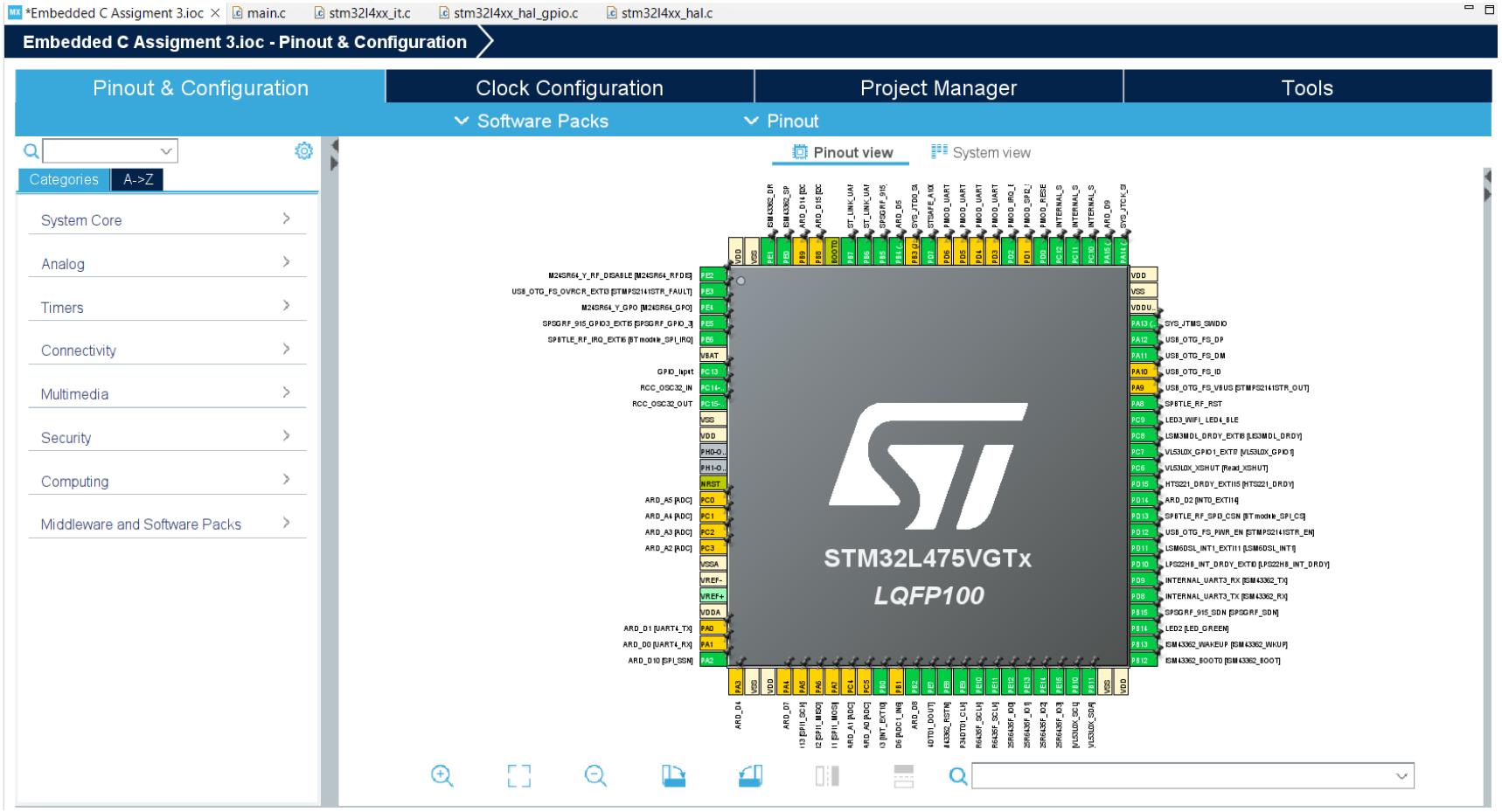
Step 4. See the firmware package name and version



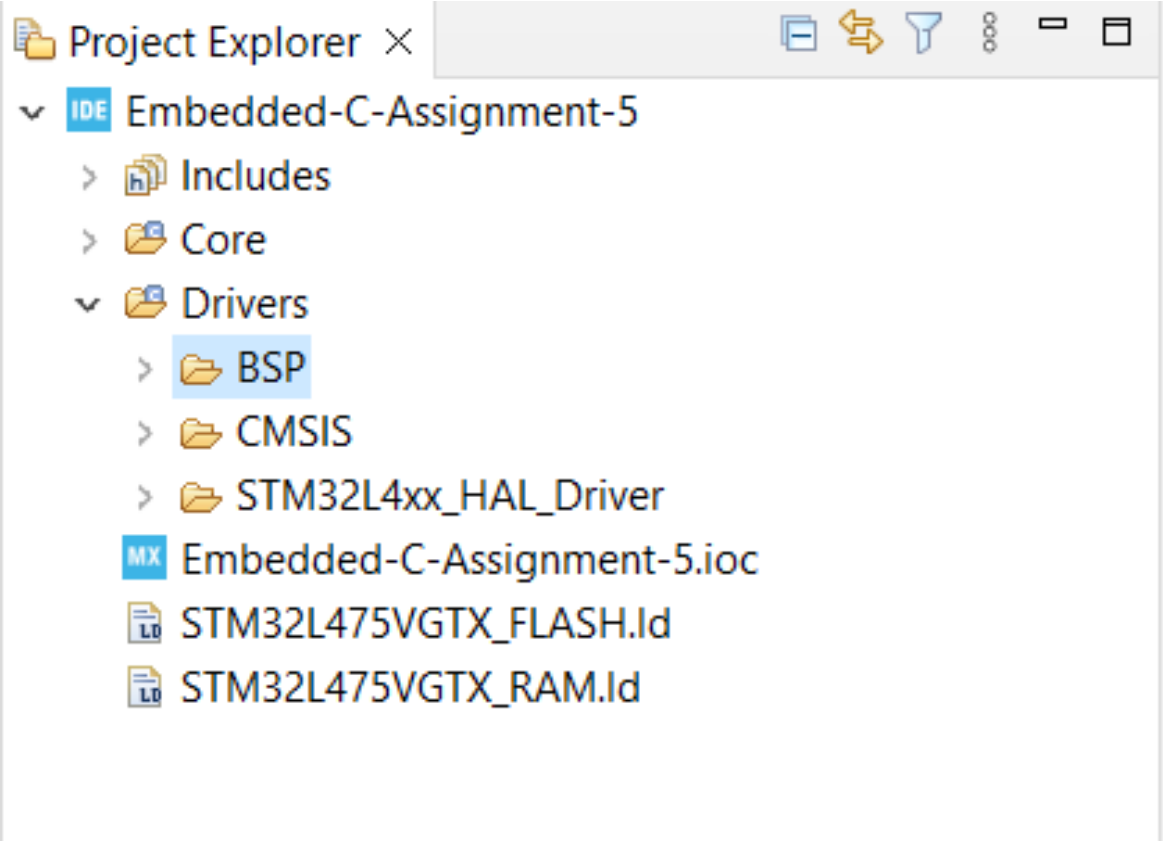
Step 5. Click yes to initialize all peripherals to default



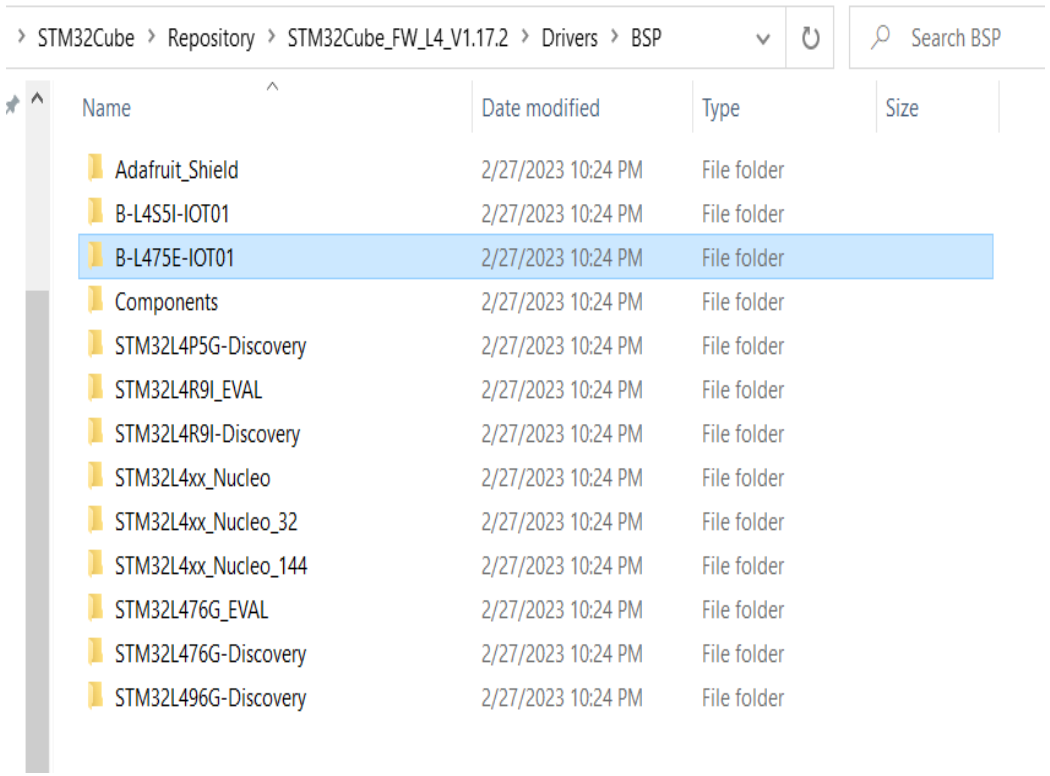
Step 6. When in .ioc file, click Pinout & Configurations. Use the default setting then generate code

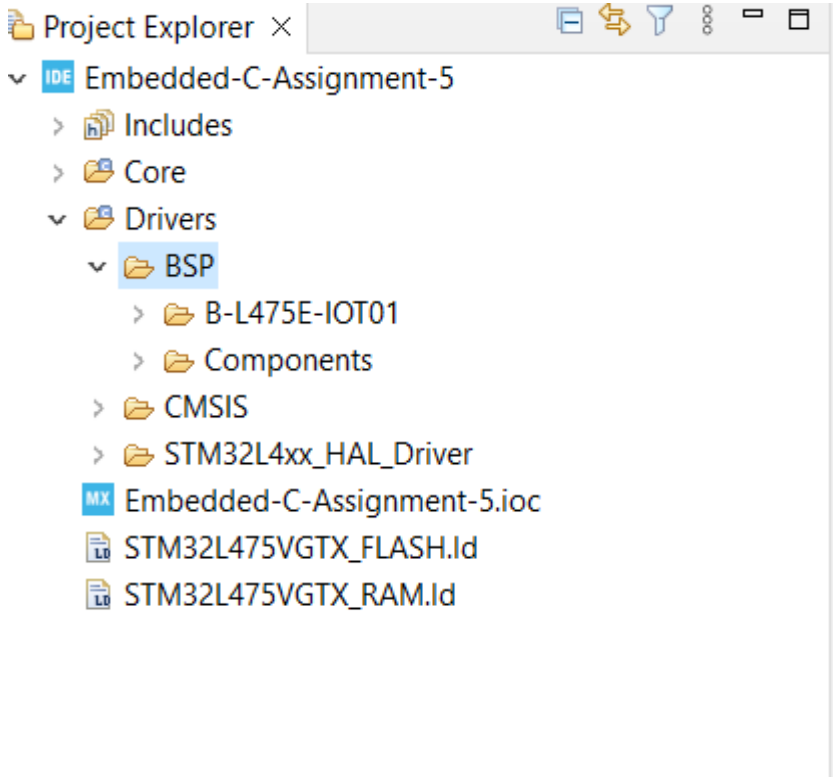


Step 7. Create BSP folder in project under drivers

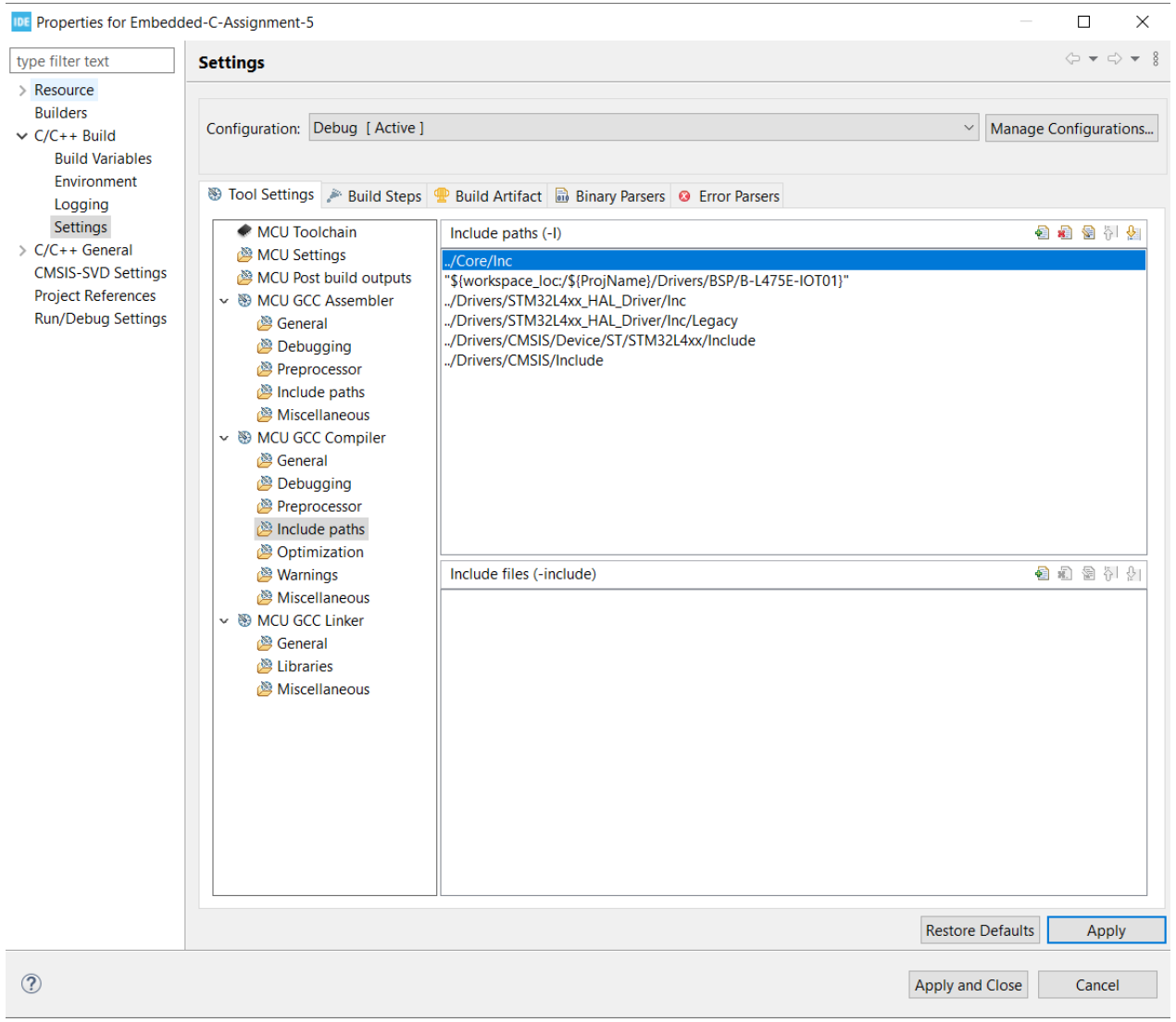


Step 8. Find the BSP location in the system, then copy B-L475E-IOT01 and Components folder into BSP folder in the project





Step 9. Add the include path



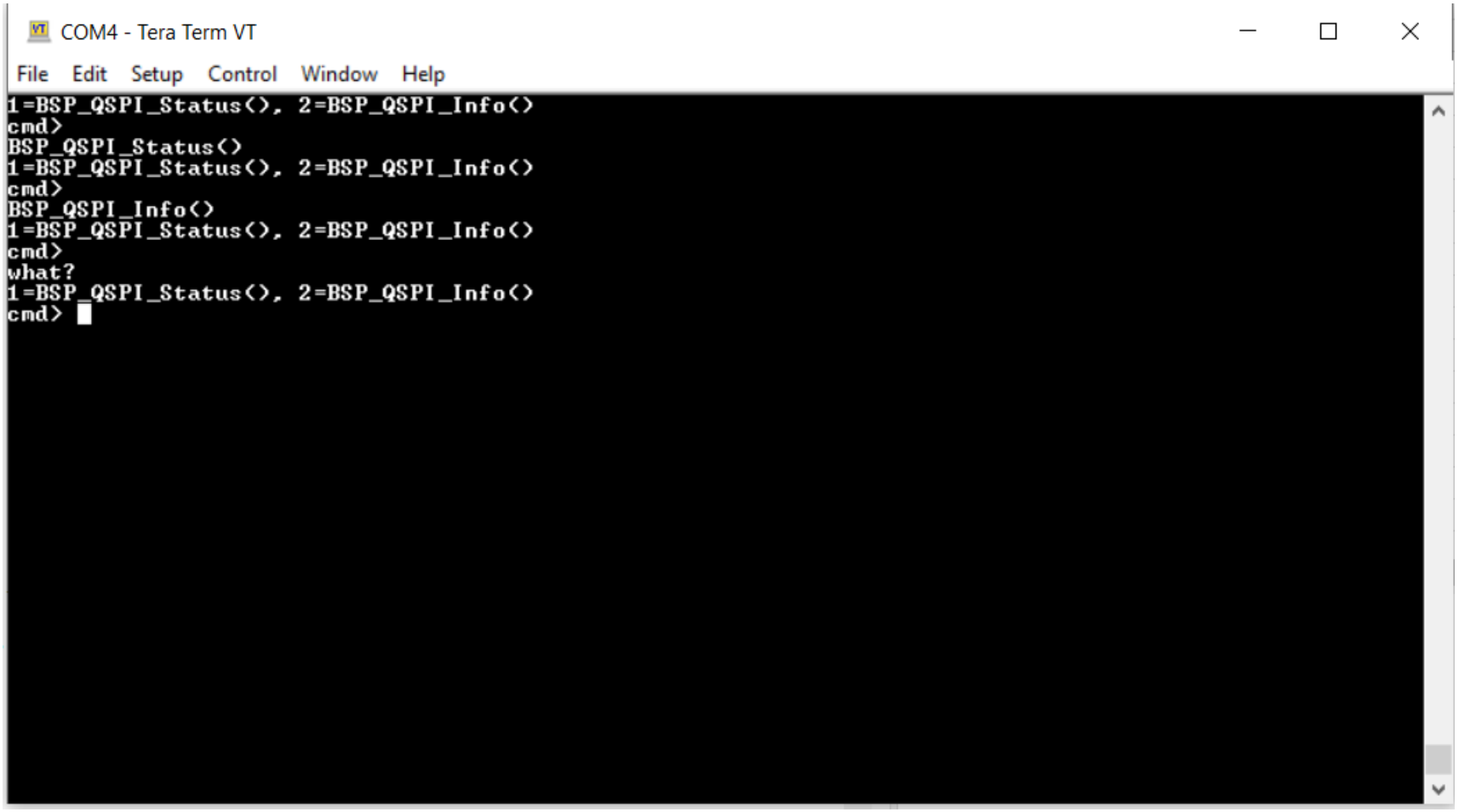
Step 10. User Story 1. CLI. Create a CLI (Command Line Interface) on UART1

*main.c × stm32l475e_iot01.h

```c
  /* Infinite loop */
  /* USER CODE BEGIN WHILE */
  while (1)
  {
    /* USER CODE END WHILE */

    /* USER CODE BEGIN 3 */
    // Issue command prompt
    char *prompt = "1=BSP_QSPI_Status(), 2=BSP_QSPI_Info()\n\rcmd> ";
    HAL_UART_Transmit(&huart1, (uint8_t*) prompt, strlen(prompt), 1000);
    // Wait for a single number entry
    char ch;
    HAL_UART_Receive(&huart1, (uint8_t*)&ch, 1, HAL_MAX_DELAY);
    char *msg = "what?";
    switch(ch)
    {
        case '1':
        {
            msg = "\r\nBSP_QSPI_Status()\r\n";
            HAL_UART_Transmit(&huart1, (uint8_t*) msg, strlen(msg), 1000);
            do_qspi_status();
            break;
        }
        case '2':
        {
            msg = "\r\nBSP_QSPI_Info()\r\n";
            HAL_UART_Transmit(&huart1, (uint8_t*) msg, strlen(msg), 1000);
            do_qspi_info();
            break;
        }
        default:
        {
            msg = "\r\nwhat?\r\n";
            HAL_UART_Transmit(&huart1, (uint8_t*) msg, strlen(msg), 1000);
        }
    }
  }
  /* USER CODE END 3 */
```

```c
/* USER CODE END Header */
/* Includes ------------------------
#include "main.h"

/* Private includes ----------------
/* USER CODE BEGIN Includes */
#include "stm32l475e_iot01.h"
#include "stm32l475e_iot01_qspi.h"
#include <string.h>
#include <stdio.h>
/* USER CODE END Includes */

void do_qspi_status()
{

}

void do_qspi_info()
{

}
/* USER CODE END 0 */
```

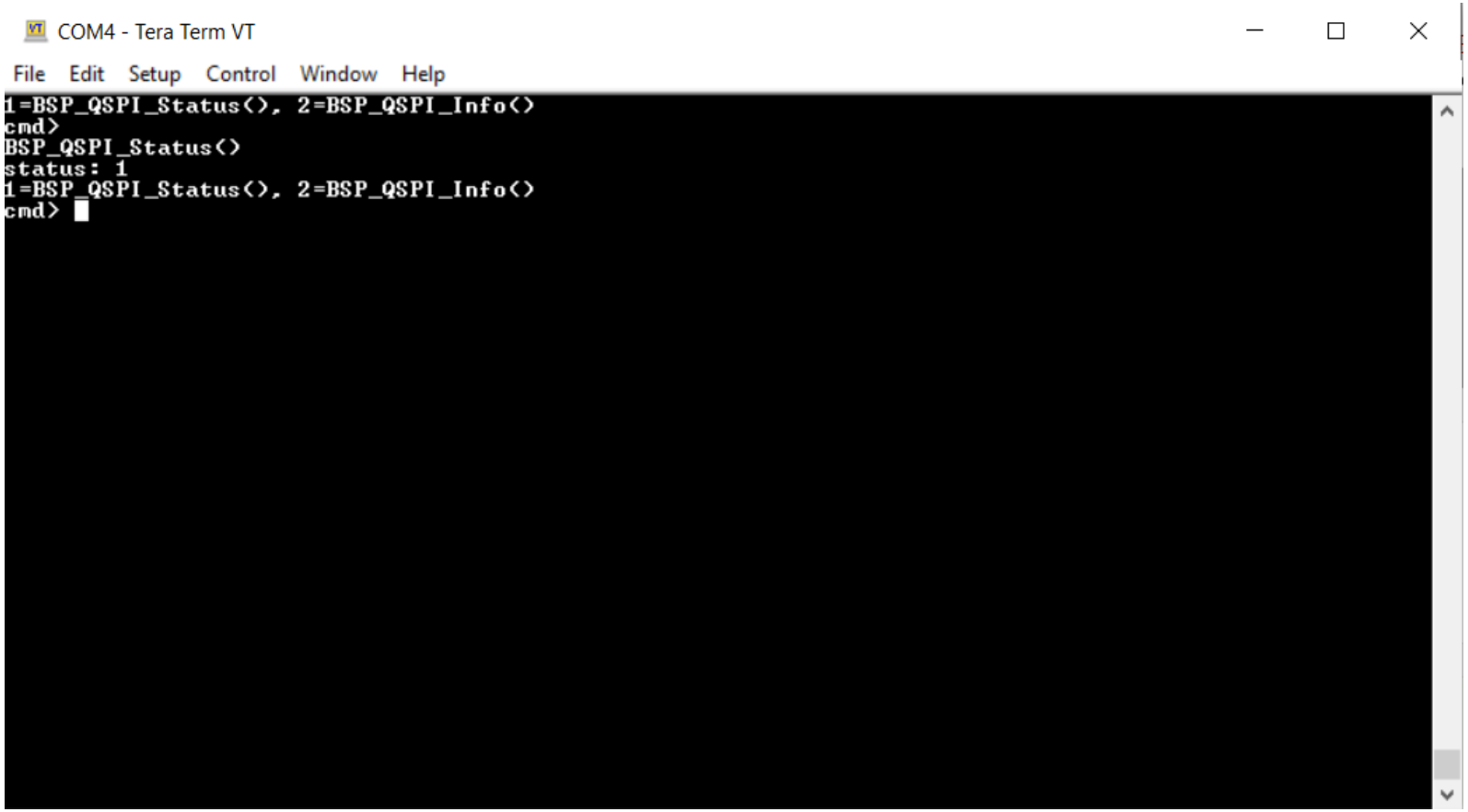
Step 11. User Story 1: Build and run the code, test is successful



Step 12. User Story 2. BSP_QSPI_GetStatus(). When the user selects 1, display the results of calling BSP_QSPI_GetStatus() on the console.

```c
void do_qspi_status()
{
    uint8_t status = BSP_QSPI_GetStatus();
    char buf[100];
    snprintf(buf, sizeof(buf), "status: %d\r\n", status);

    HAL_UART_Transmit(&huart1, (uint8_t*)buf, strlen(buf), 1000);
}
```

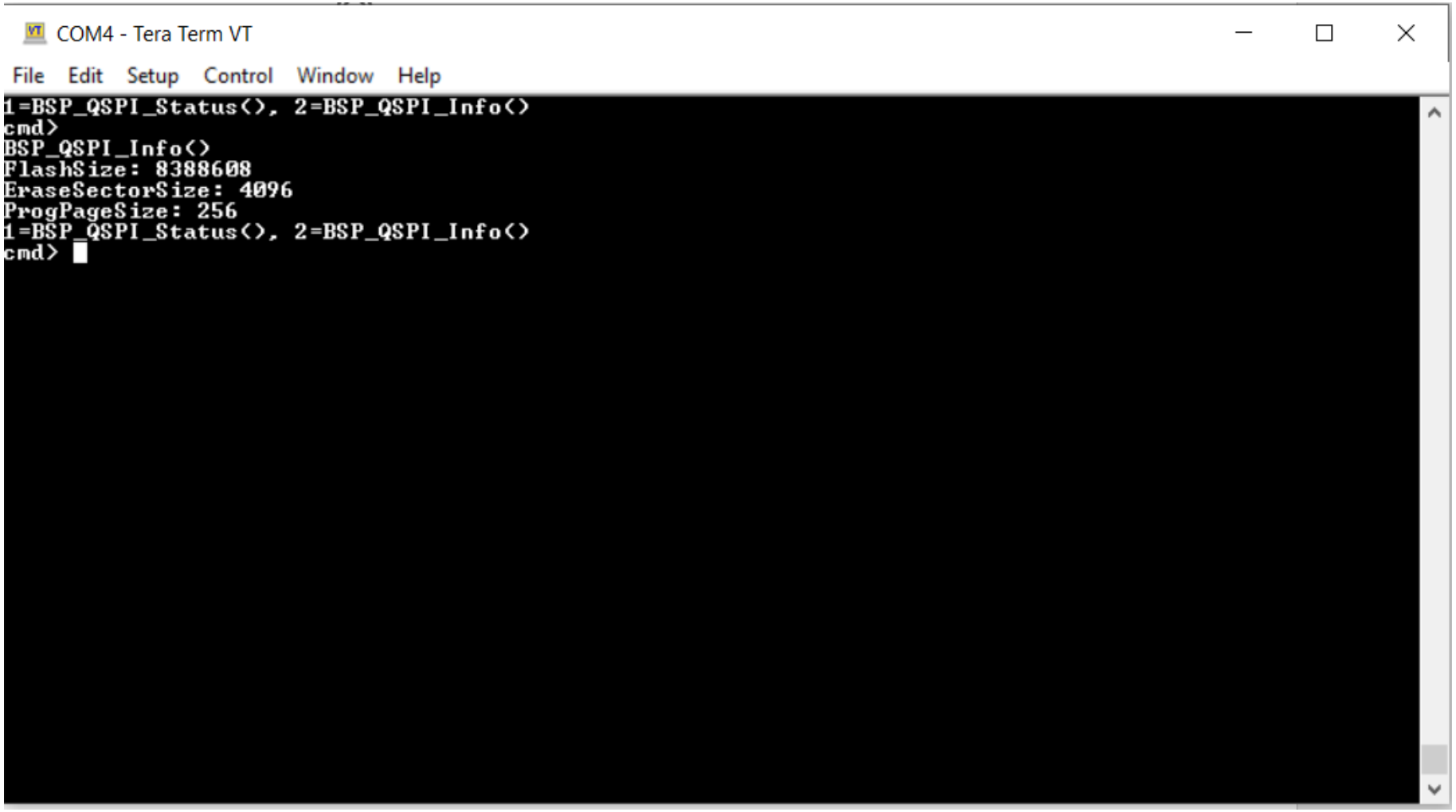
Step 13. User Story 2: Build and run the code, test is successful



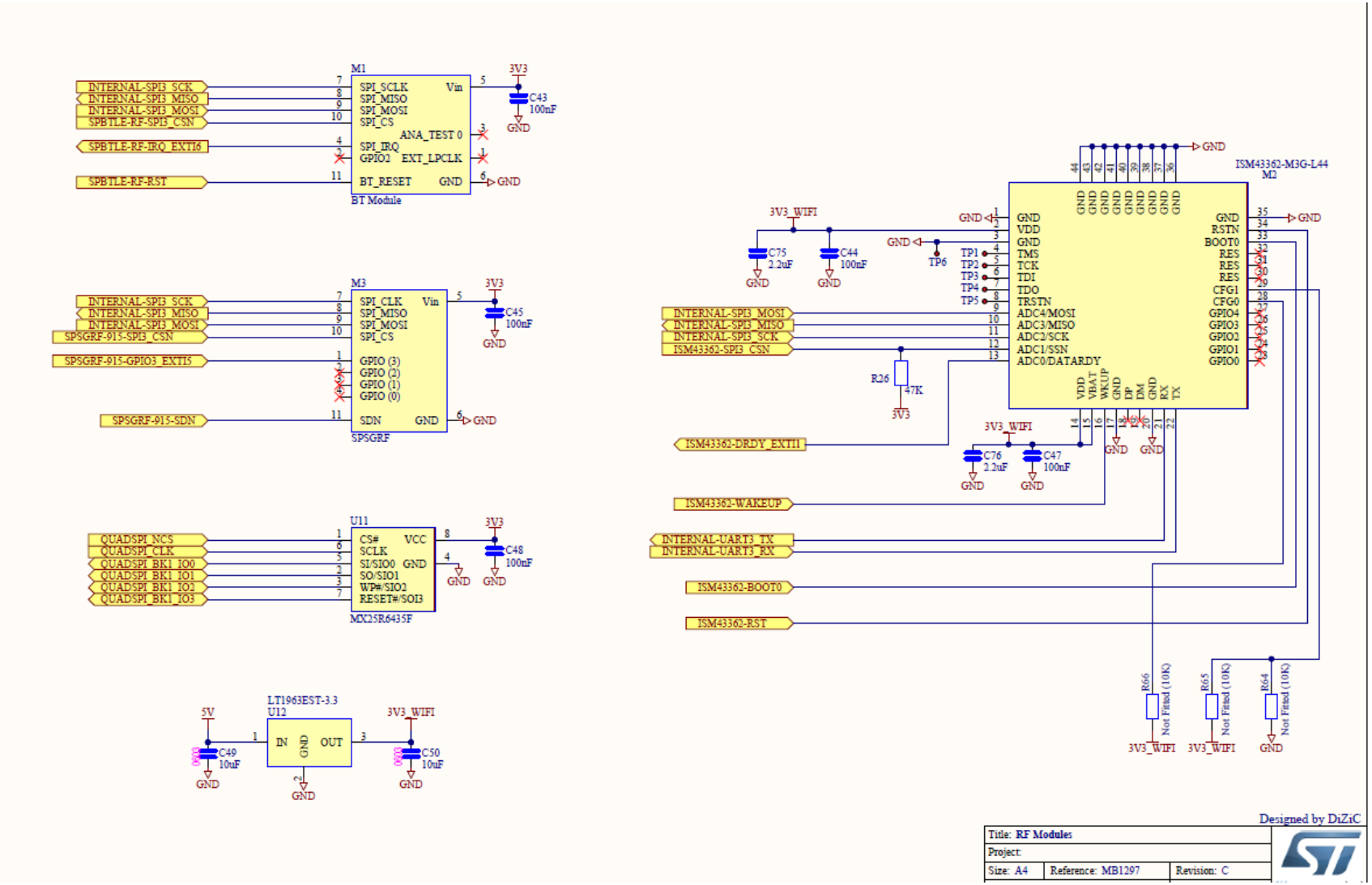
Step 14. User Story 3. BSP_QSPI_GetInfo(). When the user selects 2, display the results of calling BSP_QSPI_GetStatus() on the console.

```c
void do_qspi_info()
{
    QSPI_Info info = {0};
    BSP_QSPI_GetInfo(&info);

    char buf[100];
    snprintf(buf, sizeof(buf), "FlashSize: %lu\r\n"
                               "EraseSectorSize: %lu\r\n"
                               "ProgPageSize: %lu\r\n",
                               info.FlashSize,
                               info.EraseSectorSize,
                               info.ProgPageSize);
    HAL_UART_Transmit(&huart1, (uint8_t*)buf, strlen(buf), 1000);
}
```

Step 15. User Story 3. Build and run the code, test is successful



Appendix, schematic for the module on the board that use SPI as connection



Appendix, Ardunio connector that has SPI connection