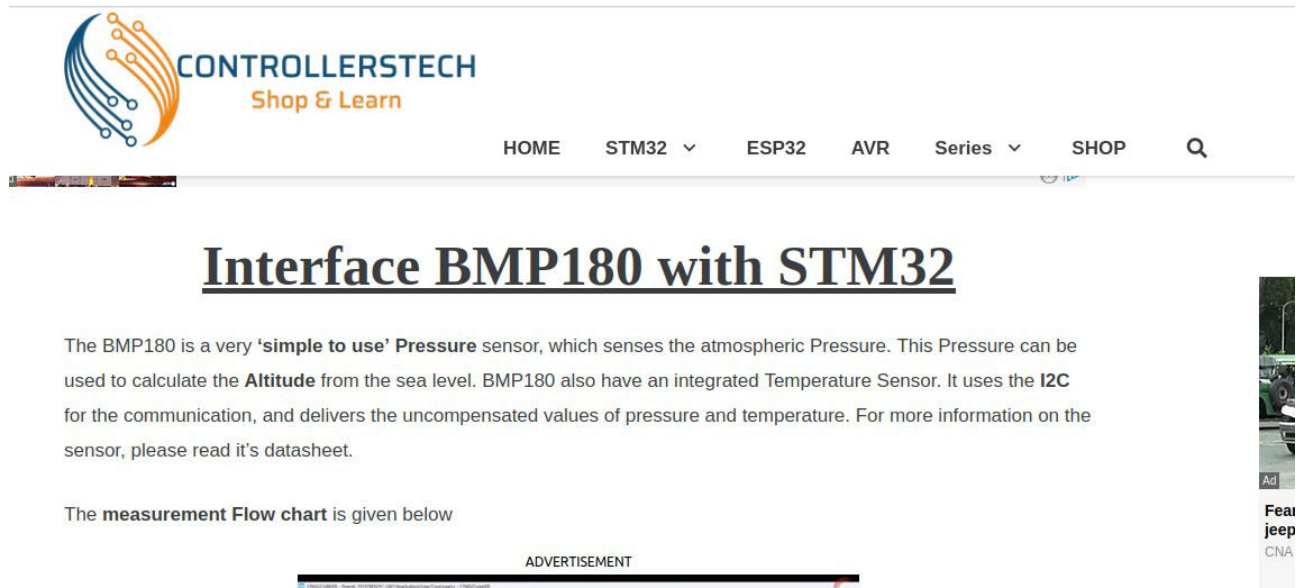


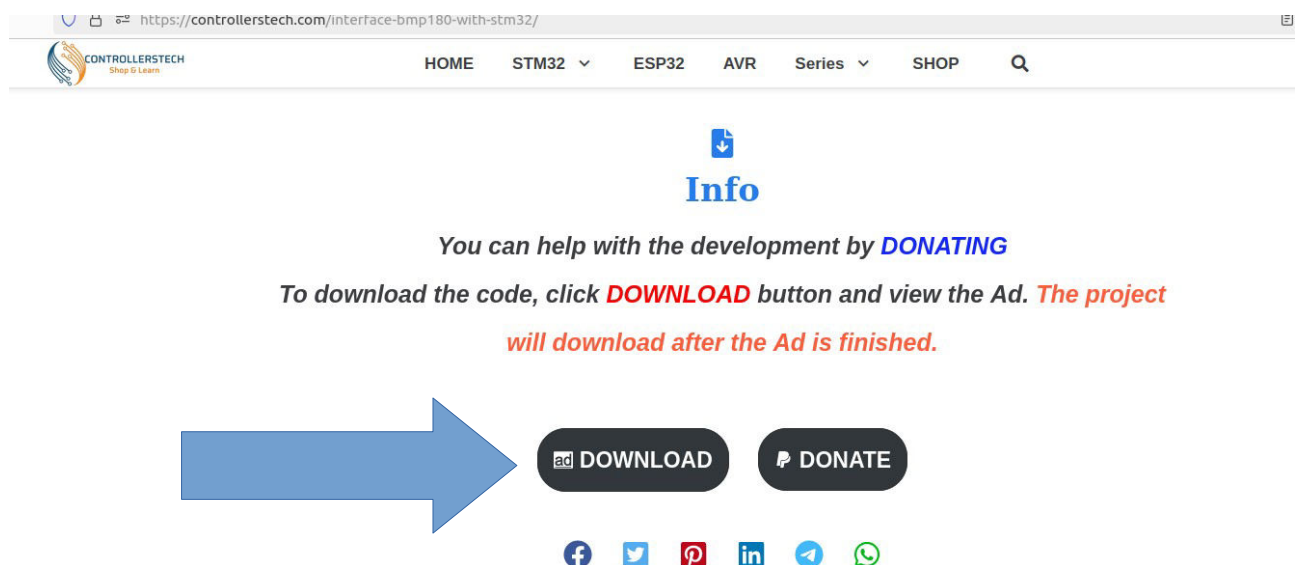
Interfacing with bmp180 sensor with stm32. Essential step has been given below

Step 1: We have to download the BMP180 header file as well as bmp.c so first visit this site <https://controllerstech.com/interface-bmp180-with-stm32>



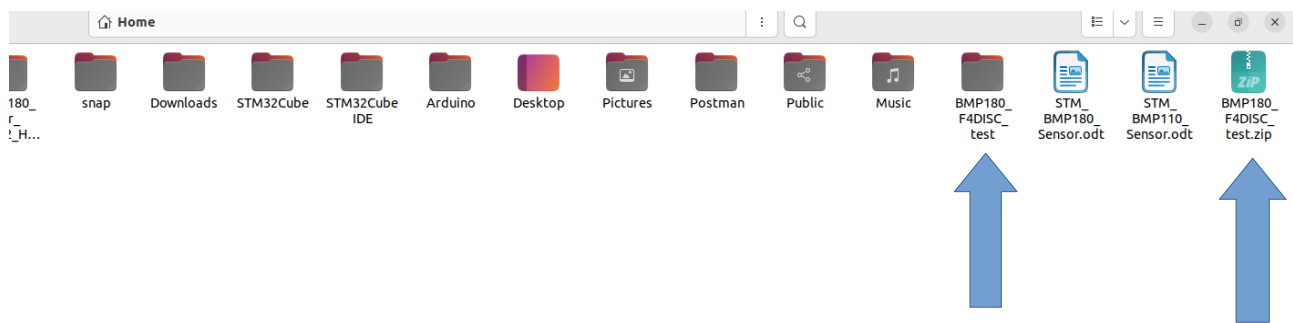
The screenshot shows the Controllerstech website with the article 'Interface BMP180 with STM32'. The website has a navigation bar with links for HOME, STM32, ESP32, AVR, Series, and SHOP. The article text describes the BMP180 as a 'simple to use' Pressure sensor that can calculate altitude from sea level. It also mentions an integrated Temperature Sensor and I2C communication. A measurement flow chart is referenced but not shown. An advertisement for 'Fear jeep CNA' is visible on the right side.

Step 2: Download zip file for BMP180 package and library

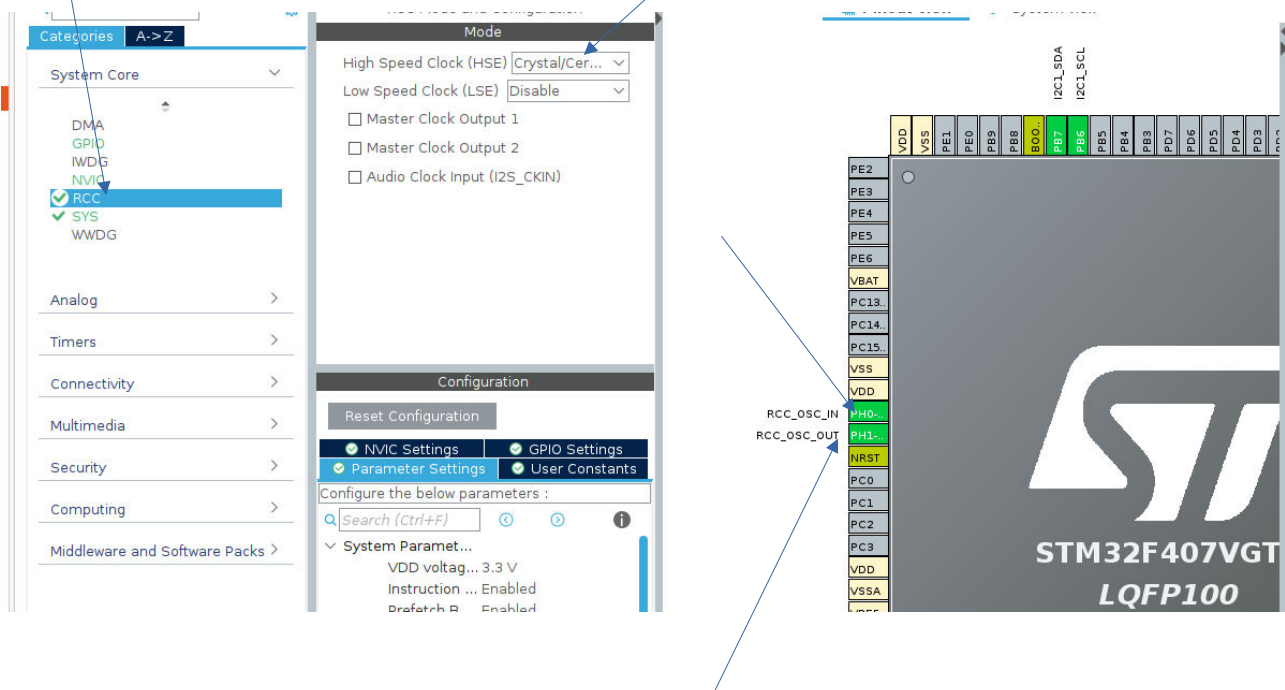


The screenshot shows the 'Info' section of the Controllerstech website. It includes a download icon and the text 'You can help with the development by DONATING'. Below this, it says 'To download the code, click DOWNLOAD button and view the Ad. The project will download after the Ad is finished.' There are two buttons: 'DOWNLOAD' and 'DONATE'. A large blue arrow points to the 'DOWNLOAD' button. At the bottom, there are social media icons for Facebook, Twitter, Pinterest, LinkedIn, Telegram, and WhatsApp.

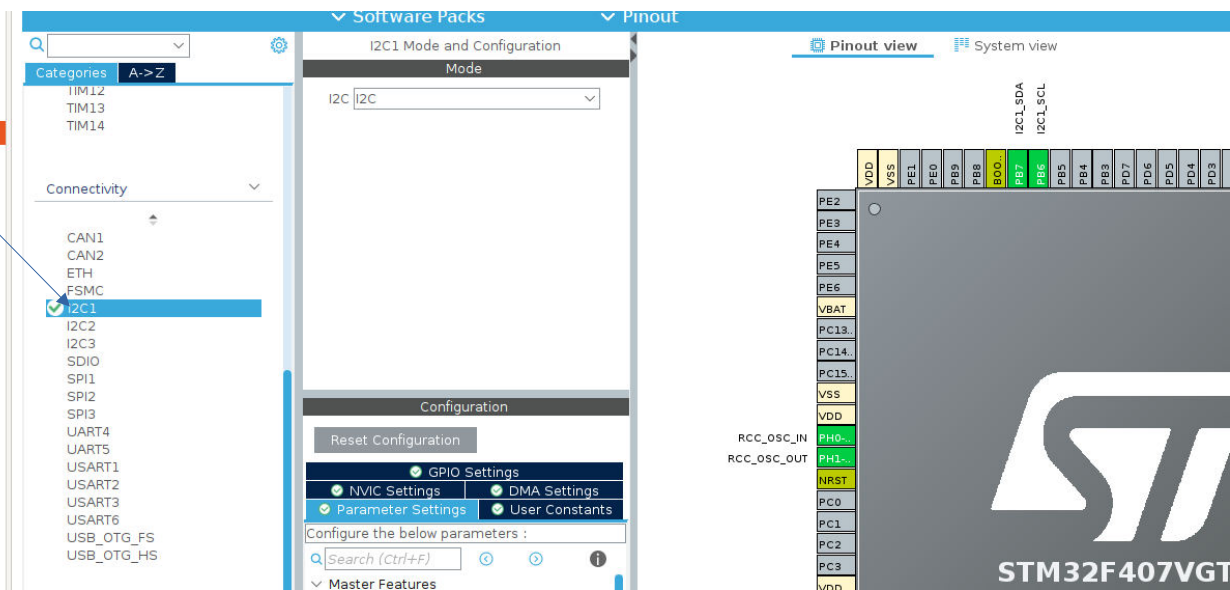
Step 3: Unzip the file for using this



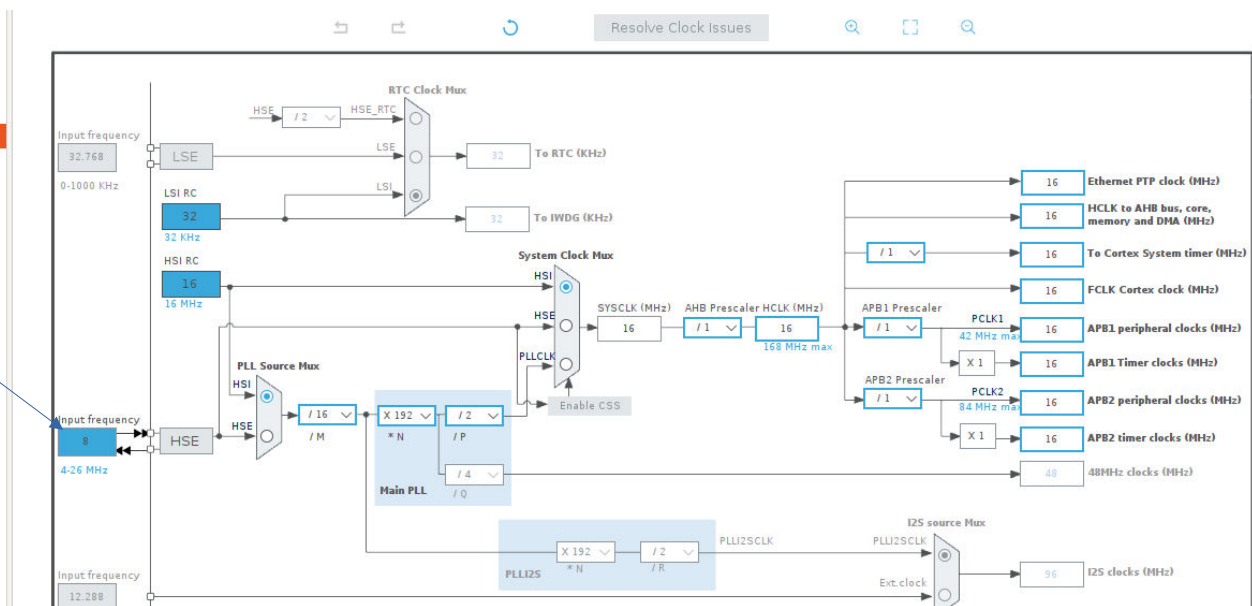
Step 4: Make a project on the stm2 cube ide and then make changes in ioc file something like this
RCC as Crystal/ceramic



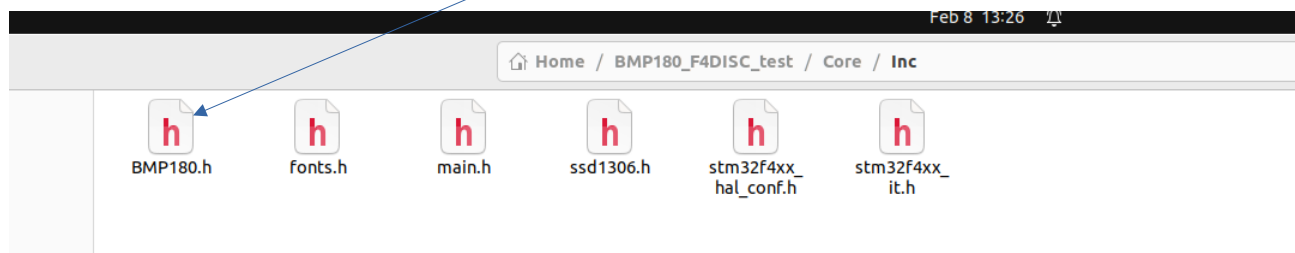
Step 5: Enable i2c feature



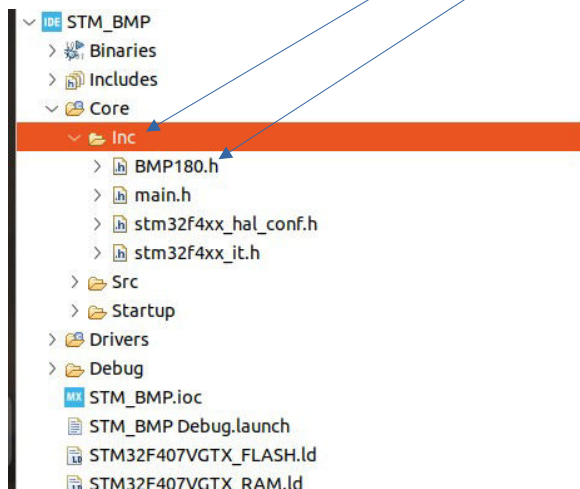
Step 6: Clock configuration has been given below. There is only one change



Step 7: After above step copy BMP180.h



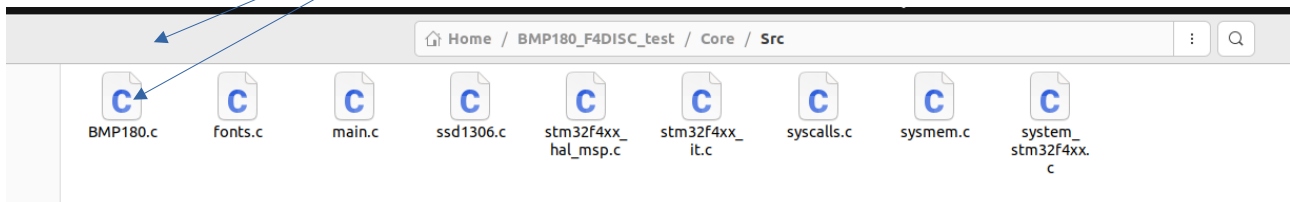
Step 7: Paste it into the inc path



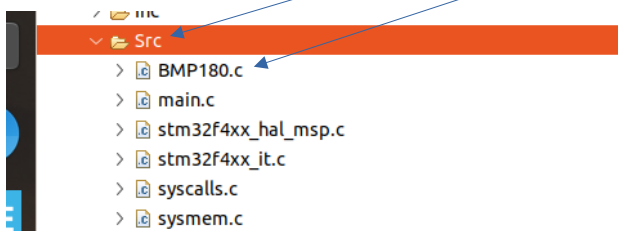
```
4  * @file          : main.c
5  * @brief         : Main program body
6  * *****
7  * @attention
8  *
9  * Copyright (c) 2024 STMicroelectronics.
10 * All rights reserved.
11 *
12 * This software is licensed under terms that can be
13 * in the root directory of this software component.
14 * If no LICENSE file comes with this software, it is
15 *
16 * *****
17 */
18 /* USER CODE END Header */
19 /* Includes -----
20 #include "main.h"
21
22 #include "BMP180.h"
23 #include "stdio.h"
24 /* Private includes -----
25 /* USER CODE BEGIN Includes */
```

Step
8 :

Then copy the BMP180.c from downloaded folder

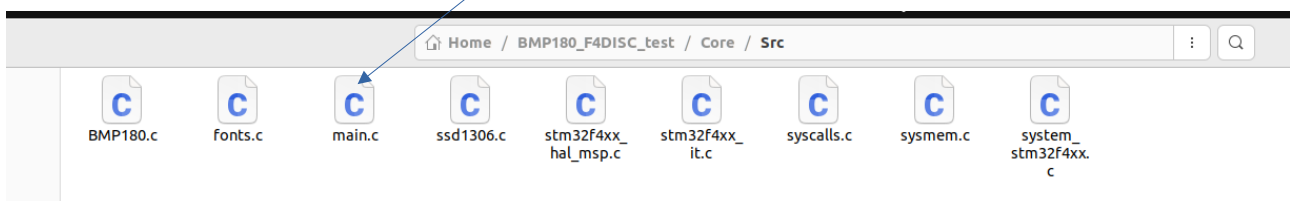


Step 8: Paste into the src path



```
10 * All rights reserved.
11 *
12 * This software is licensed under terms that can be found
13 * in the root directory of this software component.
14 * If no LICENSE file comes with this software, it is provided
15 *
16 * *****
17 */
18 /* USER CODE END Header */
19 /* Includes -----
```

Step 9: make changes in the main.c file according to downloaded file as per main.c



Step 9 : Changes highlighted with red color

```
/* USER CODE BEGIN Header */
/**
 * *****
 * @file           : main.c
 * @brief          : Main program body
 * *****
 * @attention
 *
 * Copyright (c) 2024 STMicroelectronics.
 * All rights reserved.
 *
 * This software is licensed under terms that can be found in the LICENSE file
 * in the root directory of this software component.
 * If no LICENSE file comes with this software, it is provided AS-IS.
 *
 * *****
 */
/* USER CODE END Header */
/* Includes -----*/
#include "main.h"

#include "BMP180.h"
#include "stdio.h"
/* Private includes -----*/
/* USER CODE BEGIN Includes */

/* USER CODE END Includes */

/* Private typedef -----*/
/* USER CODE BEGIN PTD */

/* USER CODE END PTD */

/* Private define -----*/
/* USER CODE BEGIN PD */

/* USER CODE END PD */

/* Private macro -----*/
/* USER CODE BEGIN PM */

/* USER CODE END PM */

/* Private variables -----*/
I2C_HandleTypeDef hi2c1;

/* USER CODE BEGIN PV */

/* USER CODE END PV */

/* Private function prototypes -----*/
void SystemClock_Config(void);
static void MX_GPIO_Init(void);
static void MX_I2C1_Init(void);
/* USER CODE BEGIN PFP */

/* USER CODE END PFP */

/* Private user code -----*/
/* USER CODE BEGIN 0 */
float Temperature = 0;
```

```

float Pressure = 0;
float Altitude = 0;

char Temperature1[10];
char Pressure1[10];
char Altitude1[10];
/* USER CODE END 0 */

/**
 * @brief The application entry point.
 * @retval int
 */
int main(void)
{
    /* USER CODE BEGIN 1 */

    /* USER CODE END 1 */

    /* MCU Configuration-----*/

    /* Reset of all peripherals, Initializes the Flash interface and the Systick.
    */
    HAL_Init();

    /* USER CODE BEGIN Init */

    /* USER CODE END Init */

    /* Configure the system clock */
    SystemClock_Config();

    /* USER CODE BEGIN SysInit */

    /* USER CODE END SysInit */

    /* Initialize all configured peripherals */
    MX_GPIO_Init();
    MX_I2C1_Init();
    /* USER CODE BEGIN 2 */
    BMP180_Start();
    /* USER CODE END 2 */

    /* Infinite loop */
    /* USER CODE BEGIN WHILE */
    while (1)
    {
        Temperature = BMP180_GetTemp();

        Pressure = BMP180_GetPress (0);

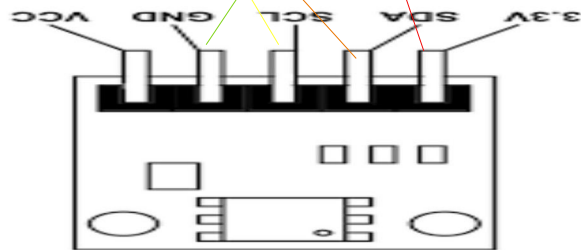
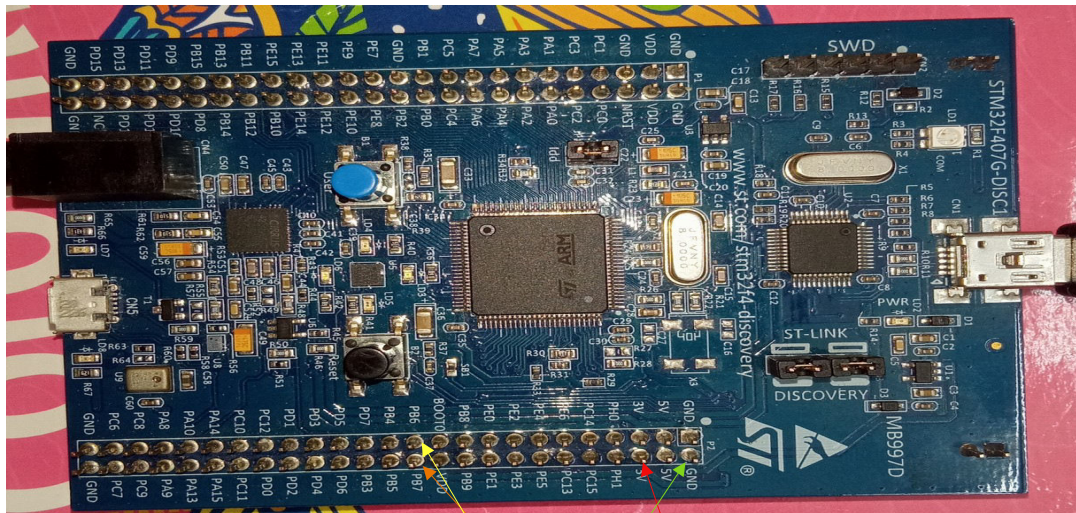
        Altitude = BMP180_GetAlt(0);
        /* USER CODE END WHILE */

        /* USER CODE BEGIN 3 */
    }
    /* USER CODE END 3 */
}

```

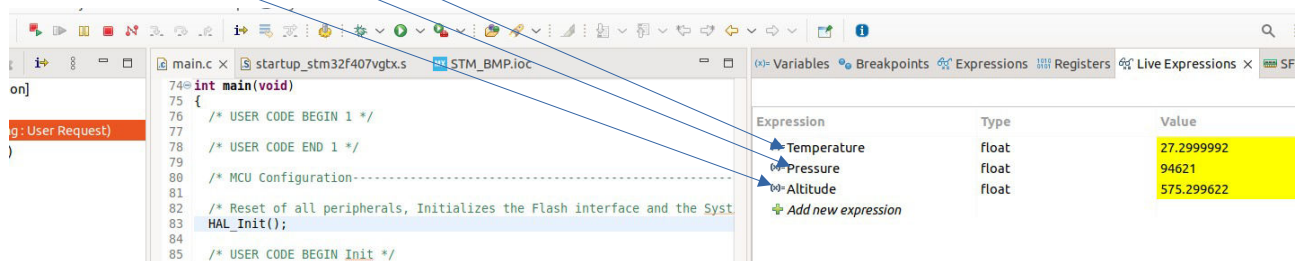
Note: No further changes in this code

Step 10: Do not connect vcc and 3.3 together always use one pin and according to i2c configuration your pin may change configuration



Step 11 : During the program execution you need to declare 3 variable in the Live expression such as

- 1.Temprature
- 2.Presssure
- 3.Altitude

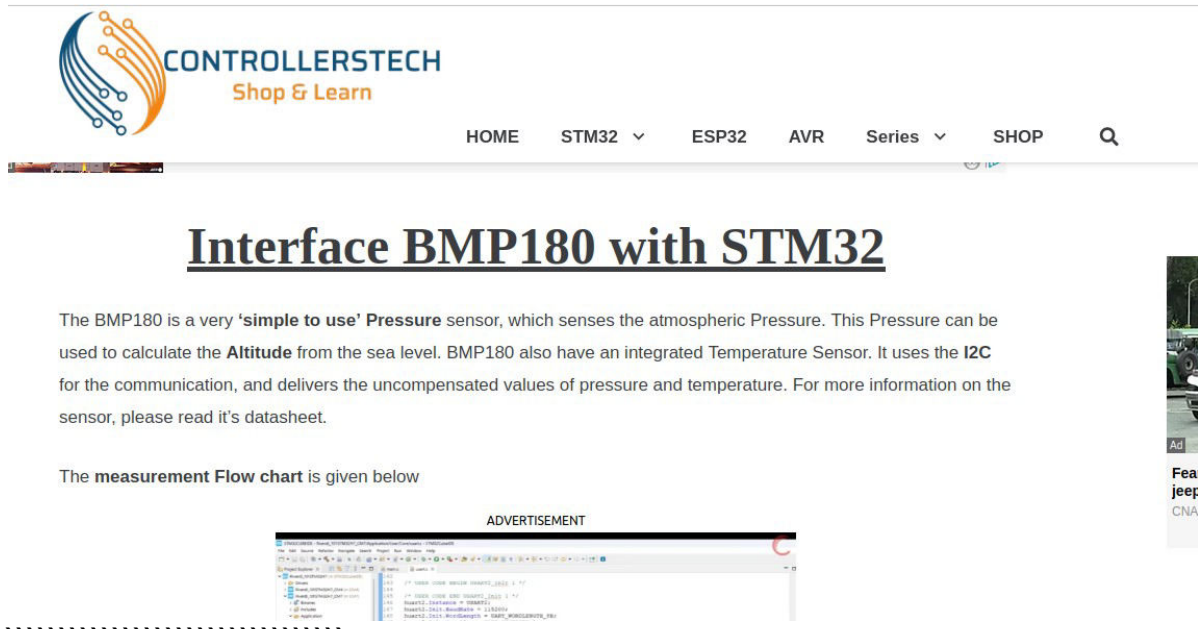


INTERFACING BMP180 WITH OLED DISPLAY MODULE(0.96 inch I2C-WHITE)

Objective : In this project we will interface with oled display for showing the temprature ,pressure ,and altitude data with STM32.Essential step has been given below

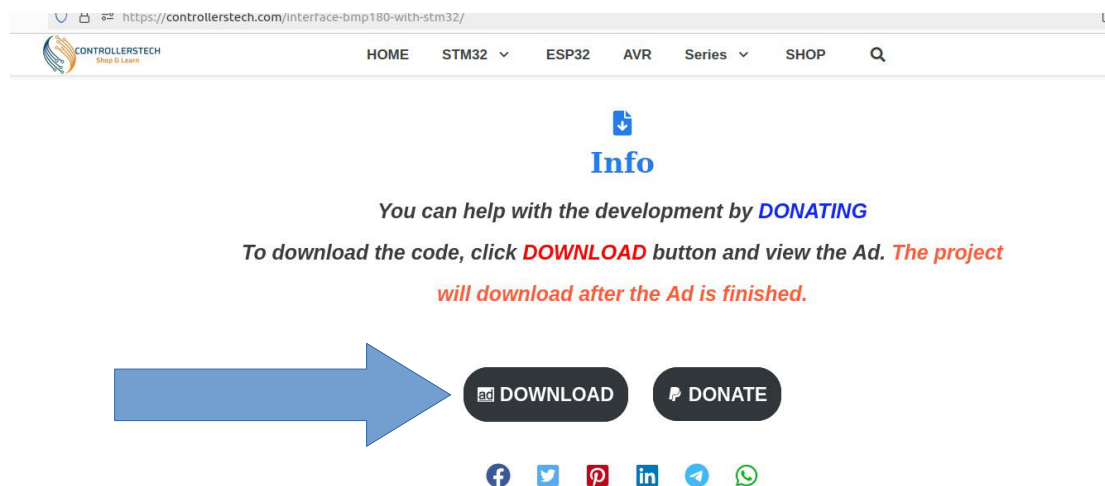
Step 1: We have to download the BMP180 library zip file so first visit this site

<https://controllerstech.com/interface-bmp180-with-stm32>



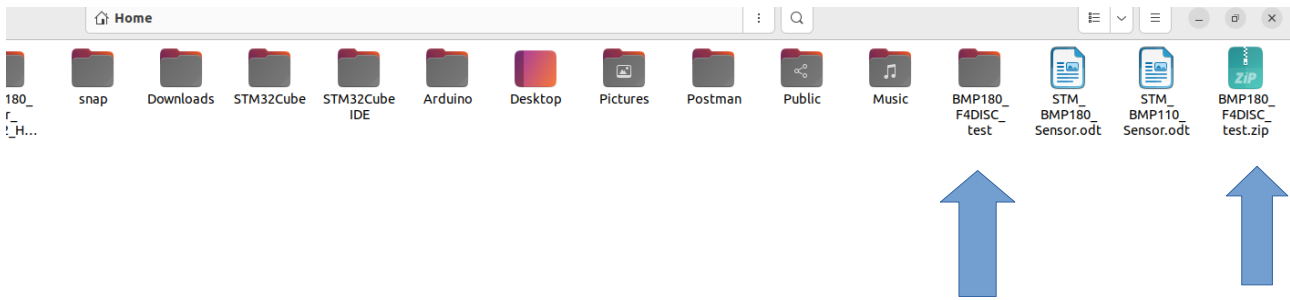
The screenshot shows the Controllerstech website with the article title "Interface BMP180 with STM32". The article text describes the BMP180 sensor as a 'simple to use' Pressure sensor that can calculate altitude from sea level pressure. It mentions the sensor uses I2C for communication and provides uncompensated pressure and temperature values. A measurement flow chart is referenced but not shown. An advertisement for a Jeep CNA is visible on the right side of the page.

Step 2: Download zip file for BMP180 package and library

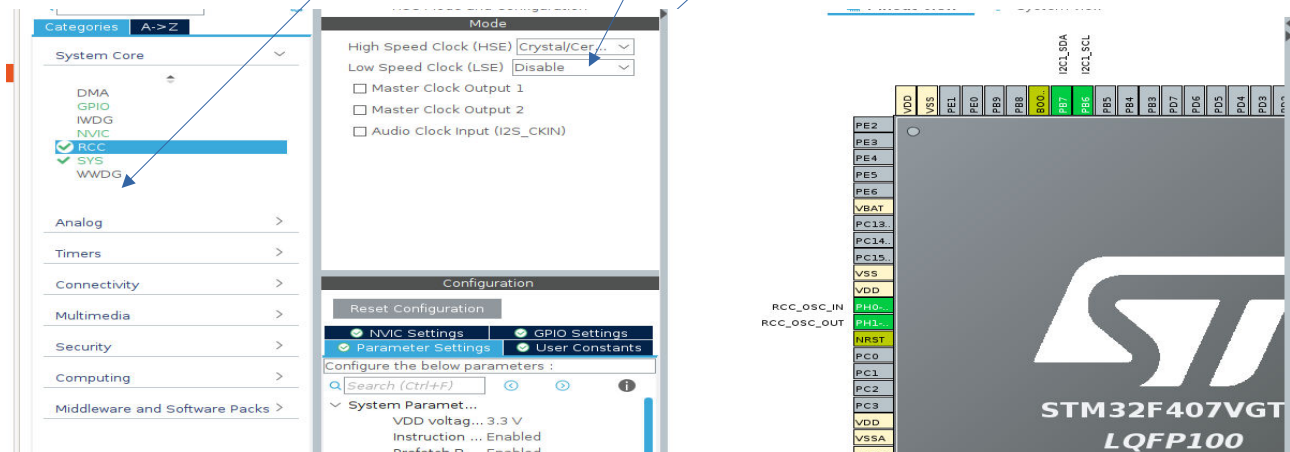


The screenshot shows the Controllerstech website with the article title "Interface BMP180 with STM32". Below the article title, there is a blue arrow pointing down to the word "Info". Below "Info", there is a line of text: "You can help with the development by DONATING". Below this, there is a line of text: "To download the code, click **DOWNLOAD** button and view the Ad. **The project will download after the Ad is finished.**". Below this text, there is a large blue arrow pointing right towards a button labeled "ad DOWNLOAD" and a button labeled "DONATE". Below these buttons, there are social media icons for Facebook, Twitter, Pinterest, LinkedIn, Telegram, and WhatsApp.

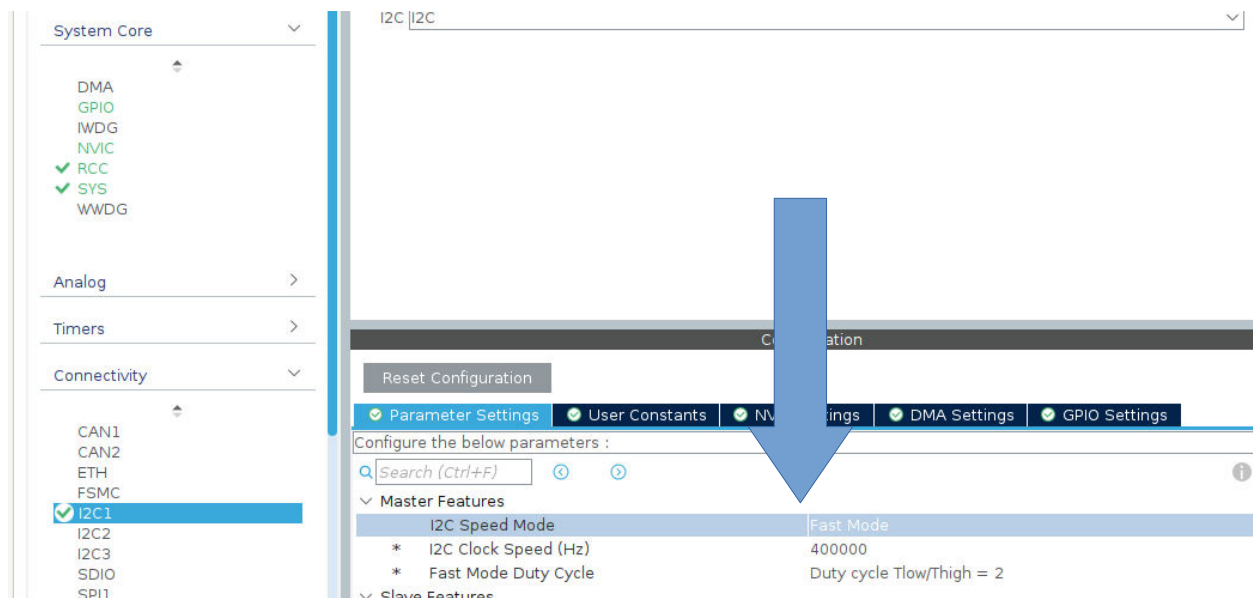
Step 3: Unzip the file



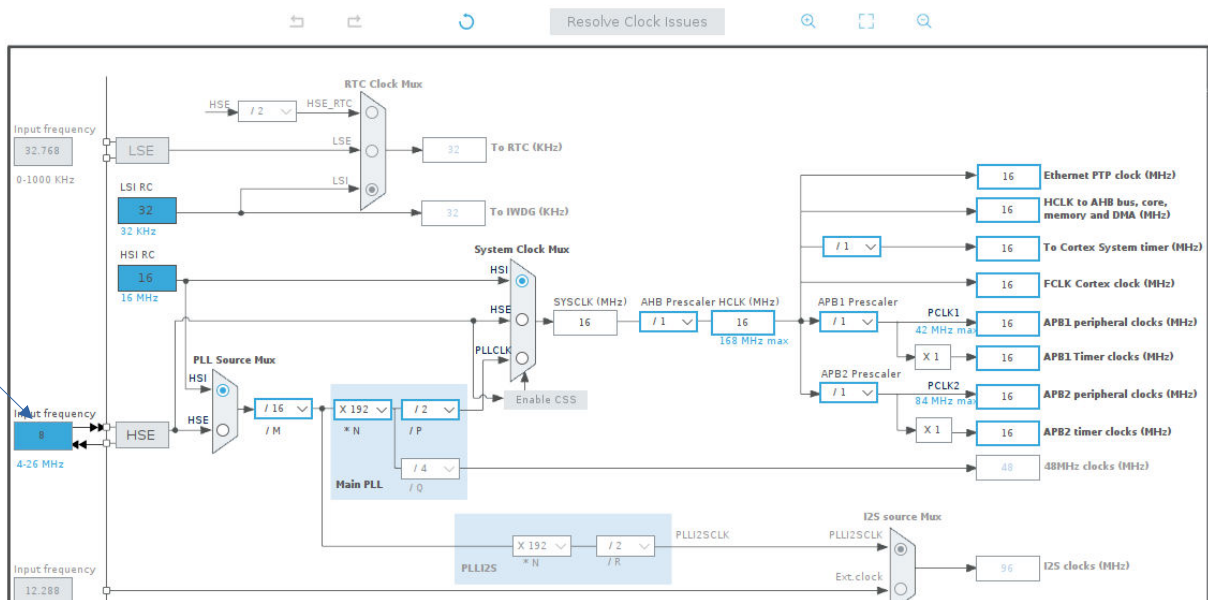
Step 4: Make a project on the stm2 cube ide and then make changes in ioc file something like this



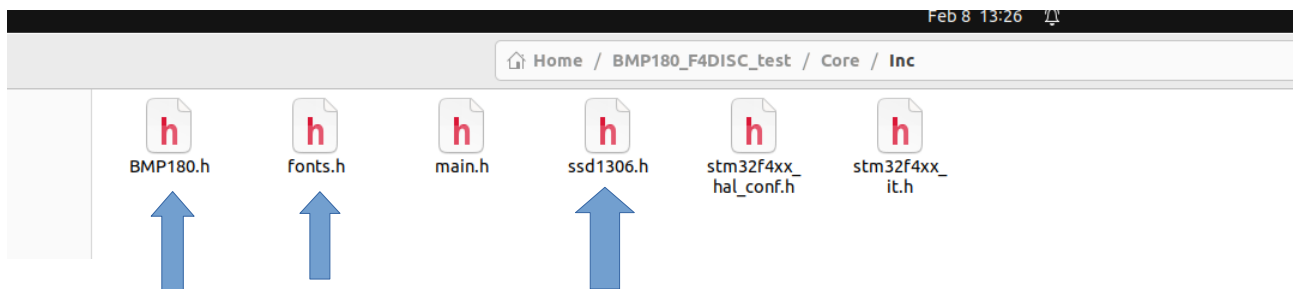
Step 5: Enable i2c feature in **Fast mode**



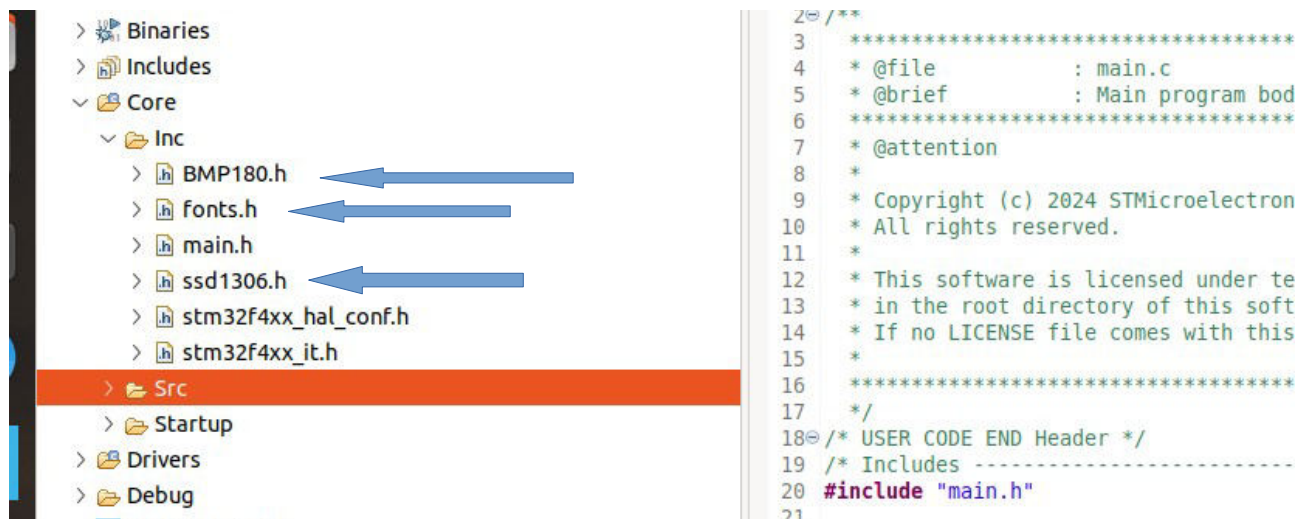
Step 5 : Clock Configuration only one change here you can use any frequency as per use



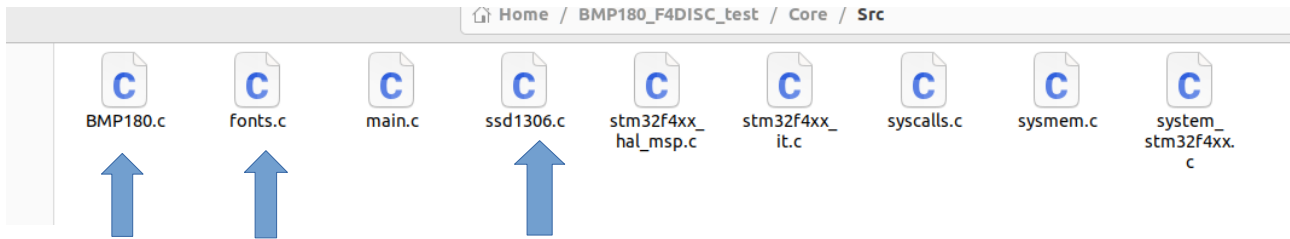
Step 7 : After above step copy BMP180.h fonts.h nad ssd1306.h and paste in the inc path.Take this file from downloaded file



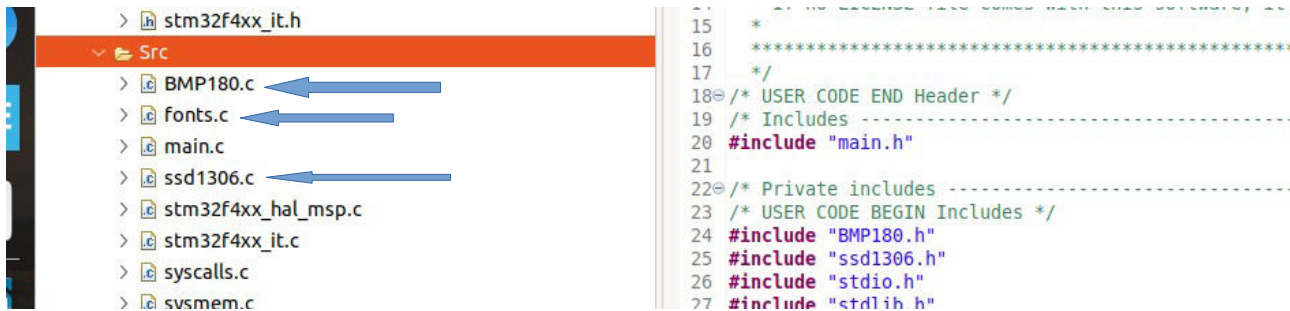
Step 8 : Paste in inc path



Step 9: Copy three file from downloaded zip file



Step 9: Paste in src path



Step 10 :

```

/* USER CODE BEGIN Header */
/**
 * @file           : main.c
 * @brief          : Main program body
 * @attention
 *
 * Copyright (c) 2024 STMicroelectronics.
 * All rights reserved.
 *
 * This software is licensed under terms that can be found in the LICENSE file
 * in the root directory of this software component.
 * If no LICENSE file comes with this software, it is provided AS-IS.
 */
/* USER CODE END Header */
/* Includes -----*/
#include "main.h"

/* Private includes -----*/
/* USER CODE BEGIN Includes */
#include "BMP180.h"
#include "ssd1306.h"
#include "stdio.h"
#include "stdlib.h"
#include "string.h"

/* USER CODE END Includes */

/* Private typedef -----*/
/* USER CODE BEGIN PTD */

```

```

/* USER CODE END PTD */

/* Private define -----*/
/* USER CODE BEGIN PD */

/* USER CODE END PD */

/* Private macro -----*/
/* USER CODE BEGIN PM */

/* USER CODE END PM */

/* Private variables -----*/
I2C_HandleTypeDef hi2c1;

/* USER CODE BEGIN PV */

/* USER CODE END PV */

/* Private function prototypes -----*/
void SystemClock_Config(void);
static void MX_GPIO_Init(void);
static void MX_I2C1_Init(void);
/* USER CODE BEGIN PFP */

/* USER CODE END PFP */

/* Private user code -----*/
/* USER CODE BEGIN 0 */
float Temperature = 0;
float Pressure = 0;
float Altitude = 0;

char Temperature1[10];
char Pressure1[10];
char Altitude1[10];
/* USER CODE END 0 */

/**
 * @brief The application entry point.
 * @retval int
 */
int main(void)
{
    /* USER CODE BEGIN 1 */

    /* USER CODE END 1 */

    /* MCU Configuration-----*/

    /* Reset of all peripherals, Initializes the Flash interface and the Systick.
    */
    HAL_Init();

    /* USER CODE BEGIN Init */

    /* USER CODE END Init */

    /* Configure the system clock */
    SystemClock_Config();

    /* USER CODE BEGIN SysInit */

```

```

/* USER CODE END SysInit */

/* Initialize all configured peripherals */
MX_GPIO_Init();
MX_I2C1_Init();
/* USER CODE BEGIN 2 */

BMP180_Start();

SSD1306_Init();

SSD1306_GotoXY (35,0);
SSD1306_Puts ("BMP180", &Font_11x18, 1);
SSD1306_GotoXY (10,20);
SSD1306_Puts ("Barometric", &Font_11x18, 1);
SSD1306_GotoXY (30,40);
SSD1306_Puts ("Sensor", &Font_11x18, 1);
SSD1306_GotoXY (20,40);
SSD1306_UpdateScreen(); //display
HAL_Delay(2000);
SSD1306_Clear();
/* USER CODE END 2 */

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

    Temperature = BMP180_GetTemp();

    Pressure = BMP180_GetPress (0);

    Altitude = BMP180_GetAlt(0);

    /*****To display on OLED*****/

    SSD1306_GotoXY (35,0);
    SSD1306_Puts ("BMP180", &Font_11x18, 1);
    SSD1306_GotoXY (0,0);
    SSD1306_Puts ("Temperature", &Font_11x18, 1);
    SSD1306_GotoXY (20,40);
    sprintf(Temperature1, "%.2f", Temperature);
    SSD1306_Puts(Temperature1, &Font_11x18, 1);
    SSD1306_DrawCircle(80, 40, 2, 1); //To print degree only
    SSD1306_GotoXY (85,40); //To print celcius
    SSD1306_Puts ("C", &Font_11x18, 1);
    SSD1306_UpdateScreen(); //display
    HAL_Delay(2000);
    SSD1306_Clear();

    SSD1306_GotoXY (20,0);
    SSD1306_Puts ("Pressure", &Font_11x18, 1);
    SSD1306_GotoXY (10,40);
    sprintf(Pressure1, "%.2f", Pressure);
    SSD1306_Puts(Pressure1, &Font_11x18, 1);
    SSD1306_GotoXY (100,40);
    SSD1306_Puts ("pa", &Font_11x18, 1);
    SSD1306_UpdateScreen(); //display
    HAL_Delay(2000);
    SSD1306_Clear();

    SSD1306_GotoXY (20,0);
    SSD1306_Puts ("Altitude", &Font_11x18, 1);

```

```

SSD1306_GotoXY (15,40);
printf(Altitude1, "%.2f", Altitude);
SSD1306_Puts(Altitude1, &Font_11x18, 1);
SSD1306_GotoXY (90,40);
SSD1306_Puts ("m", &Font_11x18, 1);
SSD1306_UpdateScreen(); //display
HAL_Delay(2000);
SSD1306_Clear();

    HAL_Delay (2000);
/* USER CODE END WHILE */

/* USER CODE BEGIN 3 */
}
/* USER CODE END 3 */
}

=====No further changes =====

```

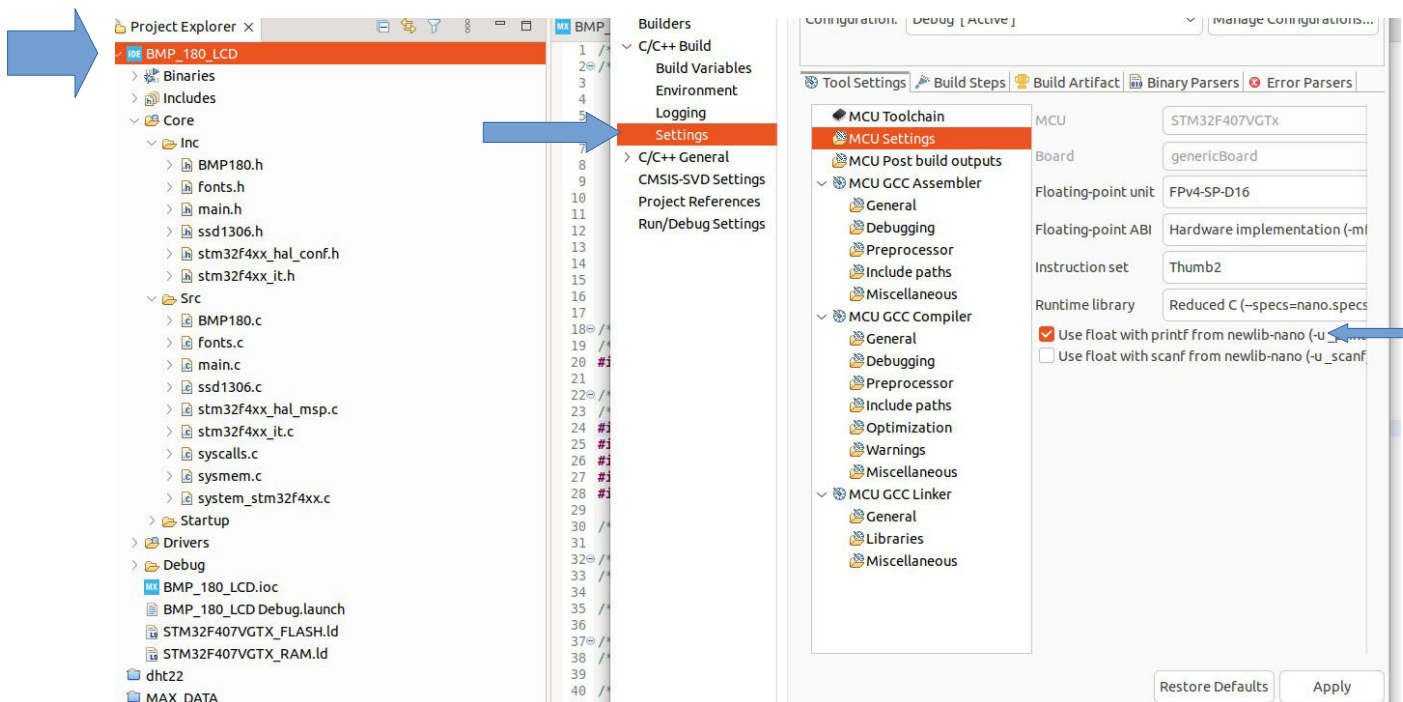
Step 11 : you may get one problem regarding below

```

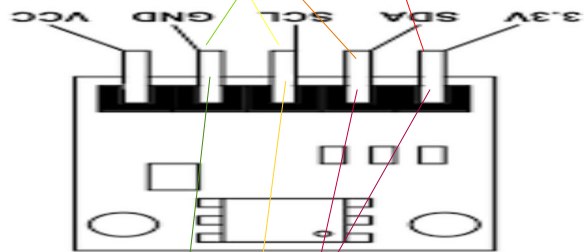
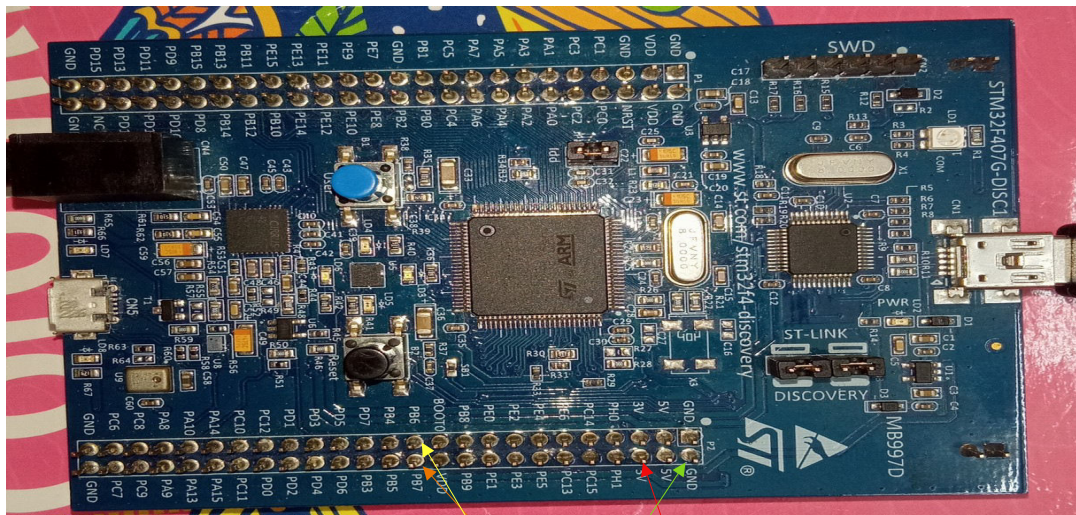
printf(Temperature1, "%.2f", Temperature);
printf(Pressure1, "%.2f", Pressure);
printf(Altitude1, "%.2f", Altitude);

```

for solving this make changes in setting (BMP180_LCD then Properties then settings then MCU setting then use float with printf option click and apply



Step 10: Do not connect vcc and 3.3 together always use one pin and according to i2c configuration your pin may change configuration



Step 11 : During the program execution you need to declare 3 variable in the Live expression such as

- 1.Temprature
- 2.Pressure
- 3.Altitude

The screenshot shows an IDE with a C code editor and a live expressions table. The code is in a file named `main.c` and includes comments for user code and MCU configuration. The live expressions table displays the values of `Temperature`, `Pressure`, and `Altitude` as floats.

Expression	Type	Value
<code>Temperature</code>	float	27.2999992
<code>Pressure</code>	float	94621
<code>Altitude</code>	float	575.299622