Design a Datalogger code forMPU6050 6-axis motion sensor for STM32

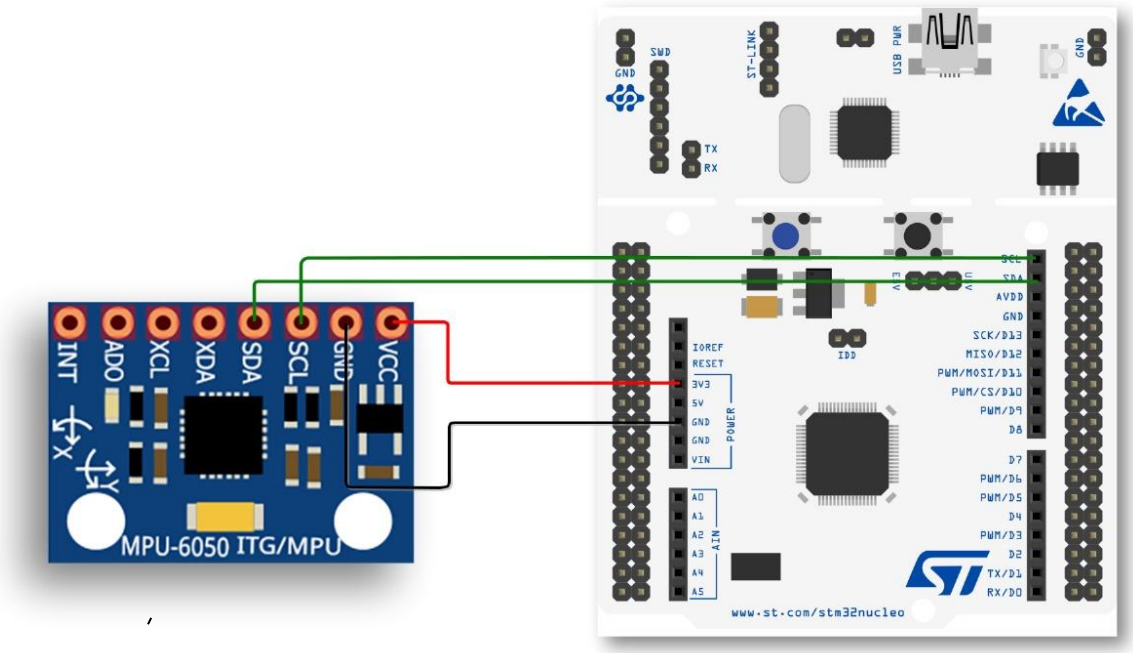
Objective:

The Objective of this experiment is to create a datalogger code for 6-axis motion sensor (MPU6050). The datalogger code will create a buffer where all the motion sensor data sample will be stored, using which we will be able to create datasets of motion samples to build a machine learning model in the NanoEdge AI Studio.

Requirements:

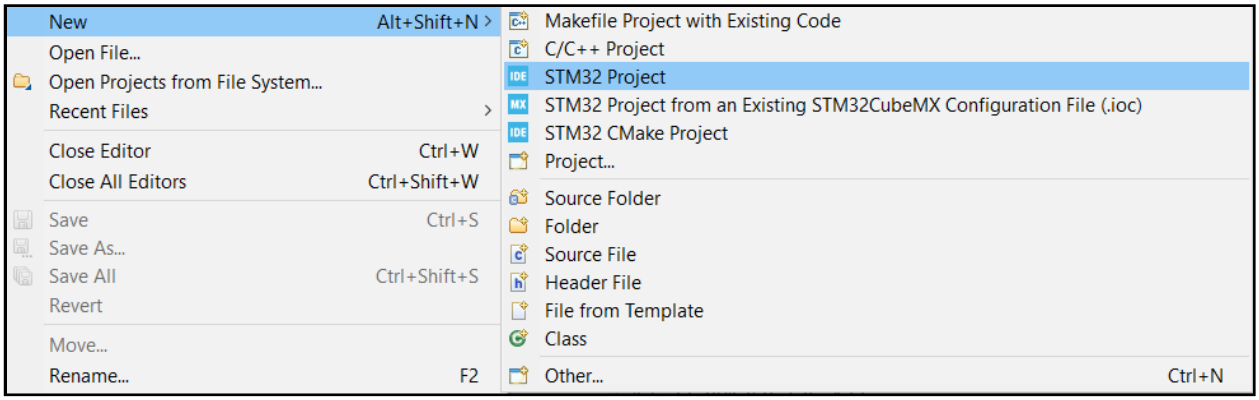
1. STM32 Cube IDE software.
2. MPU6050 sensor (I2C).
3. STM32 Microcontroller.
4. USB Cable for the microcontroller.
5. Jumper Wires.
6. Laptop or PC

Connection Diagram:

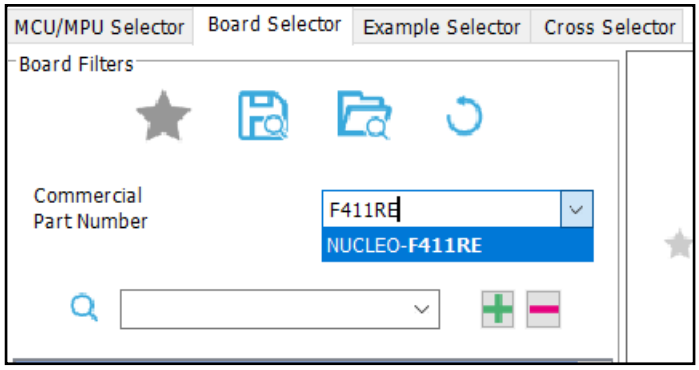


Procedure:

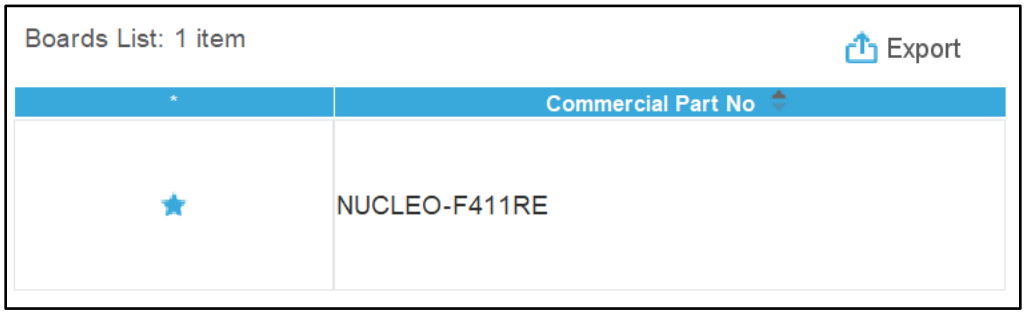
1. Click on **File→New→STM32 Project**to start your project on Cube IDE.



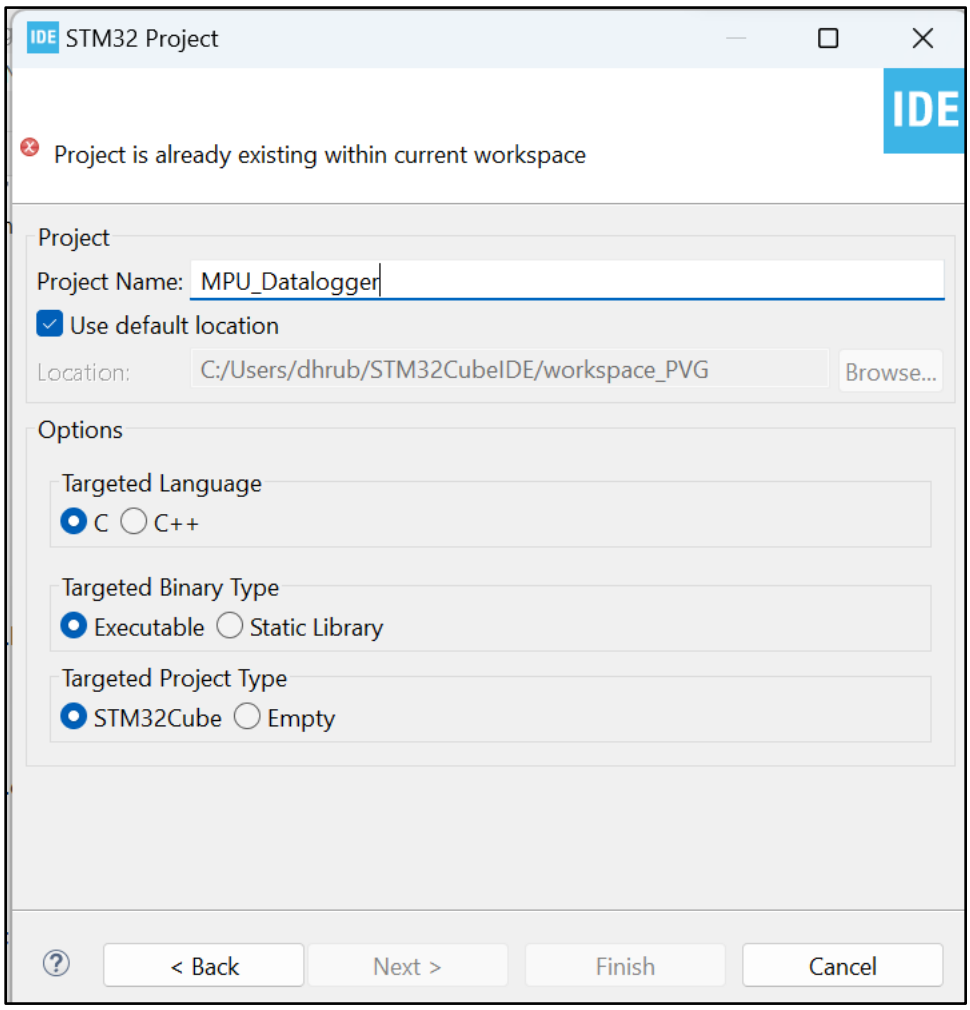
2. A **Target Selection** window will open. Click on **Board Selector**, where you need to select the microcontroller board you are working with.
(**NB:** If you are having Nucleo-F401RE, you have to select the said Commercial Part Number)



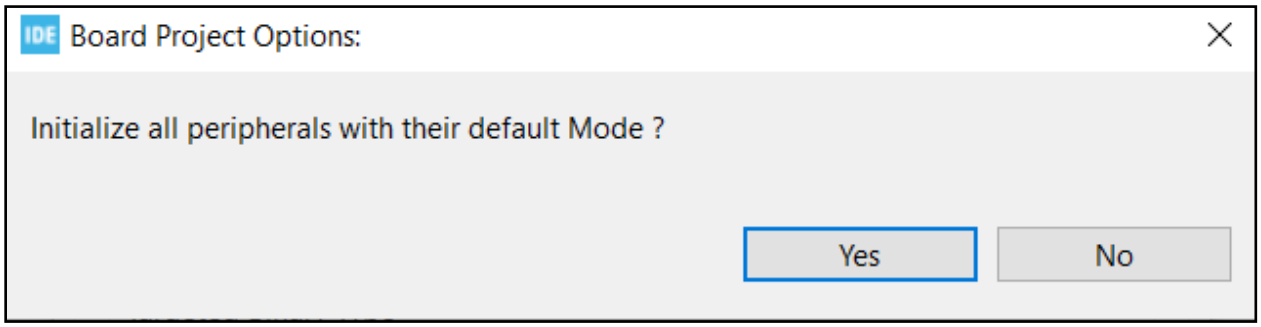
3. After this on the right-hand side of the window, under**Board List** you will see the board you have selected. Click on the board and then click on **Next.**



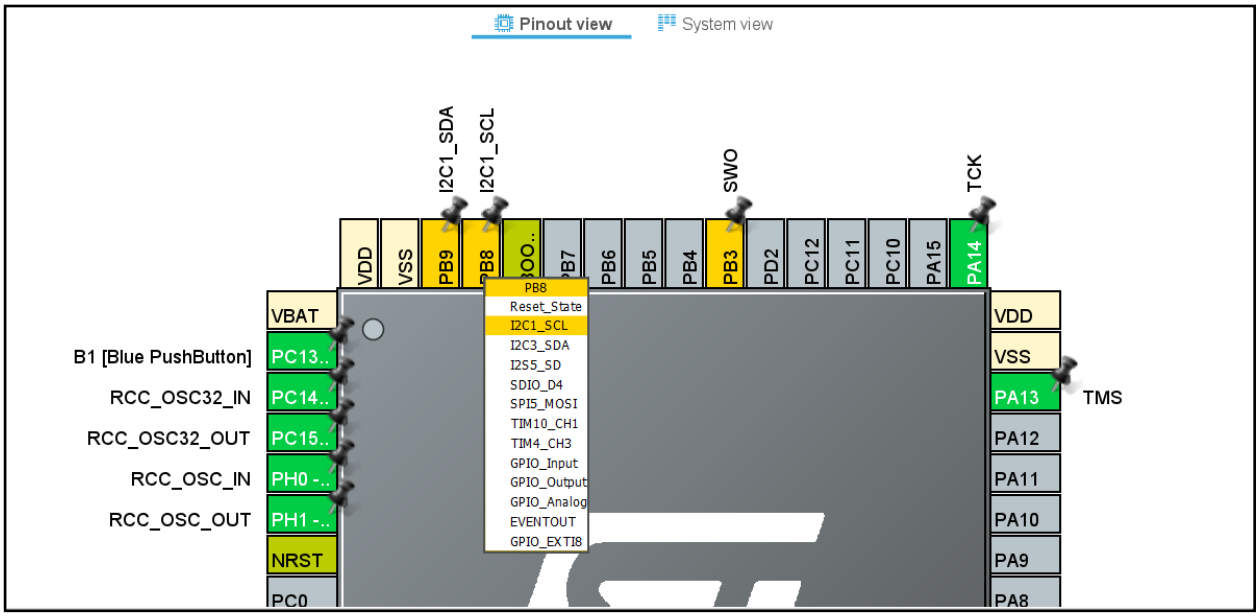
4. In the next window give your project a name, rest of the things will remain by default as it is for now. Click on **Finish.**



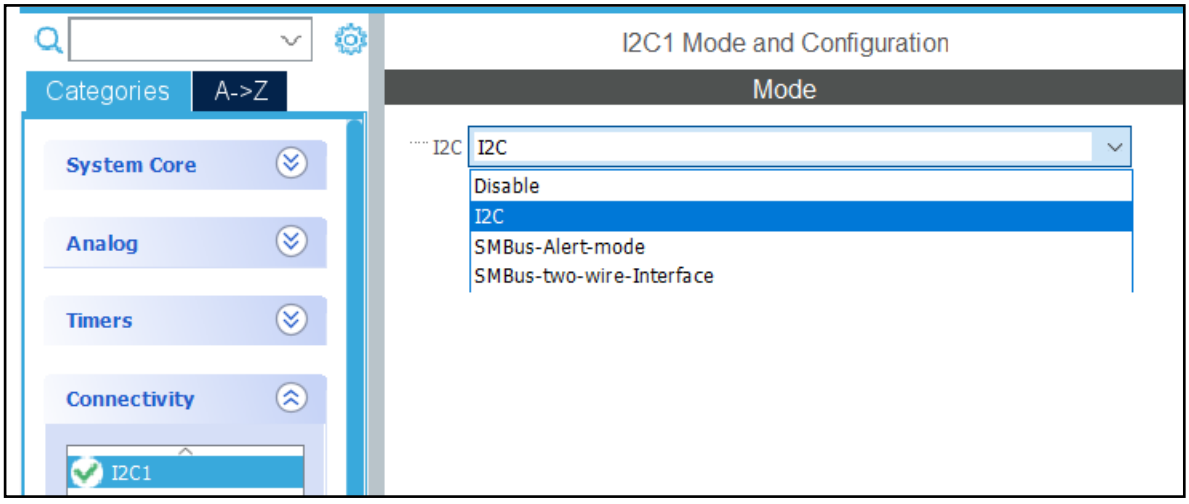
5. Cube IDE will ask if you want to initialize all peripherals with their default mode, click on **Yes.**



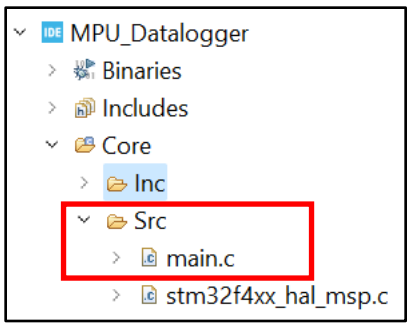
6. In the **Pinout & Configuration** tab, click on **PB8** pinand select it as an **I2C1_SCL** and **PB9** pinas an **I2C1_SDA**.



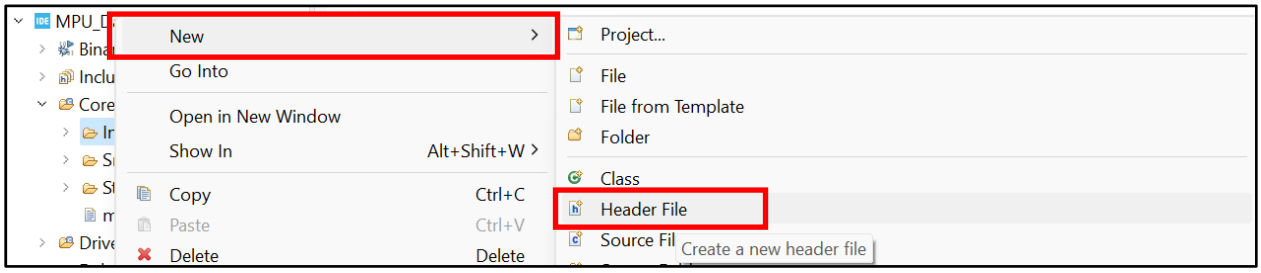
7. Next on the left-hand side under **Categories →Connectivity**, select I2C1and enable it.



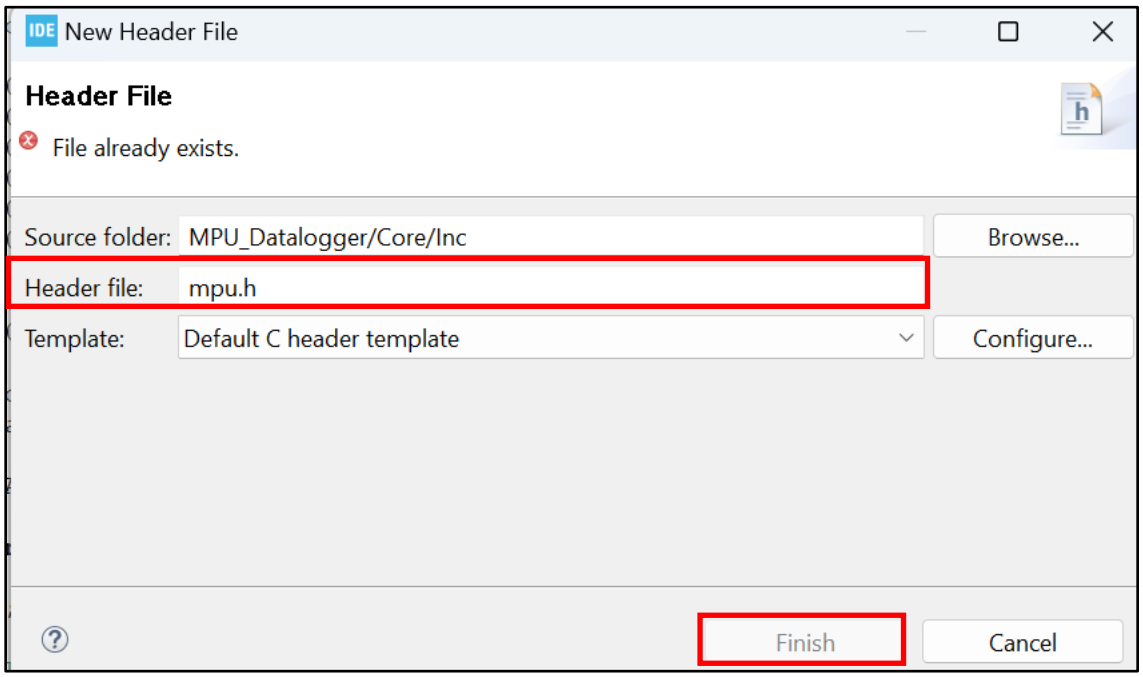
8. Press C**trl+S** to generate your code. On the left-hand side of the Cube IDE, under **Project Explorer** go to the project you have created (For example, I have named my project as (MPU_Datalogger)**MPU_Datalogger→Core →Src→main.c (**double click to load the code).



9. Now open your project tree **MPU_Datalogger→Core →Inc.** Right click on your **Inc** folder and create a new **Header File**.



10. Name the Header File as **mpu.h**and select on **Finish**.



11. Below are the code snippets, please put your code in the appropriate places in the **mpu.h** file.

```
#ifndef INC_MPU_H_
#define INC_MPU_H_

extern I2C_HandleTypeDef hi2c1;

#define MPU6050_ADDR            0xD0
#define MPU6050_PWR_MGMT_1      0x6B
#define MPU6050_ACCEL_XOUT_H    0x3B
#define MPU6050_ACCEL_YOUT_H    0x3D
#define MPU6050_ACCEL_ZOUT_H    0x3F
#define MPU6050_TEMP_OUT_H      0x41
#define MPU6050_GYRO_XOUT_H     0x43
#define MPU6050_GYRO_YOUT_H     0x45
#define MPU6050_GYRO_ZOUT_H     0x47

int16_t accel_data[3];
int16_t gyro_data[3];

float Ax, Ay, Az, Gx, Gy, Gz;

void MPU6050_Init(void)
{
  uint8_t data;

  // Wake up MPU6050
  data = 0x00;
  HAL_I2C_Mem_Write(&hi2c1, MPU6050_ADDR, MPU6050_PWR_MGMT_1, 1, &data, 1, HAL_MAX_DELAY);
}
```

```
void MPU6050_Read_Accel(int16_t* accel_data)
{
  uint8_t buffer[6];

  HAL_I2C_Mem_Read(&hi2c1, MPU6050_ADDR, MPU6050_ACCEL_XOUT_H, 1, buffer, 6, HAL_MAX_DELAY);

  accel_data[0] = (int16_t)((buffer[0] << 8) | buffer[1]);
  accel_data[1] = (int16_t)((buffer[2] << 8) | buffer[3]);
  accel_data[2] = (int16_t)((buffer[4] << 8) | buffer[5]);

/*  AFS_SEL       Full Scale Range       LSB Sensitivity
       0               ±2g                16384 LSB/g
       1               ±4g                 8192 LSB/g
       2               ±8g                 4096 LSB/g
       3               ±16g                2048 LSB/g */

    Ax = accel_data[0]/2048.0;
    Ay = accel_data[1]/2048.0;
    Az = accel_data[2]/2048.0;
}
```

```
52⊖void MPU6050_Read_Gyro(int16_t* gyro_data)
53 {
54   uint8_t buffer[6];
55
56   HAL_I2C_Mem_Read(&hi2c1, MPU6050_ADDR, MPU6050_GYRO_XOUT_H, 1, buffer, 6, HAL_MAX_DELAY);
57
58   gyro_data[0] = (int16_t)((buffer[0] << 8) | buffer[1]);
59   gyro_data[1] = (int16_t)((buffer[2] << 8) | buffer[3]);
60   gyro_data[2] = (int16_t)((buffer[4] << 8) | buffer[5]);
61
62⊖ /*  FS_SEL       Full Scale Range       LSB Sensitivity
63       0              ± 250 °/s             131 LSB/°/s
64       1              ± 500 °/s            65.5 LSB/°/s
65       2              ± 1000 °/s           32.8 LSB/°/s
66       3              ± 2000 °/s           16.4 LSB/°/s */
67
68    Gx = gyro_data[0]/131.0;
69    Gy = gyro_data[1]/131.0;
70    Gz = gyro_data[2]/131.0;
71 }
72
73 #endif
```

12. Cube IDE automatically generates a code format based on the configurations you have done. Cube IDE uses HAL libraries. Below are the code snippets, please put your code in the appropriate places in the **main.c** file.

```
22⊖/* Private includes ----------
23 /* USER CODE BEGIN Includes */
24 #include "mpu.h"
25 #include "stdio.h"
26 #include "string.h"
27 /* USER CODE END Includes */
```

```
34⊖/* Private define ----------
35 /* USER CODE BEGIN PD */
36 #define DATA_INPUT_USER 256
37 #define AXIS_NUMBER 3
38 /* USER CODE END PD */
```

```
50 /* USER CODE BEGIN PV */
51 float mpu_buffer[AXIS_NUMBER * DATA_INPUT_USER] = {0};
52 /* USER CODE END PV */
```

```
59 /* USER CODE BEGIN PFP */
60 int __io_putchar(int);
61 void fill_mpu_buffer();
62 void Log();
63 /* USER CODE END PFP */
```

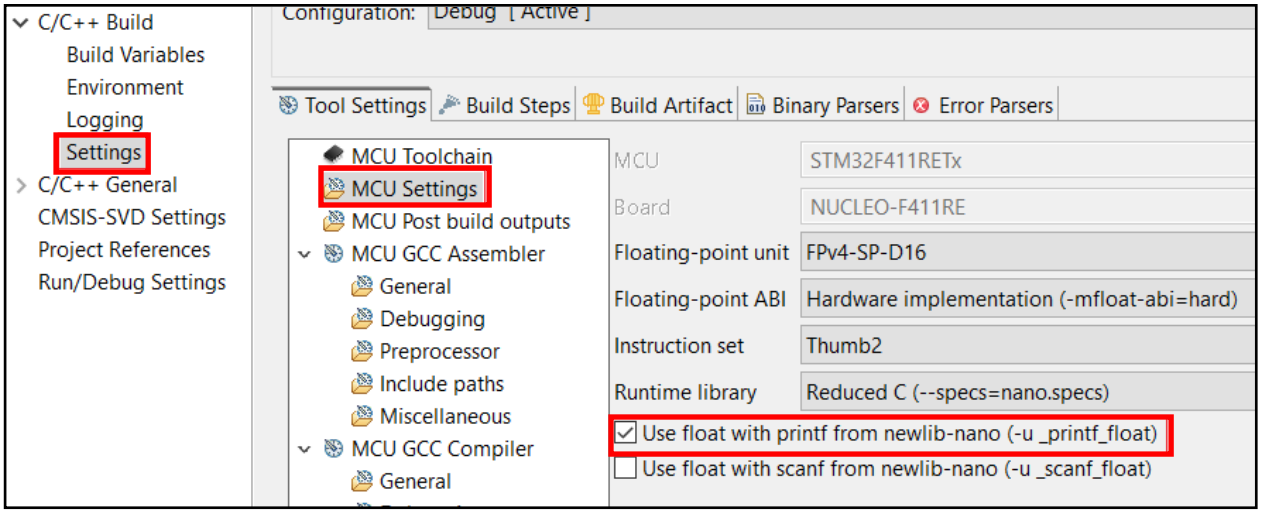
```
100   /* USER CODE BEGIN 2 */
101   MPU6050_Init();
102   HAL_Delay(1000);
103   /* USER CODE END 2 */
```

```
106   /* USER CODE BEGIN WHILE */
107   while (1)
108   {
109       Log();
110     /* USER CODE END WHILE */
```

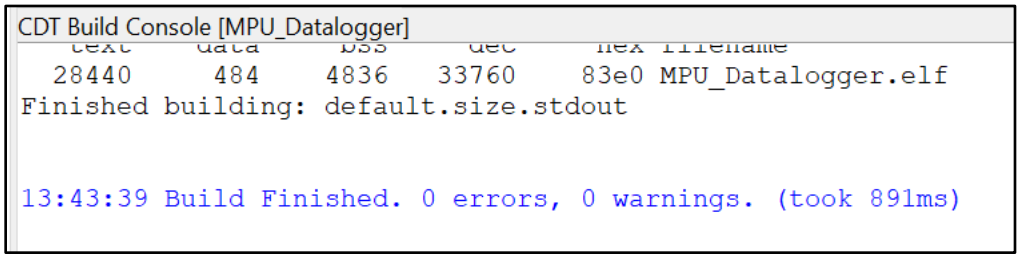
```
/* USER CODE BEGIN 4 */
int __io_putchar(int ch)
{
    HAL_UART_Transmit(&huart2, (uint8_t *)&ch, 1, HAL_MAX_DELAY);
    return ch;
}
void fill_mpu_buffer()
{

    for(int i=0; i<DATA_INPUT_USER; i++)
    {
        MPU6050_Read_Accel(accel_data);
        mpu_buffer[AXIS_NUMBER *i] = Ax;
        mpu_buffer[AXIS_NUMBER *i + 1] = Ay;
        mpu_buffer[AXIS_NUMBER *i + 2] = Az;

    }
}
```

```
void Log()
{
    fill_mpu_buffer();
    for(int i=0; i<DATA_INPUT_USER; i++)
    {
        printf("%.2f", mpu_buffer[AXIS_NUMBER * i]);
        printf(" ");
        printf("%.2f", mpu_buffer[AXIS_NUMBER * i + 1]);
        printf(" ");
        printf("%.2f", mpu_buffer[AXIS_NUMBER * i + 2]);
        printf(" ");
    }
    printf("\r\n");
    HAL_Delay(100);
}
```

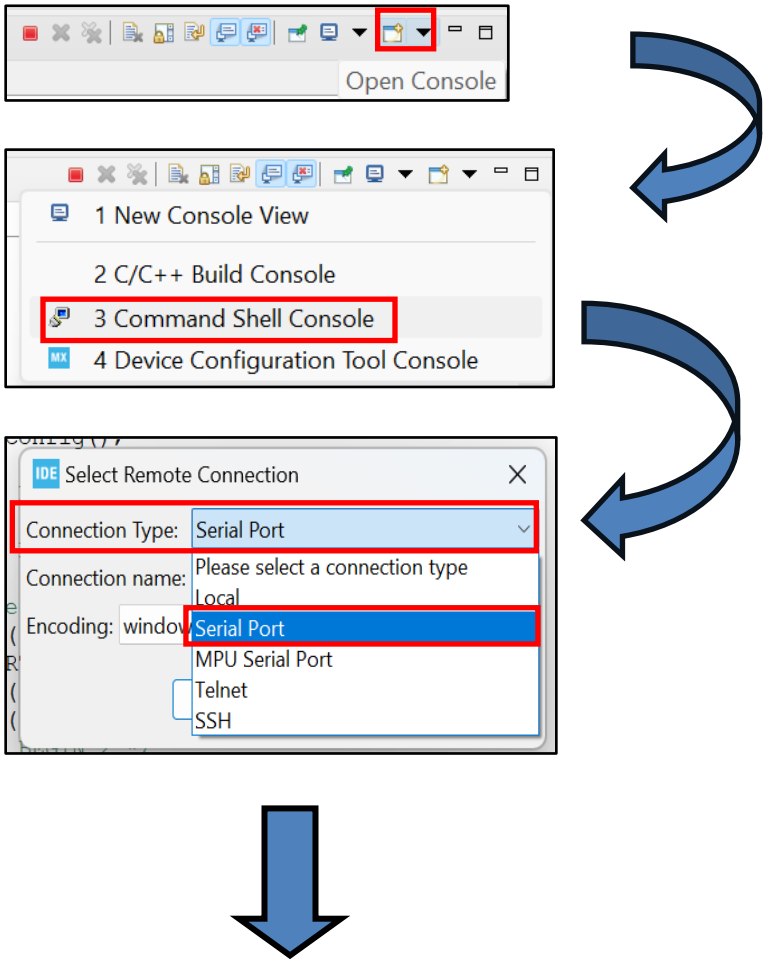
13. Right click on the Audio_Classification project and select **Properties**. Go to **C/C++ Build →Settings.** Next select **MCU Settings** and enable the option **Use float with printf from newlib-nano (-u_printf_float).**

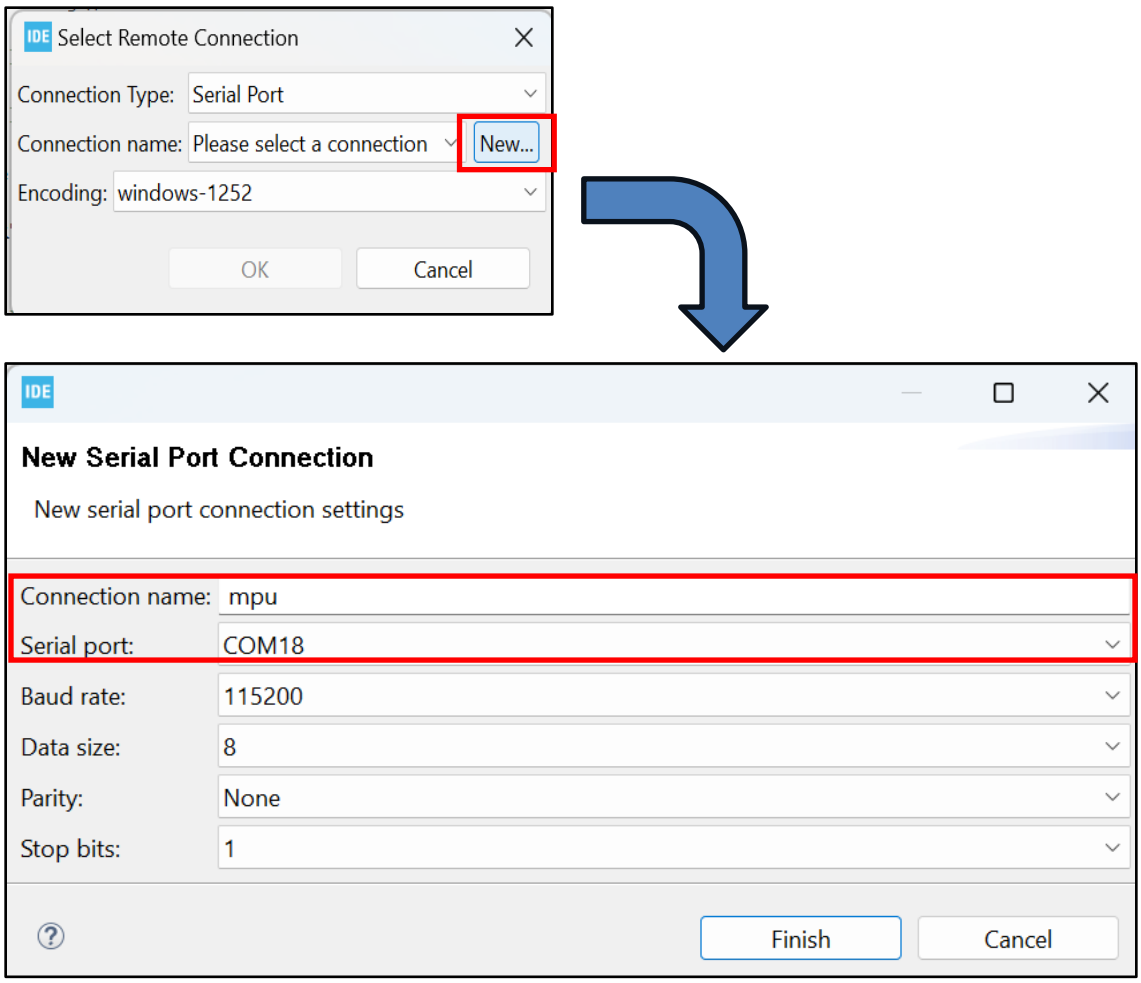


14. Now click on the build  symbol on the top left corner on your Cube IDE. If you have done everything correctly your code should be built without any errors.

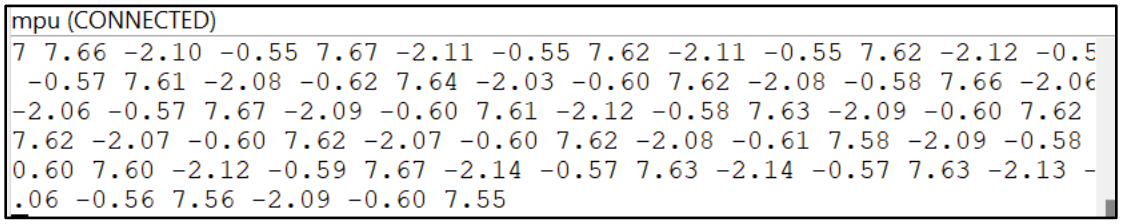


15. Next connect your STM32 board with your audio sensor connect to it to your PC and click on the **Debug**  icon to start the Debugging process. An**Edit Configuration** window will open, click on **OK,** without making any changes.
16. In the debug mode, go to the bottom right hand side corner, click on open console. Selectthe **Connection Type** as **Serial Port**, then click on **New.** In the new window, in **Connection name** give some name to your new connection, and select the **Serial port** correctly. Then click on **Finish** and then **Ok.** A console with the given name will be opened at the bottom of your screen.





17. Click on the **Resume** icon  to run your code. You should be able to see the values ofmpu6050sensor in the **Console**.



18. Before moving out of the debugging mode, click on **Disconnect**  and close the console then click on the **Terminate**  icon. You will be moved out of the debugging mode.

Note: All important steps and parts are highlighted with a red color box for the proper understanding of the user. This document is for the use of education purpose only.