

EXPERIMENT

TLV493D 3D MAGNETOMETER INTERFACING WITH DEV BOARD/NODE

What will you learn from this module:

- Interfacing with the help of I2C protocol.
- Measurement of 3D movements using TLV493D sensor using nrf dev board/node.
- Configuration of overlay file, device tree and prj file for enabling hardware device.

Requirements:

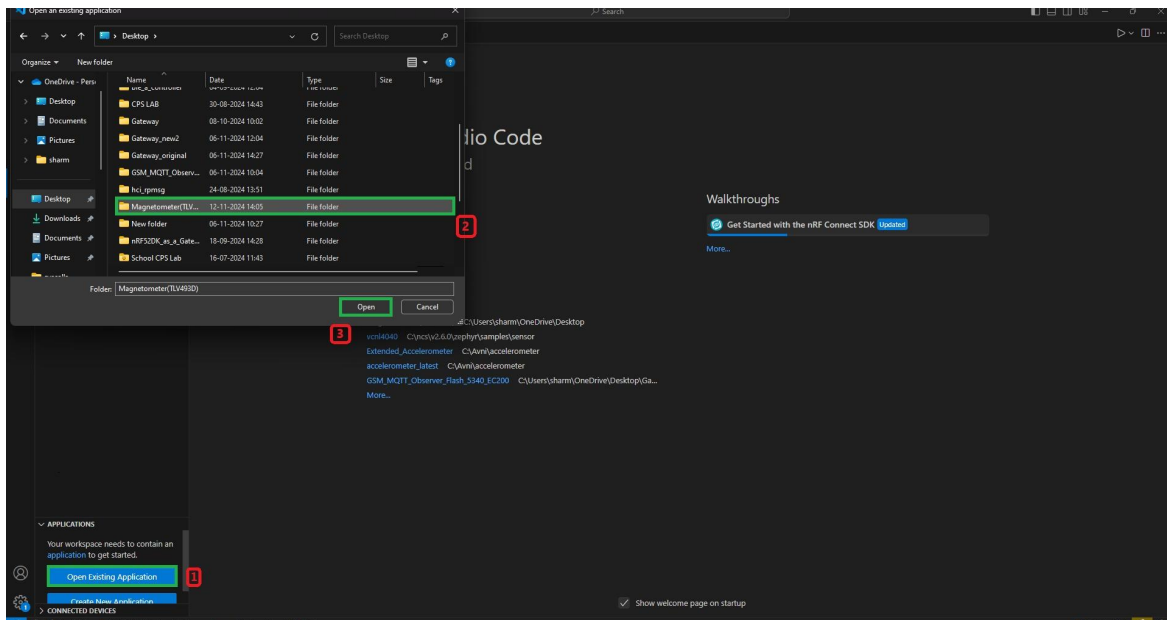
- nRF connect desktop software.
- nRF Command line tools.
- Visual studio code.
- USB cable.
- nRF52832 Development Board/Node.
- TLV493D Sensor.
- TTL Device.

Prerequisites:

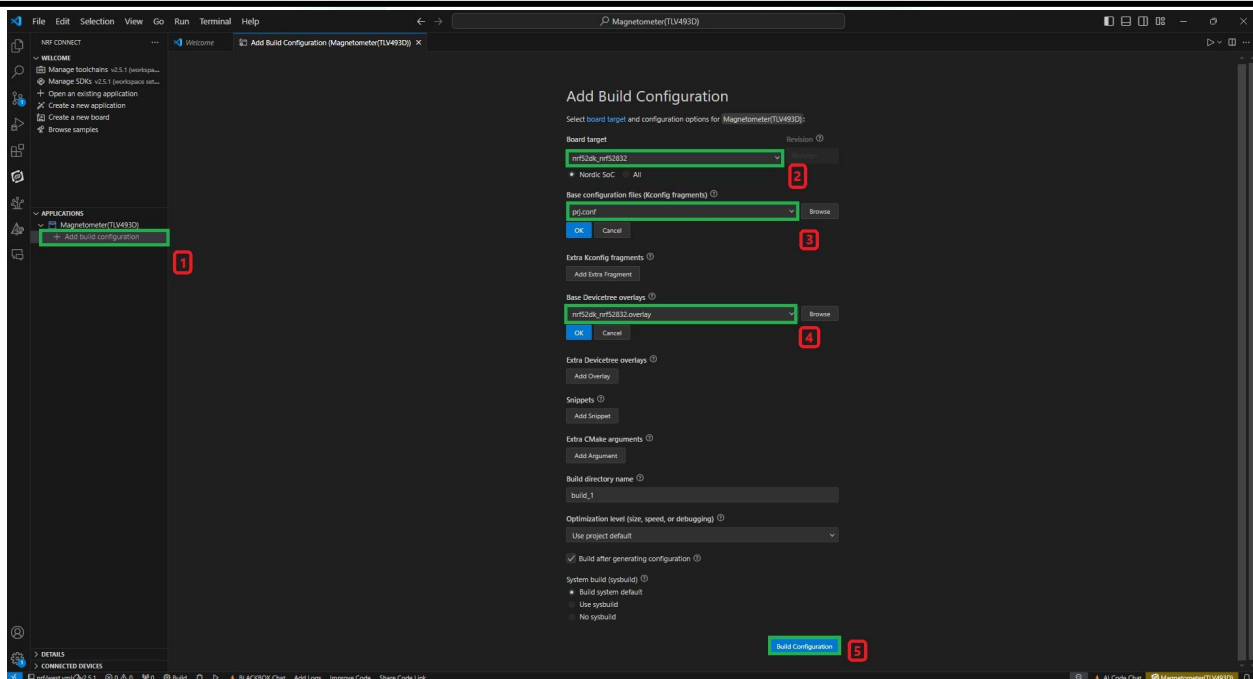
- Basic knowledge of C/C++.
- Basic knowledge of communication protocol.
- Basic project setup.

➤ Setup and Configuration:

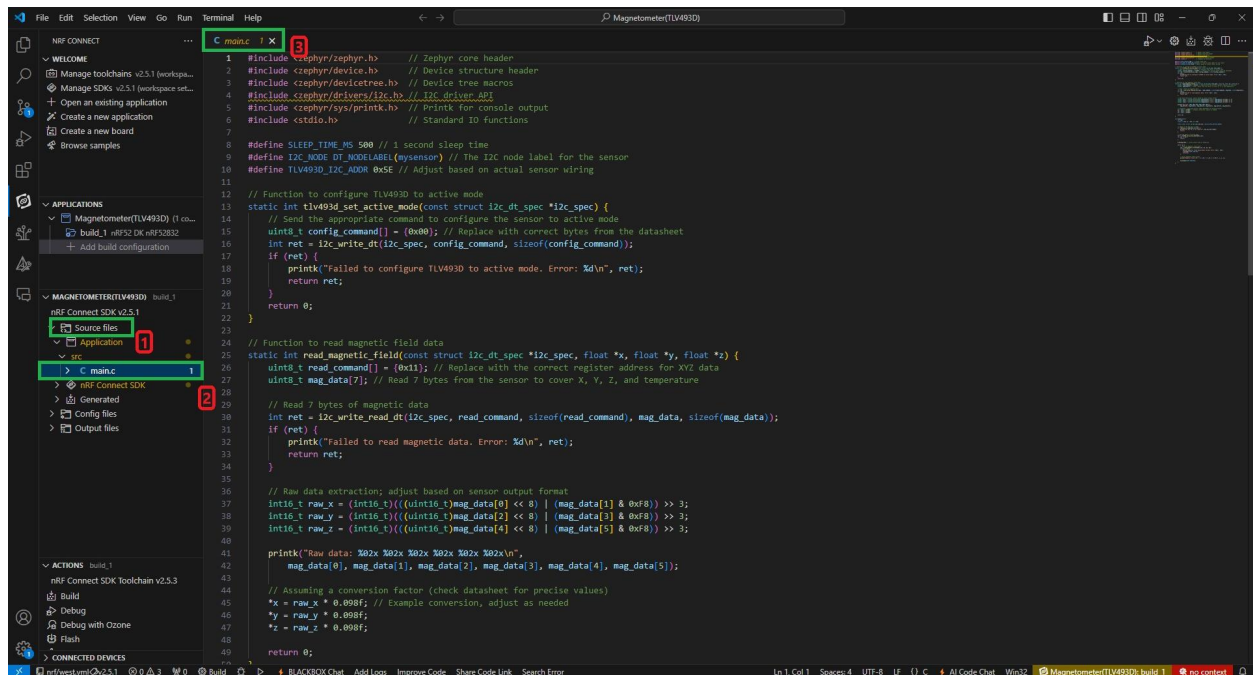
- Open VS Code and click on **Open Existing Application [1]** > click on TLV493D [2] > **Open [3]** as shown in the picture below.



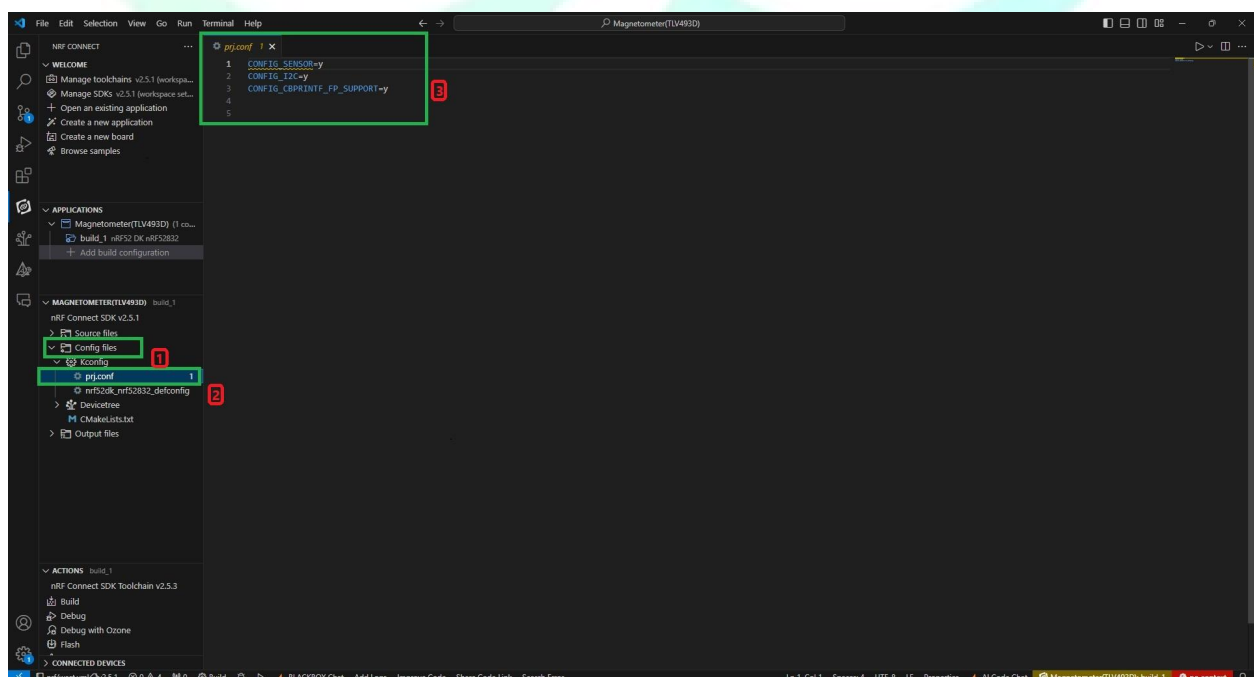
- Click on **Create new build configuration [1]**. Here you can change the board version, if you are using nRF52832, then select **nrf52dk_nrf52832 [2]** or you can change from dropdown menu for another version like nRF52833 etc.
- Click on the Configuration and select **prj.conf [3]** from dropdown menu and then click on the device tree overlays and select **nrf52dk_nrf52832.overlay[4]**.
- Then click on the **Build Configuration [5]** as shown below in the picture.



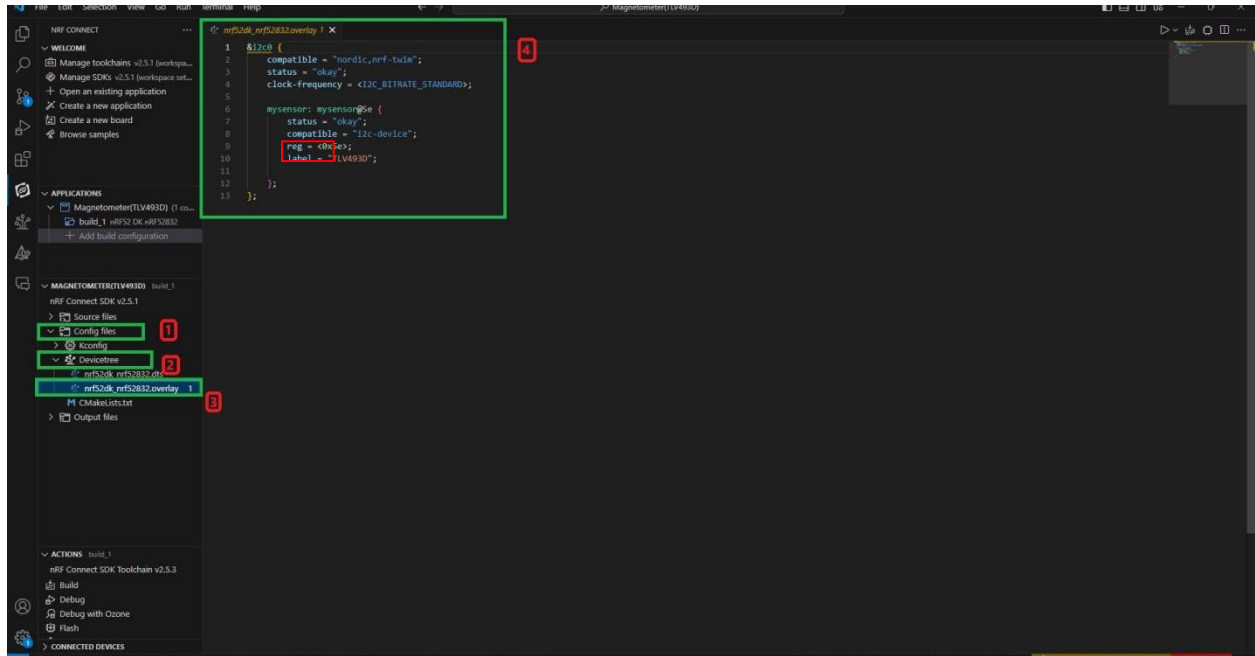
- Go to source file, click **source file [1]** > click on **Application** > click on **src** > click on **main.c [2]**.
- By Clicking on **main.c** file and you will see the code on your screen **[3]**.



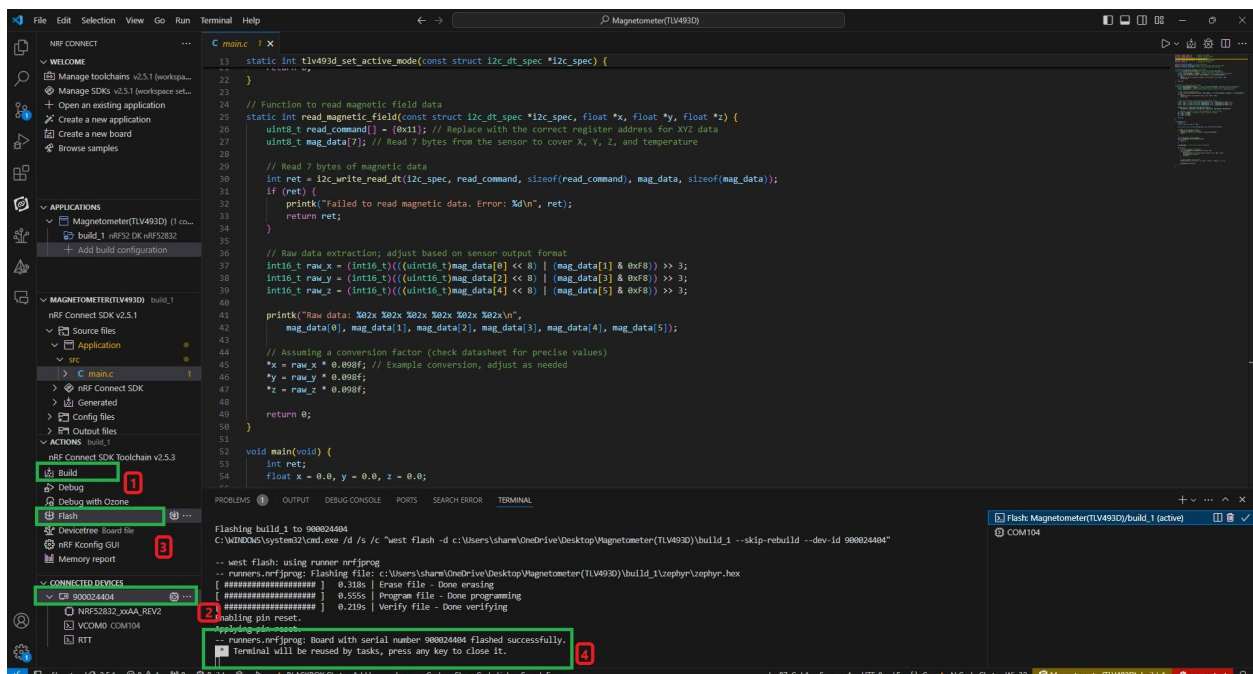
- To configure the prj configuration, click on **Config files [1]** > click on **Kconfig** > click on **prj.conf [2]**.
- The prj configuration will appear on your screen **[3]** as shown in the picture below.

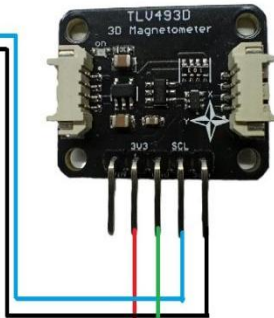


- To configure the i2c protocol, you have to enable it in the **.overlay** file.
- Click on the **Config files [1]** > click on **Kconfig** > click on **Devicetree [2]** > click on **nrf52dk_nrf52832.overlay [3]**.
- The .overlay file will appear on your screen and add the given code to the .overlay file as shown in the picture given below [4].



- Click on **Build [1]** configuration again and check the **CONNECTED DEVICES [2]**.
- If device id is visible, then **Flash [3]** the code in Dev Kit.
- If **flashed successfully [4]** message is displayed on serial terminal, then flash process is complete.





GND -> GND

VDD(3.3V) -> VDD

PO.26 -> SDA

PO.27 -> SCL

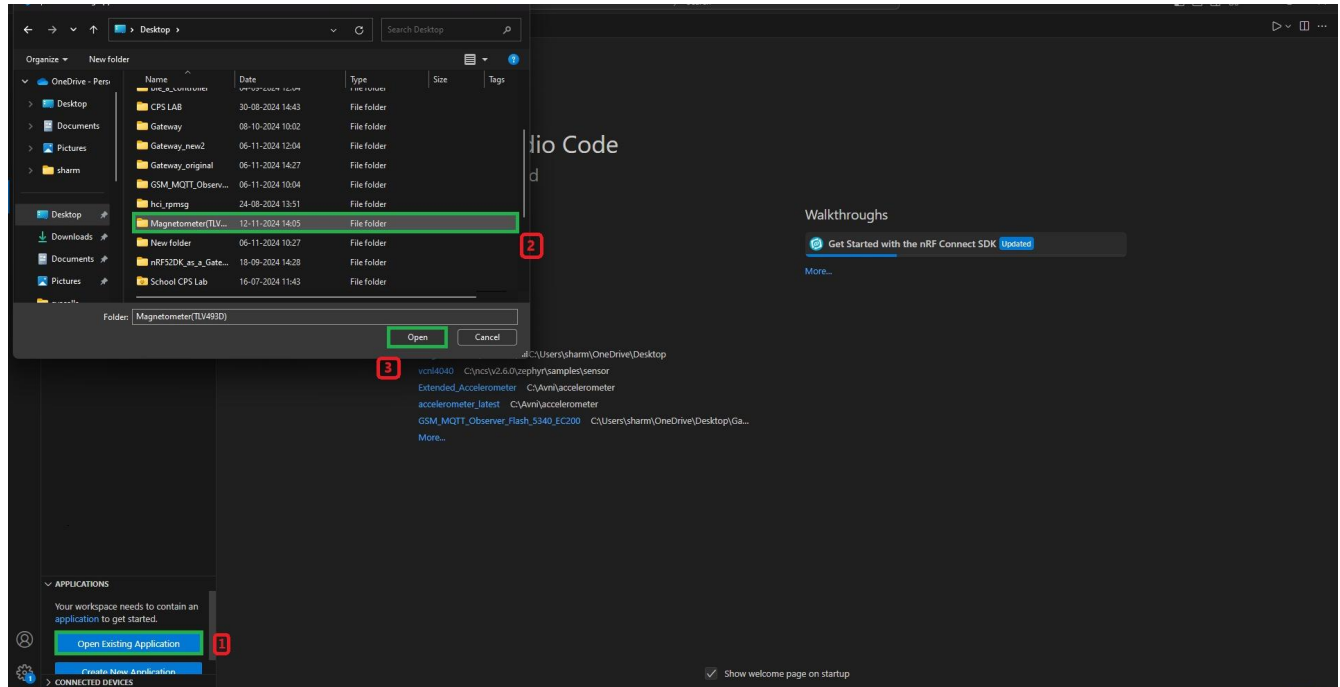
VDD(3.3V) -> VDD

PO.27 -> SCL

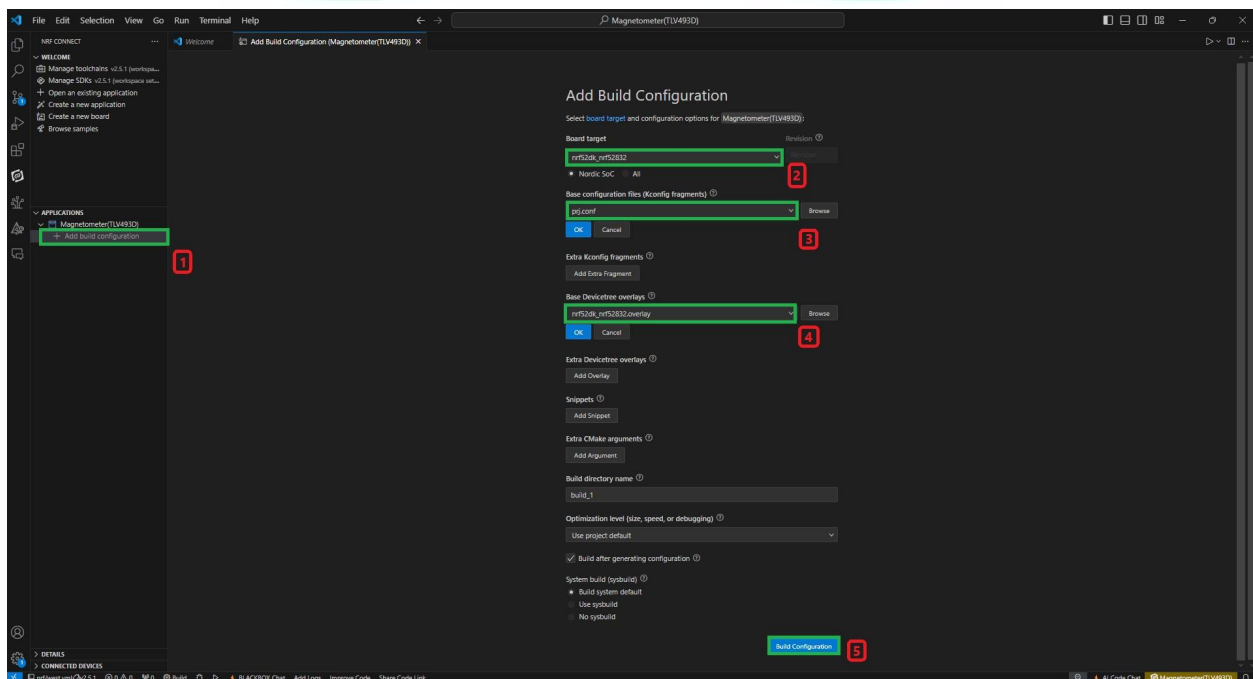
[illegible]

INTERFACING WITH THE HELP OF NODE

- Open VS Code and click on **Open Existing Application [1]** > click on **TLV493D_node_code [2]** > **Open [3]** as shown in the picture below.



- Click on **Create new build configuration [1]**. Here you can change the board version, if you are using nRF52832, then select **nrf52dk_nrf52832 [2]** or you can change from dropdown menu for another version like nRF52833 etc.
- Click on the Configuration and select **prj.conf [3]** from dropdown menu and then click on the device tree overlays and select **nrf52dk_nrf52832.overlay[4]**.
- Then click on the **Build Configuration [5]** as shown below in the picture.



- Go to source file, click **source file [1]** > click on **Application** > click on **src** > click on **main.c [2]**.
- By clicking on **main.c** file and you will see the code on your screen.

```

1 #include <zephyr/zephyr.h> // Zephyr core header
2 #include <zephyr/device.h> // Device structure header
3 #include <zephyr/devicetree.h> // Device tree macros
4 #include <zephyr/drivers/i2c.h> // I2C driver API
5 #include <zephyr/sys/printk.h> // Printk for console output
6 #include <stdio.h> // Standard IO functions
7
8 #define SLEEP_TIME_MS 500 // 1 second sleep time
9 #define I2C_NODE DT_MODELABEL(mysensor) // The I2C node label for the sensor
10 #define TLV493D_I2C_ADDR 0x5E // Adjust based on actual sensor wiring
11
12 // Function to configure TLV493D to active mode
13 static int tlv493d_set_active_mode(const struct i2c_dt_spec *i2c_spec) {
14     // Send the appropriate command to configure the sensor to active mode
15     uint8_t config_command[] = {0x00}; // Replace with correct bytes from the datasheet
16     int ret = i2c_write_dt(i2c_spec, config_command, sizeof(config_command));
17     if (ret) {
18         printk("Failed to configure TLV493D to active mode. Error: %d\n", ret);
19         return ret;
20     }
21     return 0;
22 }
23
24 // Function to read magnetic field data
25 static int read_magnetic_field(const struct i2c_dt_spec *i2c_spec, float *x, float *y, float *z) {
26     uint8_t read_command[] = {0x10}; // Replace with the correct register address for X/Y/Z data
27     uint8_t mag_data[7]; // Read 7 bytes from the sensor to cover X, Y, Z, and temperature
28
29     // Read 7 bytes of magnetic data
30     int ret = i2c_read_dt(i2c_spec, read_command, sizeof(read_command), mag_data, sizeof(mag_data));
31     if (ret) {
32         printk("Failed to read magnetic data. Error: %d\n", ret);
33         return ret;
34     }
35
36     // Raw data extraction; adjust based on sensor output format
37     int16_t raw_x = (int16_t)((uint16_t)mag_data[0] << 8) | (mag_data[1] << 8) >> 3;
38     int16_t raw_y = (int16_t)((uint16_t)mag_data[2] << 8) | (mag_data[3] << 8) >> 3;
39     int16_t raw_z = (int16_t)((uint16_t)mag_data[4] << 8) | (mag_data[5] << 8) >> 3;
40
41     printk("Raw data: %02x %02x %02x %02x %02x\n",
42           mag_data[0], mag_data[1], mag_data[2], mag_data[3], mag_data[4], mag_data[5]);
43
44     // Assuming a conversion factor (check datasheet for precise values)
45     *x = raw_x * 0.008f; // Example conversion, adjust as needed
46     *y = raw_y * 0.008f;
47     *z = raw_z * 0.008f;
48
49     return 0;
50 }

```

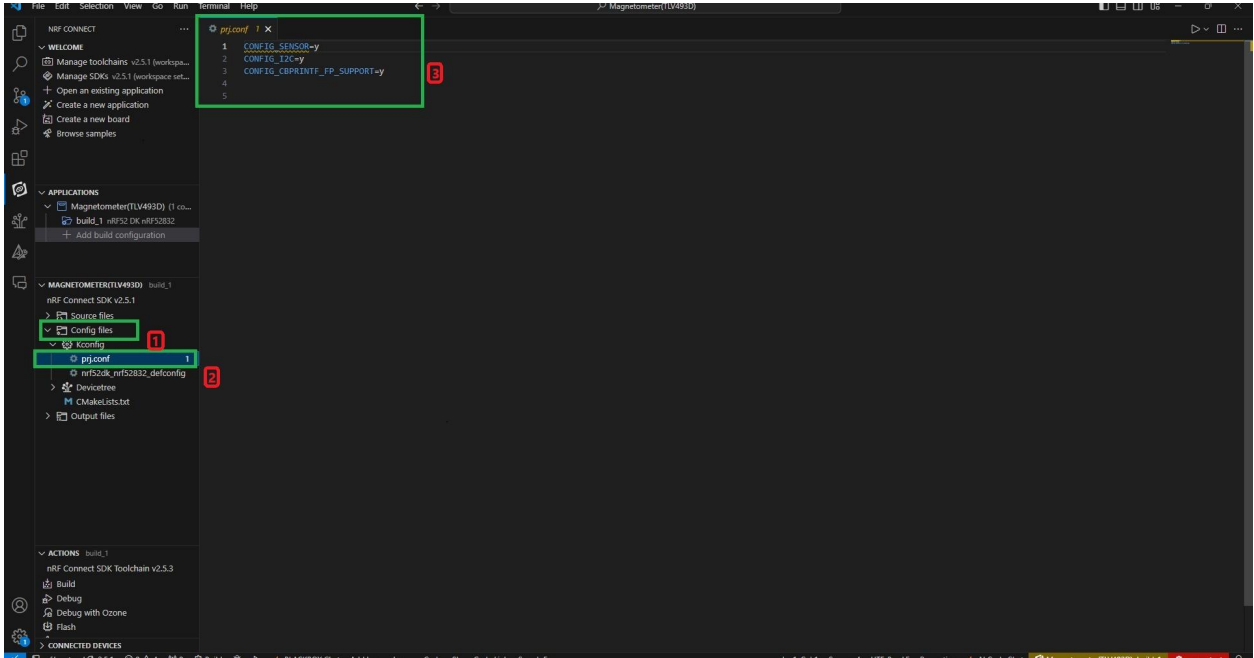
- To configure the i2c & UART protocols, you have to enable it in the **overlay file**.
- Click on the **Config files[1]** > click on **Kconfig** > click on **Devicetree** > click on **nrf52dk_nrf52832.overlay [2]**.
- The overlay file will appear on your screen and add the given code to the **overlay file** as shown in the picture given below [3].

```

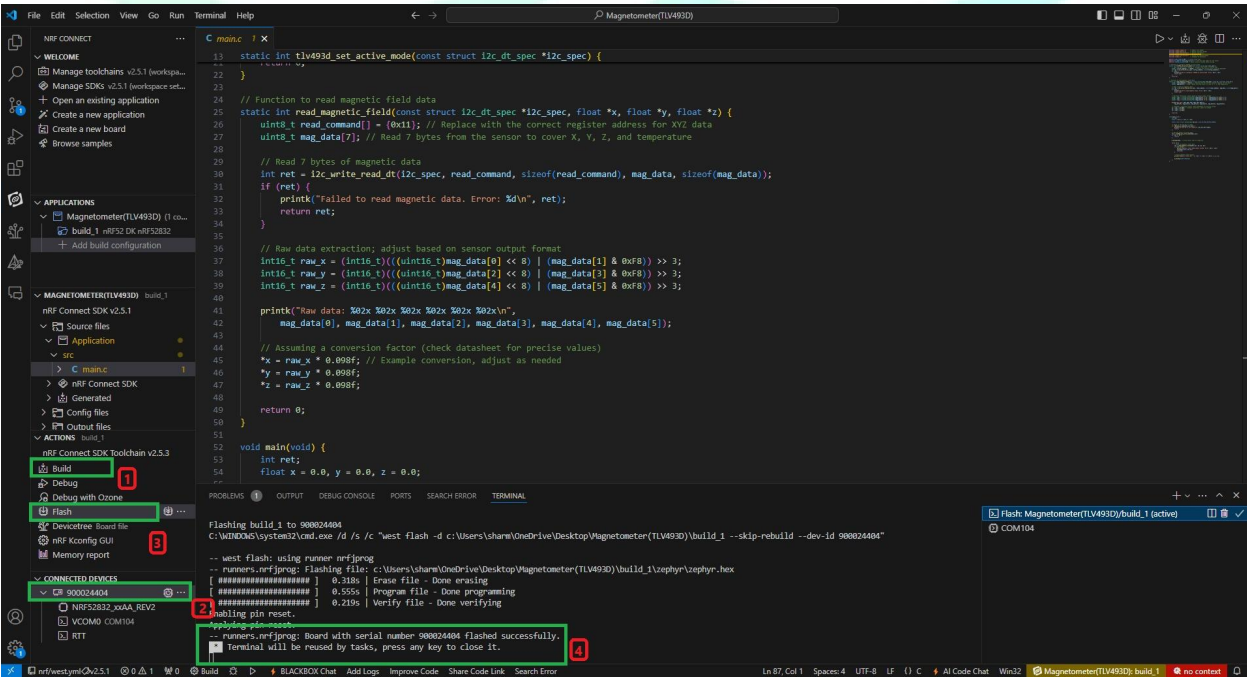
1 #include <dt-bindings/gpio/gpio.h>
2 #include <dt-bindings/i2c/i2c.h>
3 #include <dt-bindings/uart/uart.h>
4
5 // UART configuration
6 #define UART0_DEFAULT_CONFIG {
7     .psels = {<NRF_PSEL_UART_TX, 0, 6>},
8     .chrf_psel = {<NRF_PSEL_UART_RX, 0, 8>},
9     .chrf_psel = {<NRF_PSEL_UART_RTS, 0, 5>},
10     .chrf_psel = {<NRF_PSEL_UART_CTS, 0, 7>},
11 };
12
13 // UART sleep configuration
14 #define UART0_SLEEP_CONFIG {
15     .psels = {<NRF_PSEL_UART_TX, 0, 6>},
16     .chrf_psel = {<NRF_PSEL_UART_RX, 0, 8>},
17     .chrf_psel = {<NRF_PSEL_UART_RTS, 0, 5>},
18     .chrf_psel = {<NRF_PSEL_UART_CTS, 0, 7>},
19     .low-power-enable = 1;
20 };
21
22 // I2C configuration
23 #define I2C0_DEFAULT_CONFIG {
24     .psels = {<NRF_PSEL_I2C_SDA, 0, 26>},
25     .chrf_psel = {<NRF_PSEL_I2C_SCL, 0, 27>},
26 };
27
28 // I2C sleep configuration
29 #define I2C0_SLEEP_CONFIG {
30     .psels = {<NRF_PSEL_I2C_SDA, 0, 26>},
31     .chrf_psel = {<NRF_PSEL_I2C_SCL, 0, 27>},
32     .low-power-enable = 1;
33 };

```

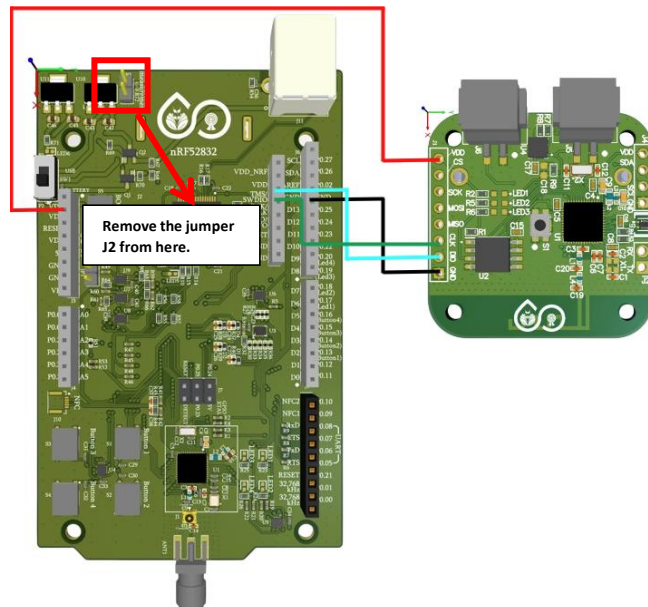
- You need to enable sensor in prj file for communication as shown below.
- Click on **Config files [1]** > then click on **Kconfig files** > click on **prj.conf [2]**.
- The **prj.conf** will appear on the screen **[3]** as shown below in the picture.



- Click on **Build [1]** configuration again and check the **CONNECTED DEVICES [2]**.
- If device id is visible, then **Flash [3]** the code in Dev Kit.



- For Node programming remove the jumper **J2** from the development board.
- Now flash the code with the help of nRF52832 development board as shown below in the figure.



Board Pins -> NODE Pins

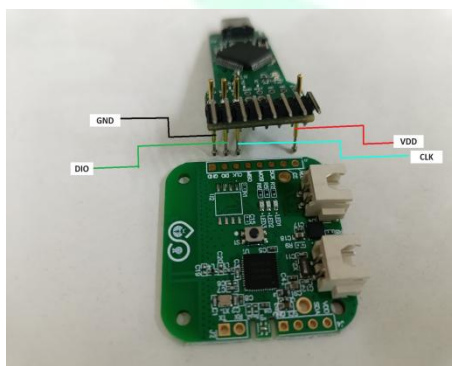
VDD(3.3V) -> VDD

GND -> GND

CLK -> CLK

DIO -> DIO

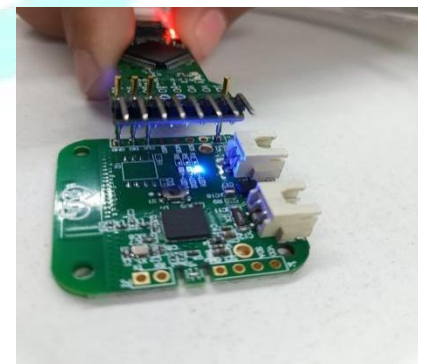
- There is another way of flashing the code with the help of Node Programmer as shown in the picture below.



➤ NODE without connection.

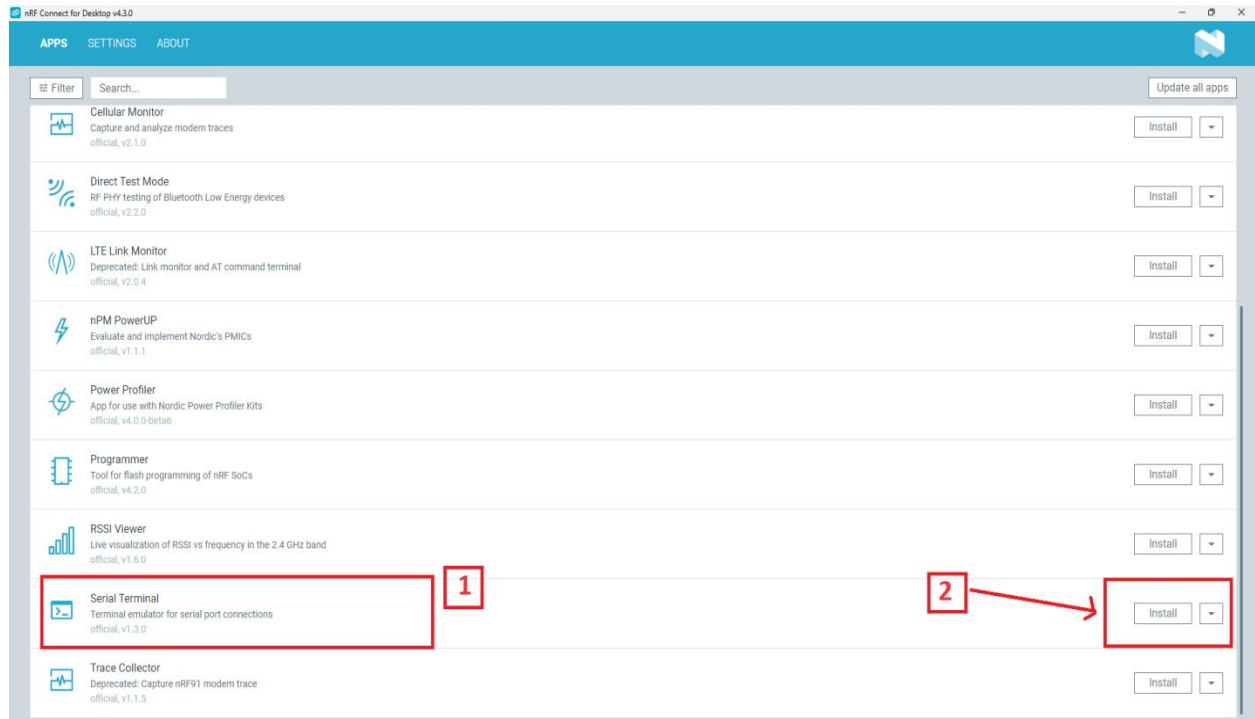


➤ NODE with connection.

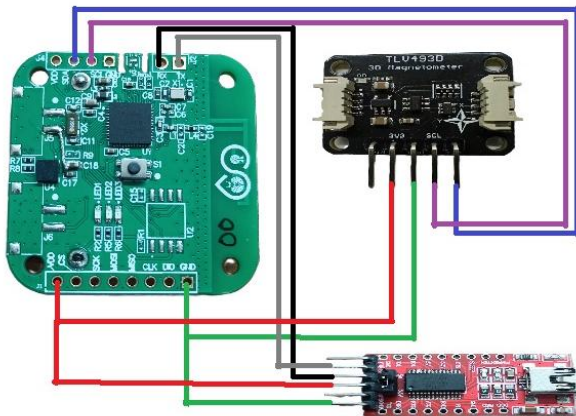


➤ NODE after program.

- Firstly, you have to **Install [2]** the nRF **Serial Terminal [1]** in nRF Connect for Desktop application as shown below.



- Connect the **TTL Device** for UART communication so that the data must appear on the serial terminal.
- Connect the **TTL Device** as shown below in the picture.



Node Pins -> TTL Pins

Tx -> Rx

Rx -> Tx

VDD -> VDD

GND -> GND

Node Pins -> Sensor Pins

SDA -> SDA

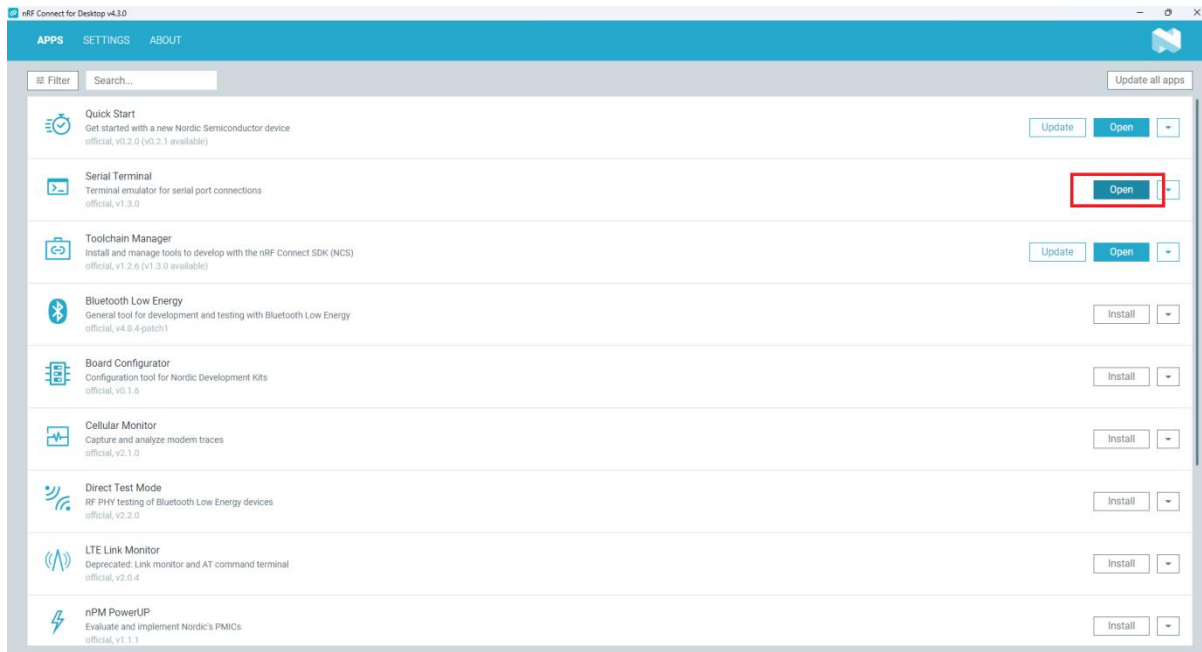
SCL -> SCL

VDD(3V3) -> VDD

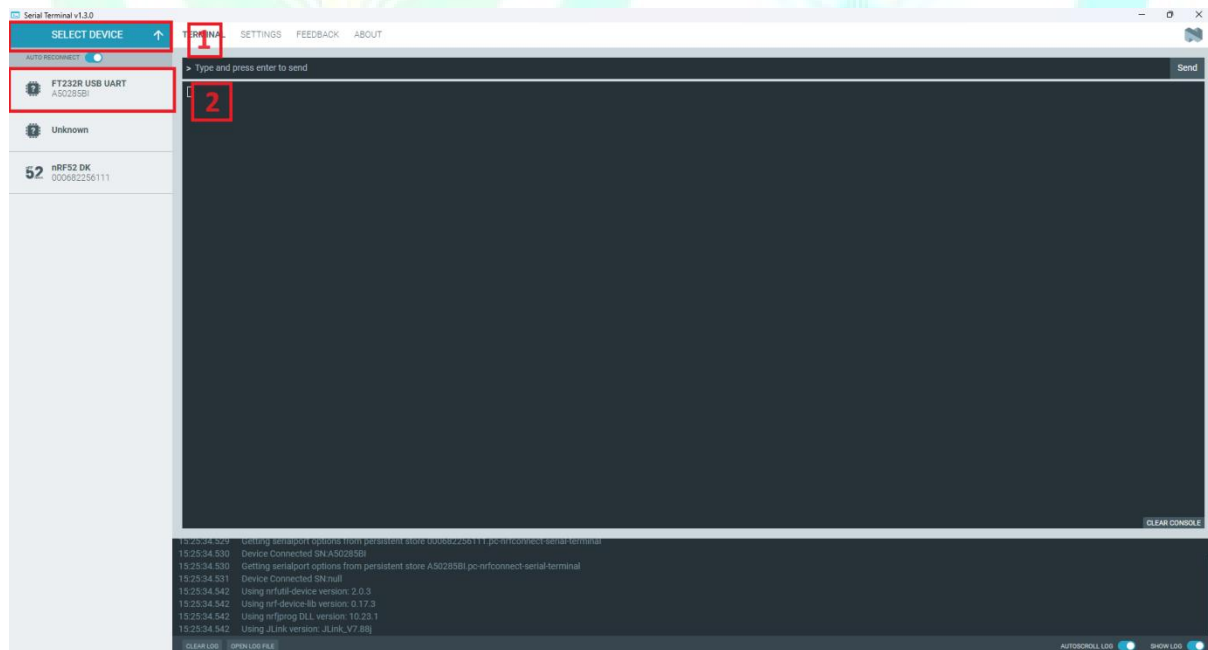
GND -> GND

Note: - Do not supply 5 volts current to board through TTL. Otherwise, board will damage.

- Click on **Open** as shown below in the picture.



➤ Click on **Select Device** [1] > click on **FT232R USB UART** [2] as shown below in the picture.



➤ Now the output will appear on your screen as shown below.

OUTPUT

