Application note

How to use a sensor on a DIL 24 socket in X-CUBE-MEMS1 package applications

Introduction

The X-CUBE-MEMS1 software package provides example applications for STM32 Nucleo development platforms connected to an X-NUCLEO expansion board with inertial and environmental MEMS sensors.

Examples of expansion boards are X-NUCLEO-IKS01A1, X-NUCLEO-IKS01A2, X-NUCLEO-IKS01A3 (the latest, embedding consumer sensors) and X-NUCLEO-IKS02A1 (embedding industrial sensors).

The expansion board can be further extended by plugging an additional sensor board, such as STEVAL-MKI194V1 with LSM6DSR, onto the DIL 24 socket.

1 Example description

In this document, we build an example application for the NUCLEO-L476RG development board, stacked with the X-NUCLEO-IKS01A3 expansion board, on which the STEVAL-MKI194V1 is plugged on the DIL 24 socket.

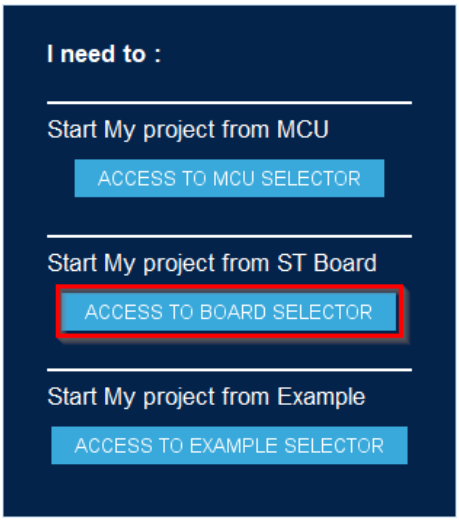
The application reads the sensor data (accelerometer, gyroscope, magnetometer) and transmit them to the MotionFX sensor fusion library which performs the orientation estimation and computes the corresponding quaternion and Euler angles (roll, pitch, and yaw).

2 Create a new project

Step 1. Run STM32CubeMX and create the new project.

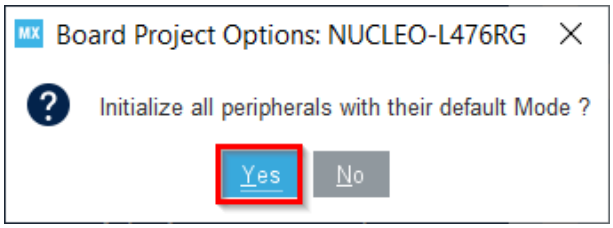
Step 2. In the main window choose [**ACCESS TO BOARD SELECTOR**] and select NUCLEO-L476RG development board.

Figure 1. Board selection



Step 3. Accept [**Initialize all peripherals with their default Mode**].

Figure 2. Default peripheral settings



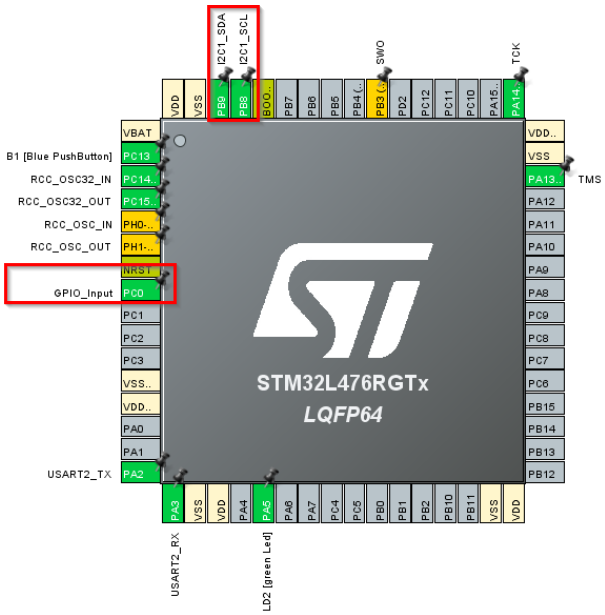
3 Pin-out setup

In the *Pinout view* you have to set pins as follows:

- PB9: I2C_SDA
- PB8: I2C_SCL
- PC0: GPIO_Input

The other pins should be already set as shown in the picture below:

Figure 3. Pin assignment



4 Peripheral configuration

4.1 DMA configuration

Referring to Figure 4, follow the steps below.

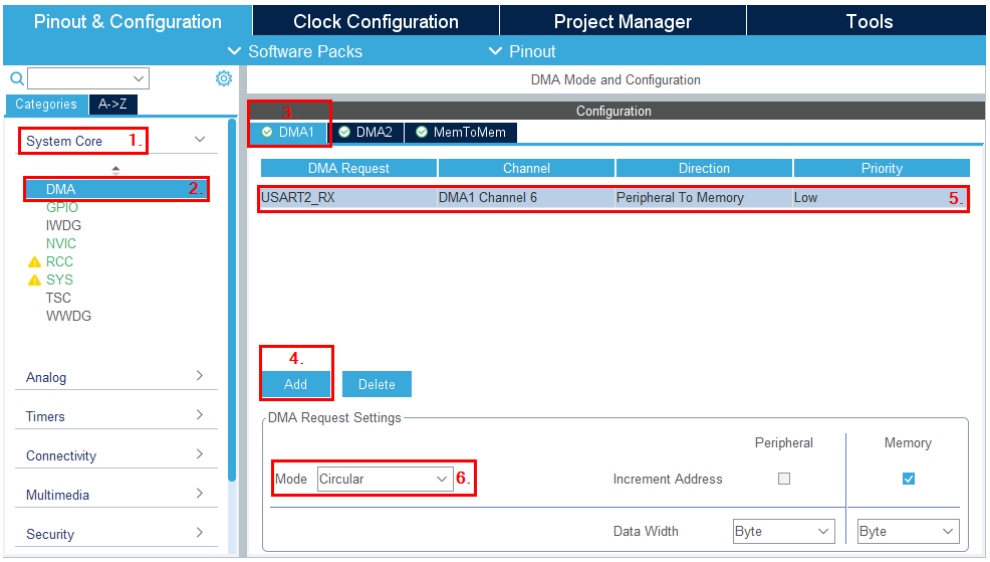
Step 1. In [**Pinout & Configuration**] tab open [**System Core group**] (1).

Step 2. Choose [**DMA**] peripheral (2) - part [**DMA1**] (3).

Step 3. Click on [**Add**] button (4) to add [**DMA Request: USART2_RX**] (5).

Step 4. Set [**Circular**] mode (6) for DMA.

Figure 4. DMA configuration



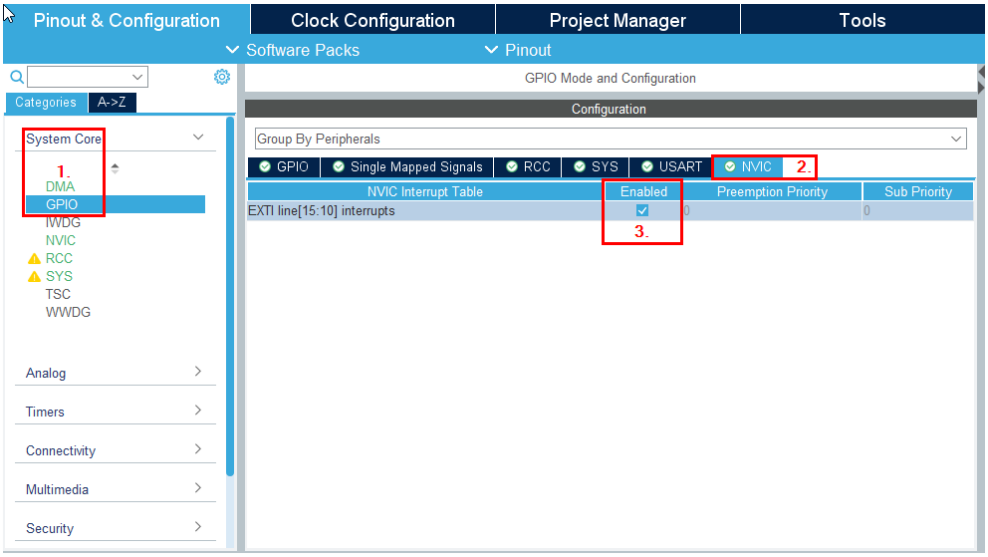
4.2 GPIO configuration

Referring to Figure 5, follow the steps below.

Step 1. For [**System Core - GPIO**] (1), select [**NVIC**] tab (2).

Step 2. Check [**Enabled**] (3) for `EXTI line[15:10] interrupts`

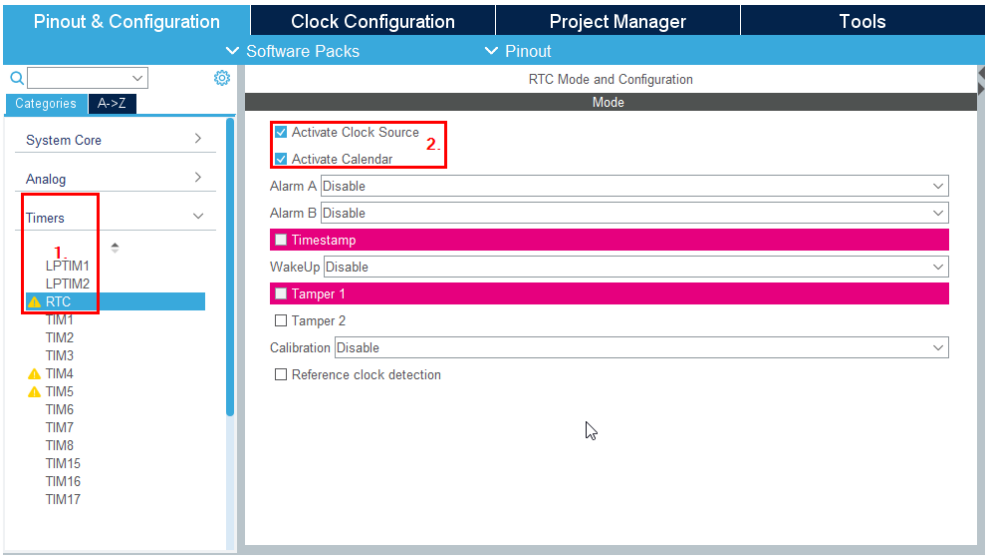
Figure 5. **GPIO configuration**



4.3 RTC configuration

For [**Timers - RTC**] (1), check [**Activate Clock Source**] and [**Activate Calendar**] (2).

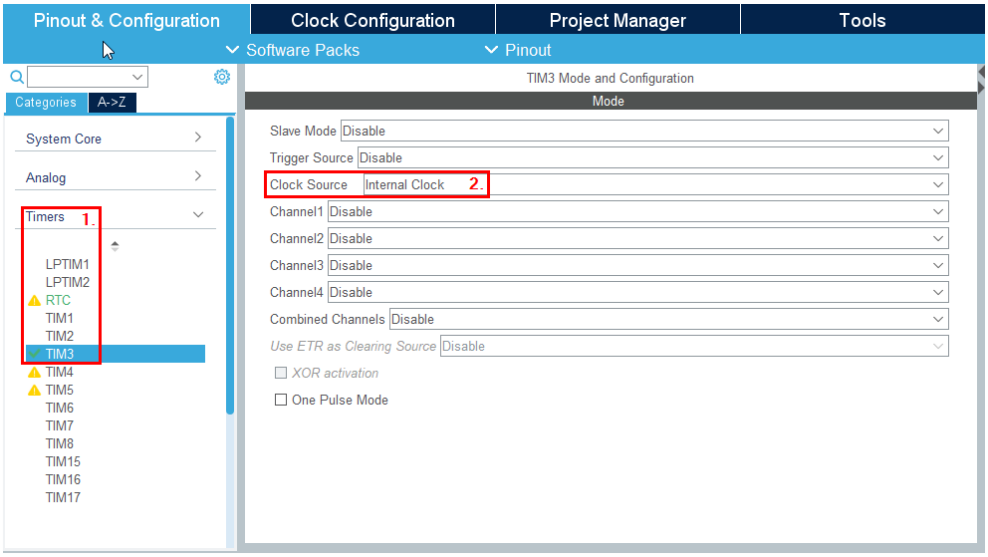
Figure 6. **RTC configuration**



4.4 TIM3 configuration

For [**Timers - TIM3**] (1), select [**Internal Clock**] for [**Clock Source**] (2).

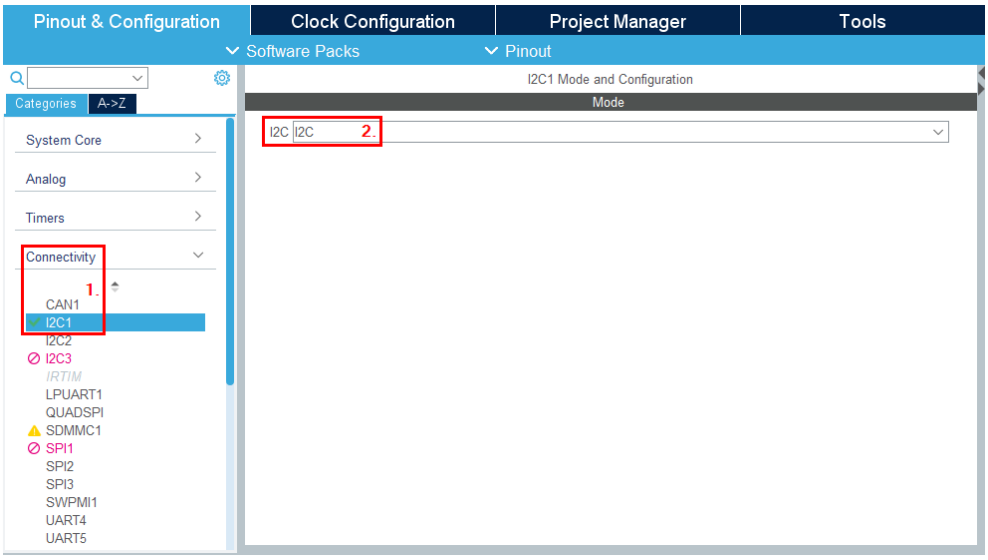
Figure 7. **TIM3 configuration**



4.5 I2C1 configuration

For [**Connectivity - I2C1**] (1), select [**I2C**] mode (2).

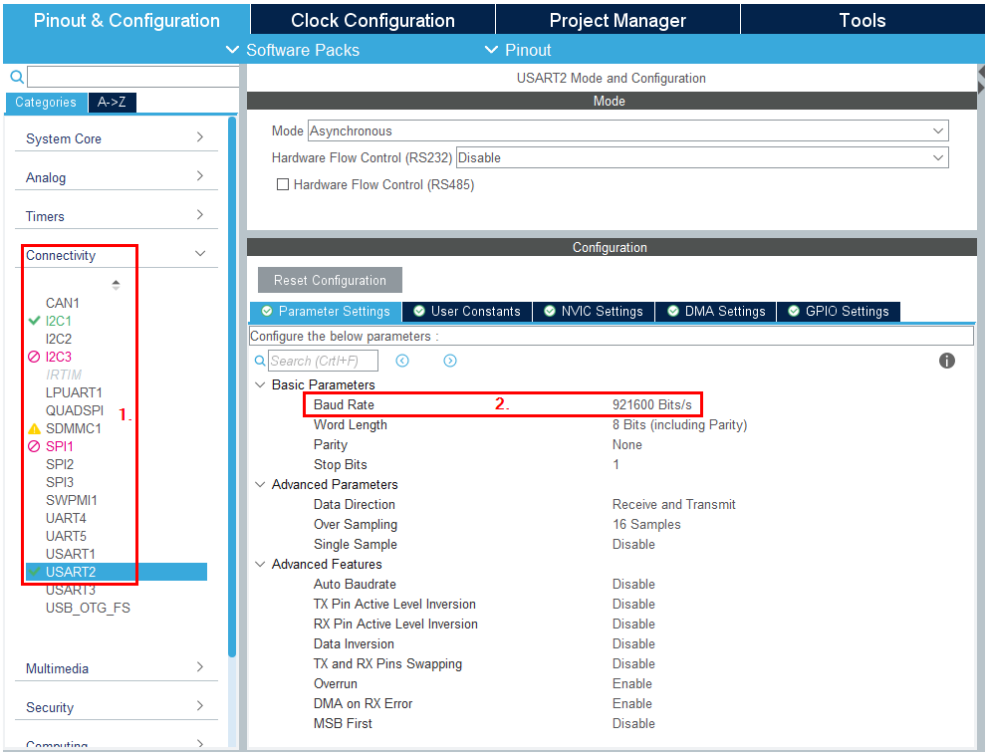
Figure 8. **I2C1 configuration**



4.6 USART2 configuration

For [**Connectivity - USART2**] (1), change [**Baud Rate**] to `921600 Bits/s` (2) in [**Parameter Settings**] tab.

Figure 9. **USART2 configuration**

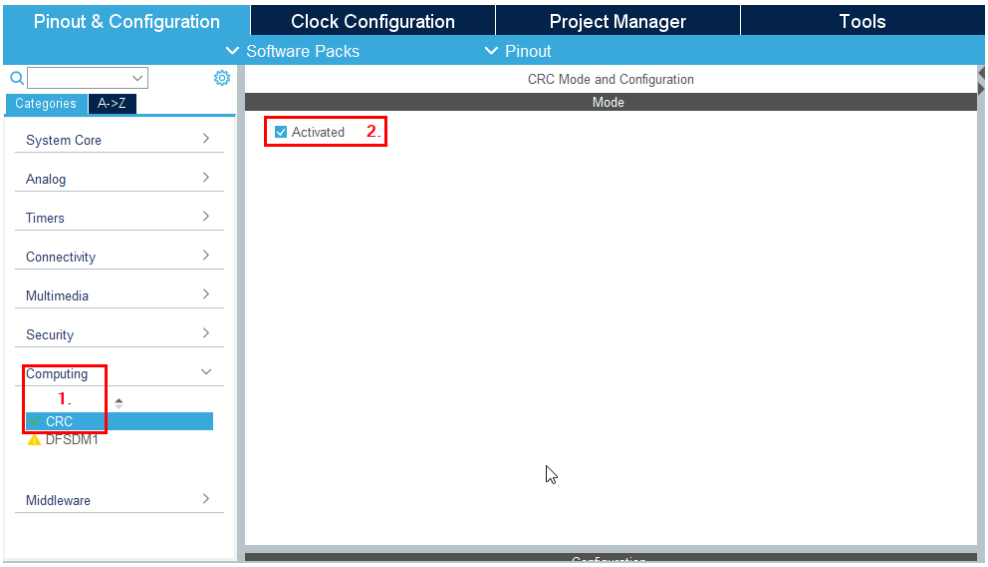


4.7 CRC configuration

Referring to Figure 10. CRC configuration, follow the procedure below.

Step 1. For [**Computing - CRC**] (1), check [**Activated**] (2).

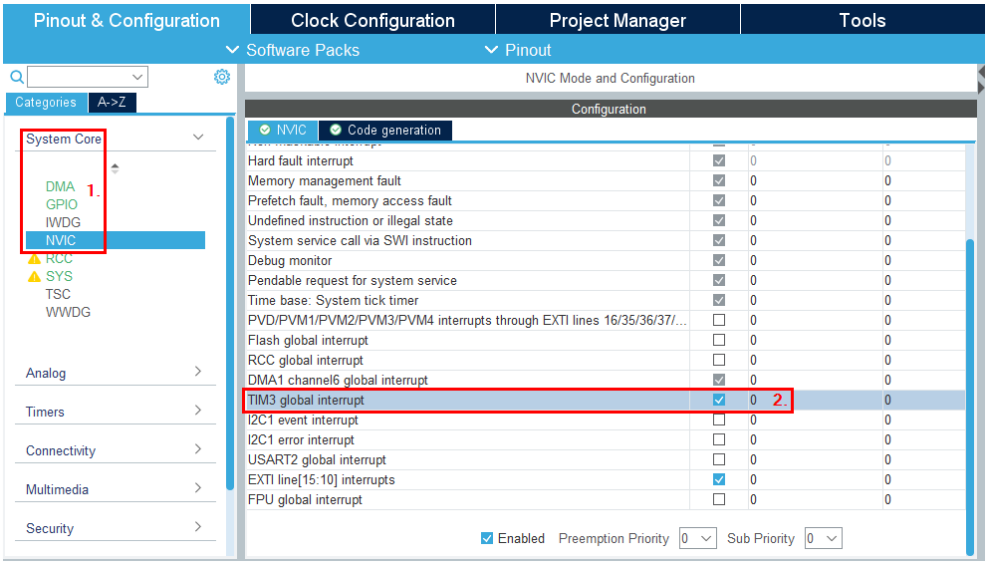
Figure 10. **CRC configuration**



4.8 NVIC configuration

For [**System Core - NVIC**] (1), check [**Enabled**] for `TIM3 global interrupt` (2).

Figure 11. **NVIC configuration**

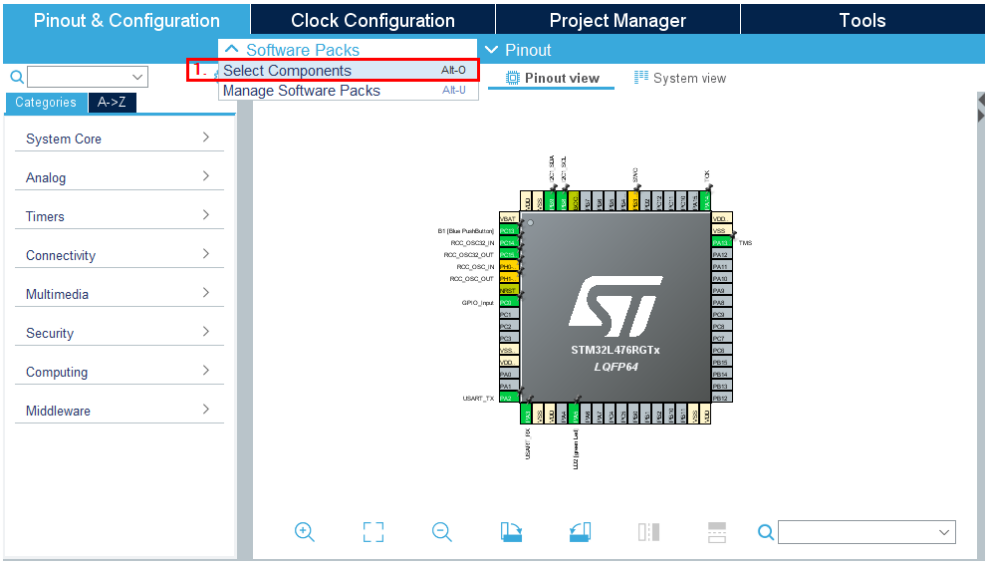


5 Software pack setup

5.1 Software Packs menu

From [**Software Packs**], choose [**Select Components**] (1).

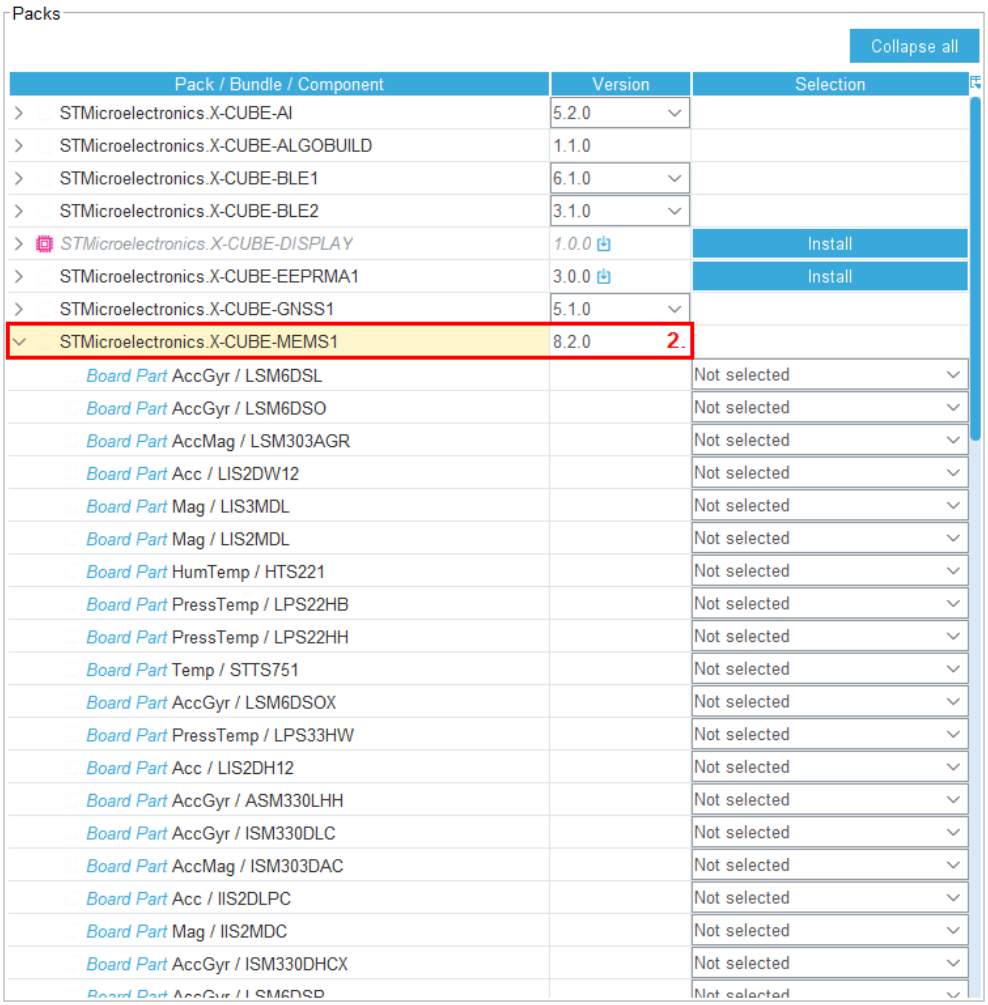
Figure 12. Software components selection



5.2 Software pack selection

From [**Packs**], choose `STMicroelectronics.X-CUBE-MEMS1 8.2.0` (2).

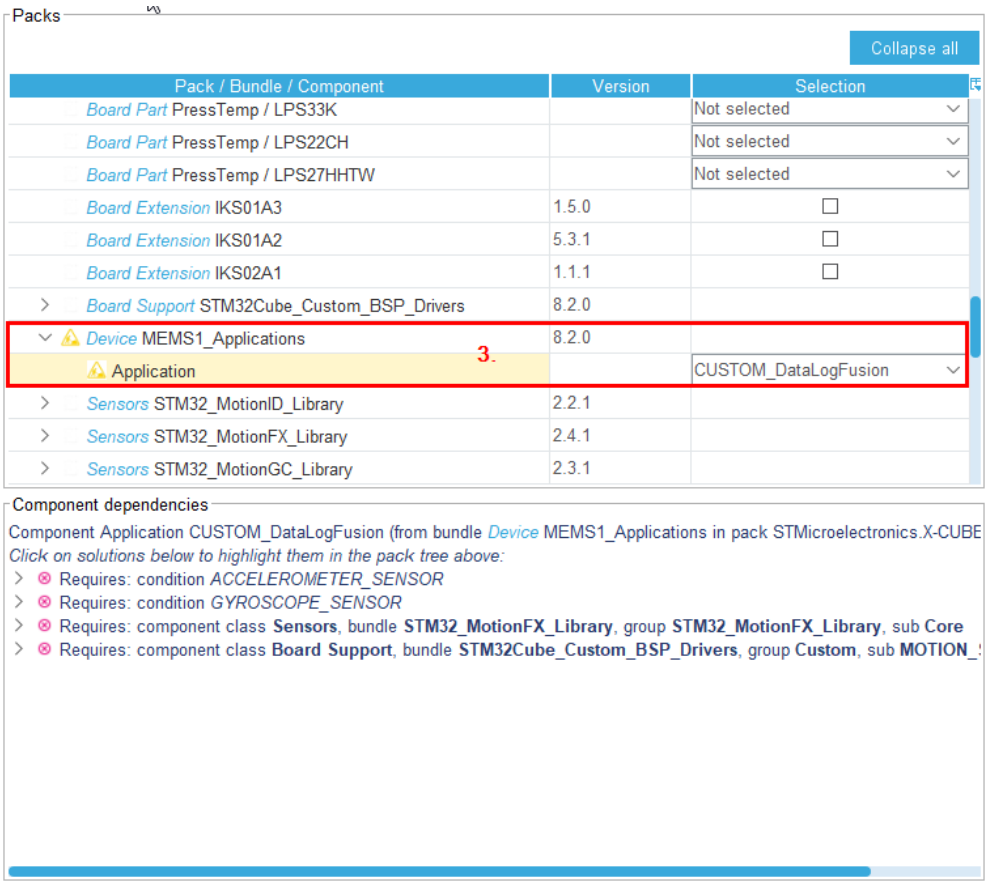
Figure 13. X-CUBE-MEMS1 pack selection



5.3 Application selection

From [**STMicroelectronics.X-CUBE-MEMS1 8.2.0**], choose [**Device MEMS1_Applications**]>[**Application: CUSTOM_DataLogFusion**] (3).

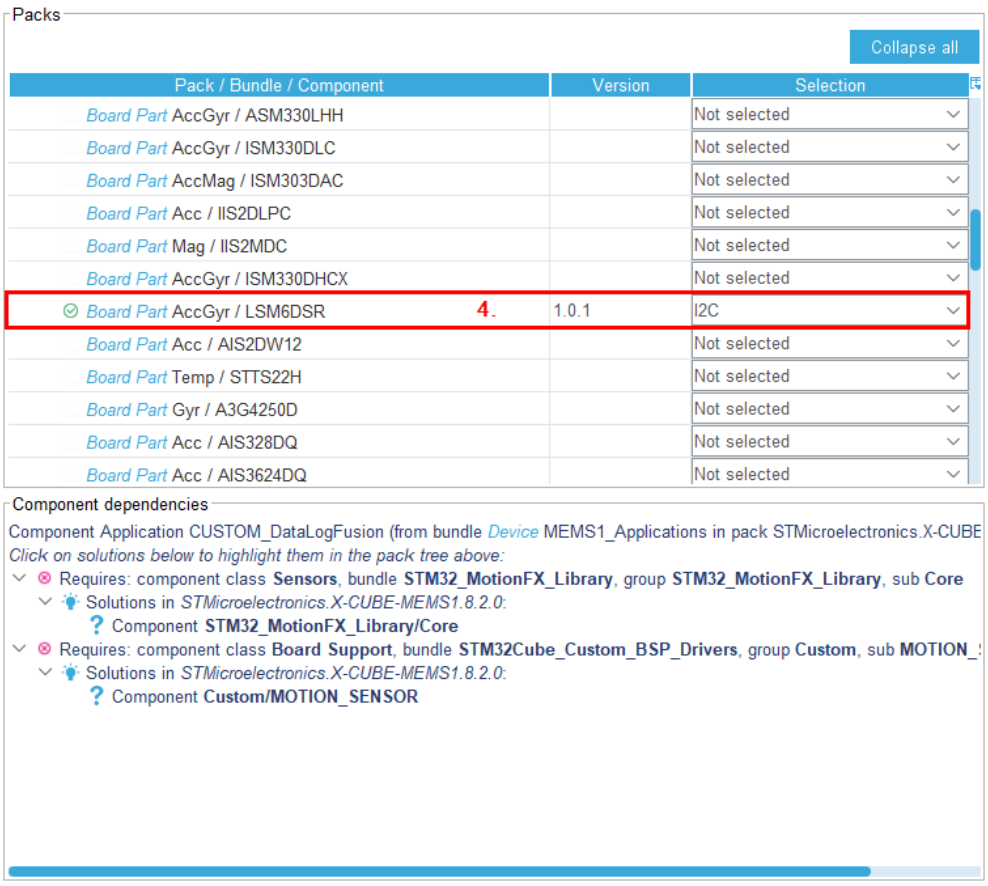
Figure 14. Application selection



5.4 Accelerometer and gyroscope selection

From [**STMicroelectronics.X-CUBE-MEMS1 8.2.0**], choose [**Board Part AccGyr / LSM6DSR 1.0.1 I2C**] (4). We will use I^2C communication in this example.

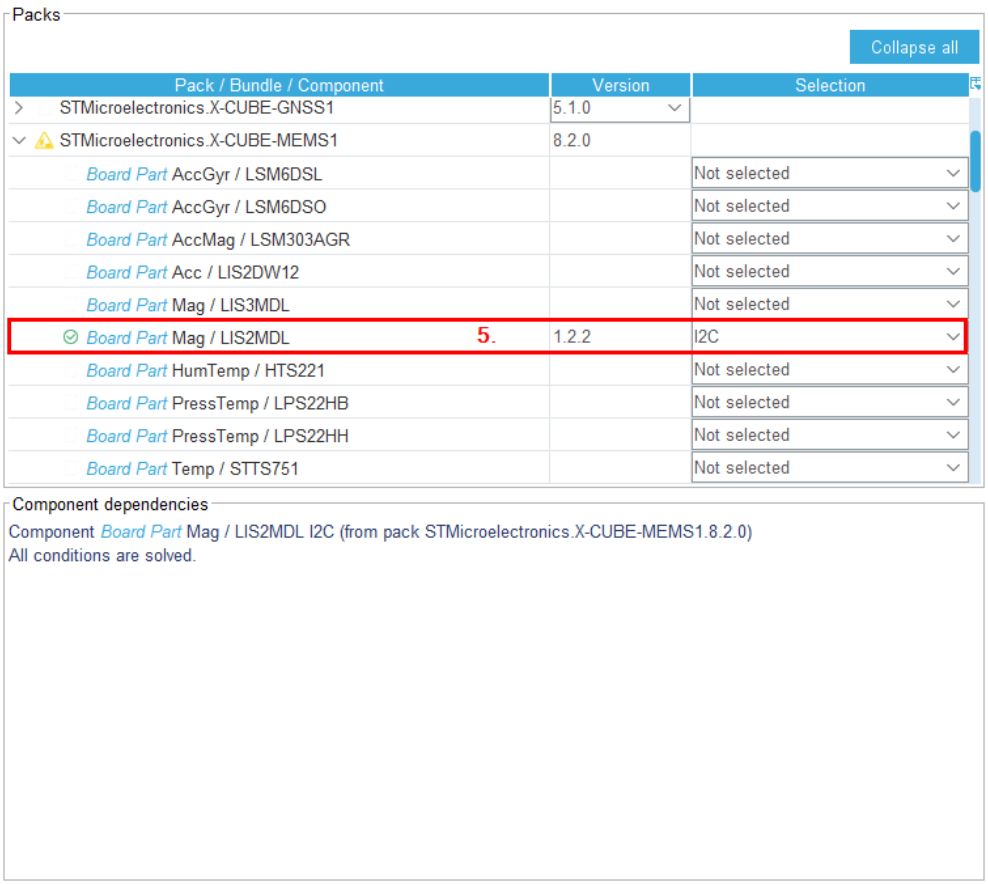
Figure 15. Accelerometer and gyroscope selection



5.5 Magnetometer selection (optional)

From [**STMicroelectronics.X-CUBE-MEMS1 8.2.0**], choose [**Board Part Mag / LIS2MDL 1.2.2 I2C**] (5). We will use I^2C communication in this example.

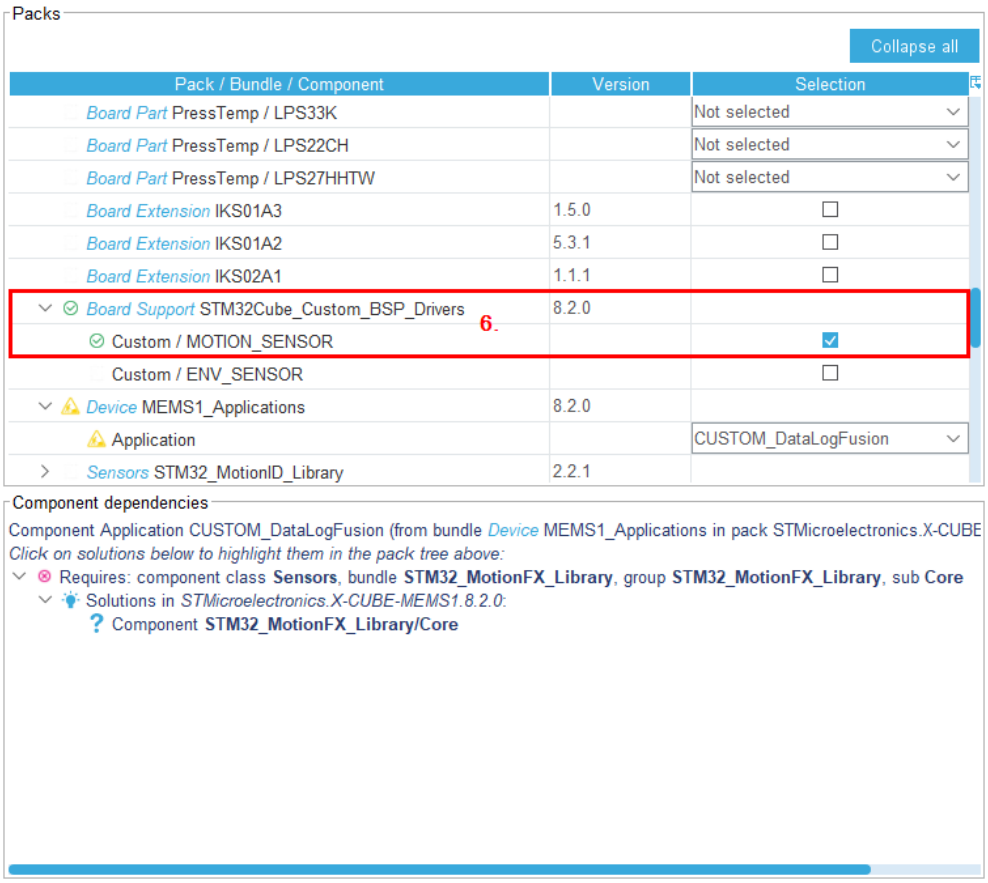
Figure 16. Magnetometer selection



5.6 BSP driver selection

From [**STMicroelectronics.X-CUBE-MEMS1 8.2.0**], choose [**Board Support STM32Cube_custom_BSP_Drivers**]>[**Custom**]>[**MOTION_SENSOR**] (6).

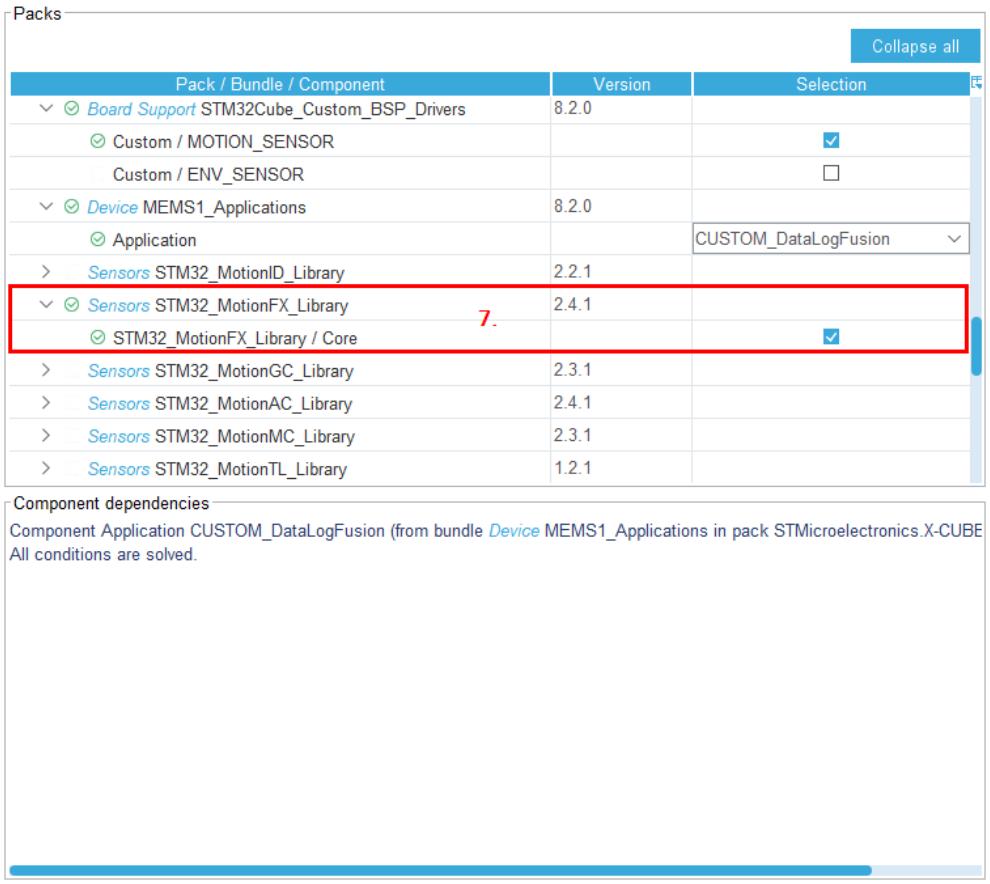
Figure 17. BSP driver selection



5.7 Algorithm library selection

Step 1. From [**STMicroelectronics.X-CUBE-MEMS1 8.2.0**], choose [**Sensors STM32_MotionFX_Library**]>[**STM32_MotionFX_Library/Core**] (7).

Figure 18. Middleware selection

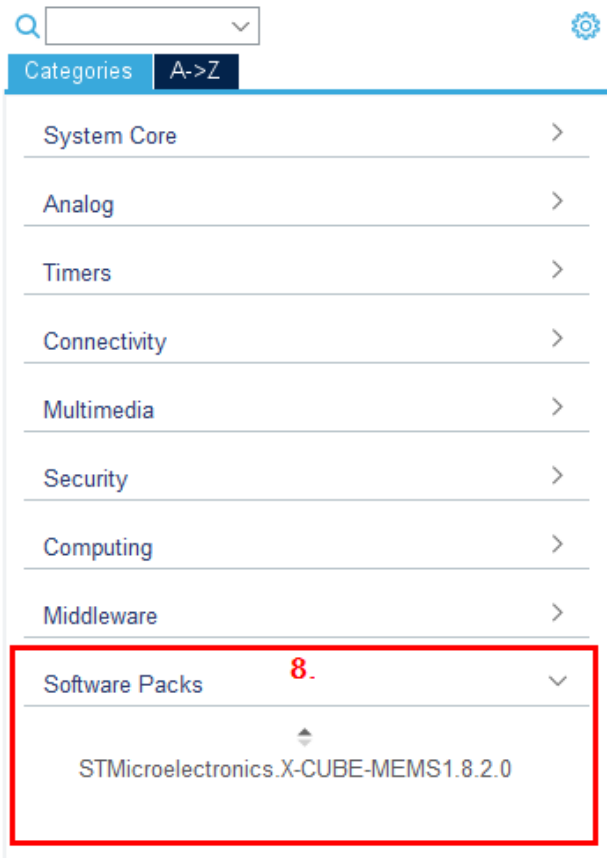


Step 2. Click [**OK**] to confirm the selected [**Software Packs**] setup.

5.8 Software pack configuration

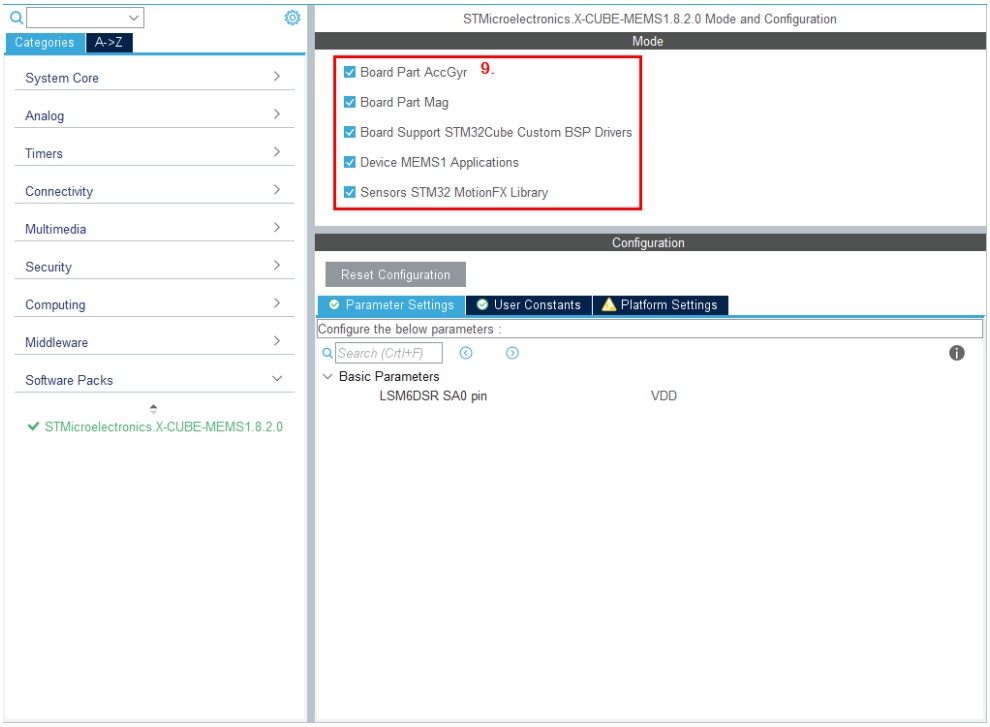
Step 1. Select [**Software Packs**]>[**STMicroelectronics.X-CUBE-MEMS1.8.2.0**] (8).

Figure 19. Software pack



Step 2. Check all check-boxes (9).

Figure 20. Software pack mode selection

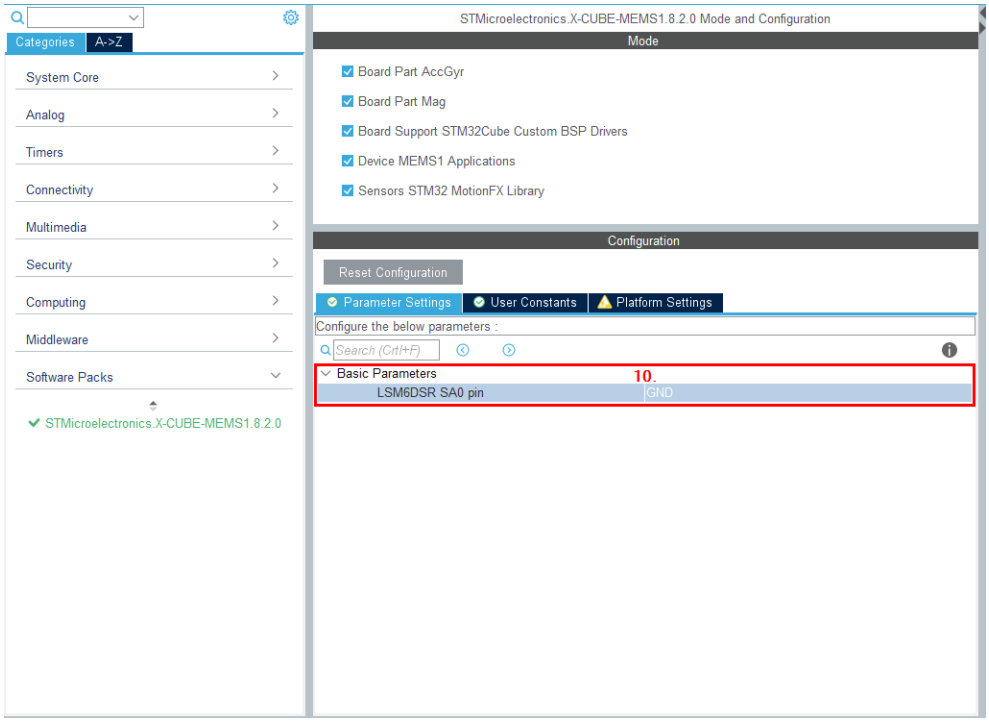


5.9 DIL 24 component custom configuration

Step 1. Change `LSM6DSR SA0 pin` to `GND - DIL24` components by using `SA0 = 0` (10).

I²C address for sensors is determined by the SA0 pin. The sensors on the board have SA0 pull-up. The sensor on the DIL24 socket must use SA0 pull-down to avoid conflicts (a conflict is when two different sensors have the same I²C address and try to talk over each other).

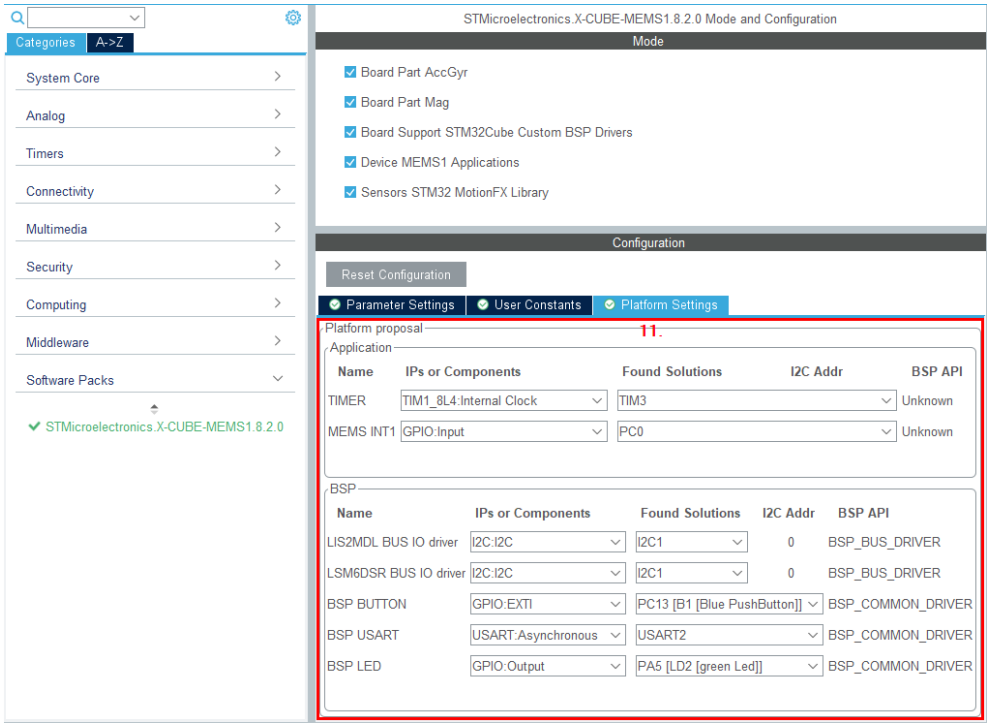
Figure 21. LSM6DSR SA0 pin configuration



5.10 Platform configuration

Step 1. Assign previously configured peripherals to required peripherals by the application example in [**Platform Settings**] (11).

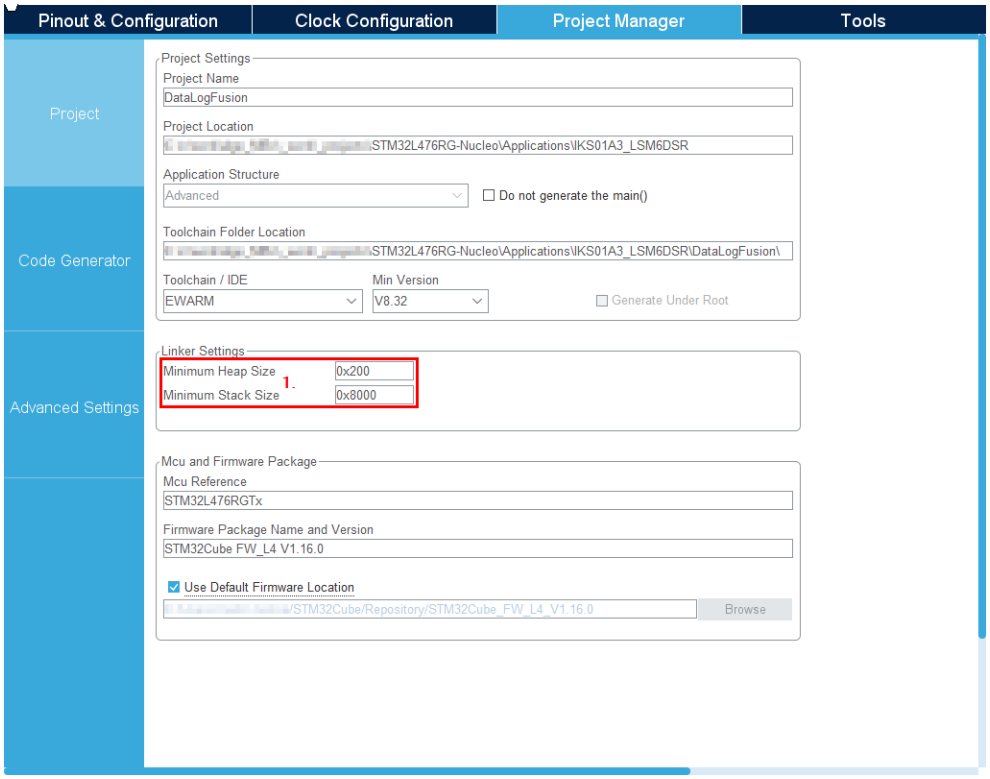
Figure 22. Platform configuration



6 Project setup

Step 1. In [**Project Manager**]>[**Project**], set [**Minimum Heap Size/Minimum Stack Size**] as shown in the picture below (1).

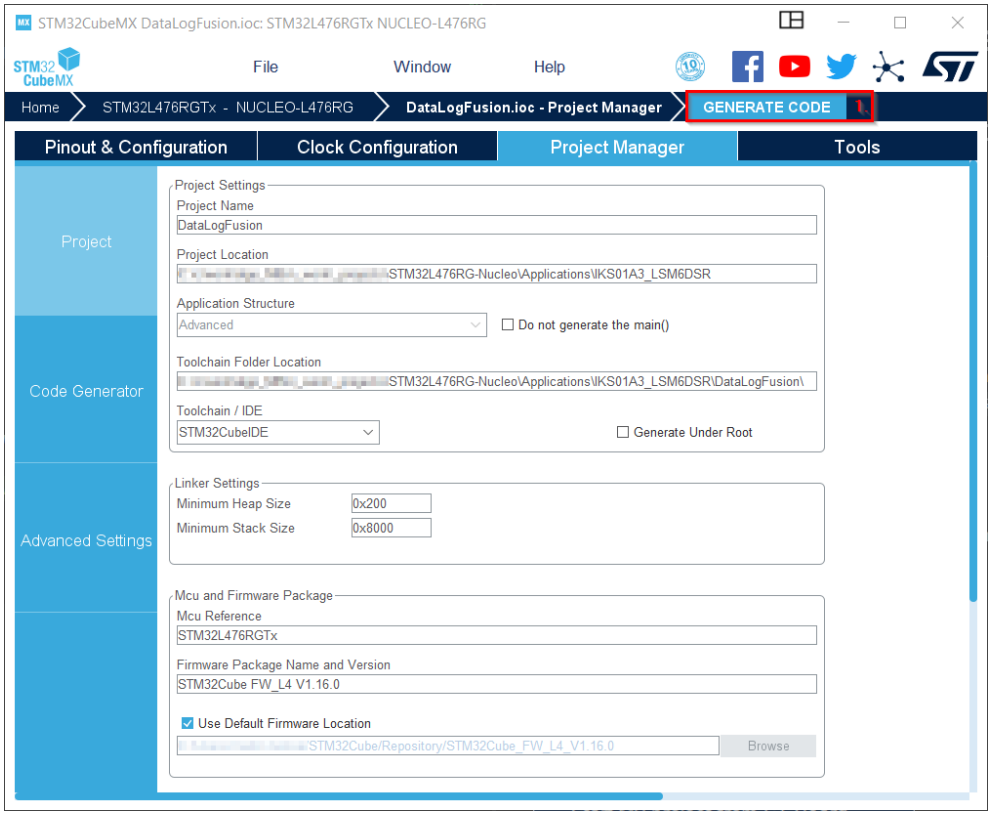
Figure 23. Heap and Stack size configuration



7 Project generation

Step 1. Click on [**GENERATE CODE**] (1) to generate project files and sources.

Figure 24. Code generation



8 Code update (optional)

It is necessary modify the source code for algorithms and sensors whose correct functionality depends on the orientation. In our case the `DataLogFusion` application depends on the correct orientation of MEMS sensors.

When building a project, the following warning messages might appear:

```
#warning Function BSP_SENSOR_ACC_GetOrientation is not implemented
#warning Function BSP_SENSOR_GYR_GetOrientation is not implemented
#warning Function BSP_SENSOR_MAG_GetOrientation is not implemented
```

These warnings are built into the project to set forced orientation according to the actual hardware setup. For example, concerning the accelerometer, part of the source code is:

```
/**
  * @brief  Get accelerometer sensor orientation
  * @param  Orientation Pointer to sensor orientation
  * @retval None
  */
void BSP_SENSOR_ACC_GetOrientation(char *Orientation)
{
#if (defined BSP_MOTION_SENSORS)
  #ifdef CUSTOM_ACC_INSTANCE_0
  #warning Function BSP_SENSOR_ACC_GetOrientation is not implemented
/*
  Example:
  Orientation[0] = 's';
  Orientation[1] = 'e';
  Orientation[2] = 'u';
*/
  #endif
#endif
}
```

The hardware configuration for this example is:

- STEVAL-MKI194V1 DIL24 module - LSM6DSR accelerometer and gyroscope sensor in DIL 24 socket
- X-NUCLEO-IKS01A3 expansion board

Figure 25. LSM6DSR sensor orientation

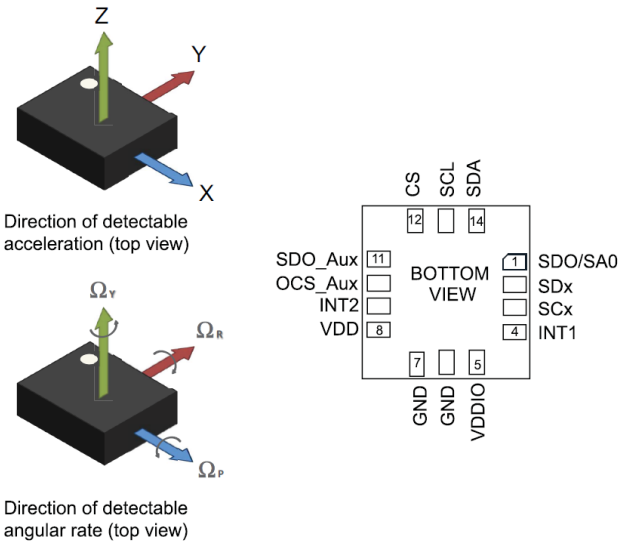


Figure 26. **STEVAL-MKI194V1 evaluation board**

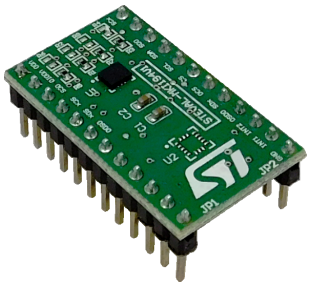
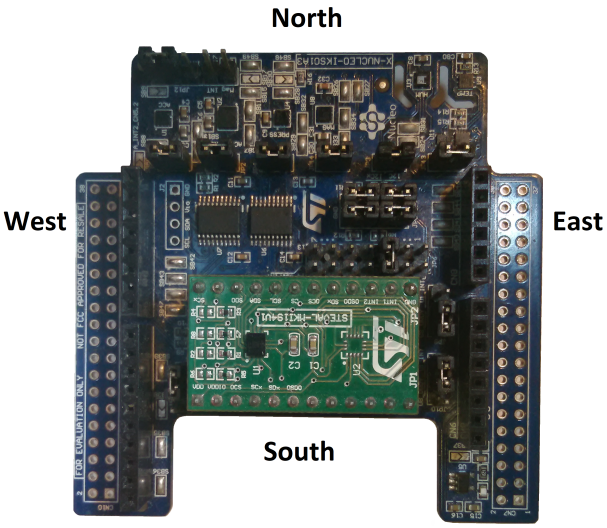


Figure 27. **STEVAL-MKI194V1 stacked on top of X-NUCLEO-IKS01A3**



The orientation of the accelerometer (LSM6DSR) is North-West-Up (`NWU`). The above code should be modified as follows:

```
/**
  * @brief  Get accelerometer sensor orientation
  * @param  Orientation Pointer to sensor orientation
  * @retval None
  */
void BSP_SENSOR_ACC_GetOrientation(char *Orientation)
{
  Orientation[0] = 'n';
  Orientation[1] = 'w';
  Orientation[2] = 'u';
}
```

A similar change has to be done for gyroscope (LSM6DSR, orientation = NWU) and magnetometer (LIS2MDL, orientation = NEU).

9 Sensors with I3C

9.1 Description

Some sensors have the option of using the I3C interface unlike the X-NUCLEO expansion boards and the STM32 Nucleo development boards which use I2C interface only.

Due to the connection to ST2378E level shifter (Figure 28), the sensor (in DIL 24) interrupt pins (Figure 29) are pulled high through a 9 kOhm resistor (Figure 30): thus, devices with I3C bus enable the I3C interface. As the boards use the I2C only, the I3C must be disabled.

The procedures described hereafter are available to ensure that I3C is disabled and I2C is enabled.

Figure 28. Level shifter circuit

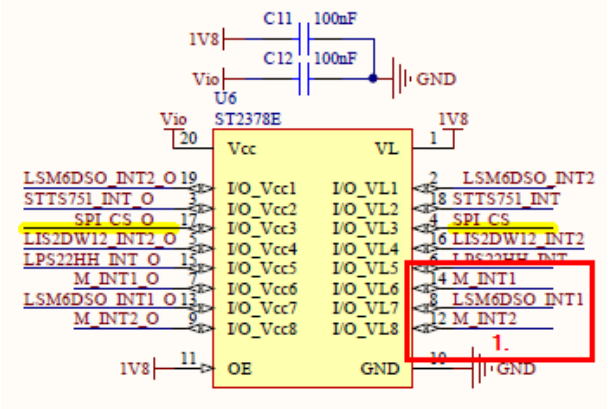


Figure 29. DIL 24 socket wiring

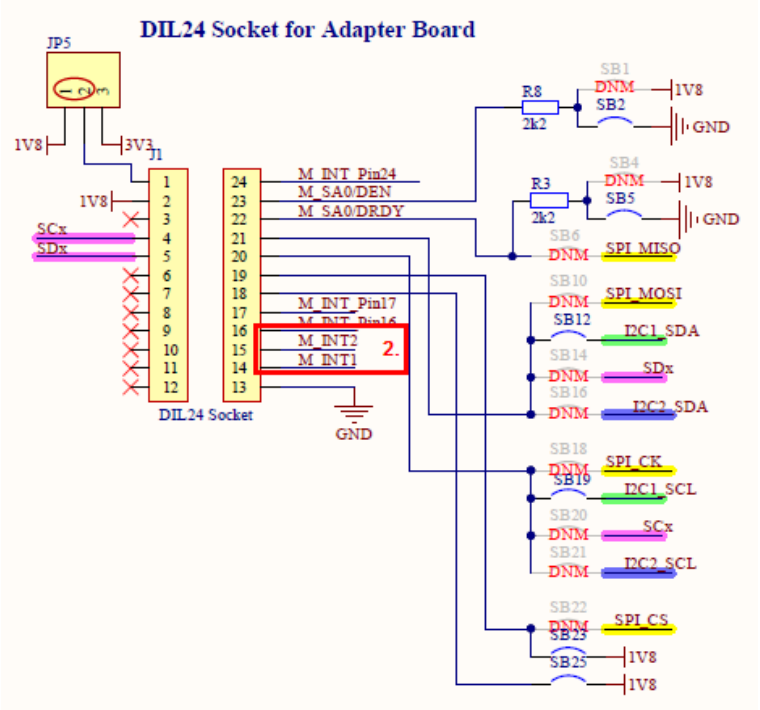
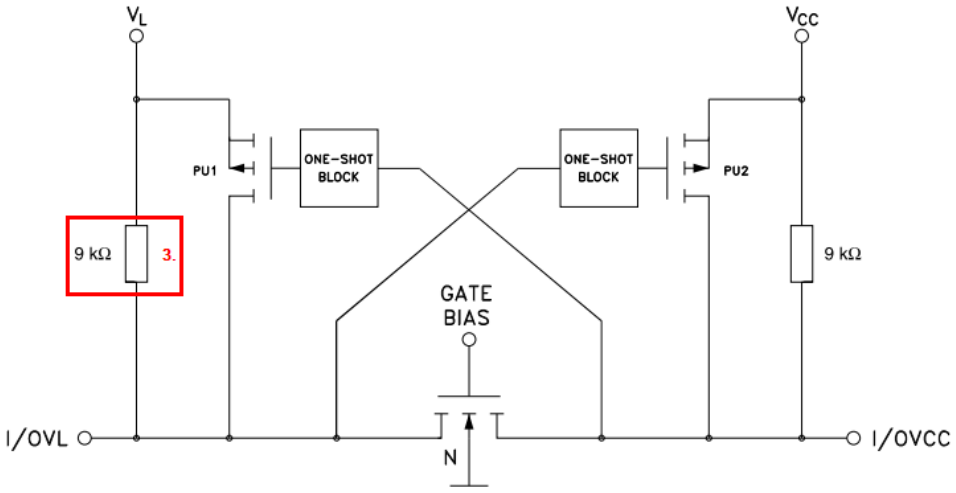


Figure 30. **Level shifter internal circuitry**



9.2 Hardware solution

Connect a strong external pull-down resistor (< 1 kOhm) to `INT1` pin.

9.3 Software solution

9.3.1 Motion sensors

To disable I3C via software, for motion sensors follow the procedure below.

Step 1. Configure `GPIO` for `INT1` pin to output and set the output value to low.

Step 2. During the sensor initialization procedure, disable I3C in the sensor register.

Step 3. Reconfigure the `GPIO` for `INT1` pin to input.

9.3.2 Environmental sensors

To disable I3C via software, for environmental sensors (without hot-join, e.g.: LPS22HH) follow the procedure below.

Step 1. Configure `GPIO` for `INT1` pin to output and set the output value to low.

Step 2. Manually generate 9 clock pulses on `SCL` to unlock the bus.

Step 3. During the sensor initialization procedure, disable I3C in the sensor register.

Step 4. Reconfigure the `GPIO` for `INT1` pin to input.

9.4 Solution used in STM32CubeMX

The following solution has been applied in the project generated by STM32CubeMX as described hereafter.

In lines 166 .. 173 of MEMS/APP/app_mems.c file:

```
#ifdef BSP_IP_MEMS_INT1_PIN_NUM
  /* Force MEMS INT1 pin of the sensor low during startup in order to disable I3C and enable
I2C. This function needs
   * to be called only if user wants to disable I3C / enable I2C and didn't put the pull-
down resistor to MEMS INT1 pin
   * on his HW setup. This is also the case of usage X-NUCLEO-IKS01A2 or X-NUCLEO-IKS01A3
expansion board together with
   * sensor in DIL24 adapter board where the LDO with internal pull-up is used.
   */
  MEMS_INT1_Force_Low();
#endif
```

and in lines 196 .. 199:

```
#ifdef BSP_IP_MEMS_INT1_PIN_NUM
  /* Initialize MEMS INT1 pin back to it's default state after I3C disable / I2C enable */
  MEMS_INT1_Init();
#endif
```

Revision history

Table 1. **Document revision history**

Date	Revision	Changes
11-Jun-2019	1	Initial release
19-Mar-2021	2	Updated all content to add guidelines on how to create example applications for sensors in DIL 24 socket.

Contents

List of figures

List of tables