Using the Teseo-LIV3F/3R/3FL as an I2C positioning sensor

ADG/Positioning – Mar 2022

Introduction

- The Teseo-LIV3F/3R/3FL GNSS module embeds a 3rd generation of ST's single-die standalone positioning receiver ICs (Teseo III) able to work simultaneously on multiple constellations (GPS/Galileo/Glonass/BeiDou/QZSS).
- The Teseo-LIV3F/3R/3FL ’s I2C interface emits the same NMEA stream available on its UART interface.
- The Host MCU should continuously poll the Teseo III GNSS IC’s I2C bus to access real-time positioning data
- Certain applications only need to read the current position on-demand (with no interest in the NMEA stream) using the Teseo-LIV3F/3R/3FL as an I2C sensor (for example, as a gyroscope, accelerometer, etc)

Proposal

- The Host can use the $PSTMNMEAREQUEST command to request on-demand only specific NMEA data based on a message-list-bitmap:

```
$PSTMNMEAREQUEST,<msg-low>,<msg-high>*<checksum>
```

- Use case:
  - Host requests the required NMEA message;
  - Teseo-LIV3F/3R/3FL responds with the requested NMEA message;
  - Host parses only the NMEA messages it wants (when it wants)

Prepare the module

- Reset the I2C-MessageList

```
$PSTMCFGMSGL,3,1,0,0
$PSTMSAVEPAR
```

- Disable echo-ing commands (CDB-ID 227[0])

```
$PSTMSETPAR,1227,1,2
$PSTMSAVEPAR
```

- Now the I2C-MessageList is empty.
- This means than TeseoLIV-I2C_Readbuffer isn’t filled with an autonomous NMEA message stream

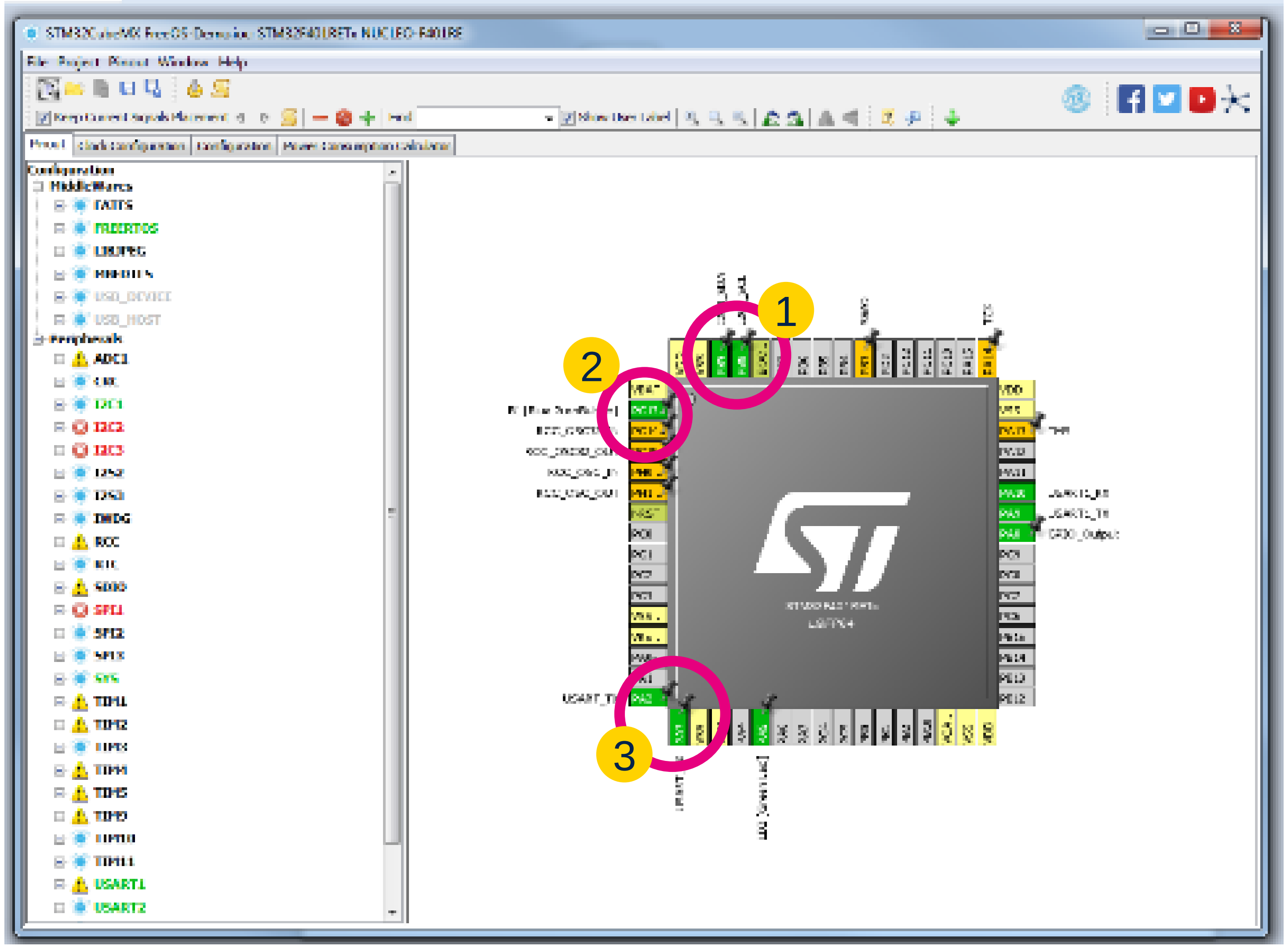
Prepare the stm32cubemx project [1/4]

Using the **STM32CubeMX** graphical software configuration tool to configure your application

1 Enable I2C on PB9-PB8

2 Enable Blue-Button on PC13

3 Enable USART-2

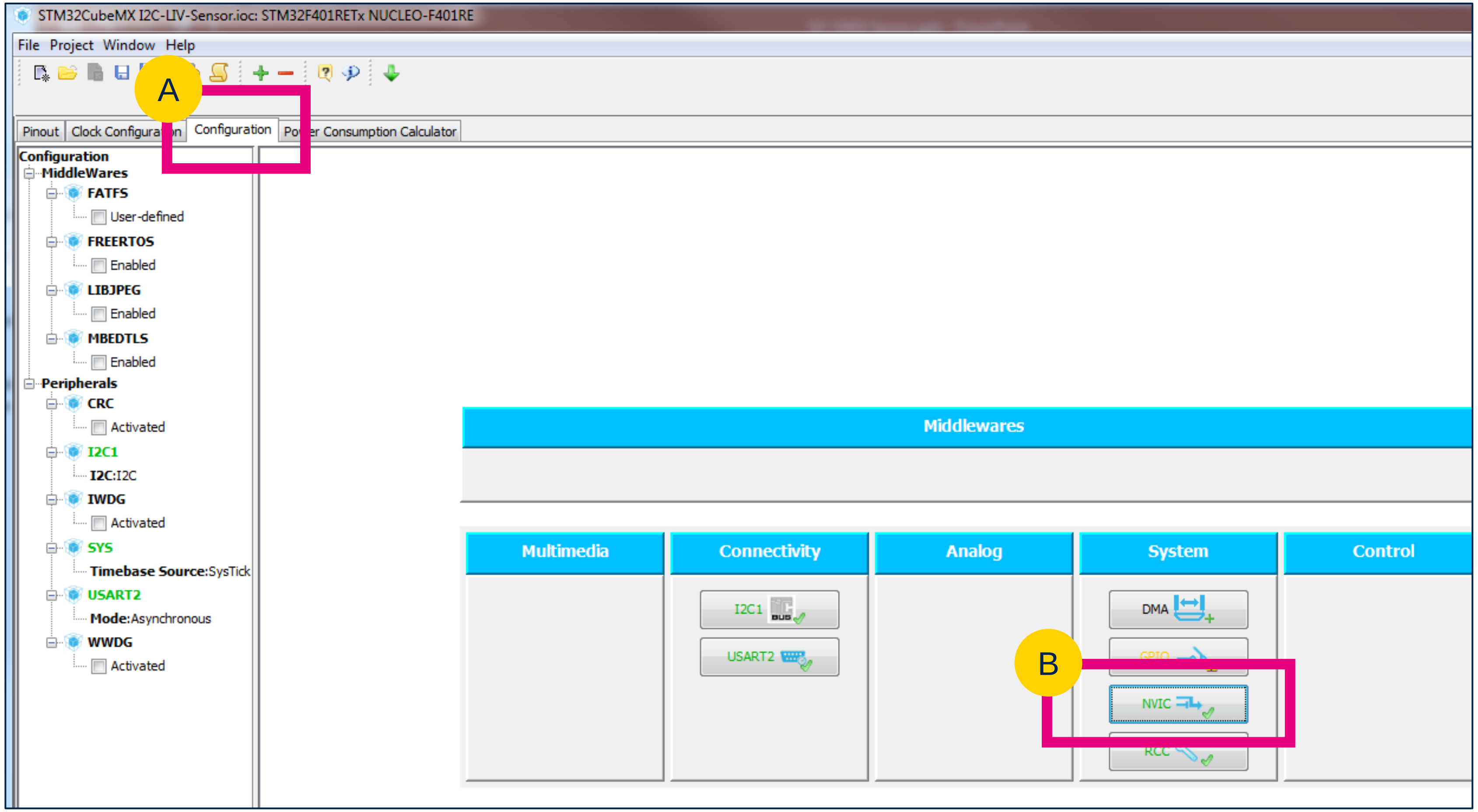


Prepare the stm32cubemx project [2/4]

3 Enable Blue-Button Interrupt (EXTI interrupt)

A Open Configuration Tab

B Press NVIC button



STM32CubeMX I2C-LIV-Sensor.ioc: STM32F401RETx NUCLEO-F401RE

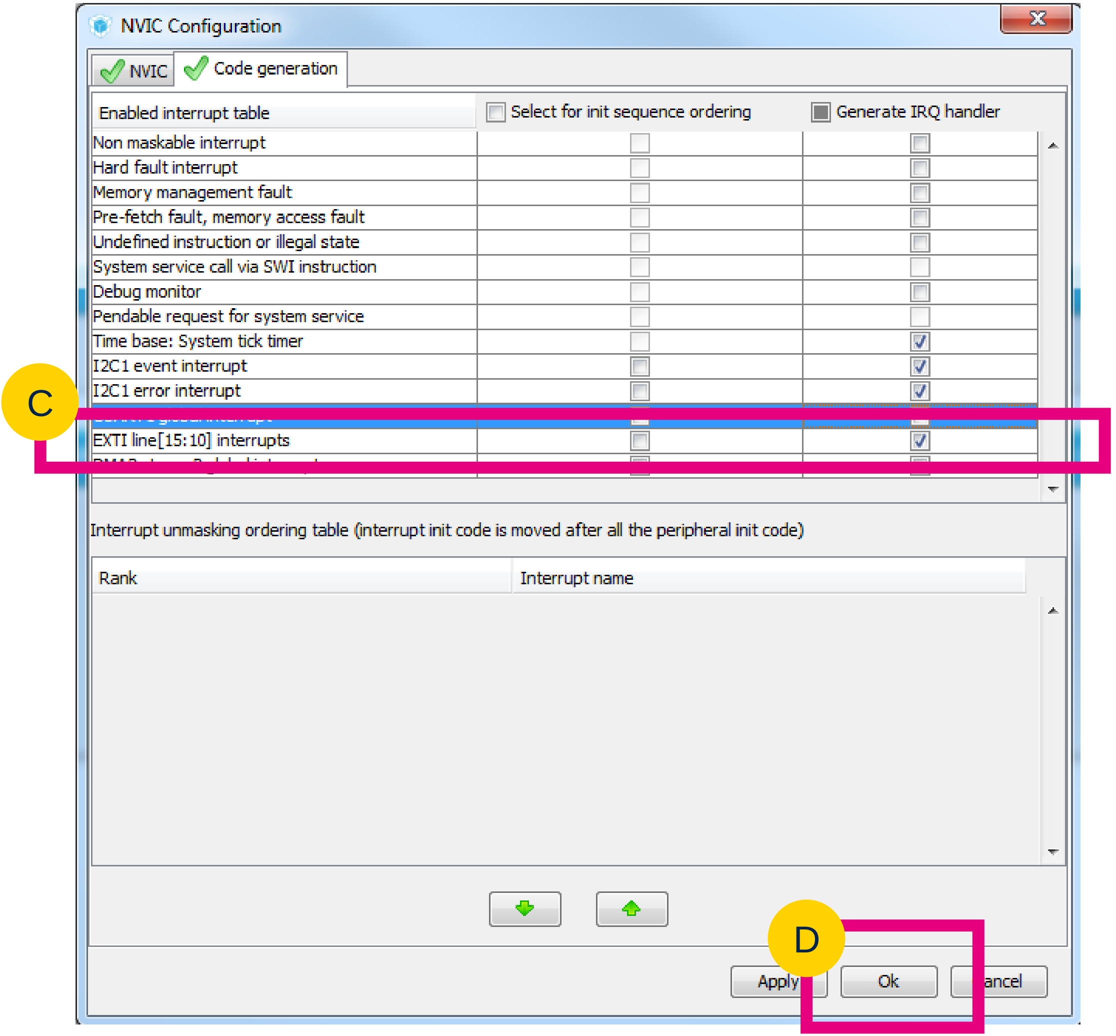
Configuration
- MiddleWares
  - FATFS
    - □ User-defined
  - FREERTOS
    - □ Enabled
  - LIBJPEG
    - □ Enabled
  - MBEDTLS
    - □ Enabled
- Peripherals
  - CRC
    - □ Activated
  - I2C1
    - I2C:I2C
  - IWDG
    - □ Activated
  - SYS
    - Timebase Source:SysTick
  - USART2
    - Mode:Asynchronous
  - WWDG
    - □ Activated

life.augmented

Prepare the stm32cubemx project [3/4]

3 Enable Blue-Button Interrupt (EXTI interrupt)

- C Enable EXTI IRQ handler
- D Press OK button



NVIC Configuration

NVIC | Code generation

Enabled interrupt table	□ Select for init sequence ordering	■ Generate IRQ handler
Non maskable interrupt	□	□
Hard fault interrupt	□	□
Memory management fault	□	□
Pre-fetch fault, memory access fault	□	□
Undefined instruction or illegal state	□	□
System service call via SWI instruction	□	□
Debug monitor	□	□
Pendable request for system service	□	□
Time base: System tick timer	□	☑
I2C1 event interrupt	□	☑
I2C1 error interrupt	□	☑
[illegible]	□	□
EXTI line[15:10] interrupts	□	☑
[illegible]	□	□

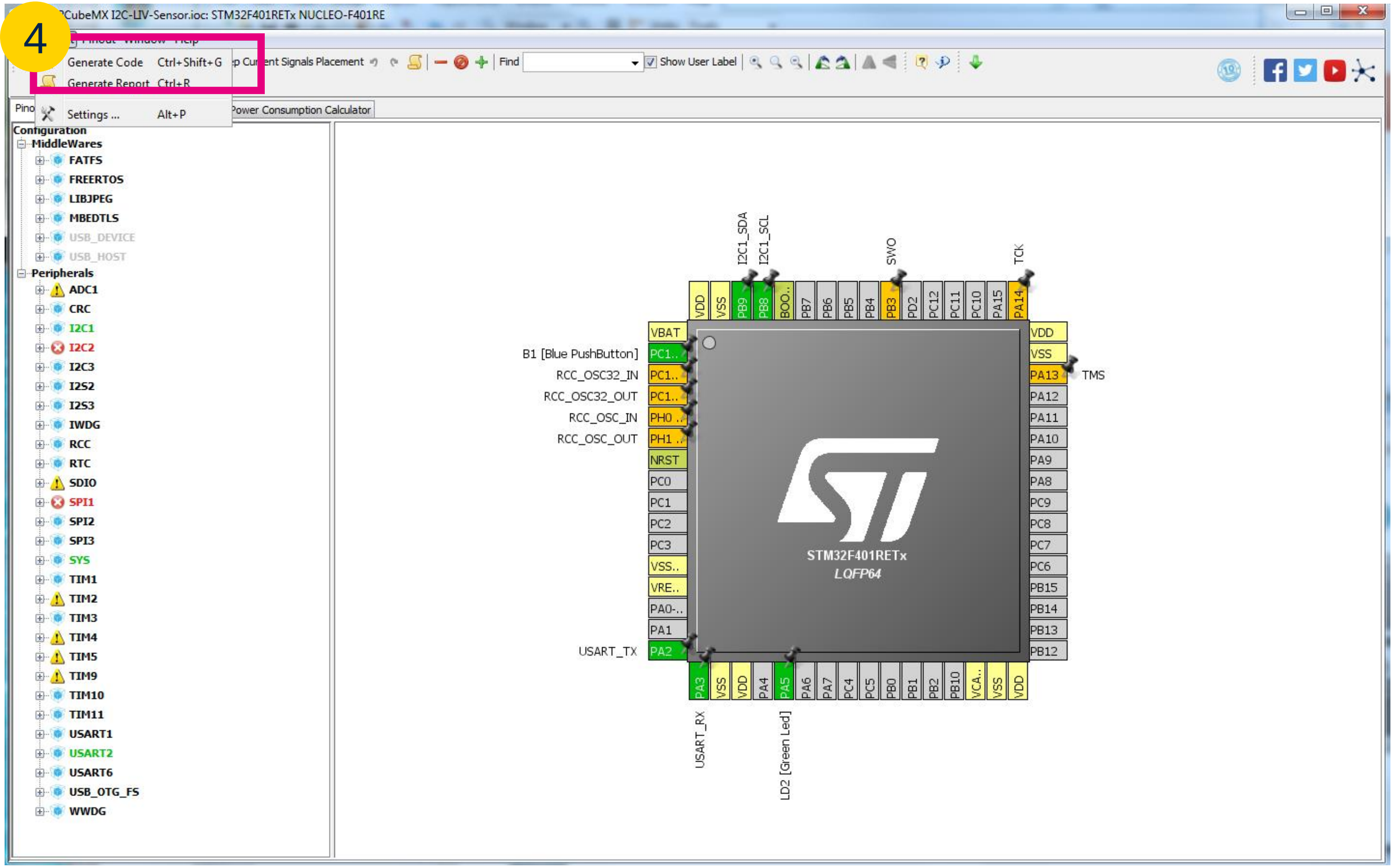
Interrupt unmasking ordering table (interrupt init code is moved after all the peripheral init code)

Rank	Interrupt name

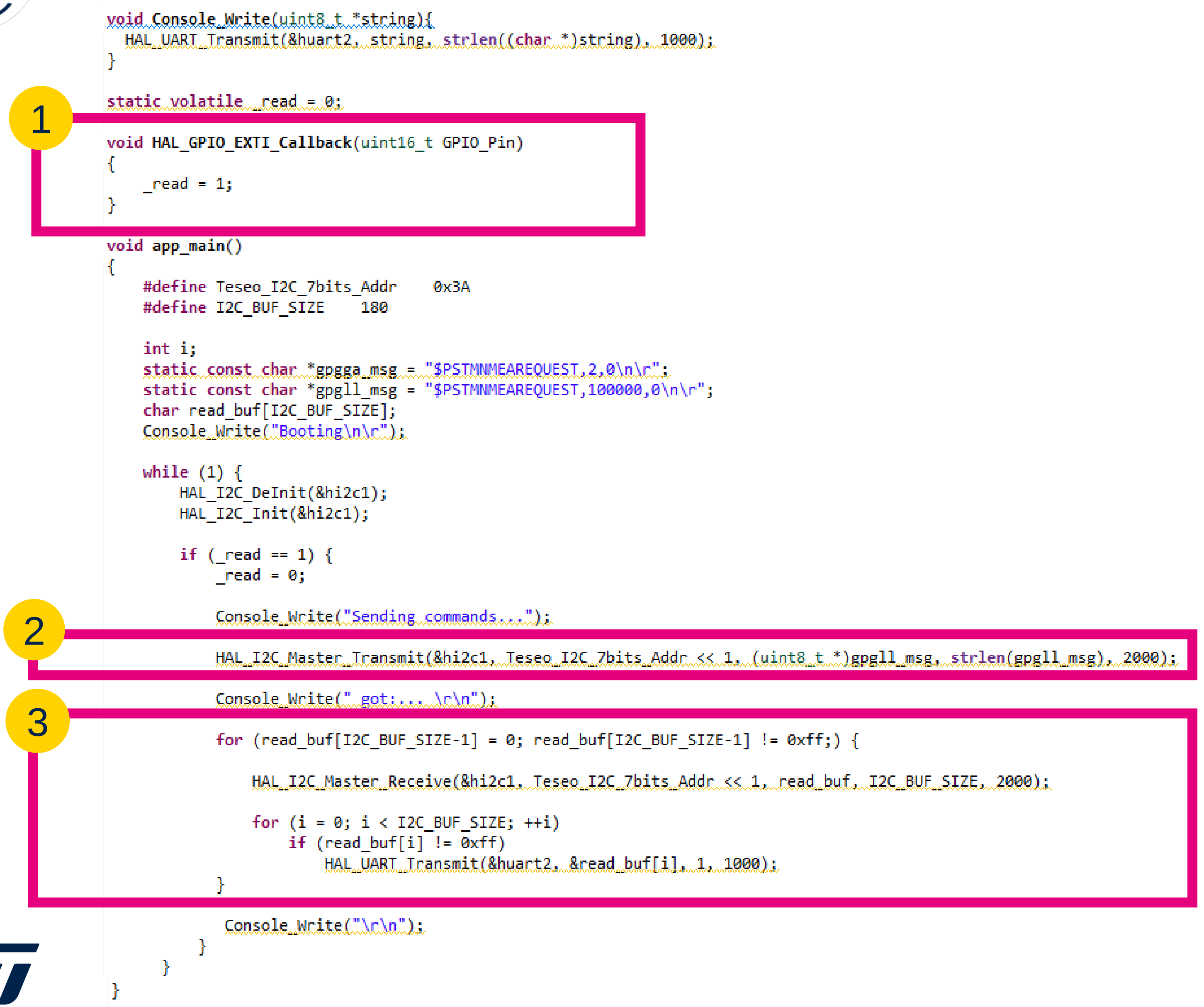
Prepare the stm32cubemx project [4/4]

4 Generate the C initialization code

Now you can open your IDE.


4
CubeMX I2C-LIV-Sensor.ioc: STM32F401RETx NUCLEO-F401RE
Pinout Window Help
Generate Code Ctrl+Shift+G
Generate Report Ctrl+R
Settings ... Alt+P
p Current Signals Placement
Find
Show User Label
Pino
Power Consumption Calculator
Configuration
MiddleWares
FATFS
FREERTOS
LIBJPEG
MBEDTLS
USB_DEVICE
USB_HOST
Peripherals
ADC1
CRC
I2C1
I2C2
I2C3
I2S2
I2S3
IWDG
RCC
RTC
SDIO
SPI1
SPI2
SPI3
SYS
TIM1
TIM2
TIM3
TIM4
TIM5
TIM9
TIM10
TIM11
USART1
USART2
USART6
USB_OTG_FS
WWDG
I2C1_SDA
I2C1_SCL
SWO
TCK
TMS
B1 [Blue PushButton]
RCC_OSC32_IN
RCC_OSC32_OUT
RCC_OSC_IN
RCC_OSC_OUT
USART_TX
USART_RX
LD2 [Green Led]
STM32F401RETx
LQFP64
VDD VSS PB9 PB8 BOO.. PB7 PB6 PB5 PB4 PB3 PD2 PC12 PC11 PC10 PA15 PA14
VBAT PC1.. PC1.. PC1.. PH0 .. PH1 .. NRST PC0 PC1 PC2 PC3 VSS.. VRE.. PA0-.. PA1 PA2
VDD VSS PA13 PA12 PA11 PA10 PA9 PA8 PC9 PC8 PC7 PC6 PB15 PB14 PB13 PB12
PA3 VSS VDD PA4 PA5 PA6 PA7 PC4 PC5 PB0 PB1 PB2 PB10 VCA.. VSS VDD

Edit the C-Code



```c
void Console_Write(uint8_t *string){
  HAL_UART_Transmit(&huart2, string, strlen((char *)string), 1000);
}

static volatile _read = 0;

void HAL_GPIO_EXTI_Callback(uint16_t GPIO_Pin)
{
    _read = 1;
}

void app_main()
{
    #define Teseo_I2C_7bits_Addr    0x3A
    #define I2C_BUF_SIZE    180

    int i;
    static const char *gpgga_msg = "$PSTMNMEAREQUEST,2,0\n\r";
    static const char *gpgll_msg = "$PSTMNMEAREQUEST,100000,0\n\r";
    char read_buf[I2C_BUF_SIZE];
    Console_Write("Booting\n\r");

    while (1) {
        HAL_I2C_DeInit(&hi2c1);
        HAL_I2C_Init(&hi2c1);

        if (_read == 1) {
            _read = 0;

            Console_Write("Sending commands...");

            HAL_I2C_Master_Transmit(&hi2c1, Teseo_I2C_7bits_Addr << 1, (uint8_t *)gpgll_msg, strlen(gpgll_msg), 2000);

            Console_Write(" got:... \r\n");

            for (read_buf[I2C_BUF_SIZE-1] = 0; read_buf[I2C_BUF_SIZE-1] != 0xff;) {

                HAL_I2C_Master_Receive(&hi2c1, Teseo_I2C_7bits_Addr << 1, read_buf, I2C_BUF_SIZE, 2000);

                for (i = 0; i < I2C_BUF_SIZE; ++i)
                    if (read_buf[i] != 0xff)
                        HAL_UART_Transmit(&huart2, &read_buf[i], 1, 1000);
            }
             Console_Write("\r\n");
          }
      }
}
```

1 Blue-button Call-back

2 I2C-Write to request the message

3 I2C-Read to read the message

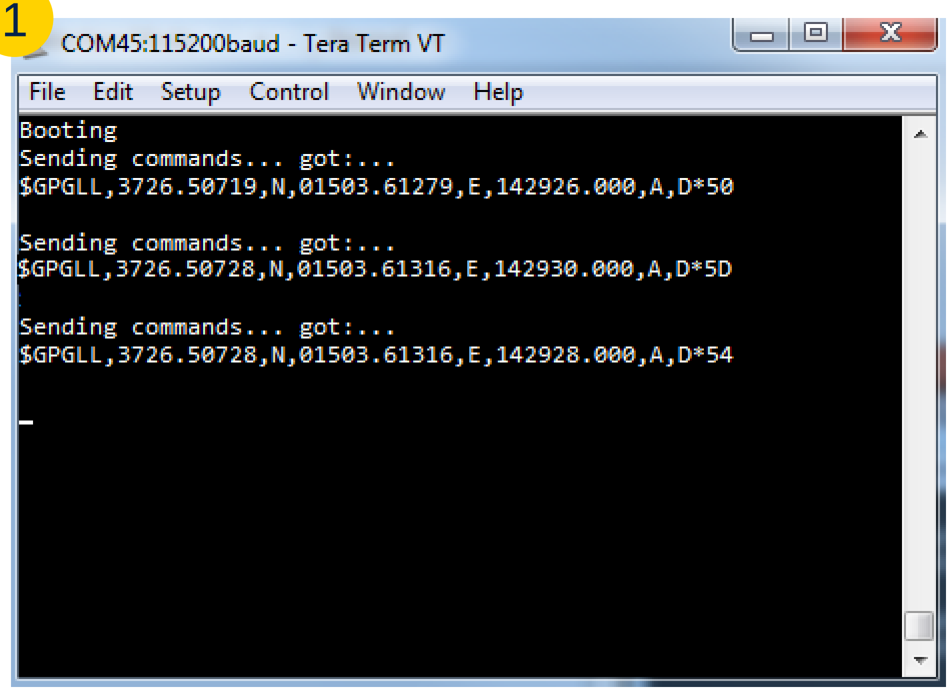
Run and final conclusion

1 Run and view result on a terminal emulator

2 Final conclusion:

- The Host MCU doesn’t need to continuously poll the Teseo-LIV module
- The Host MCU can request any NMEA message it wants (when it wants)
- The Host MCU has to parse only the needed NMEA message

1


COM45:115200baud - Tera Term VT
File Edit Setup Control Window Help

```
Booting
Sending commands... got:...
$GPGLL,3726.50719,N,01503.61279,E,142926.000,A,D*50

Sending commands... got:...
$GPGLL,3726.50728,N,01503.61316,E,142930.000,A,D*5D

Sending commands... got:...
$GPGLL,3726.50728,N,01503.61316,E,142928.000,A,D*54

_
```