

Interfacing a Temperature & Humidity Sensor

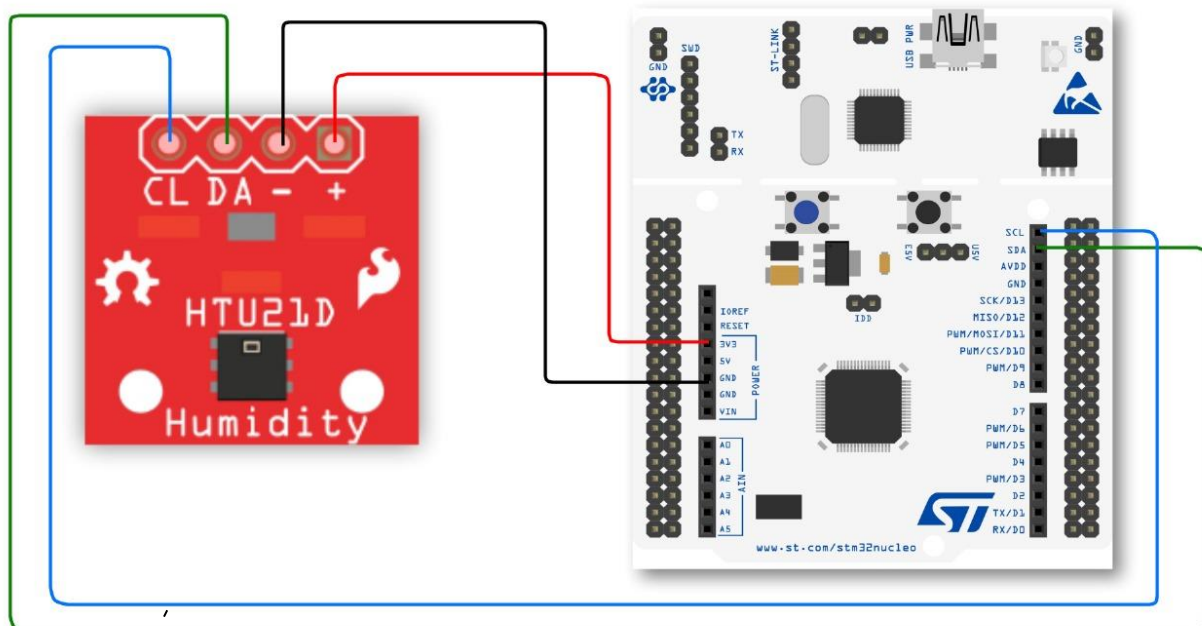
Objective:

The Objective of this experiment is to interface a temperature & humidity sensor to an STM32 microcontroller.

Requirements:

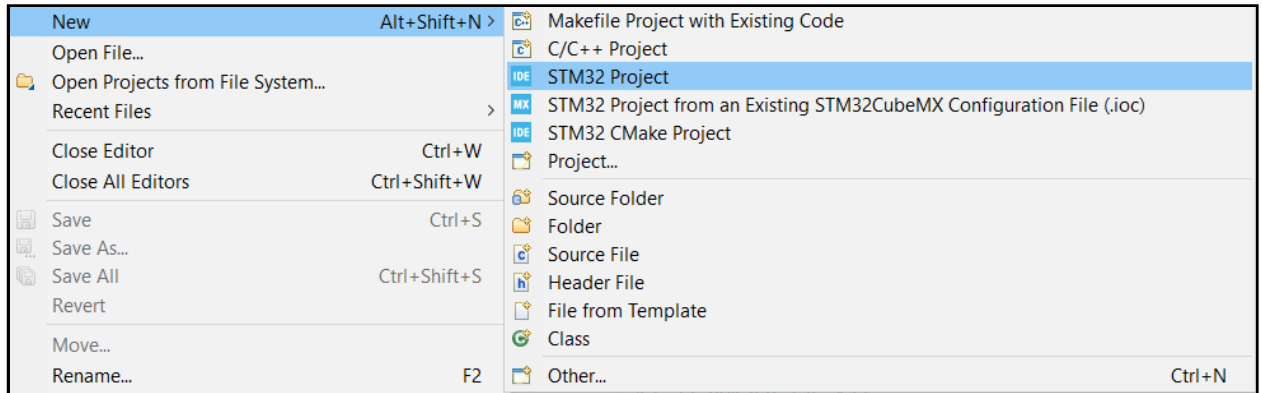
1. STM32 Cube IDE software.
2. HTU21DF Sensor (I2C).
3. STM32 Microcontroller.
4. USB Cable for the microcontroller.
5. Jumper Wires.

Connection Diagram:

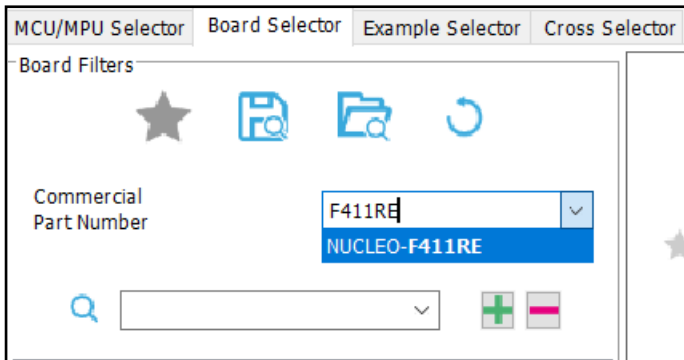


Procedure:


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



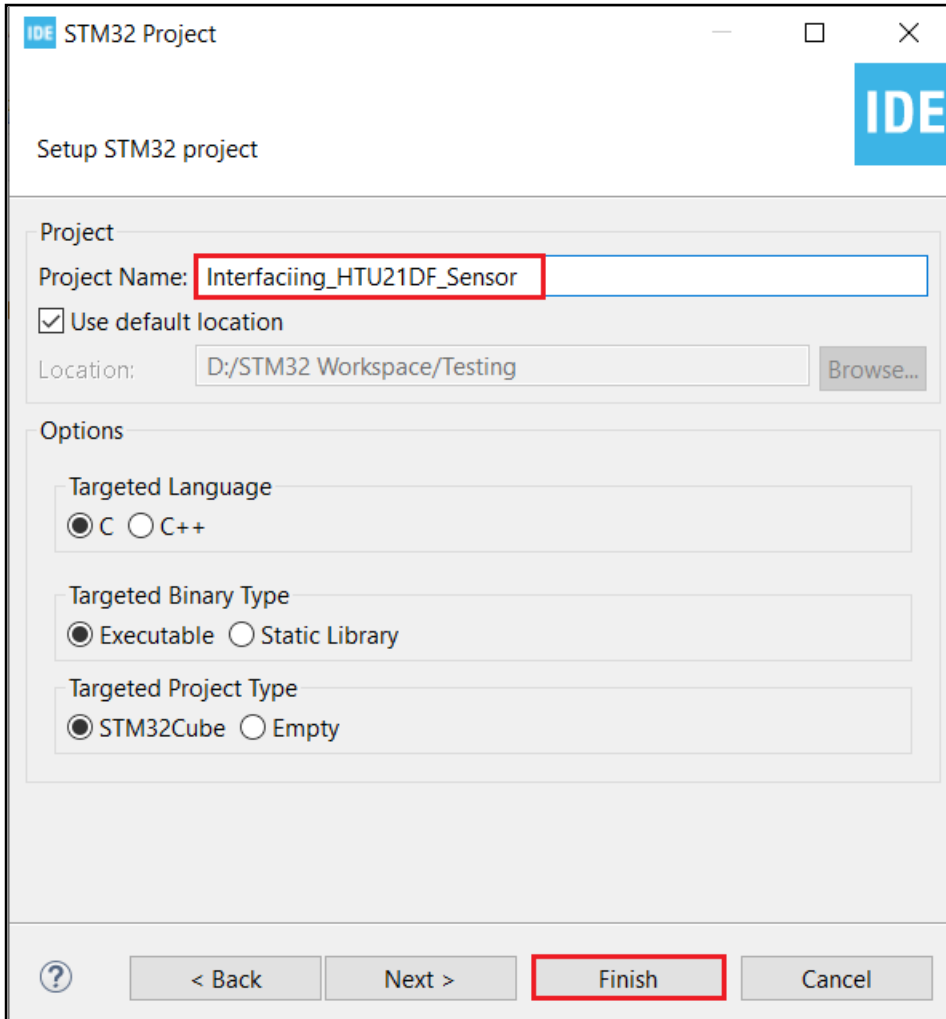
2. A **Target Selection** window will open. Click on **Board Selector**, where you need to select the microcontroller board you are working with.
(NB: If you are having Nucleo-F401RE, you have to select the said Commercial Part Number)



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**.

Overview	Commercial Part No	Type	Marketing Status	Unit Price (US\$)	Mounted Device
	NUCLEO-F411RE	Nucleo-64	Active	13.0	STM32F411RET6

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



The image shows the 'Setup STM32 project' dialog box in the IDE. The 'Project' section has 'Project Name' set to 'Interfaciing_HTU21DF_Sensor' (highlighted with a red box). The 'Use default location' checkbox is checked, and the 'Location' is 'D:/STM32 Workspace/Testing'. The 'Options' section has 'Targeted Language' set to 'C', 'Targeted Binary Type' set to 'Executable', and 'Targeted Project Type' set to 'STM32Cube'. At the bottom, the 'Finish' button is highlighted with a red box.

IDE STM32 Project

Setup STM32 project

Project

Project Name: Interfaciing_HTU21DF_Sensor

☒ Use default location

Location: D:/STM32 Workspace/Testing Browse...

Options

Targeted Language

☒ C ☐ C++

Targeted Binary Type

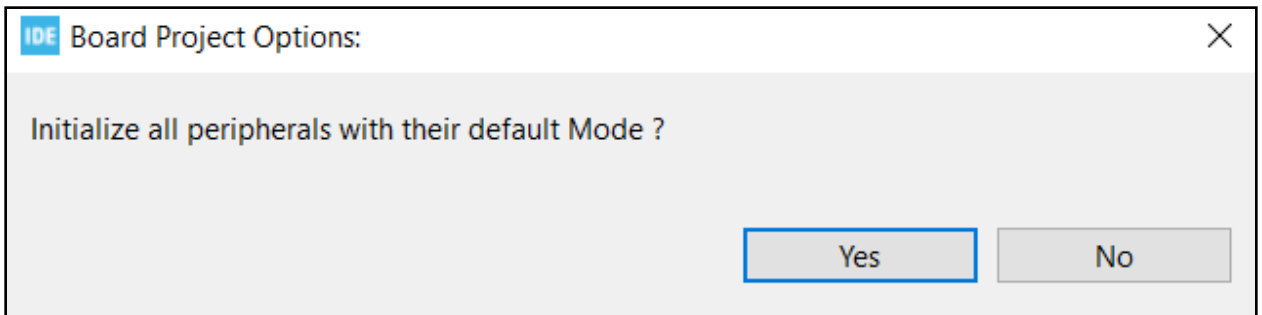
☒ Executable ☐ Static Library

Targeted Project Type

☒ STM32Cube ☐ Empty

? < Back Next > Finish Cancel

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



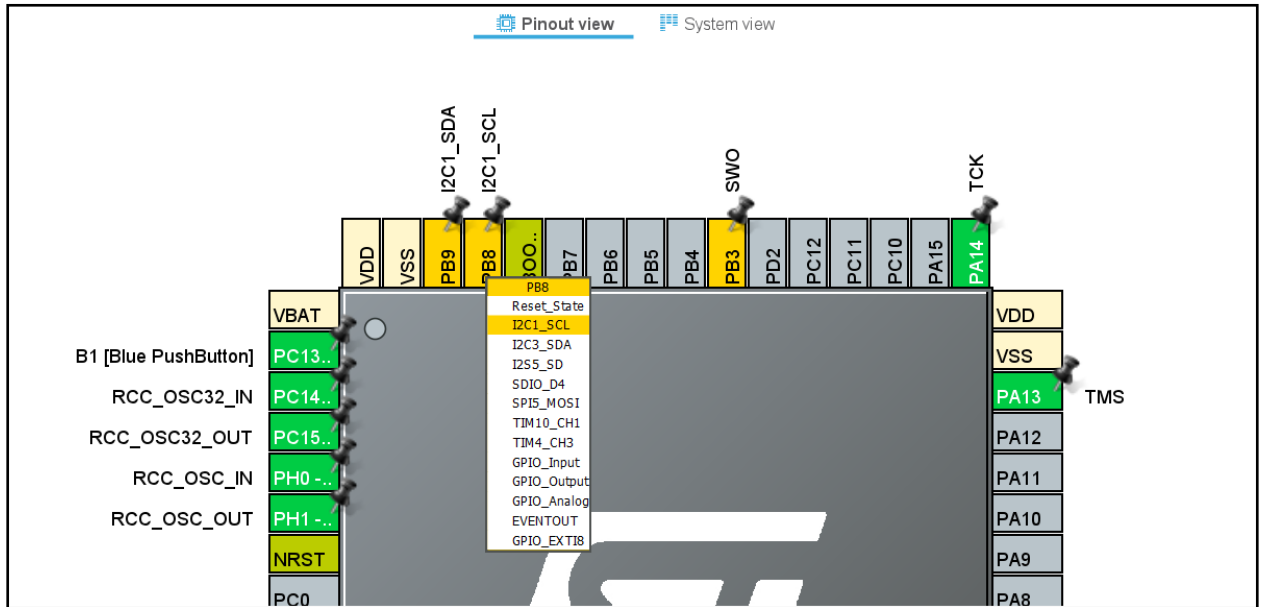
The image shows the 'Board Project Options' dialog box. It asks 'Initialize all peripherals with their default Mode ?'. The 'Yes' button is highlighted with a blue box.

IDE Board Project Options:

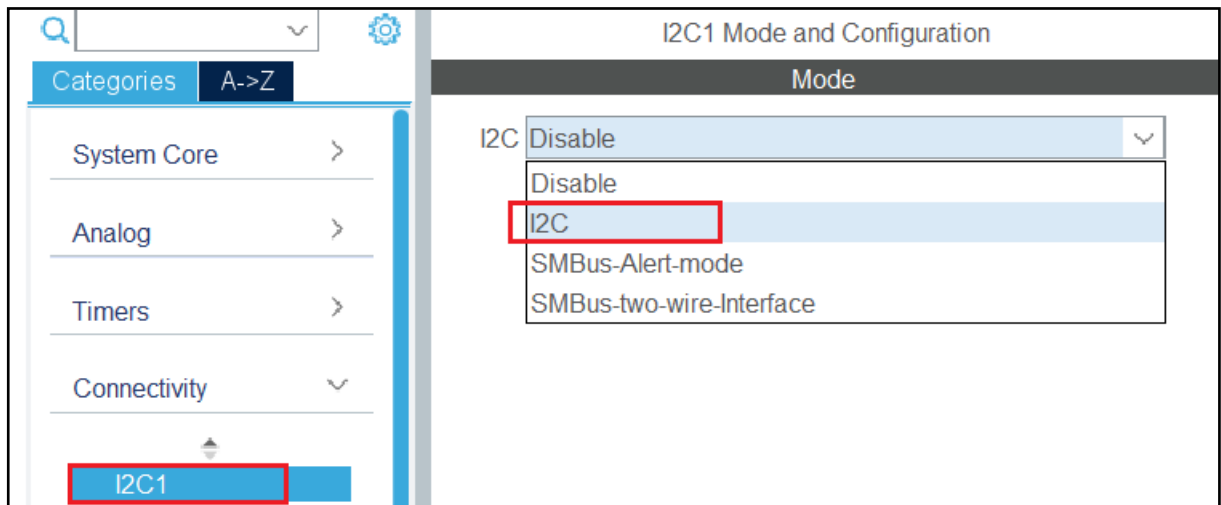
Initialize all peripherals with their default Mode ?

Yes No

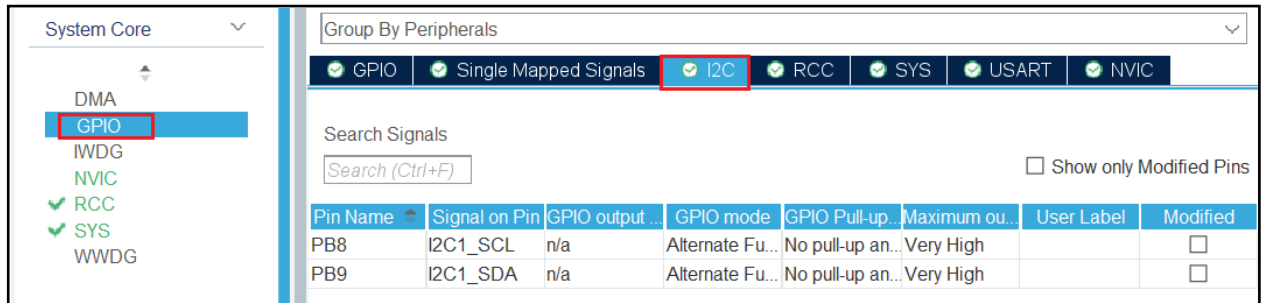
6. In the **Pinout & Configuration** tab, click on **PB8** pin and select it as an **I2C1_SCL** and **PB9** pin as an **I2C1_SDA**.



7. Next on the left-hand side under **Categories** → **Connectivity**, select **I2C1** and enable it.



8. Now go to **System Core** → **GPIO** and select **I2C** option.



System Core

Group By Peripherals

GPIO Single Mapped Signals **I2C** RCC SYS USART NVIC

Search Signals

Search (Ctrl+F)

☐ Show only Modified Pins

Pin Name	Signal on Pin	GPIO output	GPIO mode	GPIO Pull-up/...	Maximum ou...	User Label	Modified
PB8	I2C1_SCL	n/a	Alternate Fu...	No pull-up an...	Very High		<input type="checkbox"/>
PB9	I2C1_SDA	n/a	Alternate Fu...	No pull-up an...	Very High		<input type="checkbox"/>

9. After selecting **I2C** option, change the GPIO pin **PB8 and PB9** as **Pull-up**.

Pin Name	Signal on Pin	GPIO output I...	GPIO mode	GPIO Pull-up/...	Maximum out...	User Label	Modified
PB8	I2C1_SCL	n/a	Alternate Fun...	No pull-up an...	Very High		<input type="checkbox"/>
PB9	I2C1_SDA	n/a	Alternate Fun...	No pull-up an...	Very High		<input type="checkbox"/>

PB8 Configuration :

GPIO mode: Alternate Function Open Drain

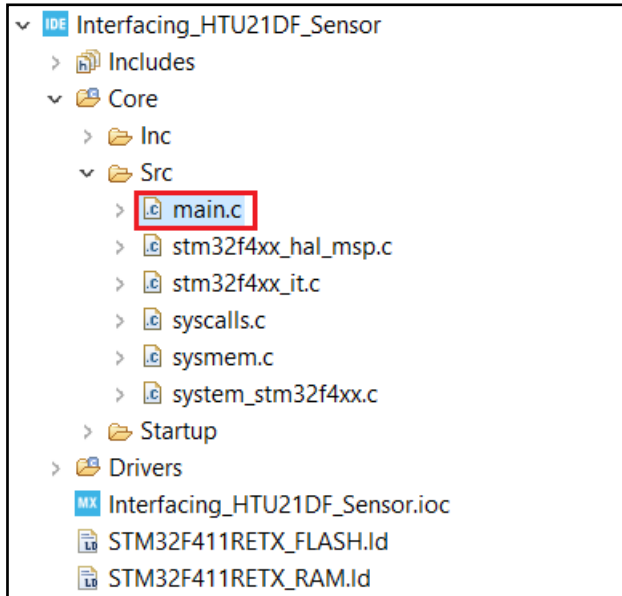
GPIO Pull-up/Pull-down: No pull-up and no pull-down

Maximum output speed: Pull-up

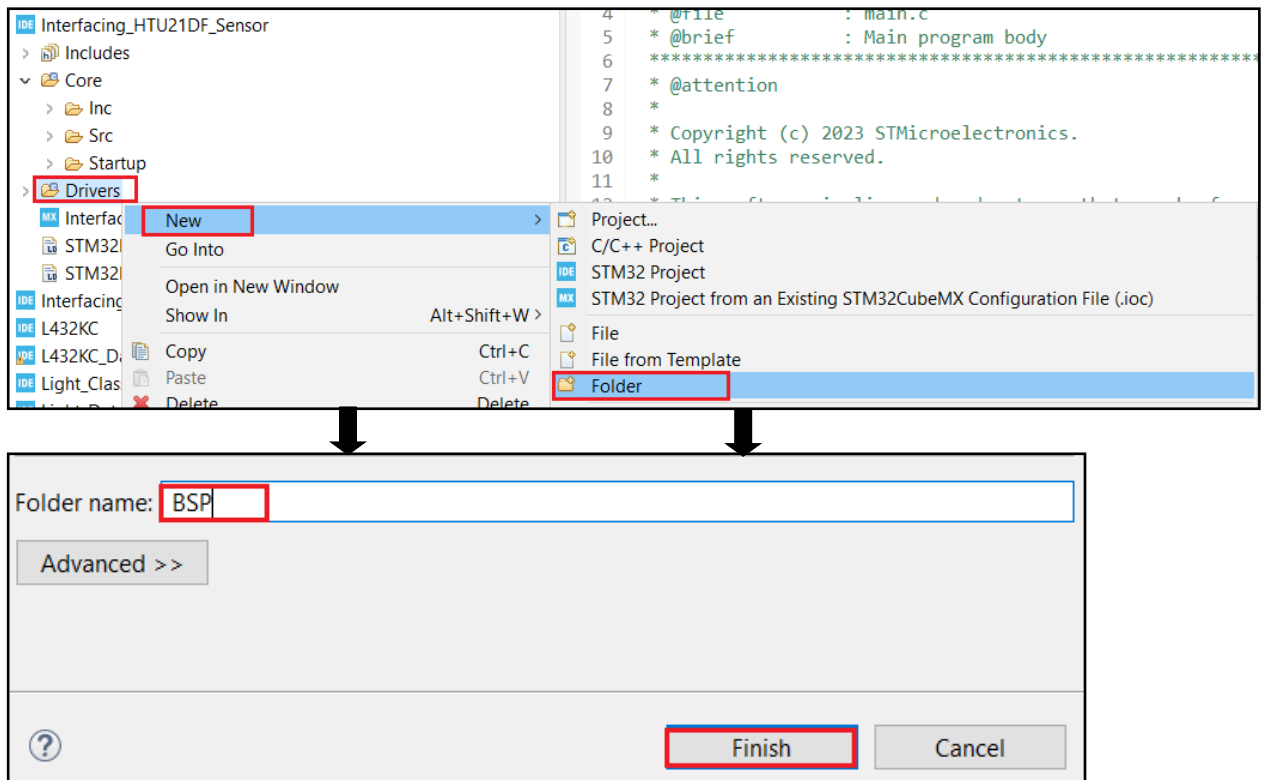
↓

Pin Name	Signal on Pin	GPIO output I...	GPIO mode	GPIO Pull-up/...	Maximum out...	User Label	Modified
PB8	I2C1_SCL	n/a	Alternate Fun...	Pull-up	Very High		<input checked="" type="checkbox"/>
PB9	I2C1_SDA	n/a	Alternate Fun...	Pull-up	Very High		<input checked="" type="checkbox"/>

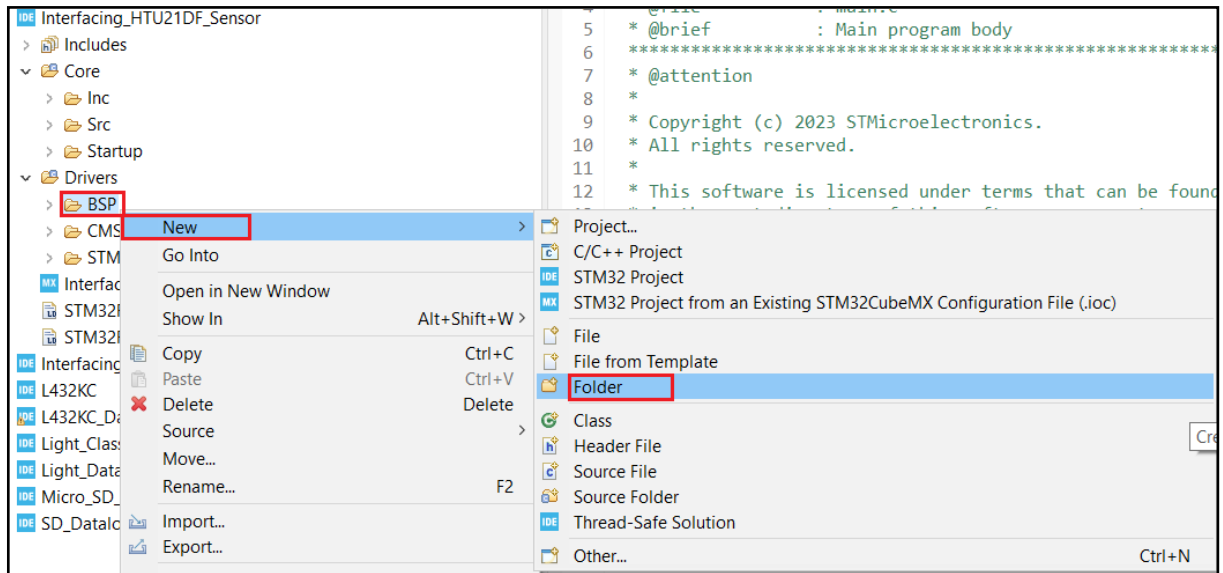
10. Press **Ctrl+S** to generate your code. On the left-hand side of the Cube IDE, under **Project Explorer** go to the project you have created (For example, I have named my project as **Interfacing_HTU21DF_Sensor**) **Interfacing_HTU21DF_Sensor**→**Core** →**Src**→**main.c** (double click to load the code).



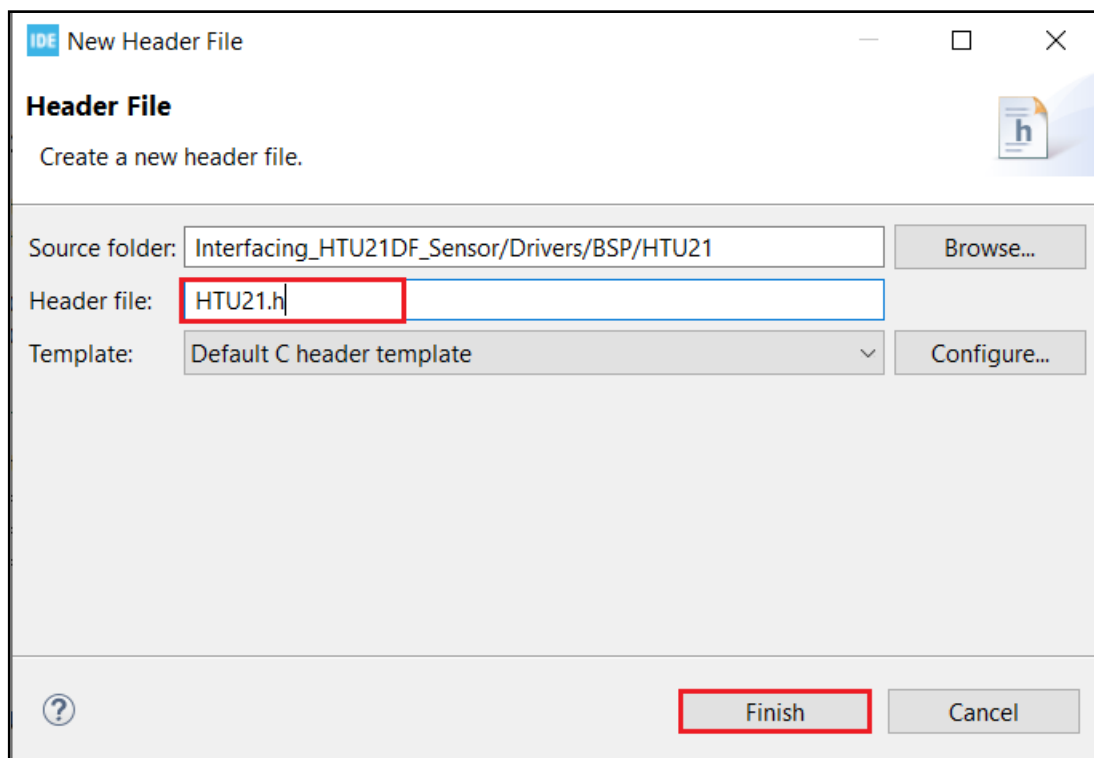
11. Open your project tree **Interfacing_HTU21DF_Sensor**. Right click on your **Drivers** folder and create a new **Folder** name as **BSP**.



12. Now right click on the **BSP** folder and create a new folder name as **HTU21**.



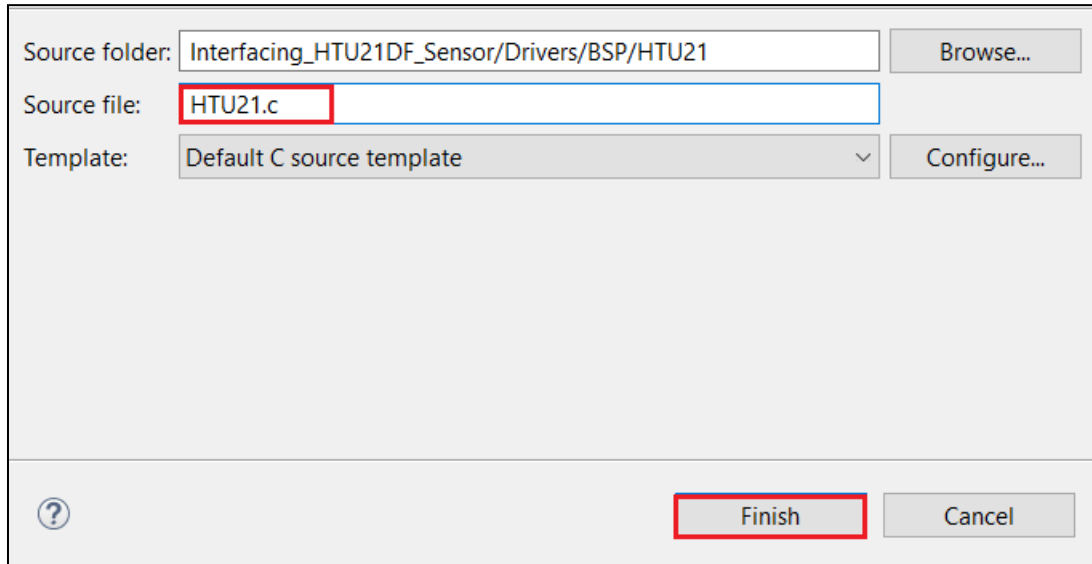
13. Right click on the HTU21 folder and create a new Source file and the name the file as **HTU21.h** and select on **Finish**.



14. Below is the code snippets, please put your code in the appropriate places in the **HTU21.h** file.

```
2
3 #ifndef SRC_HTU21_H_
4 #define SRC_HTU21_H_
5
6 #include "stm32f4xx_hal.h"
7 extern I2C_HandleTypeDef hi2c1;
8
9 /** Default I2C address for the HTU21D. */
10 #define HTU21_I2CADDR      (0x80)
11
12 /** Read temperature register. */
13 #define HTU21_READTEMP     (0xE3)
14
15 /** Read humidity register. */
16 #define HTU21_READHUM      (0xE5)
17
18 /** Write register command. */
19 #define HTU21_WRITEREG     (0xE6)
20
21 /** Read register command. */
22 #define HTU21_READREG      (0xE7)
23
24 /** Reset command. */
25 #define HTU21_RESET        (0xFE)
26
27 #ifdef __cplusplus
28 extern "C" {
29 #endif
30 char    HTU21_Init(void);
31 float   HTU21_GetTemp(void);
32 float   HTU21_GetHum(void);
33 void     HTU21_Reset(void);
34 #ifdef __cplusplus
35 }
36 #endif
37
38 #endif /* SRC_HTU21_H_ */
39
```


15. Right click on the HTU21 folder again and create a new source file and the name the file as **HTU21.c** and select on **Finish**.



Source folder: Interfacing_HTU21DF_Sensor/Drivers/BSP/HTU21 Browse...


Source file: HTU21.c

Template: Default C source template Configure...

Finish Cancel

16. Below is the code snippets, please put your code in the appropriate places in the **HTU21.c** file.

```
2 #include "HTU21.h"
3
4 void HTU21_Reset()
5 {
6     uint8_t Data=HTU21_RESET;
7
8     HAL_Delay(15);
9     HAL_I2C_Master_Transmit(&hi2c1, HTU21_I2CADDR, &Data, 1, 1000);
10    HAL_Delay(15);
11
12 }
13
14 char HTU21_Init()
15 {
16     uint8_t check;
17     uint8_t Data=HTU21_READREG;
18
19     HAL_I2C_Master_Transmit(&hi2c1, HTU21_I2CADDR, &Data, 1, 1000);
20     HAL_I2C_Master_Receive(&hi2c1, HTU21_I2CADDR, &check, 1, 1000);
21
22     return check;
23 }
24
```



```
24
25 float HTU21_GetTemp()
26 {
27     uint8_t Temp_Data[2];
28     uint8_t Data=HTU21_READTEMP;
29
30     HAL_I2C_Master_Transmit(&hi2c1, HTU21_I2CADDR, &Data, 1, 1000);
31     HAL_I2C_Master_Receive(&hi2c1, HTU21_I2CADDR, Temp_Data, 2, 1000);
32
33     uint16_t Raw_Temp = (uint16_t)(Temp_Data[0] << 8 | (Temp_Data[1]& 0b11111100));
34
35     float temp = Raw_Temp;
36     temp *= 175.72f;
37     temp /= 65536.0f;
38     temp -= 46.85f;
39
40     return temp;
41 }
42
43 float HTU21_GetHum()
44 {
45     uint8_t Hum_Data[2];
46     uint8_t Data=HTU21_READHUM;
47
48     HAL_I2C_Master_Transmit(&hi2c1, HTU21_I2CADDR, &Data, 1, 1000);
49     HAL_I2C_Master_Receive(&hi2c1, HTU21_I2CADDR, Hum_Data, 2, 1000);
50
51     uint16_t Raw_Hum = (uint16_t)(Hum_Data[0] << 8 | (Hum_Data[1]& 0b11110000));
52
53     float temp = Raw_Hum;
54     temp *= 175.72f;
55     temp /= 65536.0f;
56     temp -= 46.85f;
57
58     return temp;
59 }
60
```

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

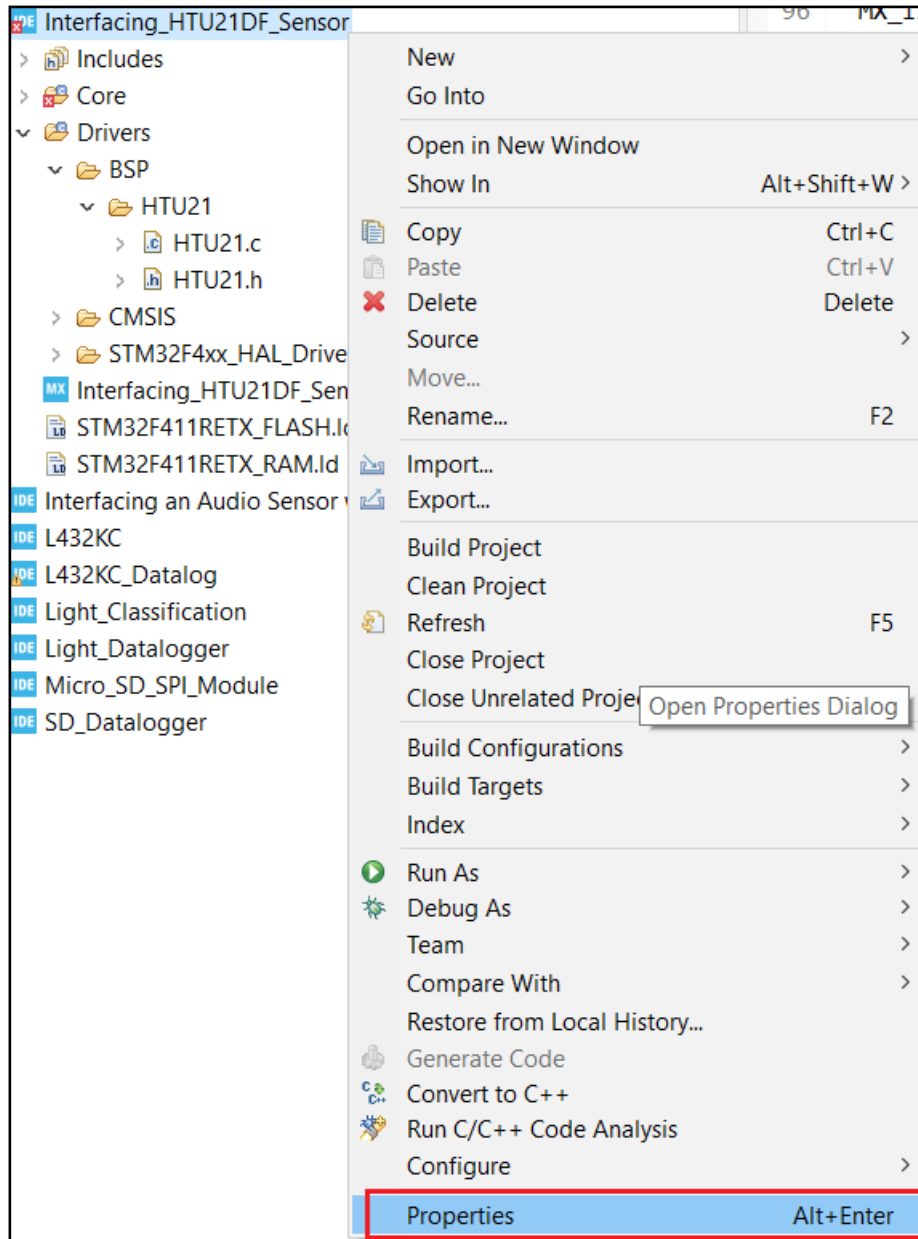
```
18 /* USER CODE END Header */
19 /* Includes -----*/
20 #include "main.h"
21
22 /* Private includes -----*/
23 /* USER CODE BEGIN Includes */
24 #include "stdio.h"
25 #include "string.h"
26 #include "HTU21.h"
27 /* USER CODE END Includes */
28
29 /* Private typedef -----*/
30 /* USER CODE BEGIN PTD */
31
32 /* USER CODE END PTD */
33
34 /* Initialize all configured peripherals */
35 MX_GPIO_Init();
36 MX_USART2_UART_Init();
37 MX_I2C1_Init();
38 /* USER CODE BEGIN 2 */
39 HAL_Delay(1000);
40 HTU21_Reset();
41
42 while (HTU21_Init() != 0x2) {
43     printf("Initializing HTU21 Sensor...\r\n");
44     HAL_Delay(100);
45 }
46 printf("HTU21 Sensor Initialization Success!\r\n");
47 /* USER CODE END 2 */
48
49 /* Infinite loop */
50 /* USER CODE BEGIN WHILE */
51 while (1)
52 {
53     float HTU21_Temp_Value = HTU21_GetTemp();
54     printf("Temperature = %.2f\r\n", HTU21_Temp_Value);
55     HAL_Delay(300);
56
57     float HTU21_Hum_Value = HTU21_GetHum();
58     printf("Humidity = %.2f\r\n", HTU21_Hum_Value);
59     HAL_Delay(300);
60
61 /* USER CODE END WHILE */
62
63 /* USER CODE BEGIN 3 */
64 }
65 }
```

```

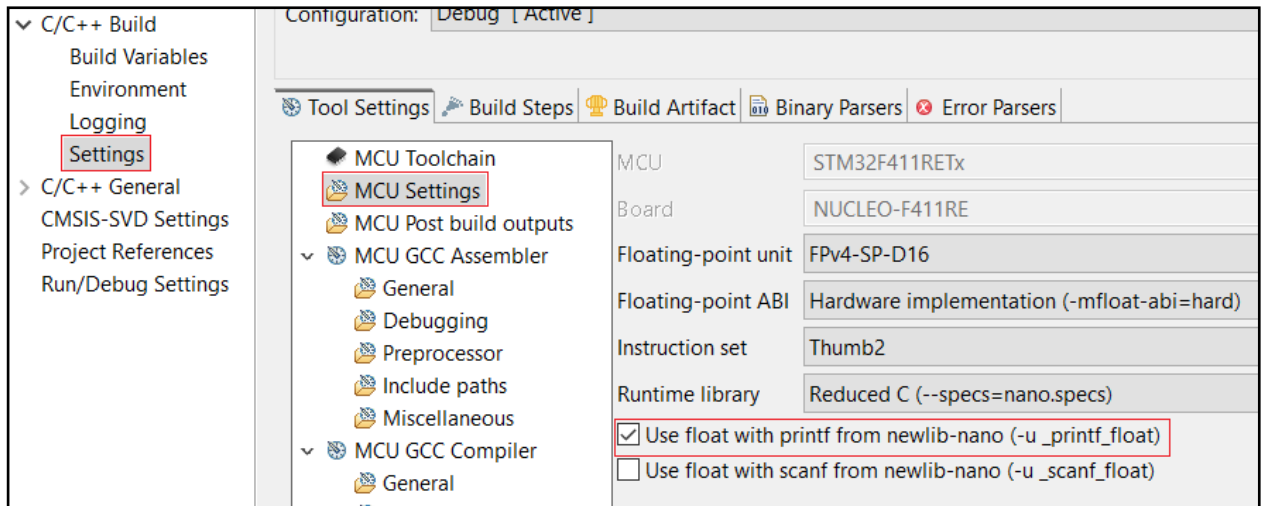
276 /* USER CODE BEGIN 4 */
277 int __io_putchar(int ch) {
278     HAL_UART_Transmit(&huart2, (uint8_t*) &ch, 1, HAL_MAX_DELAY);
279     return ch;
280 }
281 /* USER CODE END 4 */


```

18. Now right click on your project tree, go to **Properties**.




19. After that select **Settings**→**MCU Settings** and enable the **float printf**.



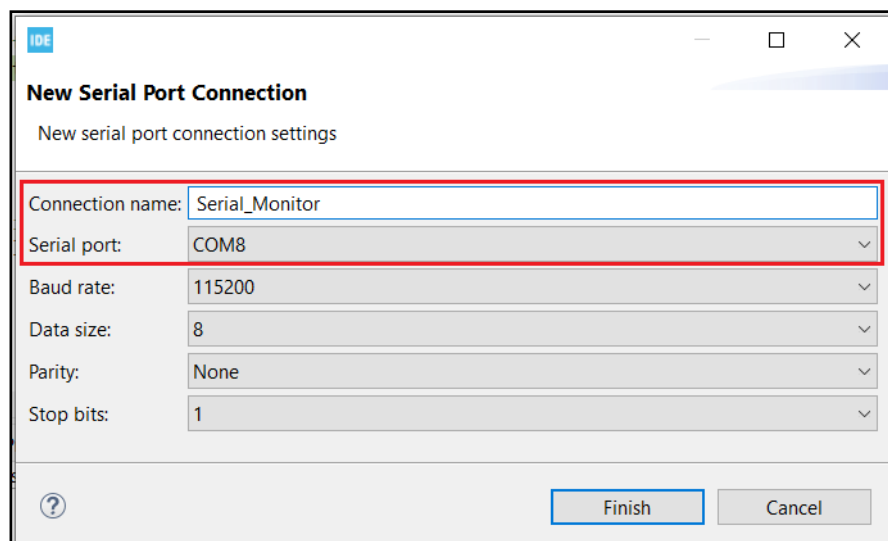
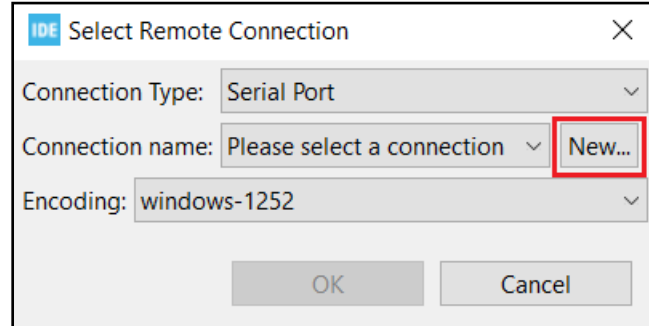
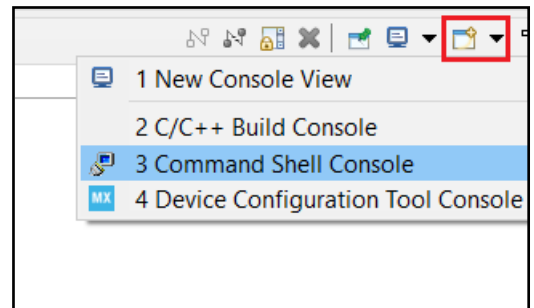
20. Now click on  the build symbol on the top left corner on your Cube IDE. If you have done everything correctly your code should be built without any errors.


```
make -j4 all
arm-none-eabi-size  Interfacing_HTU21DF_Sensor.elf
  text  data  bss  dec  hex filename
 27612   476  2060  30148  75c4 Interfacing_HTU21DF_Sensor.elf
Finished building: default.size.stdout

14:33:44 Build Finished. 0 errors, 0 warnings. (took 736ms)
```


21. 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.

22. In the debug mode, go to the bottom right hand side corner, click on open console. Select the **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.



23. Click on the **Resume** icon  to run your code. You should be able to see the value of light sensor in the **Console**.

```
Humidity = 44.53
Temperature = 28.16
Humidity = 44.53
Temperature = 28.16
Humidity = 44.57
Temperature = 28.16
Humidity = 44.57
Temperature = 28.15
Humidity = 44.66
Temperature = 28.16
Humidity = 44.66
Temperature = 28.15
Humidity = 44.66
Temperature = 28.14
Humidity = 44.70
Temperature = 28.15
Humidity = 44.74
Temperature = 28.15
Humidity = 44.74
Temperature = 28.13
Humidity = 44.79
Temperature = 28.12
Humidity = 44.79
Temperature = 28.12
Humidity = 44.79
Temperature = 28.11
```

24. 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 colour box for the proper understanding of the user. This document is for the use of education purpose only.