# Getting started with the X-CUBE-MEMS1 motion MEMS and environmental sensor software expansion for STM32Cube

## Introduction

The X-CUBE-MEMS1 software expansion pack for STM32Cube is designed to run on STM32 hardware. It includes drivers that add functionality to recognize connected sensors, configure them and collect data such as motion, temperature, humidity, pressure and QVAR.

Additionally, the pack provides middleware focused on processing sensor data. They can be divided into groups based on their general functionality: Calibration algorithms, Position tracking, Activity tracking for mobile devices, Activity tracking for wrist devices, Industrial algorithms, and Infrared algorithms. Go to Section 6: Middleware to see the full list of individual middlewares.

The expansion is built on STM32Cube software technology to ease portability across wide range of STM32 microcontrollers. The software comes with sample projects that utilize the provided middleware and drivers, featuring X-NUCLEO-IKS02A1, X-NUCLEO-IKS4A1 and X-NUCLEO-IKS5A1 expansion boards connected to NUCLEO-L073RE, NUCLEO-L152RE, NUCLEO-F401RE and NUCLEO-U575I-Q development boards. Significant portion of the preconfigured projects supports the use of custom boards. Apart from the provided projects, users can use the pack to create custom projects with help from STM32CubeMX software.

The software can be installed directly from the STM32CubeMX software or be obtained from GitHub. Users can contribute to the software's development by reporting bugs through Issues and sharing new ideas through Pull requests.

**Related links**

*Visit the STM32Cube ecosystem web page for further information*

# 1 X-CUBE-MEMS1 software expansion for STM32Cube

## 1.1 Overview

The X-CUBE-MEMS1 software package enhances the STM32Cube by extending its functionality to enable configuration, usage, and processing of data from various MEMS devices.

The key features are:

- Supports more than 50 MEMS devices – see Table 1. Supported MEMS devices for the full list
- Contains advanced libraries that process data collected by connected sensors – see Table 4. Included Middlewares for the full list
- Contains examples that demonstrate driver functionality and sensor features – see Section 5: Sensor Evaluation Examples for the full list
- Contains sample applications demonstrate features of implemented libraries and transmit real-time sensor data to a PC – see Table 5. Library evaluation applications
- Applications and select examples are compatible with the MEMS-Studio graphical PC application, which can be used to visualized the collected and processed data
- Extended support is provided for the X-NUCLEO-IKS02A1, X-NUCLEO-IKS4A1 and X-NUCLEO-IKS5A1 expansion boards – this adds preconfigured examples and applications which use the NUCLEO-L073RZ, NUCLEO-L152RE, NUCLEO-F401RE and NUCLEO-U575ZI-Q development boards
- Package is compatible with STM32CubeMX, and can be downloaded and installed directly from the application
- Free, user-friendly license terms

## 1.2 Architecture

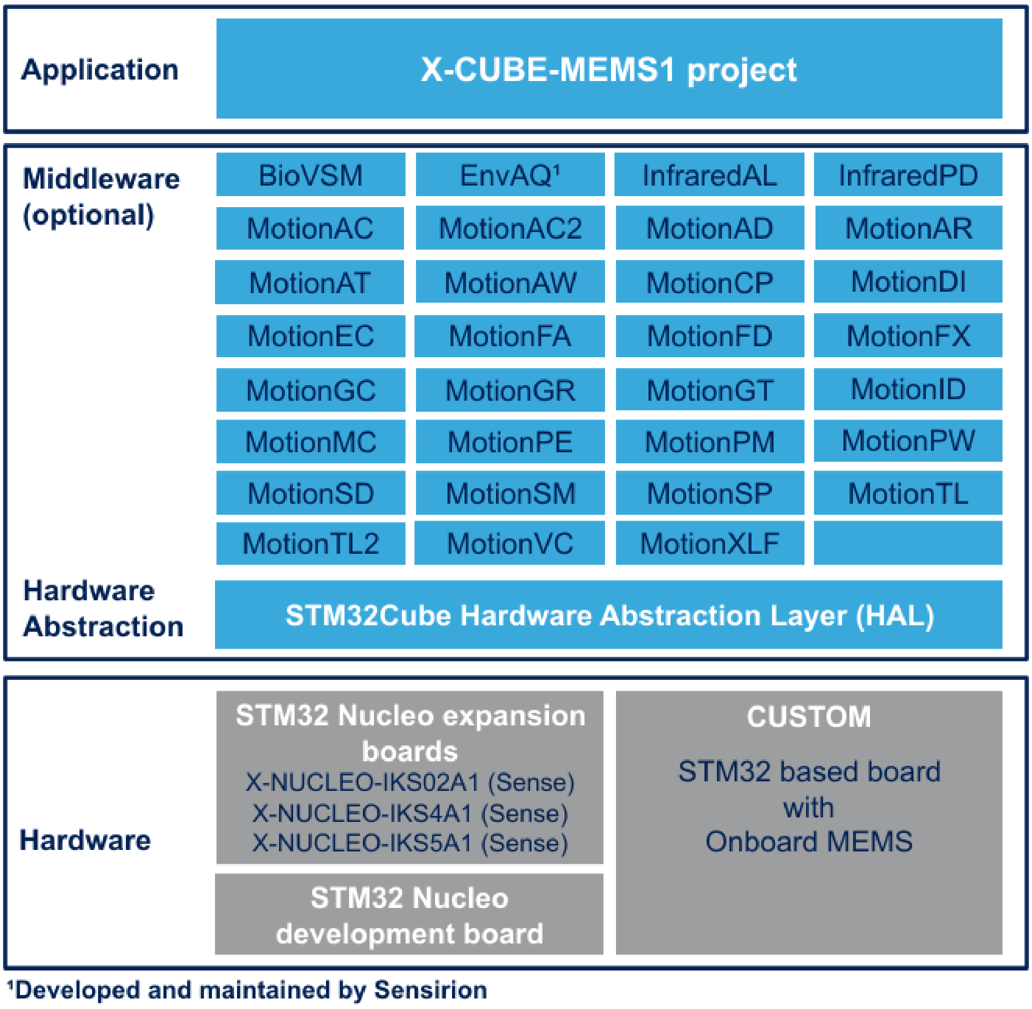
This software is a fully compliant expansion for STM32Cube enabling development of applications using inertial and environmental sensors.

The software is based on the hardware abstraction layer for the STM32 microcontroller, STM32CubeHAL. The package extends STM32Cube by providing a Board Support Package (BSP) for the sensor expansion board and a sample application for serial communication with a PC.

The software layers used by the application software to access the sensor expansion board are:

- The STM32Cube HAL driver layer provides a simple, generic and multi-instance set of APIs (application programming interfaces) to interact with the upper layers (application, libraries and stacks). It includes generic and extension APIs and is based on a generic architecture which allows the layers built on it (such as the middleware layer) to implement their functionalities without dependence on the specific hardware configuration of a given Microcontroller Unit (MCU). This structure improves library code reusability and guarantees high portability across other devices.
- The Board Support Package (BSP) layer provides supporting software for the peripherals on the STM32 Nucleo board, except for the MCU. It has a set of APIs to provide a programming interface for certain board-specific peripherals (e.g. the LED, the user button etc.) and allow identification of the specific board version. For the sensor expansion board, it provides the programming interface for various inertial and environmental sensors and provides support for initializing and reading sensor data.
- The Middleware layer provides advanced libraries used to process data collected by sensors.

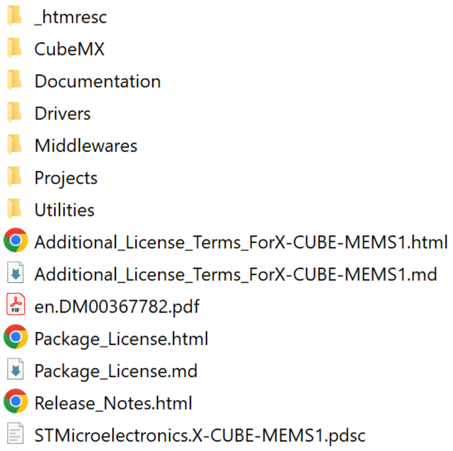
Figure 1. **X-CUBE-MEMS1 software architecture**



## 1.3 Folder structure

The software pack (if downloaded from GitHub or the official ST website) has the following folder structure:

Figure 2. **X-CUBE-MEMS1 package folder structure**



```
_htmresc
CubeMX
Documentation
Drivers
Middlewares
Projects
Utilities
Additional_License_Terms_ForX-CUBE-MEMS1.html
Additional_License_Terms_ForX-CUBE-MEMS1.md
en.DM00367782.pdf
Package_License.html
Package_License.md
Release_Notes.html
STMicroelectronics.X-CUBE-MEMS1.pdsc
```

- The **CubeMX** folder contains the STM32CubeMX templates

- The **Documentation** folder contains a compiled HTML file generated from the source code and detailed documentation regarding the software components and APIs
- The **Drivers** folder contains the HAL drivers, the board-specific drivers for each supported board or hardware platform, including those for the on-board components and the CMSIS layer, which is a vendor-independent hardware abstraction layer for the Cortex-M processor series
- The **Middlewares** folder contains the motion libraries, a platform-independent software layer provided in binary format for the supported Cortex processor series
- The **Projects** folder contains configured projects for the NUCLEO-L073RZ, NUCLEO-L152RE, NUCLEO-F401RE, and NUCLEO-U575ZI-Q platforms in specific hardware configuration. You will find them in the following location (relative to the `Projects` folder): `./<nucleo_board>/<project_type>/<expansion_board>/<project_name>`, where you replace the bracketed <> fields by your actual configuration (e.g. `./NUCLEO-F401RE/Applications/IKS4A1/AccelerometerCalibration`). You'll find project files for the supported IDEs there
- The **Utilities** folder contains a `PC_software` subfolder which in turn contains `MEMS-Studio.htm` html file with a download link for the MEMS-Studio desktop utility

## 1.4 API

Detailed technical information about the API available to the user can be found in the `X_CUBE_MEMS1.chm` compiled HTML file located in the `Documentation` folder of the software package. All the functions and parameters are fully described in it.

# 2 Supported hardware

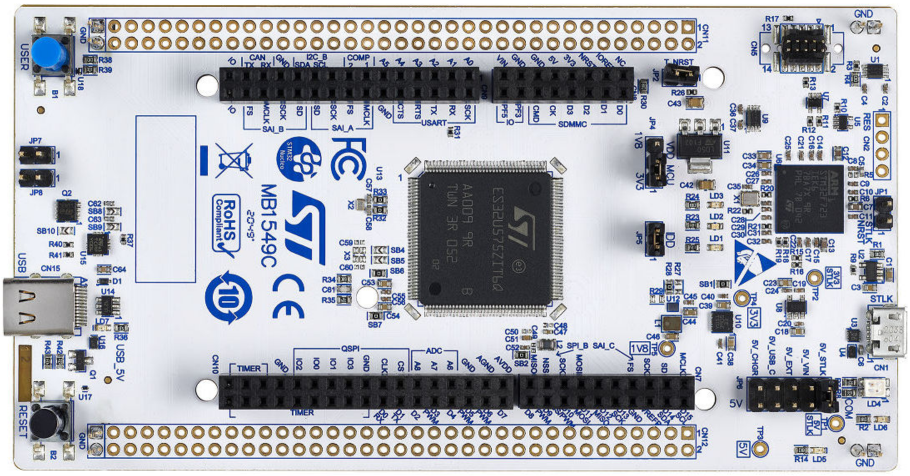
## 2.1 STM32 Nucleo

STM32 Nucleo development boards provide an affordable and flexible way for users to test solutions and build prototypes with any STM32 microcontroller line. They contain ST-LINK debugger/programmer onboard, therefore not requiring additional hardware for development.

The Arduino UNO R3 connectivity support and ST MORPHO connector make it easy to expand the board's functionality by enabling it to connect to a huge variety of available expansion boards.

Theplatform comes with the comprehensive STM32 software HAL library support together with packaged software examples supporting various IDEs (IAR EWARM, MDK-ARM, STM32CubeIDE, mbed and GCC/LLVM).

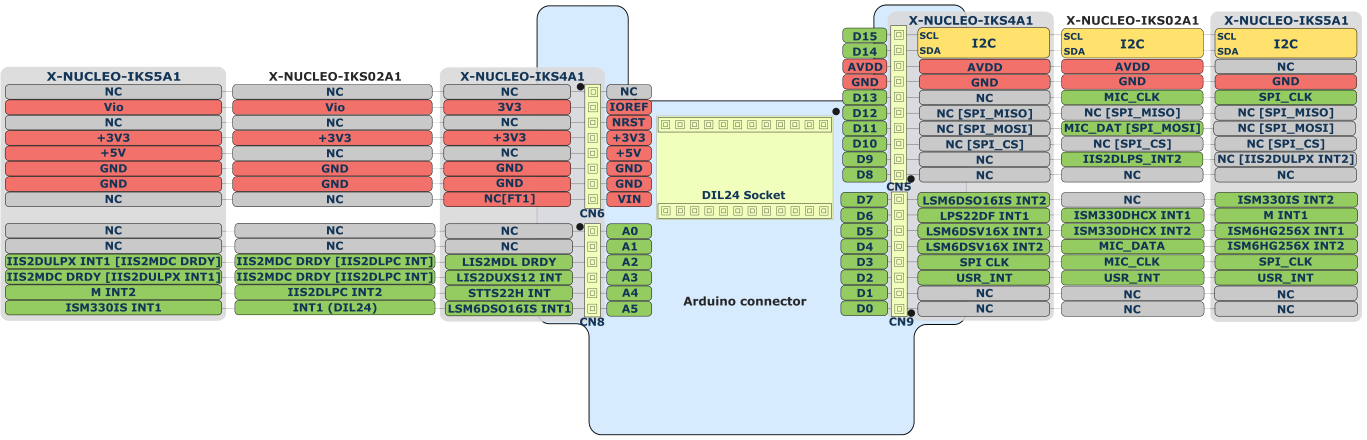
**Figure 3. STM32 Nucleo board**



## 2.2 X-NUCLEO expansion boards

X-NUCLEO expansion boards are versatile add-on boards designed to extend the functionality of STM32 Nucleo development boards. They are equipped with Arduino UNO R3 connector and ST MORPHO connector which allow you to connect it to a NUCLEO Board. Additionally these boards contain multiple solder bridges and jumpers to allow further customization. X-NUCLEO-IKS02A1, X-NUCLEO-IKS4A1 and X-NUCLEO-IKS5A1 are supported by the X-CUBE-MEMS. They all contain multiple MEMS devices and feature DIL24 connector which can house one additional sensor, provided it is supported and on a DIL24 compatible adapter board.

**Figure 4. X-NUCLEO-IKS02A1, X-NUCLEO-IKS4A1 and X-NUCLEO-IKS5A1 – pinouts**



| X-NUCLEO-IKS5A1 | X-NUCLEO-IKS02A1 | X-NUCLEO-IKS4A1 | CN6 |
|---|---|---|---|
| NC | NC | NC | NC |
| Vio | Vio | 3V3 | IOREF |
| NC | NC | NC | NRST |
| +3V3 | +3V3 | +3V3 | +3V3 |
| +5V | NC | NC | +5V |
| GND | GND | GND | GND |
| GND | GND | GND | GND |
| NC | NC | NC[FT1] | VIN |

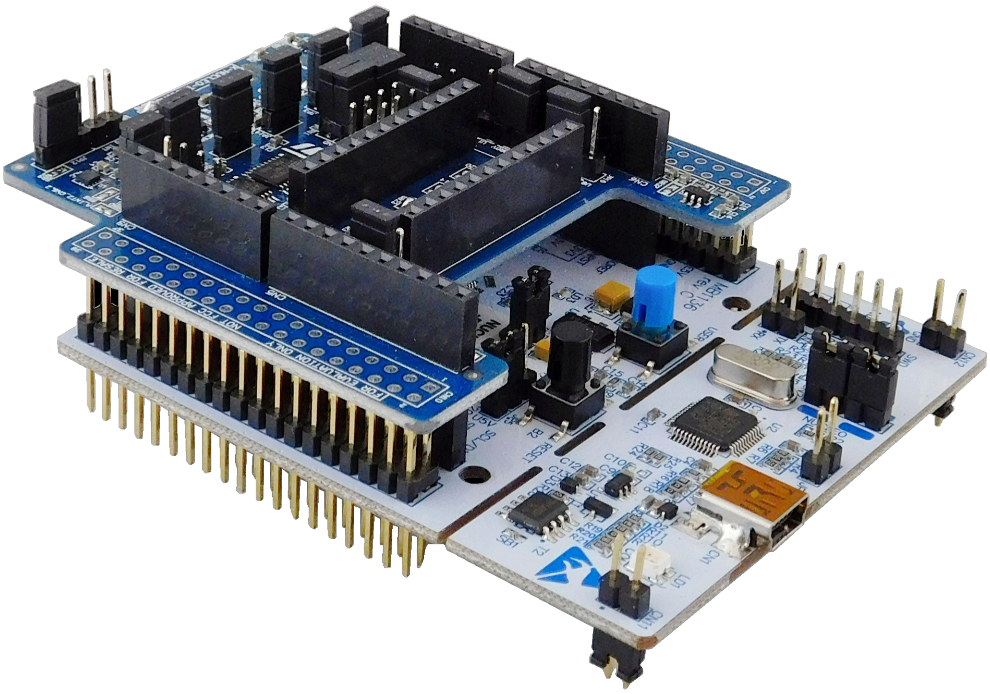
| X-NUCLEO-IKS5A1 | X-NUCLEO-IKS02A1 | X-NUCLEO-IKS4A1 | CN8 |
|---|---|---|---|
| NC | NC | NC | A0 |
| NC | NC | NC | A1 |
| IIS2DULPX INT1 [IIS2MDC DRDY] | IIS2MDC DRDY [IIS2DLPC INT] | LIS2MDL DRDY | A2 |
| IIS2MDC DRDY [IIS2DULPX INT1] | IIS2MDC DRDY [IIS2DLPC INT] | LIS2DUXS12 INT | A3 |
| M INT2 | IIS2DLPC INT2 | STTS22H INT | A4 |
| ISM330IS INT1 | INT1 (DIL24) | LSM6DSO16IS INT1 | A5 |

| CN5 | X-NUCLEO-IKS4A1 | X-NUCLEO-IKS02A1 | X-NUCLEO-IKS5A1 |
|---|---|---|---|
| D15 | SCL I2C | SCL I2C | SCL I2C |
| D14 | SDA I2C | SDA I2C | SDA I2C |
| AVDD | AVDD | AVDD | NC |
| GND | GND | GND | GND |
| D13 | NC | MIC_CLK | SPI_CLK |
| D12 | NC [SPI_MISO] | NC [SPI_MISO] | NC [SPI_MISO] |
| D11 | NC [SPI_MOSI] | MIC_DAT [SPI_MOSI] | NC [SPI_MOSI] |
| D10 | NC [SPI_CS] | NC [SPI_CS] | NC [SPI_CS] |
| D9 | NC | IIS2DLPS_INT2 | NC [IIS2DULPX INT2] |
| D8 | NC | NC | NC |

| CN9 | X-NUCLEO-IKS4A1 | X-NUCLEO-IKS02A1 | X-NUCLEO-IKS5A1 |
|---|---|---|---|
| D7 | LSM6DSO16IS INT2 | NC | ISM330IS INT2 |
| D6 | LPS22DF INT1 | ISM330DHCX INT1 | M INT1 |
| D5 | LSM6DSV16X INT1 | ISM330DHCX INT2 | ISM6HG256X INT1 |
| D4 | LSM6DSV16X INT2 | MIC_DATA | ISM6HG256X INT2 |
| D3 | SPI CLK | MIC_CLK | SPI_CLK |
| D2 | USR_INT | USR_INT | USR_INT |
| D1 | NC | NC | NC |
| D0 | NC | NC | NC |

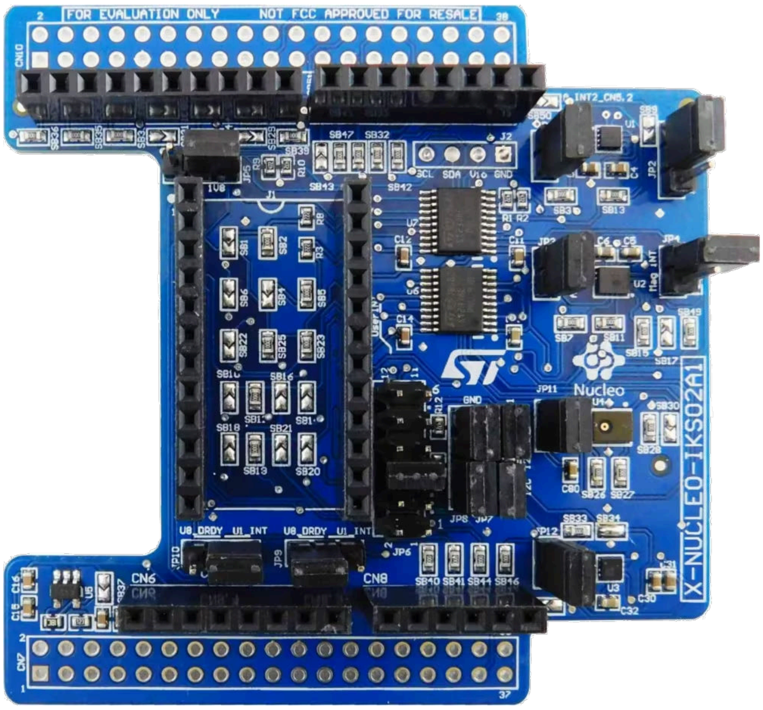
*The content of the brackets - [...] in pin description shows alternative function which can be accessed with an alternative solderbridge/jumper configuration

Figure 5. Demonstration – expansion board connected to a Nucleo64 development board



### 2.2.1 X-NUCLEO-IKS02A1

Figure 6. X-NUCLEO-IKS02A1 expansion board

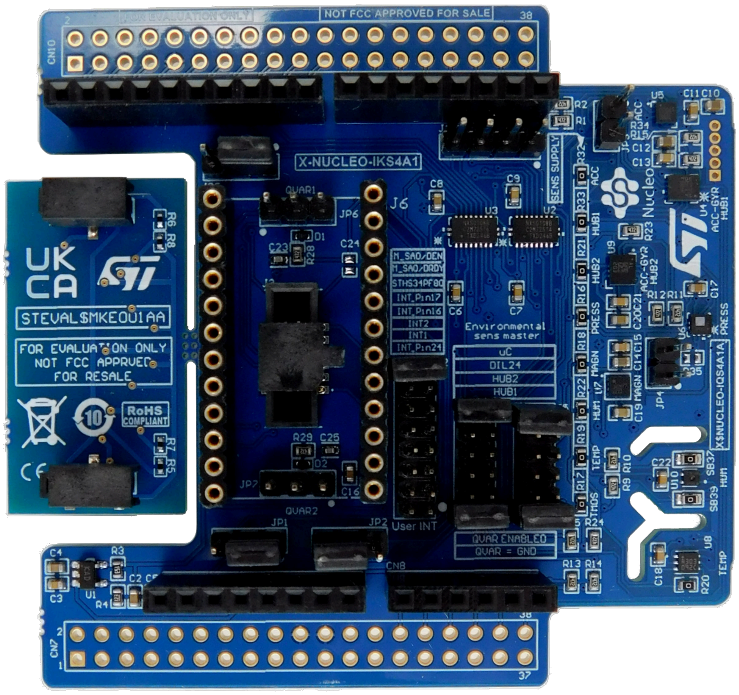


**Onboard sensors**

- ISM330DHCX – 3D Accelerometer, 3D Gyroscope
- IIS2MDC – 3D Magnetometer
- IIS2DLPC – 3D Accelerometer
- IMP34DT05 – Microphone

### 2.2.2 X-NUCLEO-IKS4A1

Figure 7. X-NUCLEO-IKS4A1 MEMS and environmental sensor expansion board



#### Integrated sensors

- LSM6DSO16IS – 3D Accelerometer, 3D Gyroscope, ISPU
- LSM6DSV16X – 3D Accelerometer, 3D Gyroscope, Qvar, FSM, MLC
- LIS2DUXS12 – 3D Accelerometer, Qvar, FSM, MLC
- LIS2MDL – 3D Magnetometer
- LPS22DF – Barometer
- SHT40AD1B – Humidity, Temperature
- STTS22H – Temperature

### 2.2.3 X-NUCLEO-IKS5A1

#### Integrated sensors

- ISM330IS – 3D Accelerometer, 3D Gyroscope, ISPU
- ISM6HG256X – 3D Accelerometer, 3D High-g Accelerometer, 3D Gyroscope, Qvar, FSM, MLC
- ILPS22QS – Pressure
- IIS2DULPX – FSM, MLC, Qvar
- IIS2MDC – 3D Magnetometer

## 2.3 Supported MEMS devices

Table 1. Supported MEMS devices

| Device name | Device type | Connection |
|---|---|---|
| A3G4250D | Angular rate | DIL24 |
| AIS2DW12 | Acceleration | DIL24 |

| Device name | Device type | Connection |
|---|---|---|
| AIS2IH | Acceleration | DIL24 |
| AIS328DQ | Acceleration | DIL24 |
| AIS3624DQ | Acceleration | DIL24 |
| ASM330LHH | Acceleration, Angular rate | DIL24 |
| ASM330LHHX | Acceleration, Angular rate, MLC | DIL24 |
| H3LIS331DL | Acceleration | DIL24 |
| HTS221 | Temperature, Humidity | DIL24 |
| IIS2DLPC | Acceleration | IKS02A1, DIL24 |
| IIS2DULPX | Acceleration, Angular rate, FSM, MLC | IKS5A1, DIL24 |
| IIS2ICLX | Acceleration, Inclinometer | DIL24 |
| IIS2MDC | Magnetometer | IKS5A1, IKS02A1, DIL24 |
| IIS3DWB | Acceleration, Vibrometer | DIL24 |
| ILPS22QS | Pressure, QVAR | IKS5A1, DIL24 |
| ILPS28QSW | Pressure, QVAR | DIL24 |
| ISM303DAC | Acceleration, Magnetometer | DIL24 |
| ISM330BX | Acceleration, Angular rate, FSM, MLC | DIL24 |
| ISM330DHCX | Acceleration, Angular rate, FSM | IKS02A1, DIL24 |
| ISM330DLC | Acceleration, Angular rate | DIL24 |
| ISM330IS | Acceleration, Angular rate, ISPU | IKS5A1, DIL24 |
| ISM6HG256X | Acceleration (high-g/low-g), Angular rate, FSM, MLC | IKS5A1, DIL24 |
| LIS2DH12 | Acceleration | DIL24 |
| LIS2DTW12 | Acceleration, Temperature | DIL24 |
| LIS2DU12 | Acceleration | DIL24 |
| LIS2DUX12 | Acceleration, FSM, MLC | DIL24 |
| LIS2DUXS12 | Acceleration, QVAR, FSM, MLC | IKS4A1, DIL24 |
| LIS2DW12 | Acceleration | DIL24 |
| LIS2MDL | Magnetometer | IKS4A1, DIL24 |
| LIS3MDL | Magnetometer | DIL24 |
| LPS22CH | Pressure | DIL24 |
| LPS22DF | Pressure | IKS4A1 |
| LPS22HB | Pressure | DIL24 |
| LPS22HH | Pressure | DIL24 |
| LPS27HHTW | Pressure, Temperature | DIL24 |
| LPS28DFW | Pressure | DIL24 |
| LPS33HW | Pressure | DIL24 |
| LPS33K | Pressure | DIL24 |
| LSM303AGR | Acceleration, Magnetometer | DIL24 |
| LSM6DSL | Acceleration, Angular rate | DIL24 |
| LSM6DSO | Acceleration, Angular rate | DIL24 |
| LSM6DSO16IS | Acceleration, Angular rate, ISPU | IKS4A1, DIL24 |
| LSM6DSO32 | Acceleration, Angular rate | DIL24 |

| Device name | Device type | Connection |
|---|---|---|
| LSM6DSO32X | Acceleration, Angular rate, FSM, MLC | DIL24 |
| LSM6DSOX | Acceleration, Angular rate | DIL24 |
| LSM6DSR | Acceleration, Angular rate | DIL24 |
| LSM6DSRX | Acceleration, Angular rate, FSM, MLC | DIL24 |
| LSM6DSV | Acceleration, Angular rate | DIL24 |
| LSM6DSV16B | Acceleration, Angular rate | DIL24 |
| LSM6DSV16BX | Acceleration, Angular rate, FSM, MLC | DIL24 |
| LSM6DSV16X | Acceleration, Angular rate, FSM, MLC | IKS4A1, DIL24 |
| LSM6DSV32X | Acceleration, Angular rate, FSM, MLC | DIL24 |
| LSM6DSV80X | Acceleration, QVAR, FSM, MLC | DIL24 |
| LSM6DSV320X | Acceleration (high-g/low-g), Angular rate, FSM, MLC | DIL24 |
| SGP40 | Gas | DIL24 |
| SHT40AD1B | Temperature, Humidity | IKS4A1, DIL24 |
| ST1VAFE3BX | Acceleration, Biopotential, FSM, MLC | DIL24 |
| ST1VAFE6AX | Acceleration, Angular rate, Biopotential, FSM, MLC | DIL24 |
| STHS34PF80 | IR motion, presence | DIL24 |
| STTS22H | Temperature | IKS4A1, DIL24 |
| STTS751 | Temperature | DIL24 |

# 3 System setup

## 3.1 Integrated Development Environment

If you wish to create custom projects or you need to modify the existing examples or applications, you will need one of the following toolchains for development.

Note: *If you just want to try the sample projects that are provided in the pack, you won't need to install an IDE, as long as your hardware configuration is available. Refer to for more information.*

- IAR Embedded Workbench for ARM® (EWARM)
- Keil uVision (MDK-ARM)
- STM32Cube IDE

## 3.2 MEMS-Studio

MEMS-Studio is a graphical tool which facilitates implementation of proof of concept projects without writing code for STM32 microcontrollers, supporting Windows, Mac and Linux operating systems. This solution allows you to configure sensors and embedded AI (machine learning and neural networks), leveraging machine learning core (MLC), neural networks for the ISPU, and finite state machines (FSM). It combines multiple embedded software libraries, into a single project. The resulting project can establish two-way communication between the application and the connected board. The data sent by the connected board can be visualized in real time using plot and display.

In the context of X-CUBE-MEMS1 pack MEMS-Studio can retrieve data collected by Section 5.3.1: DataLogExtended example or any of Section 7: Library Evaluation Applications running on the connected board. The data can be displayed in graphs, tables and visualized or processed in other ways. Basic functionality of the tool is outlined with concerned applications. You can obtain much more detailed information on how to use MEMS-Studio in the UM3233 Getting started with MEMS Studio.

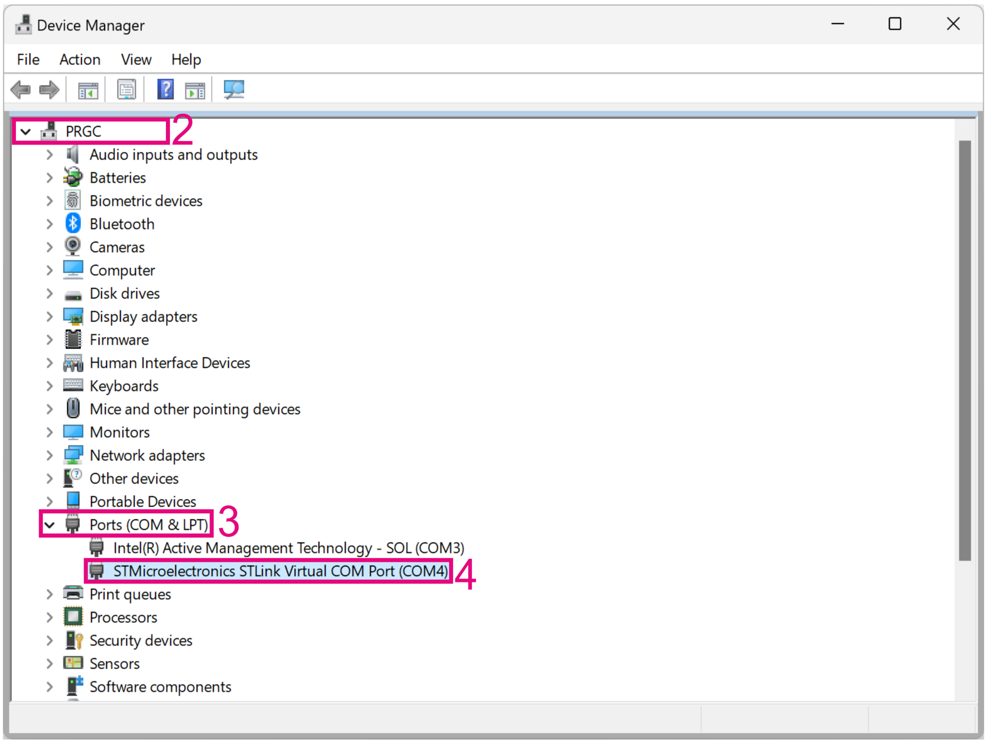
## 3.3 Determining your board's COM port

Most sensor evaluation examples and library evaluation applications utilize U(S)ART communication. You can read more on this topic in Section 4.8: U(S)ART. You can follow the steps below in order to learn which COM port the connected device occupies in your Windows PC by following the steps below.

### Finding your board's COM port

1. Open Windows Device Manager
2. Click the icon with your computer's name to show all device groups
3. Go to `Ports (COM & LPT)` section and expand it to see all `COM` ports
4. Find the item that best describes your device – in the end of the string, you'll find the name in parentheses, which normally looks something like this: `COMx`, where `x` is an index which will be specific to your computer. Remember the name to later use it in MEMS-Studio or serial hyper terminal.

Figure 8. **Windows Device Manager**



# 4 STM32 configuration

This chapter describes individual configuration steps you need to take in order to configure all required MCU peripherals to successfully set up projects that utilize X-CUBE-MEMS1. Each step is described in general terms, its utility is briefly explained, and then the configuration is shown in practice. The steps described are carried out with the help of STM32CubeMX. The hardware used for the demonstration is NUCLEO-F446RE + X-NUCLEO-IKS4A1 and you'll likely need to adjust the settings for the exact peripherals your hardware supports.

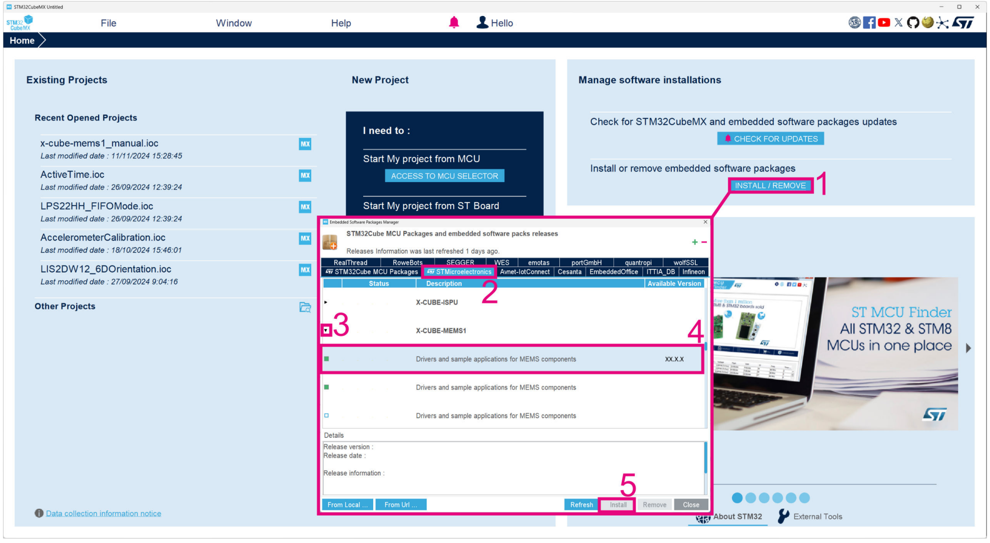
## 4.1 Installing X-CUBE-MEMS1 into STM32CubeMX

The X-CUBE-MEMS1 pack is packaged so that it can be easily installed into STM32CubeMX. The tool can be used directly to manage the download and installation of the pack and which is going to be shown here.

**Installation steps**

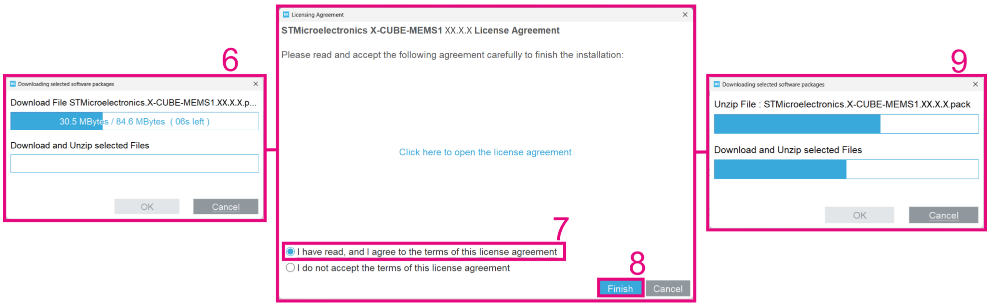
1. Go to the welcome page of STM32CubeMX or open a new instance of the software and click the `INSTALL/REMOVE` button which will open a new window
2. In the new window, switch to the `STMicroelectronics` tab
3. Locate the X-CUBE-MEMS1 and expand it by clicking the triangle on the left – choose the version you want to install and select it by clicking its corresponding checkbox
4. Finally click the `Install` button

Figure 9. **Pack installation into STM32CubeMX**


File
Window
Help
Hello
Home
Existing Projects
Recent Opened Projects
x-cube-mems1_manual.ioc
Last modified date : 11/11/2024 15:28:45
ActiveTime.ioc
Last modified date : 26/09/2024 12:39:24
LPS22HH_FIFOMode.ioc
Last modified date : 26/09/2024 12:39:24
AccelerometerCalibration.ioc
Last modified date : 18/10/2024 15:46:01
LIS2DW12_6DOrientation.ioc
Last modified date : 27/09/2024 9:04:16
Other Projects
Data collection information notice
New Project
I need to :
Start My project from MCU
ACCESS TO MCU SELECTOR
Start My project from ST Board
Manage software installations
Check for STM32CubeMX and embedded software packages updates
CHECK FOR UPDATES
Install or remove embedded software packages
INSTALL / REMOVE
1
Embedded Software Packages Manager
STM32Cube MCU Packages and embedded software packs releases
Releases Information was last refreshed 1 days ago.
RealThread
RoweBots
SEGGER
WES
emotas
portGmbH
quantropi
wolfSSL
STM32Cube MCU Packages
STMicroelectronics
Avnet-IotConnect
Cesanta
EmbeddedOffice
ITTIA_DB
Infineon
2
Status
Description
Available Version
X-CUBE-ISPU
X-CUBE-MEMS1
3
4
Drivers and sample applications for MEMS components
XX.X.X
Drivers and sample applications for MEMS components
Drivers and sample applications for MEMS components
Details
Release version :
Release date :
Release information :
5
From Local ...
From Url ...
Refresh
Install
Remove
Close
ST MCU Finder
All STM32 & STM8 MCUs in one place
About STM32
External Tools

1. Following the completion of the previous steps, new window is going to appear, showing progress of the download – after the pack is downloaded the window will automatically close and `License Agreement window` will appear – read the license agreement and if you agree, select `I have read, and I agree to the terms of this license agreement`
2. Click the `Finish` button – following this action, pack installation progress window is going to appear which will close automatically, after the installation is completed

Figure 10. **Installation progress**


6
Downloading selected software packages
Download File STMicroelectronics.X-CUBE-MEMS1.XX.X.X.p...
30.5 MBytes / 84.6 MBytes ( 06s left )
Download and Unzip selected Files
OK
Cancel
Licensing Agreement
STMicroelectronics X-CUBE-MEMS1 XX.X.X License Agreement
Please read and accept the following agreement carefully to finish the installation:
Click here to open the license agreement
7
● I have read, and I agree to the terms of this license agreement
○ I do not accept the terms of this license agreement
8
Finish
Cancel
9
Downloading selected software packages
Unzip File : STMicroelectronics.X-CUBE-MEMS1.XX.X.X.pack
Download and Unzip selected Files
OK
Cancel

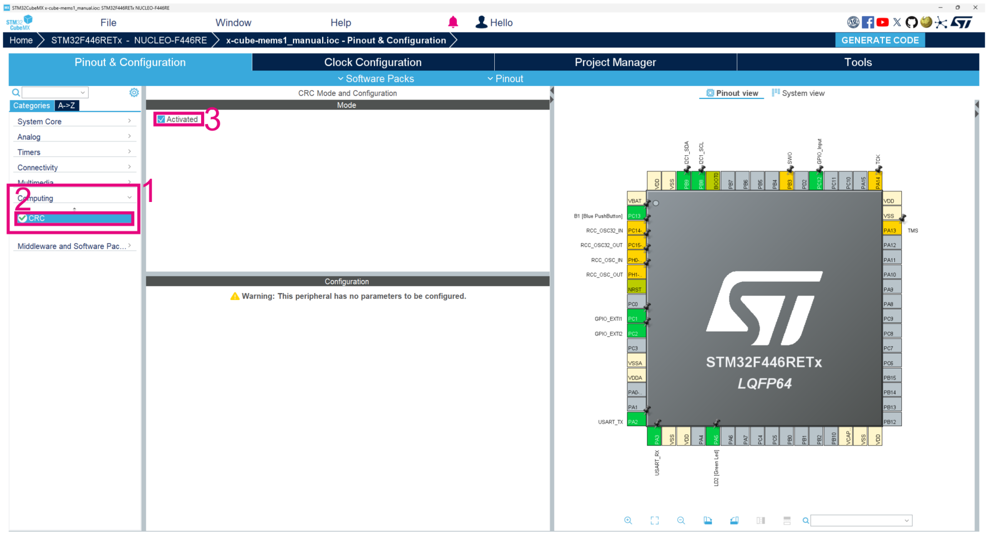
## 4.2 CRC

Projects which interface with MEMS-Studio application need to use CRC to ensure reliable communication.

### Configuring CRC with STM32CubeMX

1. Navigate to `Computing` under `Categories`
2. Click the `CRC` peripheral
3. Select `Activated`, under `CRC Mode and Configuration` > `Mode`

**Figure 11. CRC configuration**



☑ Activated

Warning: This peripheral has no parameters to be configured.

## 4.3 DMA

Projects which interface with the MEMS-Studio software use the DMA peripheral to receive data over U(S)ART and store it in memory. This offloads some of the work from the processor's core to the peripheral, saving some processing capacity of the MCU.

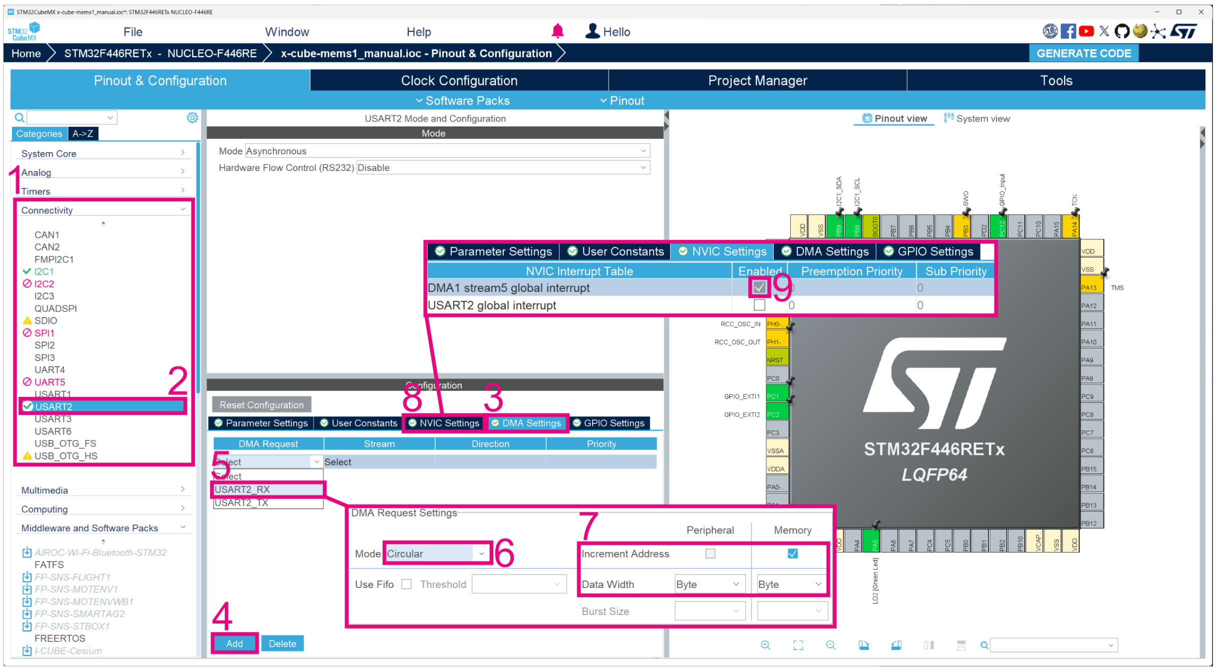
Some STM32 families (such as those with Cortex-M33) support `Linked-List` mode of DMA operation – these peripherals are more complex and generally follow a different configuration process. If this concerns your device, you may benefit from the Section 4.3.2: Configuring Linked-List capable DMA with STM32CubeMX, otherwise you can continue reading Section 4.3.1: Configuring DMA with STM32CubeMX.

### 4.3.1 Configuring DMA with STM32CubeMX

*Note:* *Make sure that the `U(S)ART` peripheral is already configured, before proceeding with configuration of DMA.*

1. Navigate to `Connectivity` under `Categories`
2. Select the `U(S)ARTx` interface which is configured for communication
3. Select `DMA Settings` in the `Configuration` section
4. Click the `Add` button to add a new request
5. Click the `DMA Request` field to select `U(S)ARTx_RX`
6. Set the `Mode` to `Circular` in the `DMA Request Settings` section which will appear below the request
7. Make sure that the `Increment Address` is set in the `Memory` side and `Data Width` is set to `Byte` for both sides
8. Switch to `NVIC Settings` tab
9. Make sure that the DMA stream's global interrupt is enabled

**Figure 12. USART DMA configuration**



Mode Asynchronous

Hardware Flow Control (RS232) Disable

| NVIC Interrupt Table | Enabled | Preemption Priority | Sub Priority |
|---|---|---|---|
| DMA1 stream5 global interrupt | ☑ | 0 | 0 |
| USART2 global interrupt | □ | 0 | 0 |

| DMA Request | Stream | Direction | Priority |
|---|---|---|---|
| Select | Select | | |

DMA Request: Select, USART2_RX, USART2_TX

DMA Request Settings

Mode Circular

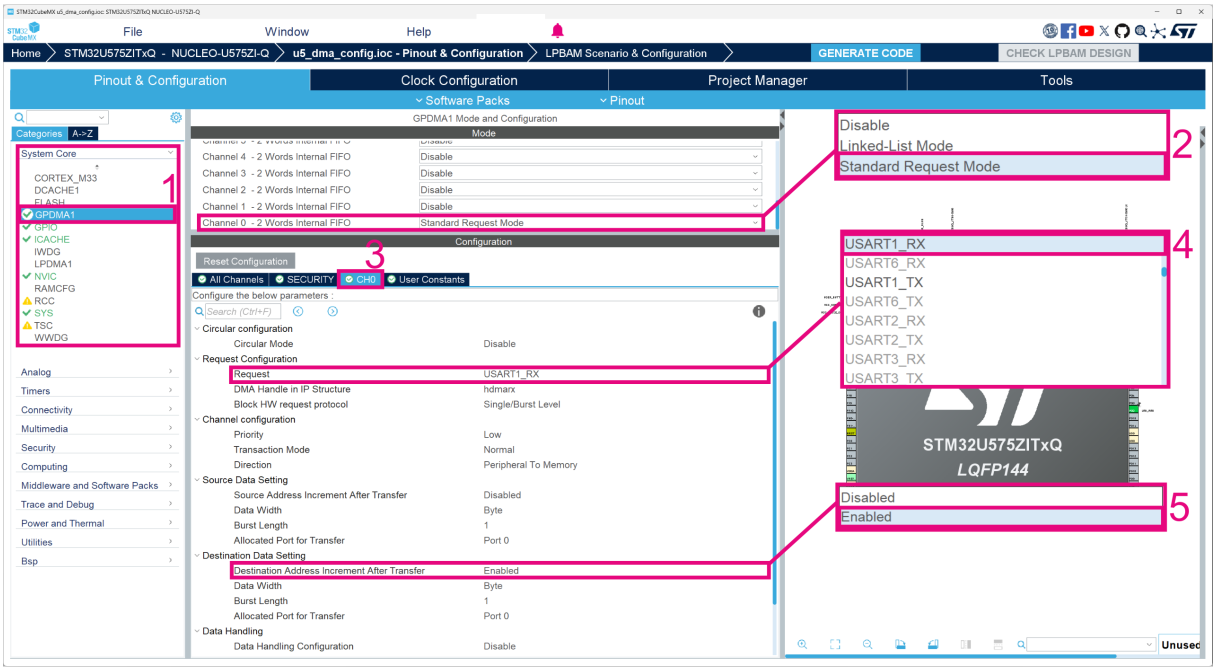
Use Fifo □ Threshold

| | Peripheral | Memory |
|---|---|---|
| Increment Address | □ | ☑ |
| Data Width | Byte | Byte |
| Burst Size | | |

### 4.3.2 Configuring Linked-List capable DMA with STM32CubeMX

*Note: Make sure that the `U(S)ART` peripheral is already configured, before proceeding with configuration of DMA.*

1. Under `Categories`, navigate to the DMA you want to use
2. Select a channel and set its mode to `Standard Request Mode`
3. Under `Configuration`, go to your channel's tab
4. Under the `Request Configuration` section, set the `Request` to `USARTx_RX` which corresponds to the peripheral as used in your application
5. Under `Destination Data Setting`, set the `Destination Address Increment After Transfer` to `Enabled`



GPDMA1 Mode and Configuration

Mode

Channel 4 - 2 Words Internal FIFO: Disable
Channel 3 - 2 Words Internal FIFO: Disable
Channel 2 - 2 Words Internal FIFO: Disable
Channel 1 - 2 Words Internal FIFO: Disable
Channel 0 - 2 Words Internal FIFO: Standard Request Mode

Configuration

Configure the below parameters :

| | |
|---|---|
| Circular configuration | |
| Circular Mode | Disable |
| Request Configuration | |
| Request | USART1_RX |
| DMA Handle in IP Structure | hdmarx |
| Block HW request protocol | Single/Burst Level |
| Channel configuration | |
| Priority | Low |
| Transaction Mode | Normal |
| Direction | Peripheral To Memory |
| Source Data Setting | |
| Source Address Increment After Transfer | Disabled |
| Data Width | Byte |
| Burst Length | 1 |
| Allocated Port for Transfer | Port 0 |
| Destination Data Setting | |
| Destination Address Increment After Transfer | Enabled |
| Data Width | Byte |
| Burst Length | 1 |
| Allocated Port for Transfer | Port 0 |
| Data Handling | |
| Data Handling Configuration | Disable |

Figure 13. **Linked-List capable DMA configuration**

## 4.4 GPIO

GPIOs are used to perform various functions in the provided Applications and Examples – they are the interface with sensors and the connected PC, they drive LEDs and detect button presses.

### Configuring GPIO with STM32CubeMX

1. Go to to the `Pinout view`. Use the search field and type full name of the pin whose function you want to set – the pin in the package view will start blinking
2. Click the blinking pin and a list of available functions is going to appear. Select the required function
3. Repeat for every pin in question

If you are not sure about the pin functions to use, then the To obtain the keyword you are looking for `Pin Function` column find its associated `STM32CubeMX Name`. Replace the `x` in the `STM32CubeMX Name` column by the actual instance of the peripheral. in use and that is the pin function keyword. You can see the Example column as examples of actual keywords. STM32CubeMX will automatically set a suitable configuration for the pin in regard to the peripheral. Repeat the previous steps for each pin you want to configure.

Figure 14. Pinout view

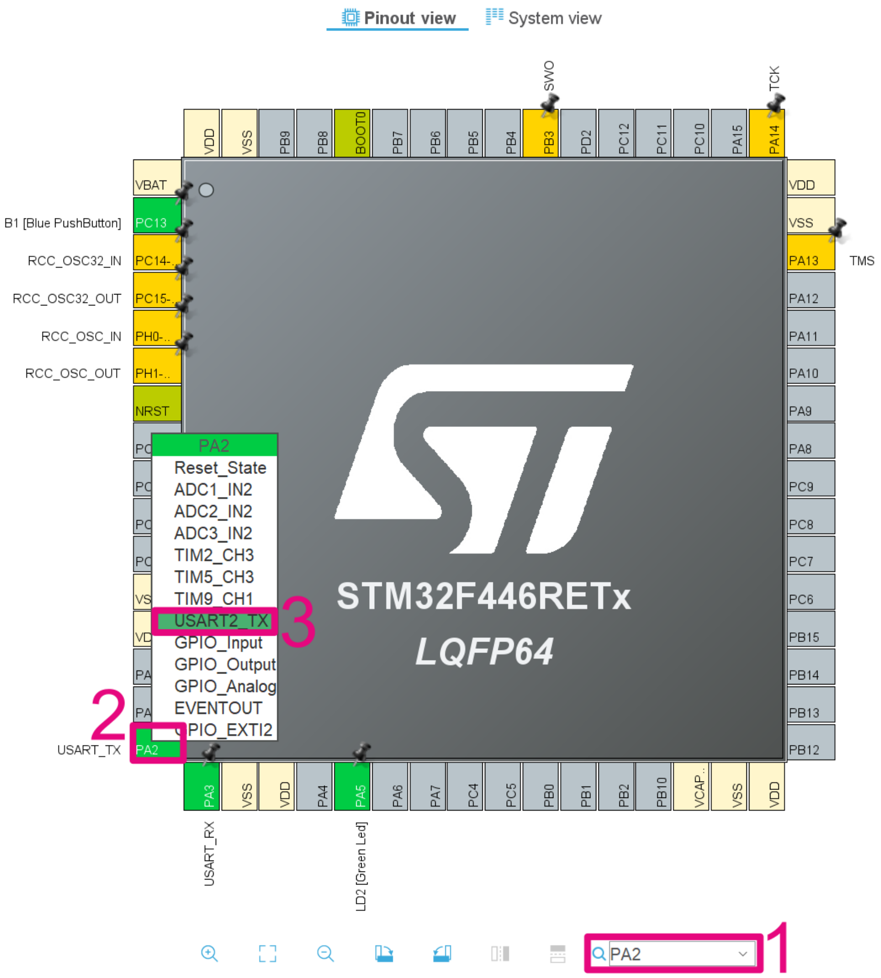


Table 2. Pin features

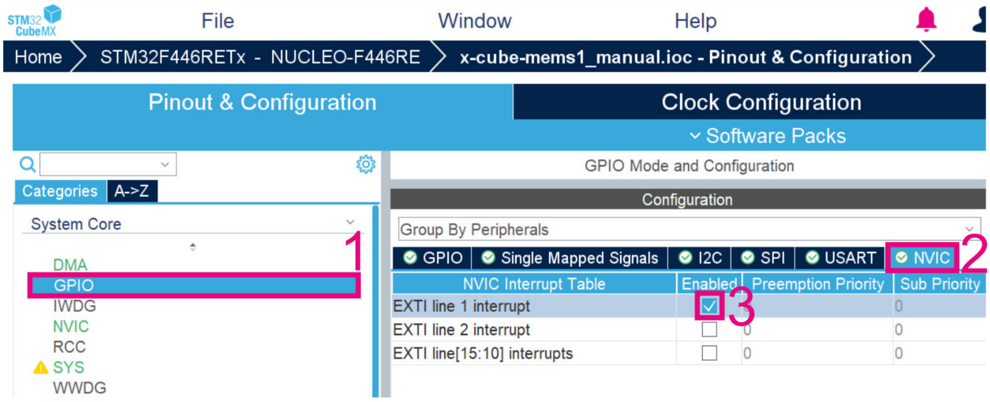
| Pin Function | STM32CubeMX Name | Example |
|---|---|---|
| U(S)ART Transmit [(1)] | `UARTx_TX` | `UART2_TX` |
| UART Receive | `UARTx_RX` | `UART2_RX` |
| I2C Clock pin | `I2Cx_SCL` | `I2C3_SCL` |
| I2C Data pin | `I2Cx_SDA` | `I2C3_SDA` |
| GPIO Input | `GPIO_Input` | `GPIO_Input` |
| GPIO Output | `GPIO_Output` | `GPIO_Output` |
| GPIO External Interrupt | `GPIO_EXTIx` | `GPIO_EXTI13` |
| SPI Clock Pin | `SPIx_SCK` | `SPI1_SCK` |
| SPI Master-in, Slave-out Pin | `SPIx_MISO` | `SPI1_MISO` |
| SPI Master-out, Slave-in Pin | `SPIx_MOSI` | `SPI1_MOSI` |

1. *To use `USART` or `LPUART` – replace `UART` with `USART` or `LPUART` respectively – e.g.: `USART3_TX`, `LPUART2_TX`*

Enabling GPIO interrupt in NVIC

If you need to configure EXTI interrupt for GPIO, then follow the steps mentioned in this section. by following the steps above you configured your pin as `GPIO_EXTIx` where the `x` stands for the line number which will be used now.

**Figure 15. Enabling GPIO interrupt in NVIC**



| NVIC Interrupt Table | Enabled | Preemption Priority | Sub Priority |
|---|---|---|---|
| EXTI line 1 interrupt | ☑ |  | 0 |
| EXTI line 2 interrupt | ☐ | 0 | 0 |
| EXTI line[15:10] interrupts | ☐ | 0 | 0 |

1. *Go to `GPIO` in the `System Core` category*
2. *Switch to the `NVIC` tab, under `Configuration` section*
3. *Enable the interrupt line going from the pin in question*

## 4.5 I2C / SPI

The communication between an STM32 MCU and a MEMS device is achieved over I2C or SPI – refer to your particular sensor to see which is supported. Based on your hardware decide which one is more suitable for your application. In general I2C bus requires fewer GPIO and lets you connect multiple devices without need for additional GPIO. SPI on the other hand, needs more GPIO but supports higher transfer speeds.

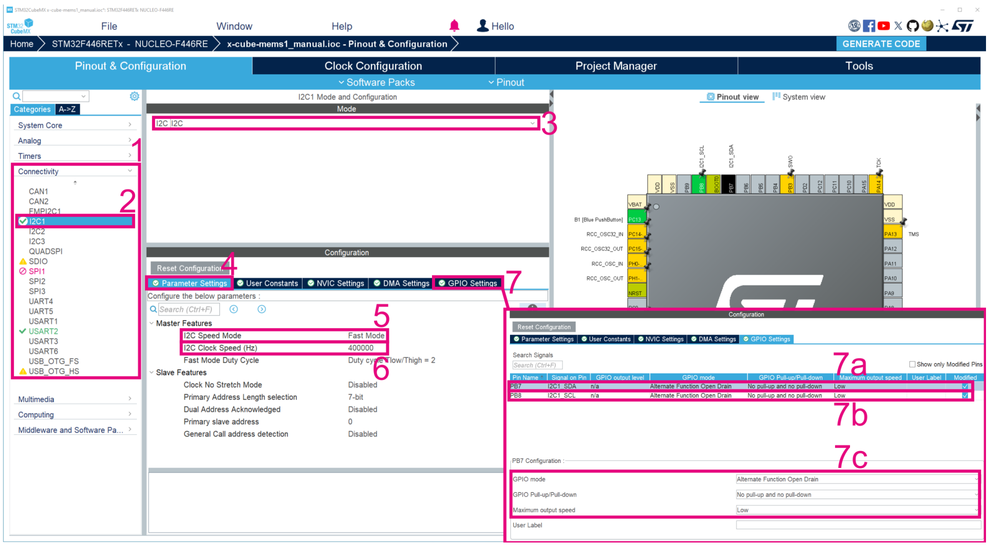
Once you've chosen between I2C and SPI, you can proceed to its corresponding configuration section.

### 4.5.1 I2C

Before you start with the I2C configuration, you need to know what MCU pins the interface connects to and which peripheral is available there. If you're using any of the supported X-NUCLEO boards then the SCL and SDA pins are connected to `D15` and `D14` respective pins of the Arduino connector.

### Configuring I2C with STM32CubeMX

1. Navigate to the `Connectivity` category under `Categories`
2. Click the `I2C` peripheral that you are going to use
3. In the `Mode` section, set `I2C` to `I2C`
4. Switch to `Parameter Settings` tab, under the `Configuration` section
5. Set the `I2C Speed Mode` to `FastMode`category, under `Master features` section
6. Set the `I2C Clock Speed (HZ)` to `400000`, under `Master features` section
7. Switch to `GPIO Settings` under `Configuration` category. Check that the correct pins are used for both `I2Cx_SDA` (7a) and `I2Cx_SCL` (7b) and their respective parameters (7c) are compatible with the hardware setup.. In the sample case – which uses the NUCLEO-F446RE, different pins were configured by default. If that happens, reconfigure the individual pins in `Pinout view` (see the Configuring GPIO with STM32CubeMX section)



I2C1 Mode and Configuration

Mode

I2C I2C

Configuration

Reset Configuration

Parameter Settings | User Constants | NVIC Settings | DMA Settings | GPIO Settings

Configure the below parameters :

Master Features

| | |
|---|---|
| I2C Speed Mode | Fast Mode |
| I2C Clock Speed (Hz) | 400000 |
| Fast Mode Duty Cycle | Duty cycle Tlow/Thigh = 2 |

Slave Features

| | |
|---|---|
| Clock No Stretch Mode | Disabled |
| Primary Address Length selection | 7-bit |
| Dual Address Acknowledged | Disabled |
| Primary slave address | 0 |
| General Call address detection | Disabled |

Configuration

Reset Configuration

Parameter Settings | User Constants | NVIC Settings | DMA Settings | GPIO Settings

Search Signals

□ Show only Modified Pins

| Pin Name | Signal on Pin | GPIO output level | GPIO mode | GPIO Pull-up/Pull-down | Maximum output speed | User Label | Modified |
|---|---|---|---|---|---|---|---|
| PB7 | I2C1_SDA | n/a | Alternate Function Open Drain | No pull-up and no pull-down | Low | | ☑ |
| PB8 | I2C1_SCL | n/a | Alternate Function Open Drain | No pull-up and no pull-down | Low | | ☑ |

PB7 Configuration :

| | |
|---|---|
| GPIO mode | Alternate Function Open Drain |
| GPIO Pull-up/Pull-down | No pull-up and no pull-down |
| Maximum output speed | Low |
| User Label | |

Figure 16. I2C configuration

## 4.5.2 SPI

To configure the SPI peripheral, you need to know which MCU pins the sensors' SPI interface is connected to. You can refer to Figure 4. X-NUCLEO-IKS02A1, X-NUCLEO-IKS4A1 and X-NUCLEO-IKS5A1 – pinouts to determine which pins of the Arduino connector of an X-NUCLEO board are used. Keep in mind that if you're using SPI to interface with multiple sensors, you are going to need to configure one GPIO output per sensor to act as the CS signal.

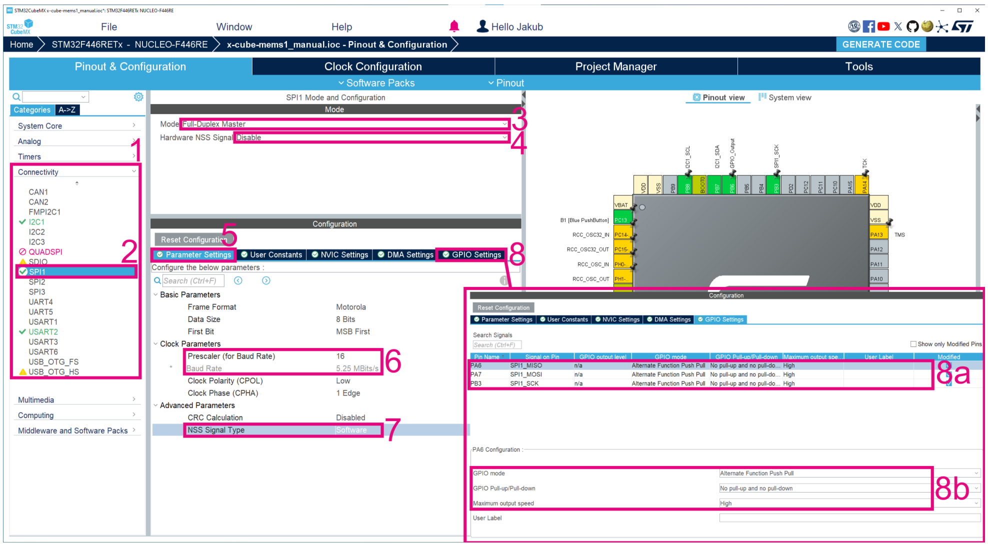
### Configuring SPI with STM32CubeMX

1. Navigate to the `Connectivity`, under `Categories`
2. Click the `SPI` peripheral you are going to use
3. Set `Mode` to `Full-Duplex Master` in the `SPIx Mode and Configuration` section
4. Set `Hardware NSS Signal` to `Disable`
5. Switch to the`Parameter Settings` tab in the `Configuration` section
6. Select the `Prescaler (for Baud Rate)` value

*Note: Make sure that automatically calculated `Baud Rate` value does not exceed 10 MHz – some sensors might not support speeds higher than that. Consult documentation for your device if you need Baud Rate higher than that.*

7. Set the `NSS Signal Type` to `Software` – meaning that the firmware will have to take care of driving the CS pin
8. Stay in the `Configuration` category but switch to `GPIO Settings` and check that the correct pins are used for all `SPIx_MISO`, `SPIx_MOSI` and `SPIx_SCK`. In the sample case – which uses the `NUCLEO-F446RE`, wrong pins were configured by default. If that happens, reconfigure the individual pins in `Pinout view` (see the Configuring GPIO with STM32CubeMX section)

**Figure 17. SPI configuration**



SPI1 Mode and Configuration

Mode

Mode: Full-Duplex Master

Hardware NSS Signal: Disable

Configuration

Reset Configuration

Parameter Settings | User Constants | NVIC Settings | DMA Settings | GPIO Settings

Configure the below parameters :

Basic Parameters

| | |
|---|---|
| Frame Format | Motorola |
| Data Size | 8 Bits |
| First Bit | MSB First |

Clock Parameters

| | |
|---|---|
| Prescaler (for Baud Rate) | 16 |
| Baud Rate | 5.25 MBits/s |
| Clock Polarity (CPOL) | Low |
| Clock Phase (CPHA) | 1 Edge |

Advanced Parameters

| | |
|---|---|
| CRC Calculation | Disabled |
| NSS Signal Type | Software |

Configuration

Reset Configuration

Parameter Settings | User Constants | NVIC Settings | DMA Settings | GPIO Settings

Search Signals

□ Show only Modified Pins

| Pin Name | Signal on Pin | GPIO output level | GPIO mode | GPIO Pull-up/Pull-down | Maximum output spe... | User Label | Modified |
|---|---|---|---|---|---|---|---|
| PA6 | SPI1_MISO | n/a | Alternate Function Push Pull | No pull-up and no pull-do... | High | | ☑ |
| PA7 | SPI1_MOSI | n/a | Alternate Function Push Pull | No pull-up and no pull-do... | High | | ☑ |
| PB3 | SPI1_SCK | n/a | Alternate Function Push Pull | No pull-up and no pull-do... | High | | ☑ |

PA6 Configuration :

| | |
|---|---|
| GPIO mode | Alternate Function Push Pull |
| GPIO Pull-up/Pull-down | No pull-up and no pull-down |
| Maximum output speed | High |
| User Label | |

*Note: Make sure to configure one CS output pin for each device that is going to use SPI – see the Configuring GPIO with STM32CubeMX. Make sure that you configure its initial output level to HIGH, meaning that the device is not selected initially.*

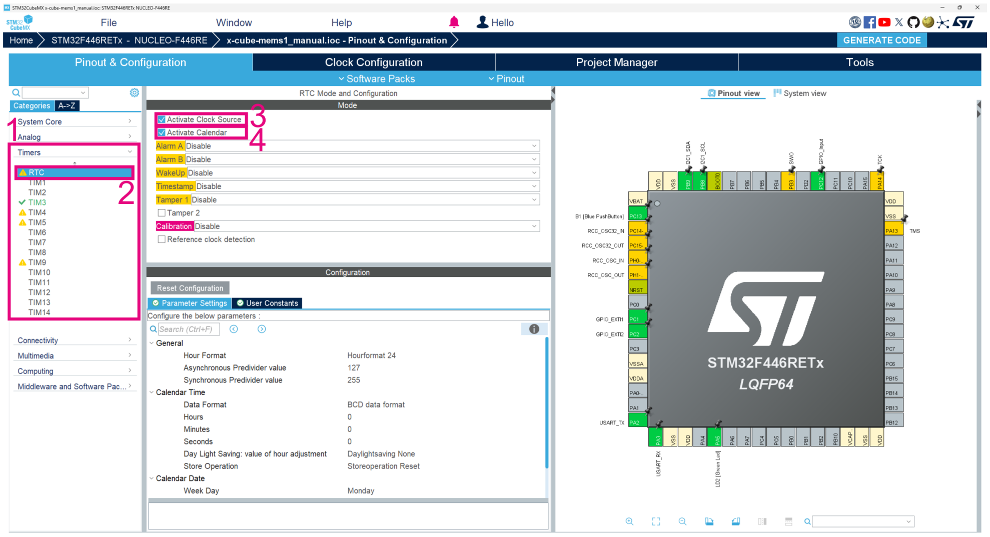
## 4.6 RTC

Projects which interface with MEMS-Studio application use RTC to keep track of time in the application. It enables time synchronization of generated sensor data with the PC application.

### Configuring RTC with STM32CubeMX

1. Navigate to `Timers` under `Categories`
2. Click the `RTC` peripheral
3. Select the `Activate Clock Source` checkbox under `RTC Mode and Configuration` > `Mode`
4. Select the `Activate Calendar` checkbox under `RTC Mode and Configuration` > `Mode`

**Figure 18. RTC configuration**



Mode

☑ Activate Clock Source
☑ Activate Calendar
Alarm A Disable
Alarm B Disable
WakeUp Disable
Timestamp Disable
Tamper 1 Disable
□ Tamper 2
Calibration Disable
□ Reference clock detection

Configure the below parameters :

- General
  - Hour Format: Hourformat 24
  - Asynchronous Predivider value: 127
  - Synchronous Predivider value: 255
- Calendar Time
  - Data Format: BCD data format
  - Hours: 0
  - Minutes: 0
  - Seconds: 0
  - Day Light Saving: value of hour adjustment: Daylightsaving None
  - Store Operation: Storeoperation Reset
- Calendar Date
  - Week Day: Monday

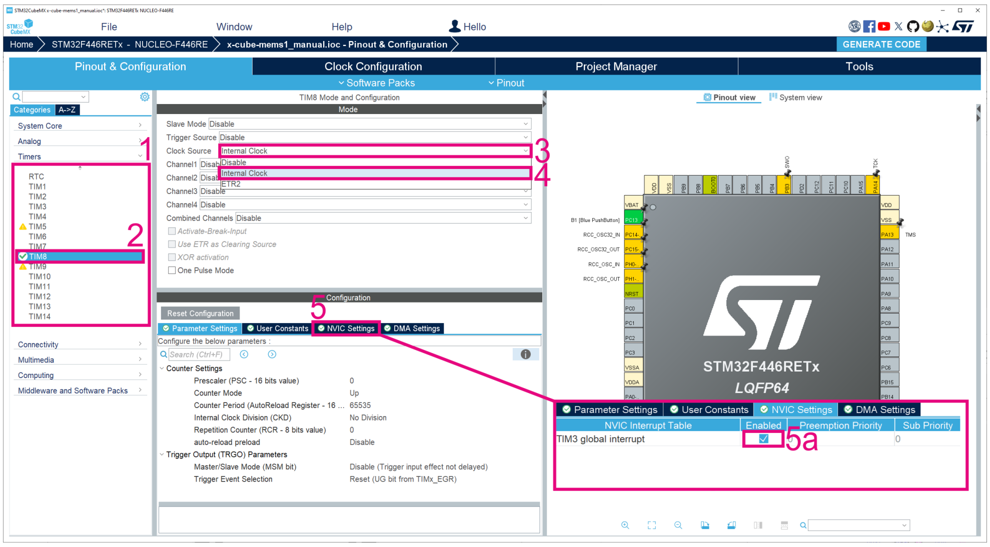
## 4.7 TIMER

Projects which interface with MEMS-Studio application use a TIMER to keep track of time in the application. It enables time synchronization of generated sensor data with the PC application.

### Configuring TIMER with STM32CubeMX

1. Navigate to `Timers` under `Categories`
2. Click the `TIMx` peripheral which you want to use
3. In the `RTC Mode and Configuration` section, under `Mode`, click the `Clock Source` selection box
4. Select the `Internal Clock`
5. Switch to the `NVIC Settings` tab and enable the `TIMx global interrupt` (5a)

**Figure 19.** **TIM configuration**



TIM8 Mode and Configuration

Mode

Slave Mode: Disable
Trigger Source: Disable
Clock Source: Internal Clock (Disable / Internal Clock / ETR2)
Channel1: Disable
Channel2: Disable
Channel3: Disable
Channel4: Disable
Combined Channels: Disable
□ Activate-Break-Input
□ Use ETR as Clearing Source
□ XOR activation
□ One Pulse Mode

Configuration

Configure the below parameters :

Counter Settings

| | |
|---|---|
| Prescaler (PSC - 16 bits value) | 0 |
| Counter Mode | Up |
| Counter Period (AutoReload Register - 16 ... | 65535 |
| Internal Clock Division (CKD) | No Division |
| Repetition Counter (RCR - 8 bits value) | 0 |
| auto-reload preload | Disable |

Trigger Output (TRGO) Parameters

| | |
|---|---|
| Master/Slave Mode (MSM bit) | Disable (Trigger input effect not delayed) |
| Trigger Event Selection | Reset (UG bit from TIMx_EGR) |

| NVIC Interrupt Table | Enabled | Preemption Priority | Sub Priority |
|---|---|---|---|
| TIM3 global interrupt | ☑ | 0 | 0 |

## 4.8 U(S)ART

U(S)ART interface, in most of the X-CUBE-MEMS1 examples and applications, is used to establish communication between the STM32 MCU and the host computer. Most modern computers, however, don't support that type of communication directly. STM32Nucleo boards contain on-board STLINK which acts as a virtual COM port bridge that bridges MCU's U(S)ART peripheral to USB. Only after that bridging can the computer access the data being sent.

Before you proceed with configuration, make sure you know what MCU's U(S)ART peripheral and pins have connection to the VCOM bridge. If you don't know, you can obtain that information from your STM32Nucleo's User Manual.

If you're using board other than STM32Nucleo, you will need to provide your own U(S)ART to USB bridge. This can be achieved by connecting an external STLINK, among other ways.

### Configuring U(S)ART in STM32CubeMX

With the knowledge of what pins and peripherals are to be used for U(S)ART communication, you can proceed with the following steps:

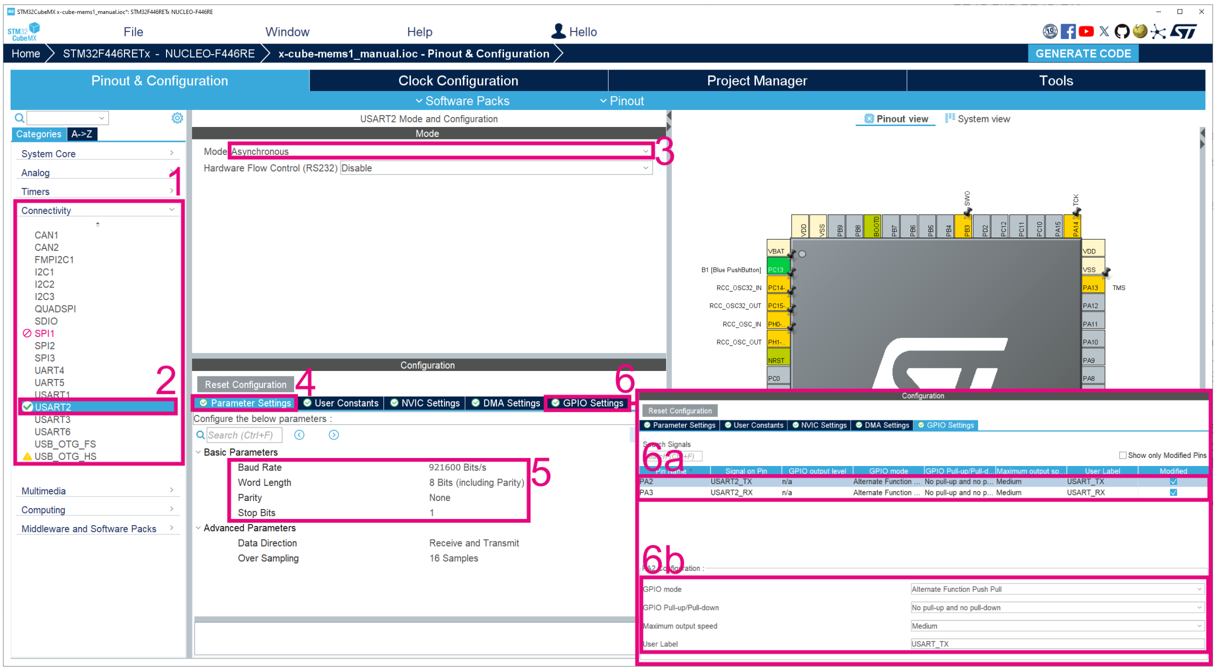
1. Navigate to `Connectivity` under `Categories`
2. Click the `U(S)ARTx` peripheral which you are going to use
3. Set `Mode` to `Asynchronous` in the `Mode` section
4. Switch to `Parameter Settings` tab in the `Configuration` section
5. Set U(S)ART parameters

*Note: If the project you are configuring establishes communication with MEMS-Studio, make sure to use the following U(S)ART settings:*

- *Baud Rate: 921 600*
- *Word Length: 8 Bits (including parity)*
- *Parity: None*
- *Stop Bits: 1*

6. Switch to `GPIO Settings` tab and check if the receive and transmit pins correspond to the ones you want to use (6a) and that the pin configuration is suitable for your communication speeds (6b)
7. If the previous step is not satisfied – the pins configured don't correspond with the ones you want to use – then follow the instructions in Section 4.4: GPIO to set the suitable `USARTx_RX` and `USARTx_TX` pins.

**Figure 20. U(S)ART configuration**



Mode: Asynchronous
Hardware Flow Control (RS232): Disable

Basic Parameters

| | |
|---|---|
| Baud Rate | 921600 Bits/s |
| Word Length | 8 Bits (including Parity) |
| Parity | None |
| Stop Bits | 1 |

Advanced Parameters

| | |
|---|---|
| Data Direction | Receive and Transmit |
| Over Sampling | 16 Samples |

Search Signals

□ Show only Modified Pins

| Pin Name | Signal on Pin | GPIO output level | GPIO mode | GPIO Pull-up/Pull-d... | Maximum output sp... | User Label | Modified |
|---|---|---|---|---|---|---|---|
| PA2 | USART2_TX | n/a | Alternate Function ... | No pull-up and no p... | Medium | USART_TX | ☑ |
| PA3 | USART2_RX | n/a | Alternate Function ... | No pull-up and no p... | Medium | USART_RX | ☑ |

PA2 Configuration :

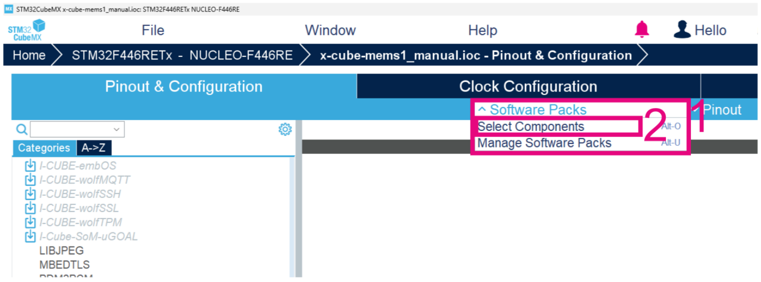
| | |
|---|---|
| GPIO mode | Alternate Function Push Pull |
| GPIO Pull-up/Pull-down | No pull-up and no pull-down |
| Maximum output speed | Medium |
| User Label | USART_TX |

## 4.9 Selecting components for configurations utilizing X-NUCLEO boards

This section assumes that your project is using a supported X-NUCLEO board. Note that you'll need to configuration for custom board in case you want to connect a sensor in the DIL24 connector.

Component selector is a tool that lets you select hardware for a given project and an application to be generated. That information is then used to include drivers for the selected components and to generate the application accordingly.

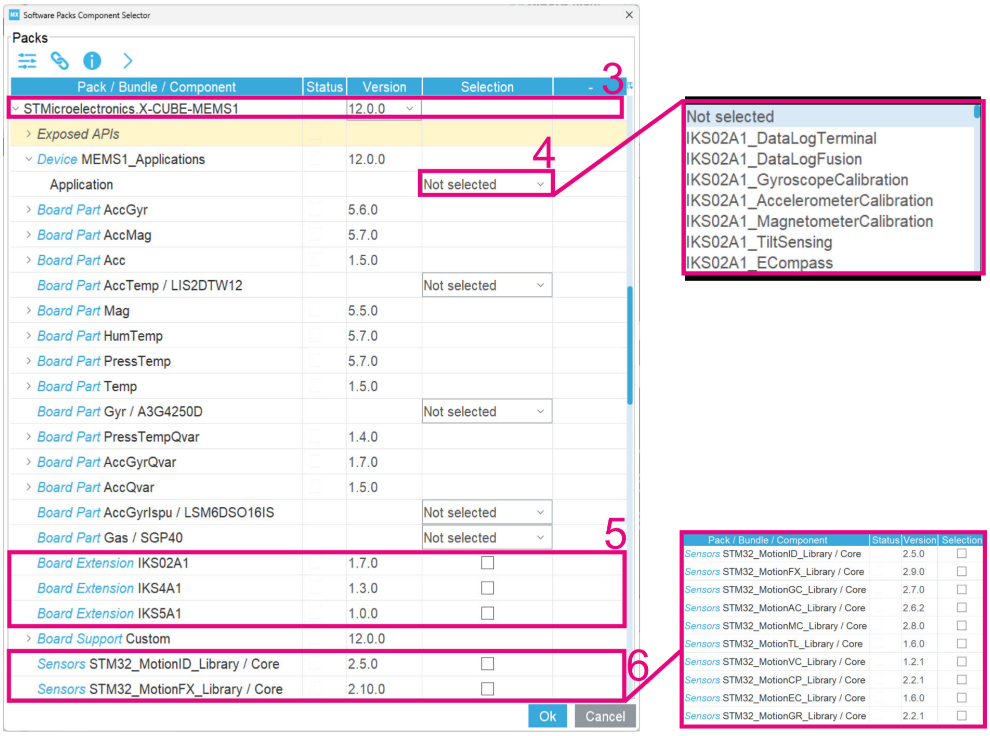
### Figure 21. Switching to component selector



**Step 1.** Expand the `Software Packs` drop-down menu

**Step 2.** Click the `Select Components` which will open a new window

### Figure 22. Selecting components



| Pack / Bundle / Component | Status | Version | Selection |
|---|---|---|---|
| STMicroelectronics.X-CUBE-MEMS1 | | 12.0.0 | |
| *Exposed APIs* | | | |
| *Device* MEMS1_Applications | | 12.0.0 | |
| Application | | | Not selected |
| *Board Part* AccGyr | | 5.6.0 | |
| *Board Part* AccMag | | 5.7.0 | |
| *Board Part* Acc | | 1.5.0 | |
| *Board Part* AccTemp / LIS2DTW12 | | | Not selected |
| *Board Part* Mag | | 5.5.0 | |
| *Board Part* HumTemp | | 5.7.0 | |
| *Board Part* PressTemp | | 5.7.0 | |
| *Board Part* Temp | | 1.5.0 | |
| *Board Part* Gyr / A3G4250D | | | Not selected |
| *Board Part* PressTempQvar | | 1.4.0 | |
| *Board Part* AccGyrQvar | | 1.7.0 | |
| *Board Part* AccQvar | | 1.5.0 | |
| *Board Part* AccGyrIspu / LSM6DSO16IS | | | Not selected |
| *Board Part* Gas / SGP40 | | | Not selected |
| *Board Extension* IKS02A1 | | 1.7.0 | □ |
| *Board Extension* IKS4A1 | | 1.3.0 | □ |
| *Board Extension* IKS5A1 | | 1.0.0 | □ |
| *Board Support* Custom | | 12.0.0 | |
| *Sensors* STM32_MotionID_Library / Core | | 2.5.0 | □ |
| *Sensors* STM32_MotionFX_Library / Core | | 2.10.0 | □ |

Not selected
IKS02A1_DataLogTerminal
IKS02A1_DataLogFusion
IKS02A1_GyroscopeCalibration
IKS02A1_AccelerometerCalibration
IKS02A1_MagnetometerCalibration
IKS02A1_TiltSensing
IKS02A1_ECompass

| Pack / Bundle / Component | Status | Version | Selection |
|---|---|---|---|
| *Sensors* STM32_MotionID_Library / Core | | 2.5.0 | □ |
| *Sensors* STM32_MotionFX_Library / Core | | 2.9.0 | □ |
| *Sensors* STM32_MotionGC_Library / Core | | 2.7.0 | □ |
| *Sensors* STM32_MotionAC_Library / Core | | 2.6.2 | □ |
| *Sensors* STM32_MotionMC_Library / Core | | 2.8.0 | □ |
| *Sensors* STM32_MotionTL_Library / Core | | 1.6.0 | □ |
| *Sensors* STM32_MotionVC_Library / Core | | 1.2.1 | □ |
| *Sensors* STM32_MotionCP_Library / Core | | 2.2.1 | □ |
| *Sensors* STM32_MotionEC_Library / Core | | 1.6.0 | □ |
| *Sensors* STM32_MotionGR_Library / Core | | 2.2.1 | □ |

**Step 3.** In the `Pack / Bundle / Component` column locate the `STMicroelectronic.X-CUBE-MEMS1` pack and expand it

**Step 4.** Expand the `Device MEMS1_Applications` and use the drop-down menu to select the Application / Example you are configuring. **Beware**, in this context, the term application is used as an umbrella term for either Library evaluation application or Sensor evaluation example. Also note that unique projects are preconfigured for each expansion board, meaning that board names are prefixed to the project names – for example the `ECompass` application exists as 3 distinct entries (`IKS02A1_ECompass`, `IKS4A1_ECompass` and `CUSTOM_ECompass`). Some projects may omit the prefix `CUSTOM_` altogether for custom boards.

**Step 5.** Select board support based on the expansion board you are using. This will automatically add support to all sensors on the board.

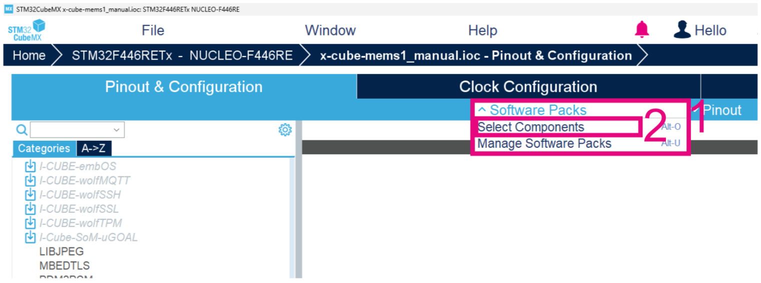
**Step 6.** If the project you are generating requires middleware, enable all that are needed.

## 4.10 Selecting components for CUSTOM boards

This section assumes that your project is using a custom board containing MEMS devices or one of the supported X-NUCLEO boards with an additional sensor plugged into the DIL24 connector.

Component selector is a tool that lets you select hardware for a given project and an application to be generated. That information is then used to include drivers for the selected components and to generate the application accordingly.

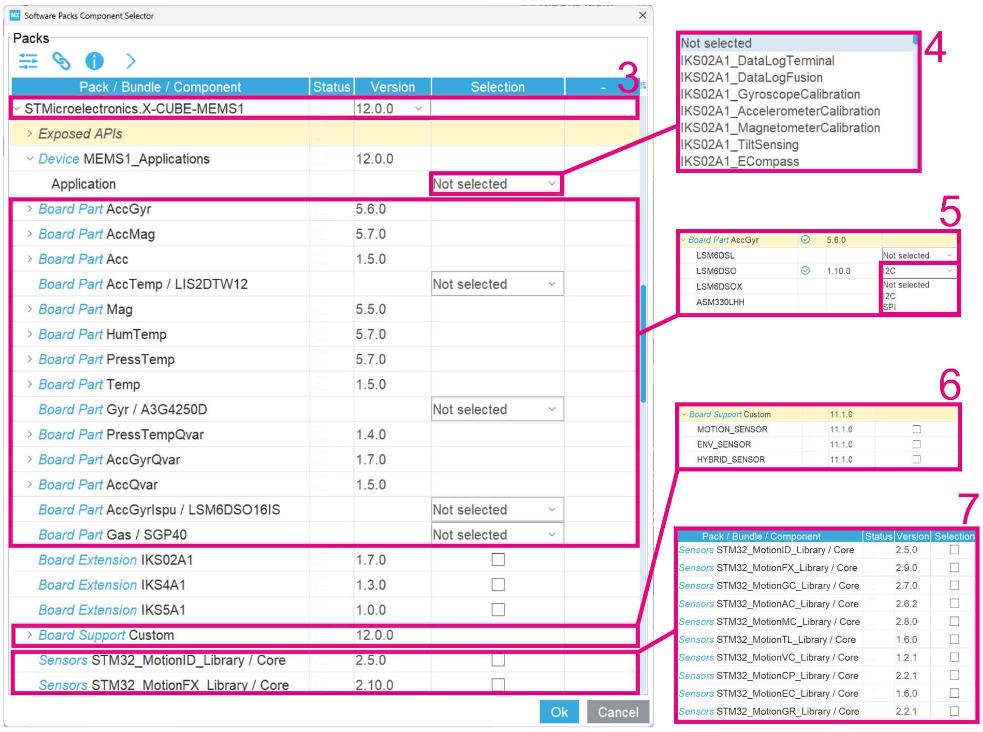
**Figure 23. Switching to component selector**



**Step 1.** Expand the `Software Packs` drop-down menu

**Step 2.** Click the `Select Components` which will open a new window

**Figure 24. Selecting components**



| Pack / Bundle / Component | Status | Version | Selection |
|---|---|---|---|
| STMicroelectronics.X-CUBE-MEMS1 | | 12.0.0 | |
| Exposed APIs | | | |
| Device MEMS1_Applications | | 12.0.0 | |
| Application | | | Not selected |
| Board Part AccGyr | | 5.6.0 | |
| Board Part AccMag | | 5.7.0 | |
| Board Part Acc | | 1.5.0 | |
| Board Part AccTemp / LIS2DTW12 | | | Not selected |
| Board Part Mag | | 5.5.0 | |
| Board Part HumTemp | | 5.7.0 | |
| Board Part PressTemp | | 5.7.0 | |
| Board Part Temp | | 1.5.0 | |
| Board Part Gyr / A3G4250D | | | Not selected |
| Board Part PressTempQvar | | 1.4.0 | |
| Board Part AccGyrQvar | | 1.7.0 | |
| Board Part AccQvar | | 1.5.0 | |
| Board Part AccGyrIspu / LSM6DSO16IS | | | Not selected |
| Board Part Gas / SGP40 | | | Not selected |
| Board Extension IKS02A1 | | 1.7.0 | □ |
| Board Extension IKS4A1 | | 1.3.0 | □ |
| Board Extension IKS5A1 | | 1.0.0 | □ |
| Board Support Custom | | 12.0.0 | |
| Sensors STM32_MotionID_Library / Core | | 2.5.0 | □ |
| Sensors STM32_MotionFX_Library / Core | | 2.10.0 | □ |

Not selected
IKS02A1_DataLogTerminal
IKS02A1_DataLogFusion
IKS02A1_GyroscopeCalibration
IKS02A1_AccelerometerCalibration
IKS02A1_MagnetometerCalibration
IKS02A1_TiltSensing
IKS02A1_ECompass

| | | | |
|---|---|---|---|
| Board Part AccGyr | ✓ | 5.6.0 | |
| LSM6DSL | | | Not selected |
| LSM6DSO | ✓ | 1.10.0 | I2C<br>Not selected<br>I2C<br>SPI |
| LSM6DSOX | | | |
| ASM330LHH | | | |

| | | | |
|---|---|---|---|
| Board Support Custom | | 11.1.0 | |
| MOTION_SENSOR | | 11.1.0 | □ |
| ENV_SENSOR | | 11.1.0 | □ |
| HYBRID_SENSOR | | 11.1.0 | □ |

| Pack / Bundle / Component | Status | Version | Selection |
|---|---|---|---|
| Sensors STM32_MotionID_Library / Core | | 2.5.0 | □ |
| Sensors STM32_MotionFX_Library / Core | | 2.9.0 | □ |
| Sensors STM32_MotionGC_Library / Core | | 2.7.0 | □ |
| Sensors STM32_MotionAC_Library / Core | | 2.6.2 | □ |
| Sensors STM32_MotionMC_Library / Core | | 2.8.0 | □ |
| Sensors STM32_MotionTL_Library / Core | | 1.6.0 | □ |
| Sensors STM32_MotionVC_Library / Core | | 1.2.1 | □ |
| Sensors STM32_MotionCP_Library / Core | | 2.2.1 | □ |
| Sensors STM32_MotionEC_Library / Core | | 1.6.0 | □ |
| Sensors STM32_MotionGR_Library / Core | | 2.2.1 | □ |

**Step 3.** In the `Pack / Bundle / Component` column locate the `STMicroelectronic.X-CUBE-MEMS1` pack and expand it

**Step 4.** Expand the `Device MEMS1_Applications` and use the drop-down menu to select the Application / Example you are configuring. **Beware**, in this context, the term application is used as an umbrella term for either Library evaluation application or Sensor evaluation example. Also note that unique projects are preconfigured for each expansion / custom board, meaning that board names are prefixed to the project names – for example the `ECompass` application exists as 3 distinct entries (`IKS02A1_ECompass`, `IKS4A1_ECompass` and `CUSTOM_ECompass`). Some projects may omit the prefix `CUSTOM_` altogether for custom boards.

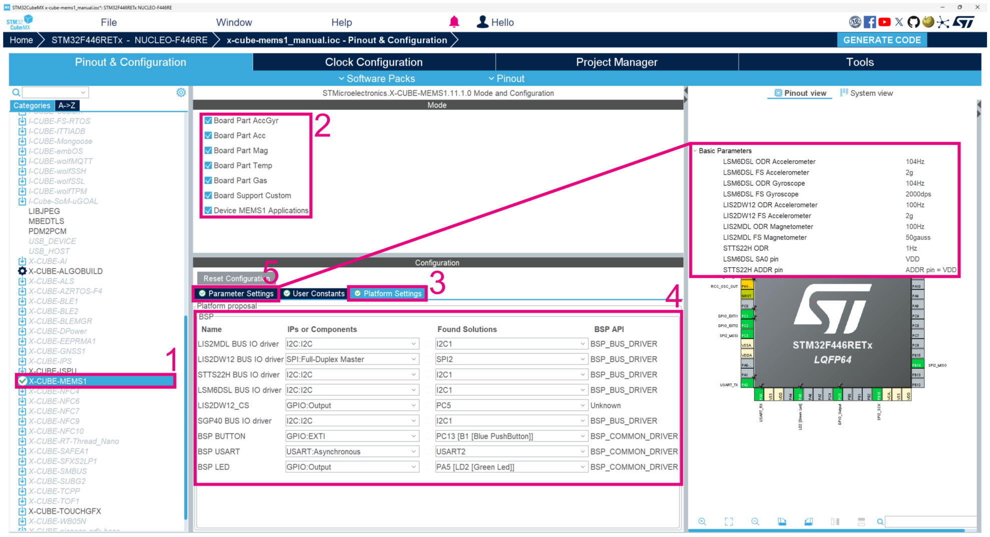
**Step 5.** Individual supported sensors are split into subsets based on the sensor capabilities. Set communication type for a sensor – either `I2C` or `SPI`, to allow the application to communicate with it. You can select multiple sensors.

**Step 6.** Apart from the previous step, you also need to select support for sensor type, which there are 3: `MOTION_SENSOR` (Accelerometer, Gyroscope, Magnetometer), `ENV_SENSOR` (Thermometer, Humidity, Pressure, Gas) or `HYBRID_SENSOR` (sensors that possess both motion and environmental measurement capabilities e.g. LIS2DTW12 which measures acceleration and temperature). You can choose multiple if required by the connected sensors.

**Step 7.** If the project you are generating requires middlewares, enable all that are needed.

## 4.11 Platform & Parameter settings

**Figure 25. Enabling used parts**



☑ Board Part AccGyr
☑ Board Part Acc
☑ Board Part Mag
☑ Board Part Temp
☑ Board Part Gas
☑ Board Support Custom
☑ Device MEMS1 Applications

Parameter Settings | User Constants | Platform Settings

Platform proposal – BSP

| Name | IPs or Components | Found Solutions | BSP API |
|---|---|---|---|
| LIS2MDL BUS IO driver | I2C:I2C | I2C1 | BSP_BUS_DRIVER |
| LIS2DW12 BUS IO driver | SPI:Full-Duplex Master | SPI2 | BSP_BUS_DRIVER |
| STTS22H BUS IO driver | I2C:I2C | I2C1 | BSP_BUS_DRIVER |
| LSM6DSL BUS IO driver | I2C:I2C | I2C1 | BSP_BUS_DRIVER |
| LIS2DW12_CS | GPIO:Output | PC5 | Unknown |
| SGP40 BUS IO driver | I2C:I2C | I2C1 | BSP_BUS_DRIVER |
| BSP BUTTON | GPIO:EXTI | PC13 [B1 [Blue PushButton]] | BSP_COMMON_DRIVER |
| BSP USART | USART:Asynchronous | USART2 | BSP_COMMON_DRIVER |
| BSP LED | GPIO:Output | PA5 [LD2 [Green Led]] | BSP_COMMON_DRIVER |

Basic Parameters

| | |
|---|---|
| LSM6DSL ODR Accelerometer | 104Hz |
| LSM6DSL FS Accelerometer | 2g |
| LSM6DSL ODR Gyroscope | 104Hz |
| LSM6DSL FS Gyroscope | 2000dps |
| LIS2DW12 ODR Accelerometer | 100Hz |
| LIS2DW12 FS Accelerometer | 2g |
| LIS2MDL ODR Magnetometer | 100Hz |
| LIS2MDL FS Magnetometer | 50gauss |
| STTS22H ODR | 1Hz |
| LSM6DSL SA0 pin | VDD |
| STTS22H ADDR pin | ADDR pin = VDD |

*Note: Note that you need to have all MCU peripherals already configured before proceeding with the configuration.*

**Step 1.** In `Categories`, go to `X-CUBE-MEMS1`

**Step 2.** In the `Mode` section enable all parts and libraries (if applicable) that the project requires

**Step 3.** In the `Configuration` section, switch to `Platform Settings` tab

**Step 4.** Assign peripherals to all required fields

**Step 5.** Switch to `Parameter Settings` tab and configure all parameters (not all applications or projects have parameters to configure in this way)

# 5 Sensor Evaluation Examples

The predefined sensor evaluation examples are designed to demonstrate the functionality of various MEMS drivers included in the X-CUBE-MEMS1 expansion pack.

Each Sensor Evaluation Example is configured to demonstrate particular functionality of a MEMS device. For an overview of available examples and the expansion boards they support, refer to the Table 3. Sensor Evaluation Examples. The firmware for each example is set up to establish communication with the connected sensors and demonstrate a particular functionality of the provided drivers. This often involves sending data over U(S)ART to a hyper terminal application running on the connected PC (Section 5.2: Tera Term gives suggestions on a good tool for this purpose). Some examples also use on-board LEDs to signal to the user. You can interact directly with some examples by pressing the on-board button.

Overview of all available examples is provided below. More in-depth description of all the examples is provided in Section 5.3: Detailed description of examples.

Table 3. **Sensor Evaluation Examples**

| Examples | NUCLEO-L073RZ | NUCLEO-L152RE | NUCLEO-F401RE | NUCLEO-U575ZI-Q | X-NUCLEO-IKS02A1 | X-NUCLEO-IKS4A1 | X-NUCLEO-IKS5A1 | CUSTOM |
|---|---|---|---|---|---|---|---|---|
| DataLogExtended | □ | ☒ | ☒ | ☒ | ☒ | ☒ | ☒ | □ |
| DataLogTerminal | ☒ | ☒ | ☒ | ☒ | ☒ | ☒ | ☒ | ☒ |
| LIS2DW12_6DOrientation | □ | □ | ☒ | ☒ | □ | □ | □ | ☒ |
| LIS2DW12_SelfTest | □ | □ | ☒ | ☒ | □ | □ | □ | ☒ |
| LIS2DW12_WakeUp | □ | □ | ☒ | ☒ | □ | □ | □ | ☒ |
| LIS2MDL_SelfTest | □ | □ | ☒ | ☒ | □ | □ | □ | ☒ |
| LPS22HB_FIFOMode | □ | □ | ☒ | ☒ | □ | □ | □ | ☒ |
| LPS22HH_FIFOMode | □ | □ | ☒ | ☒ | □ | □ | □ | ☒ |
| LSM6DSL_6DOrientation | □ | □ | ☒ | ☒ | □ | □ | □ | ☒ |
| LSM6DSL_FIFOContMode | □ | □ | ☒ | ☒ | □ | □ | □ | ☒ |
| LSM6DSL_FIFOLowPower | □ | □ | ☒ | ☒ | □ | □ | □ | ☒ |
| LSM6DSL_FIFOMode | □ | □ | ☒ | ☒ | □ | □ | □ | ☒ |
| LSM6DSL_FreeFall | □ | □ | ☒ | ☒ | □ | □ | □ | ☒ |
| LSM6DSL_MultiEvent | □ | □ | ☒ | ☒ | □ | □ | □ | ☒ |
| LSM6DSL_Pedometer | □ | □ | ☒ | ☒ | □ | □ | □ | ☒ |
| LSM6DSL_SelfTest | □ | □ | ☒ | ☒ | □ | □ | □ | ☒ |
| LSM6DSL_SingleDoubleTap | □ | □ | ☒ | ☒ | □ | □ | □ | ☒ |
| LSM6DSL_Tilt | □ | □ | ☒ | ☒ | □ | □ | □ | ☒ |
| LSM6DSL_WakeUp | □ | □ | ☒ | ☒ | □ | □ | □ | ☒ |
| LSM6DSO_6DOrientation | □ | □ | ☒ | ☒ | □ | □ | □ | ☒ |
| LSM6DSO_FIFOContMode | □ | □ | ☒ | ☒ | □ | □ | □ | ☒ |
| LSM6DSO_FIFOMode | □ | □ | ☒ | ☒ | □ | □ | □ | ☒ |
| LSM6DSO_FreeFall | □ | □ | ☒ | ☒ | □ | □ | □ | ☒ |
| LSM6DSO_Pedometer | □ | □ | ☒ | ☒ | □ | □ | □ | ☒ |

| Examples | NUCLEO-L073RZ | NUCLEO-L152RE | NUCLEO-F401RE | NUCLEO-U575ZI-Q | X-NUCLEO-IKS02A1 | X-NUCLEO-IKS4A1 | X-NUCLEO-IKS5A1 | CUSTOM |
|---|---|---|---|---|---|---|---|---|
| LSM6DSO_SelfTest | ☐ | ☐ | ☒ | ☒ | ☐ | ☐ | ☐ | ☒ |
| LSM6DSO_SingleDoubleTap | ☐ | ☐ | ☒ | ☒ | ☐ | ☐ | ☐ | ☒ |
| LSM6DSO_Tilt | ☐ | ☐ | ☒ | ☒ | ☐ | ☐ | ☐ | ☒ |
| LSM6DSO_WakeUp | ☐ | ☐ | ☒ | ☒ | ☐ | ☐ | ☐ | ☒ |

## 5.1 Example setup checklist

This chapter provides a checklist to guide you through sensor evaluation example configuration process. Additionally, the individual points link to sections that provide detailed instructions on how the configuration can be performed with the help of STM32CubeMX.

If you're using a hardware configuration that is indicated as supported in Table 3. Sensor Evaluation Examples, you can skip the manual configuration and upload the built binaries directly. See for more information. This does not apply to CUSTOM boards.

*Note:* *This checklist applies to all available examples but the Section 5.3.1: DataLogExtended which, unlike other examples, can't be generated or configured in STM32CubeMX. You, however, still get access to the source code which you can compile, modify and upload yourself. Refer to the its description directly to learn how it can be used.*

1. Configure MCU peripherals required for sensor evaluation example
   SPI / I2C – refer to Section 4.5: I2C / SPI
   U(S)ART – refer to Section 4.8: U(S)ART
   GPIO output to drive an LED – refer to Section 4.4: GPIO
   GPIO input for BUTTON input, with EXTI enabled – refer to Section 4.4: GPIO
2. In `Component Selector` select application, parts, board extensions and middlewares required by the application – depending on your HW configuration refer to Section 4.10: Selecting components for CUSTOM boards or Section 4.9: Selecting components for configurations utilizing X-NUCLEO boards respectively
3. Set `Platform settings` – select what peripherals that are going to be used by the application (peripherals need to be already configured at this point), set application specific `Parameter Settings` (if applicable) – refer to Section 4.11: Platform & Parameter settings
4. Choose IDE for which to create the project and then generate it – refer to
5. Build the project and upload the resulting binary file to your board with an IDE selected in previous step
6. Open the serial hyper terminal application of your choosing software and experiment with the example – refer to Section 5.2: Tera Term which shows how one can use that open source software for this purpose

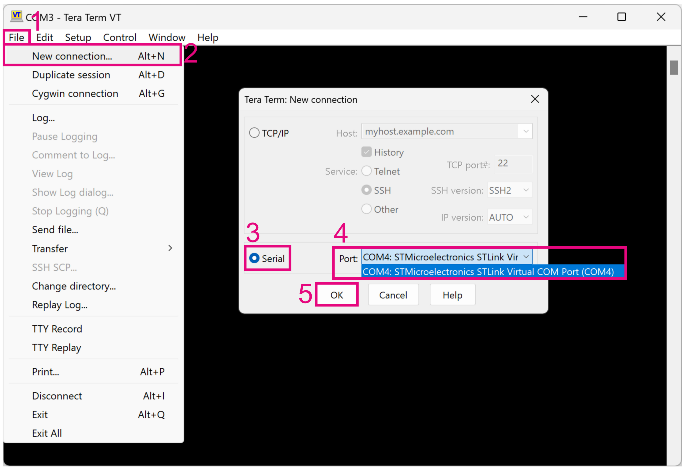
## 5.2 Tera Term

Tera Term is a user-friendly serial terminal emulator software, used for serial communication. It is published under an open source license. In the context of the X-CUBE-MEMS1 expansion pack it proves useful when trying to read data from a sensor evaluation example. The use case presented here is going to show the functionality needed to establish serial connection from the application to a connected board and to display the received data in text form.

### Creating new connection

1. Start the application and click the `File` tab
2. Click the `New connection...` option and wait until `Tera Term: New connection` window appears
3. Select the `Serial` connection type
4. Click the `Port` combobox and select the port which your device is connected to
5. Confirm your connection configuration by clicking `OK`

**Figure 26. Tera Term connection steps**

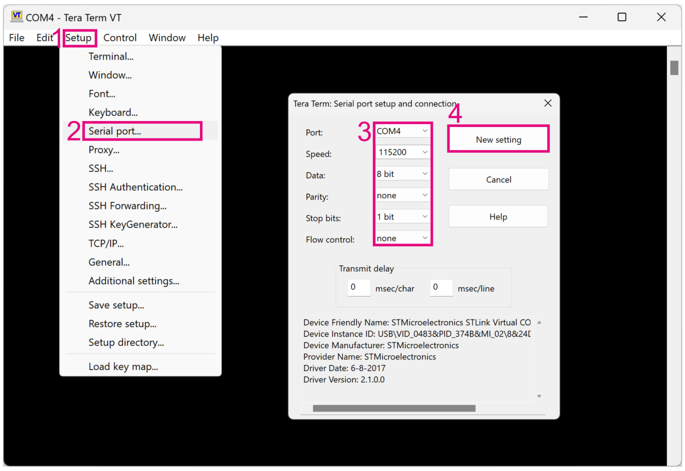


○ TCP/IP Host: myhost.example.com
☑ History
Service: ○ Telnet TCP port#: 22
● SSH SSH version: SSH2
○ Other IP version: AUTO

● Serial Port: COM4: STMicroelectronics STLink Vir
COM4: STMicroelectronics STLink Virtual COM Port (COM4)

**Configuring serial parameters**

1. Click the `Setup` tab
2. From the menu, select the `Serial port…` option – wait until new `Tera Term: Serial port setup and connection` window opens
3. Set the communication parameters – `Port`, `Speed(baudrate)`, `Data,…` – to accommodate your example. If you are using unmodified examples, then use the following settings:
   - Speed: `115200`
   - Data: `8 bit`
   - Parity: `none`
   - Stop bits: `1 bit`
   - Flow control: `none`
4. Finally click the New setting button to confirm the configuration and to establish the connection



Port: COM4
Speed: 115200
Data: 8 bit
Parity: none
Stop bits: 1 bit
Flow control: none

Transmit delay
0 msec/char 0 msec/line

Device Friendly Name: STMicroelectronics STLink Virtual CO
Device Instance ID: USB\VID_0483&PID_374B&MI_02\8&24[
Device Manufacturer: STMicroelectronics
Provider Name: STMicroelectronics
Driver Date: 6-8-2017
Driver Version: 2.1.0.0

Figure 27. **Tera Term serial port setup**

## 5.3 Detailed description of examples

### 5.3.1 DataLogExtended

**Hardware:** Nucleo + X-NUCLEO (IKS02A1 / IKS4A1 / IKS5A1)

**Functionality:** Connect your board to MEMS-Studio PC application and take advantage of various sensor data visualizations.

**Usage:** Download and install the MEMS-Studio PC application and upload the DataLogExtended application to a supported Nucleo + IKS board combination. Take advantage of the user-friendly MEMS-Studio PC software to configure the connected sensor, collect, visualize, analyze, and save data, and design no-code algorithms.

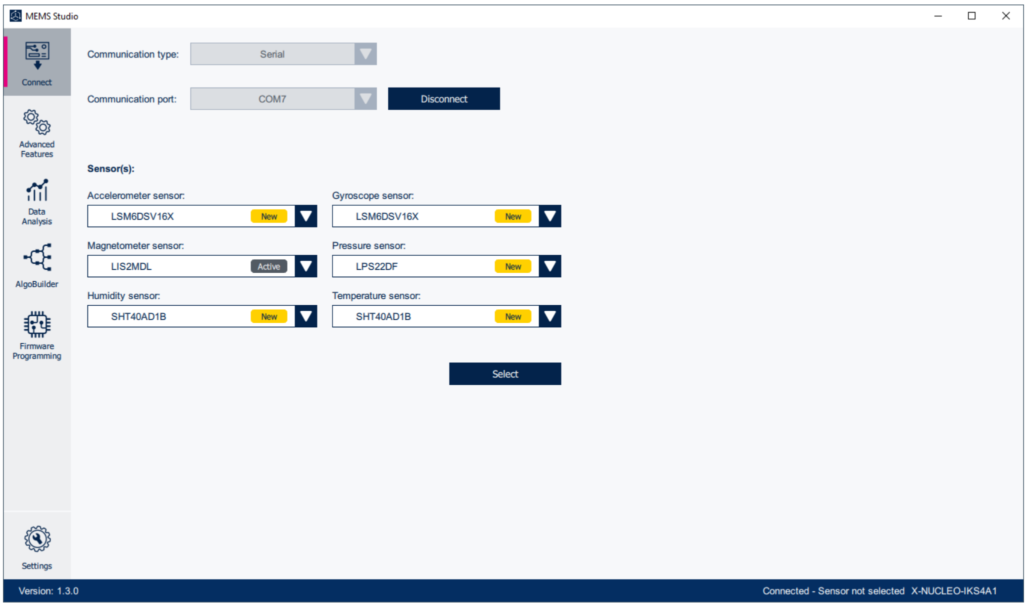
*Note:* *This application can't be generated by the user. Projects for supported IDEs are, however, provided so that you can explore the source code and build the application. You can take advantage of Compiled binaries to avoid having to build the application yourself.*

**Step 1.** Upload Table 1 or DataLogExtended sensor evaluation example to your board.

**Step 2.** Follow steps in Finding your board's COM port to find which COM port is used by your board.

**Step 3.** Launch MEMS-Studio application, ensure the COM port number for the current STM32 Nucleo board is correct and select the sensors that you want to use.

**Figure 28. MEMS-Studio main page**



Communication type: Serial

Communication port: COM7

Sensor(s):

Accelerometer sensor: LSM6DSV16X New

Gyroscope sensor: LSM6DSV16X New

Magnetometer sensor: LIS2MDL Active

Pressure sensor: LPS22DF New

Humidity sensor: SHT40AD1B New

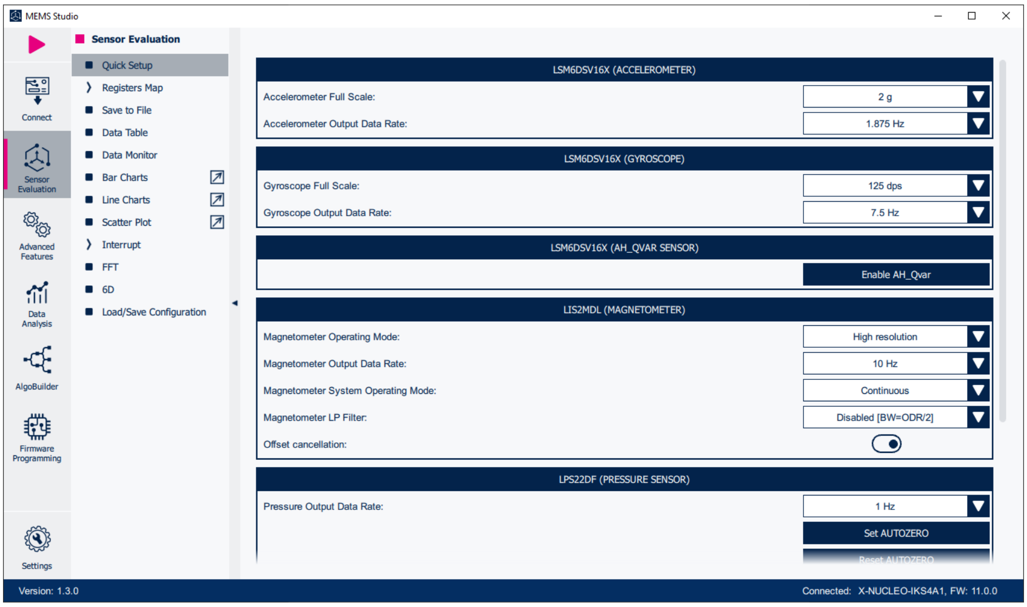
Temperature sensor: SHT40AD1B New

Version: 1.3.0

Connected - Sensor not selected X-NUCLEO-IKS4A1

**Step 4.** Setup various sensors (e.g., pressure, temperature, humidity, accelerometer, gyroscope, magnetometer) available on the expansion board.

**Figure 29. MEMS-Studio Utility sensor and interval selection**



LSM6DSV16X (ACCELEROMETER)

Accelerometer Full Scale: 2 g

Accelerometer Output Data Rate: 1.875 Hz

LSM6DSV16X (GYROSCOPE)

Gyroscope Full Scale: 125 dps

Gyroscope Output Data Rate: 7.5 Hz

LSM6DSV16X (AH_QVAR SENSOR)

LIS2MDL (MAGNETOMETER)

Magnetometer Operating Mode: High resolution

Magnetometer Output Data Rate: 10 Hz

Magnetometer System Operating Mode: Continuous

Magnetometer LP Filter: Disabled [BW=ODR/2]

Offset cancellation:

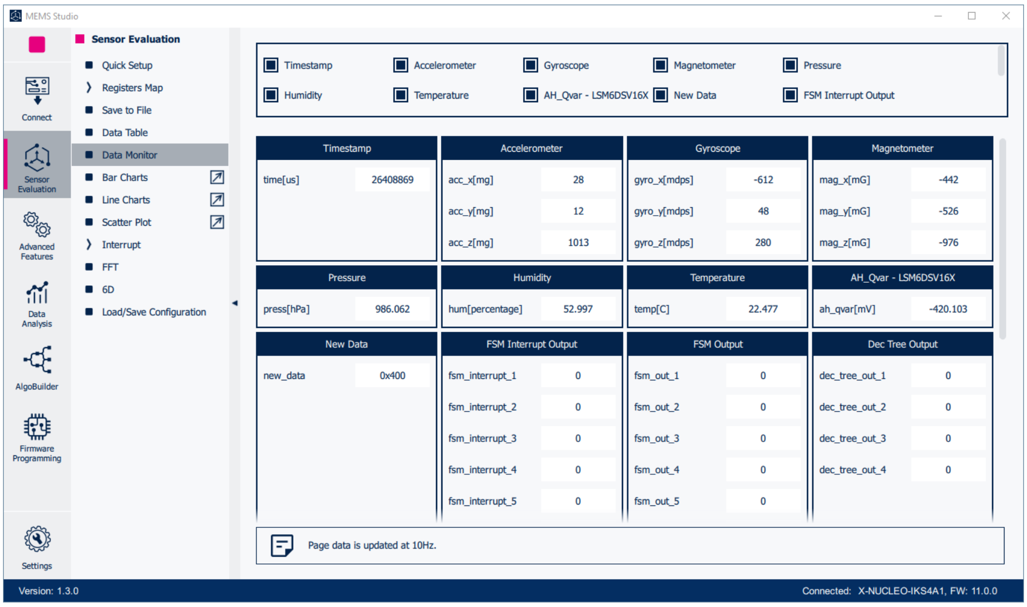
LPS22DF (PRESSURE SENSOR)

Pressure Output Data Rate: 1 Hz

Version: 1.3.0

Connected: X-NUCLEO-IKS4A1, FW: 11.0.0

**Step 5.** Press "Start" to display the data.



MEMS Studio

☒ Timestamp ☒ Accelerometer ☒ Gyroscope ☒ Magnetometer ☒ Pressure
☒ Humidity ☒ Temperature ☒ AH_Qvar - LSM6DSV16X ☒ New Data ☒ FSM Interrupt Output

| Timestamp | | Accelerometer | | Gyroscope | | Magnetometer | |
|---|---|---|---|---|---|---|---|
| time[us] | 26408869 | acc_x[mg] | 28 | gyro_x[mdps] | -612 | mag_x[mG] | -442 |
| | | acc_y[mg] | 12 | gyro_y[mdps] | 48 | mag_y[mG] | -526 |
| | | acc_z[mg] | 1013 | gyro_z[mdps] | 280 | mag_z[mG] | -976 |
| **Pressure** | | **Humidity** | | **Temperature** | | **AH_Qvar - LSM6DSV16X** | |
| press[hPa] | 986.062 | hum[percentage] | 52.997 | temp[C] | 22.477 | ah_qvar[mV] | -420.103 |
| **New Data** | | **FSM Interrupt Output** | | **FSM Output** | | **Dec Tree Output** | |
| new_data | 0x400 | fsm_interrupt_1 | 0 | fsm_out_1 | 0 | dec_tree_out_1 | 0 |
| | | fsm_interrupt_2 | 0 | fsm_out_2 | 0 | dec_tree_out_2 | 0 |
| | | fsm_interrupt_3 | 0 | fsm_out_3 | 0 | dec_tree_out_3 | 0 |
| | | fsm_interrupt_4 | 0 | fsm_out_4 | 0 | dec_tree_out_4 | 0 |
| | | fsm_interrupt_5 | 0 | fsm_out_5 | 0 | | |

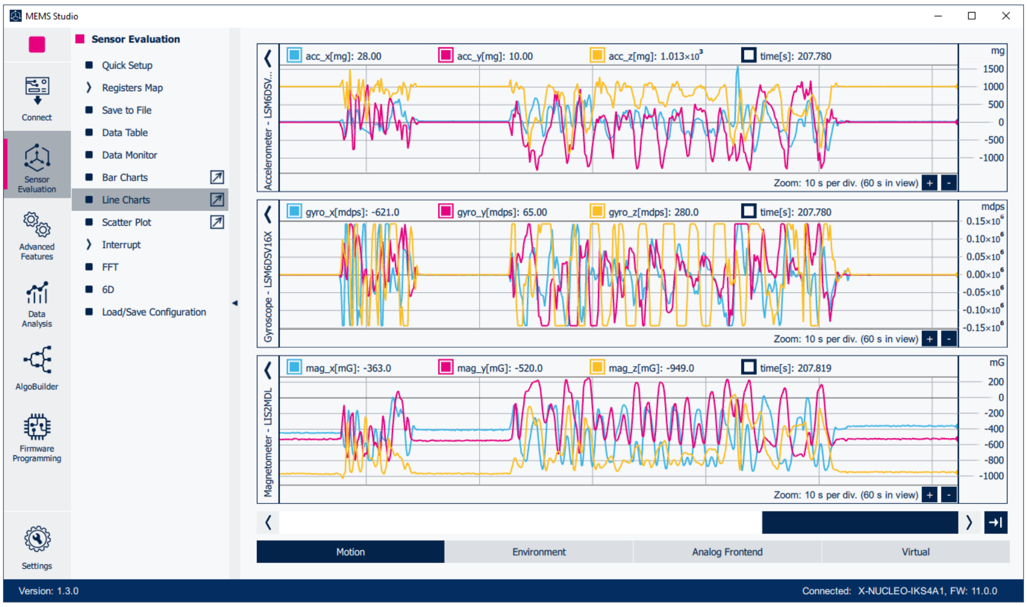
Page data is updated at 10Hz.

Version: 1.3.0

Connected: X-NUCLEO-IKS4A1, FW: 11.0.0

### Figure 30. MEMS-Studio data plot

**Step 6.** Press "Line Charts/Motion" to display inertial sensor data.



MEMS Studio

### Figure 31. MEMS-Studio Motion Sensor Plot

**Step 7.** Press "Line Charts/Environmental" to display environmental sensor data.

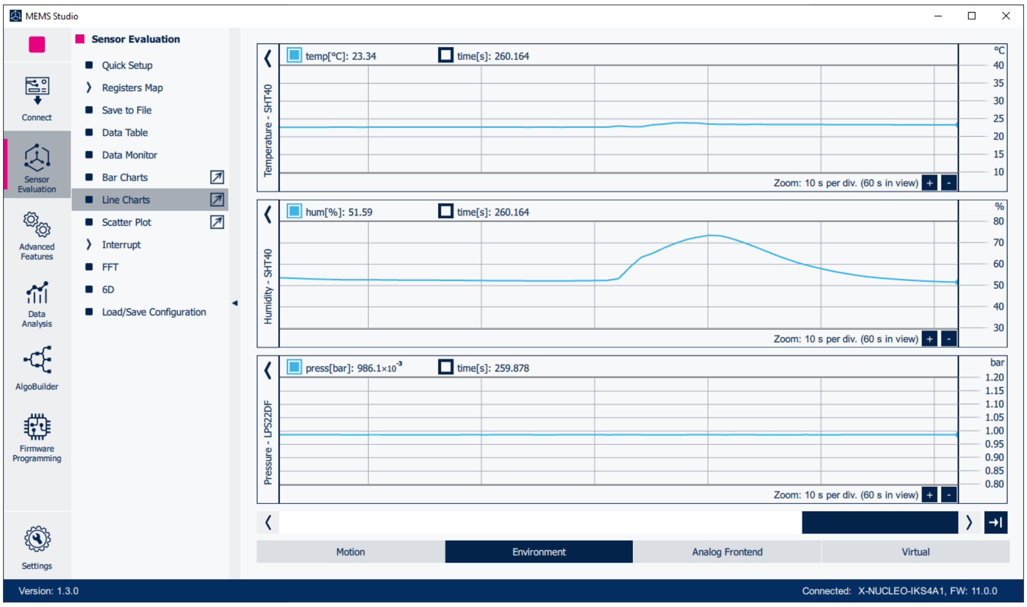


Figure 32. **MEMS-Studio Environmental Sensor Plot**

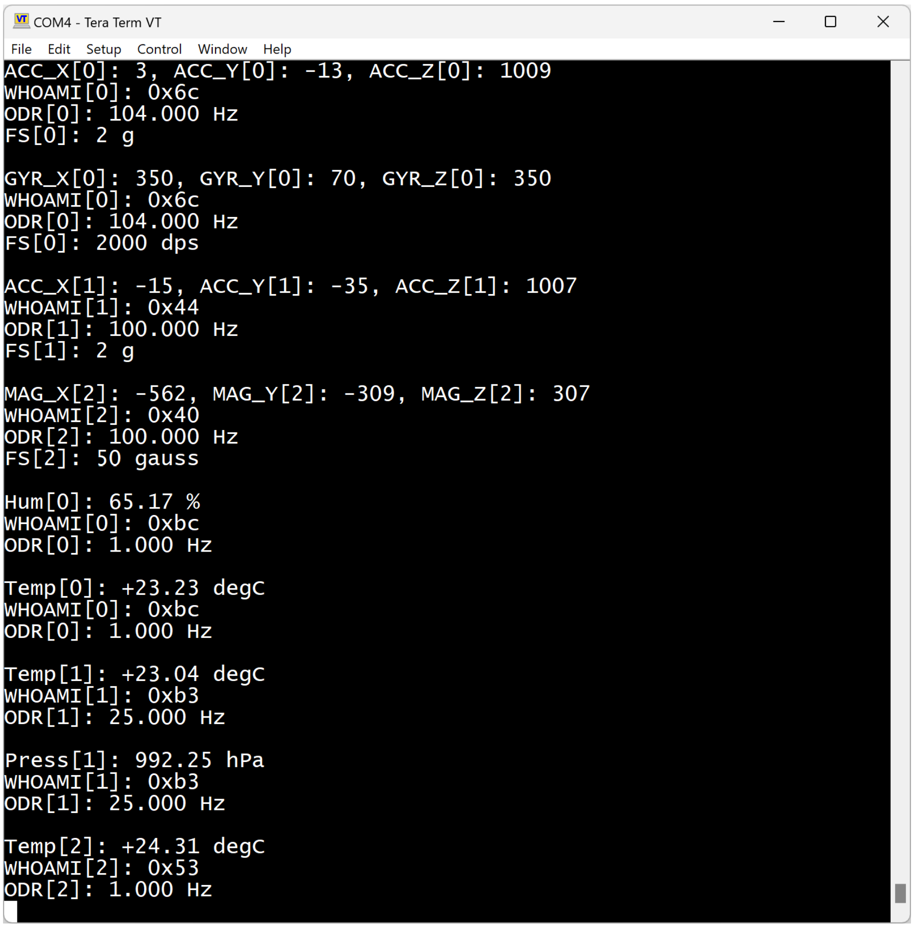
### 5.3.2 DataLogTerminal

**Hardware:** Nucleo + X-NUCLEO (IKS02A1 / IKS4A1 / IKS5A1) or CUSTOM

**Functionality:** Stream sensor data of all onboard sensors to the user's PC.

**Usage:** After starting, the application collects data from all onboard environmental (temperature, humidity, and pressure sensors) and inertial (accelerometer, gyroscope, and magnetometer) sensors and sends them to the connected PC.

Figure 33. DataLogTerminal application screenshot with X-NUCLEO-IKS4A1



```
ACC_X[0]: 3, ACC_Y[0]: -13, ACC_Z[0]: 1009
WHOAMI[0]: 0x6c
ODR[0]: 104.000 Hz
FS[0]: 2 g

GYR_X[0]: 350, GYR_Y[0]: 70, GYR_Z[0]: 350
WHOAMI[0]: 0x6c
ODR[0]: 104.000 Hz
FS[0]: 2000 dps

ACC_X[1]: -15, ACC_Y[1]: -35, ACC_Z[1]: 1007
WHOAMI[1]: 0x44
ODR[1]: 100.000 Hz
FS[1]: 2 g

MAG_X[2]: -562, MAG_Y[2]: -309, MAG_Z[2]: 307
WHOAMI[2]: 0x40
ODR[2]: 100.000 Hz
FS[2]: 50 gauss

Hum[0]: 65.17 %
WHOAMI[0]: 0xbc
ODR[0]: 1.000 Hz

Temp[0]: +23.23 degC
WHOAMI[0]: 0xbc
ODR[0]: 1.000 Hz

Temp[1]: +23.04 degC
WHOAMI[1]: 0xb3
ODR[1]: 25.000 Hz

Press[1]: 992.25 hPa
WHOAMI[1]: 0xb3
ODR[1]: 25.000 Hz

Temp[2]: +24.31 degC
WHOAMI[2]: 0x53
ODR[2]: 1.000 Hz
```

### 5.3.3 LIS2DW12_6DOrientation / LSM6DSL_6DOrientation / LSM6DSO_6DOrientation

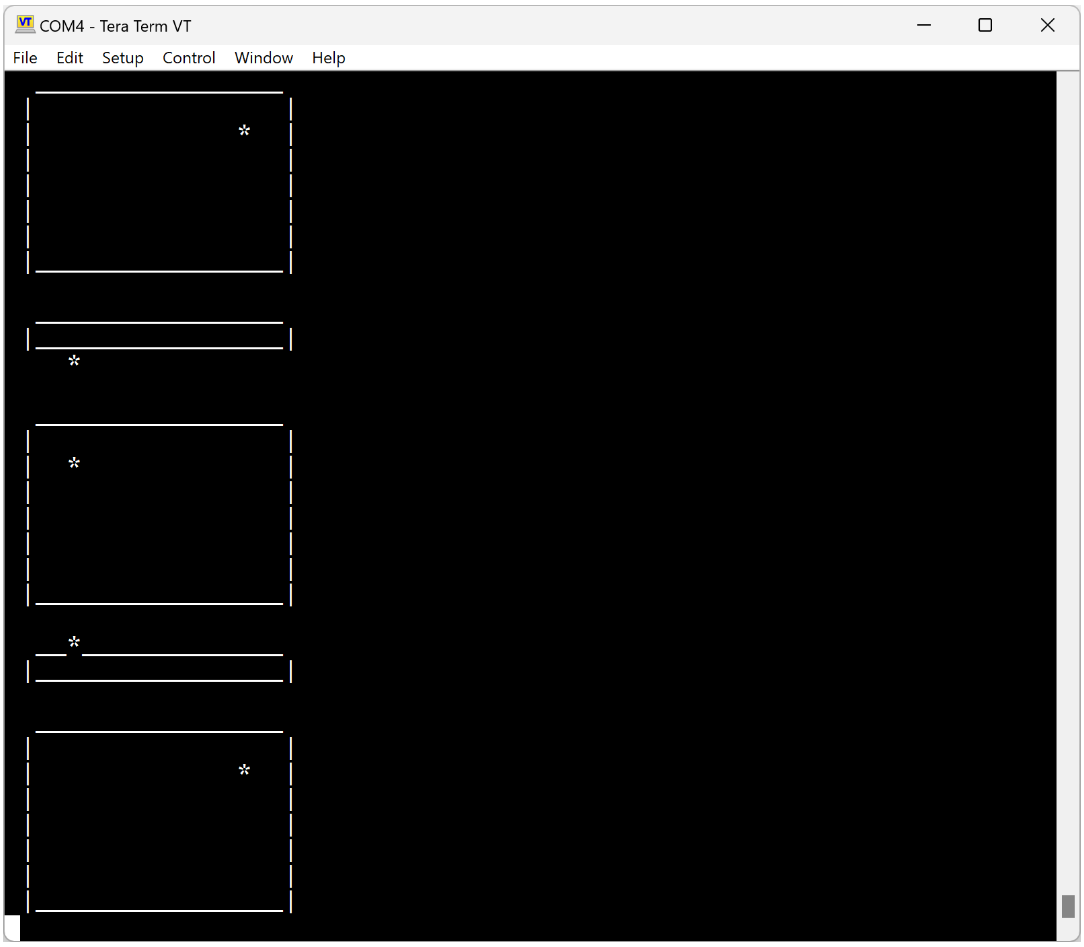
**Hardware**:

- LIS2DW12 / LSM6DSL / LSM6DSO – CUSTOM

**Functionality**: Use the connected accelerometer to determine the device's 6D orientation with reference to earth's gravity.

**Usage**: Rotate the board to change the 6D orientation. The orientation data is sent over UART to the connected PC when the 6D orientation changes or when the user button is pressed.

Figure 34. **6D orientation example for an accelerometer**



```
 _______________________
|                       |
|                   *   |
|                       |
|                       |
|                       |
|                       |
|_______________________|

 _______________________
|_______________________|
    *

 _______________________
|                       |
|   *                   |
|                       |
|                       |
|                       |
|                       |
|_______________________|

    *
 ___ ___________________
|_______________________|

 _______________________
|                       |
|                   *   |
|                       |
|                       |
|                       |
|                       |
|_______________________|
```

## 5.3.5 LIS2DW12_WakeUp / LSM6DSL_WakeUp / LSM6DSO_WakeUp

**Hardware:**

- LIS2DW12 / LSM6DSL / LSM6DSO – CUSTOM

**Functionality:** Detect the wake-up event with the connected sensor.

**Usage:** Move the sensor board from a still position. The configured device will detect the wake-up event, and the indication LED will blink briefly. Pressing the user button toggles wake-up detection on or off. This application doesn't send any data to the user.

## 5.3.6 LPS22HB_FIFOMode / LPS22HH_FIFOMode / LSM6DSL_FIFOMode / LSM6DSO_FIFOMode

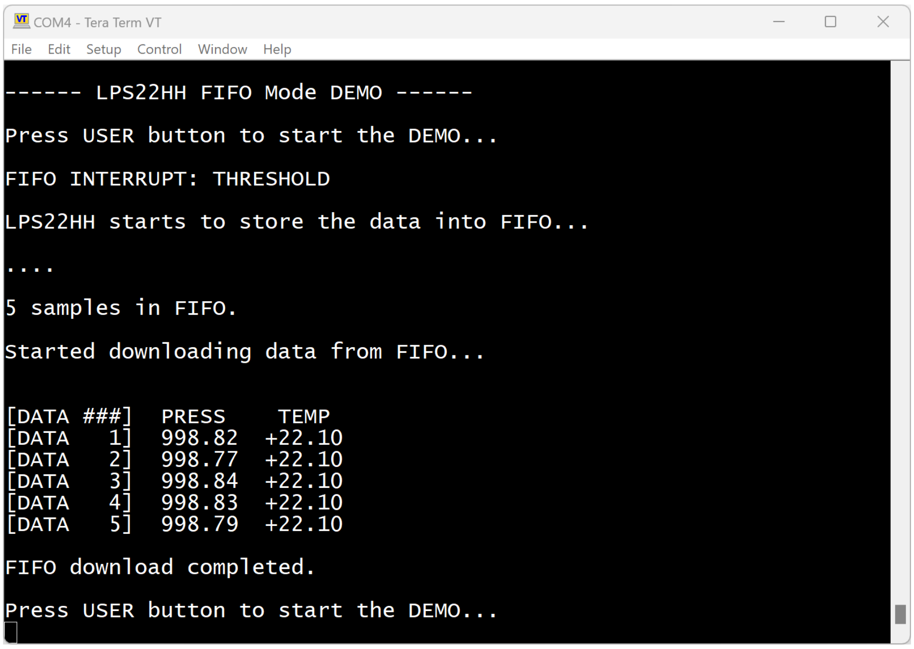
**Hardware:**

- LPS22HB / LPS22HH / LSM6DSO / LSM6DSL – CUSTOM board

**Functionality:** Collect data from a sensor in FIFO mode.

**Usage:** When the user button is pressed, the connected sensor is configured to work in FIFO mode, and the parameters of that configuration are sent to the connected PC. Available sensor data is then collected in FIFO. Each received data point is signaled to the user by a . sent to their PC. After FIFO reaches its capacity, the signaling LED turns on, and all data is read and sent to the PC. After the transfer's completion, the LED turns off, and the FIFO switches to Bypass Mode, where it remains until the application is restarted with another button press.

Figure 35. **FIFO mode example running on LPS22HH**



```
------ LPS22HH FIFO Mode DEMO ------

Press USER button to start the DEMO...

FIFO INTERRUPT: THRESHOLD

LPS22HH starts to store the data into FIFO...

....

5 samples in FIFO.

Started downloading data from FIFO...


[DATA ###]  PRESS    TEMP
[DATA   1]  998.82  +22.10
[DATA   2]  998.77  +22.10
[DATA   3]  998.84  +22.10
[DATA   4]  998.83  +22.10
[DATA   5]  998.79  +22.10

FIFO download completed.

Press USER button to start the DEMO...
```

### 5.3.7 LSM6DSL_FIFOContMode / LSM6DSO_FIFOContMode

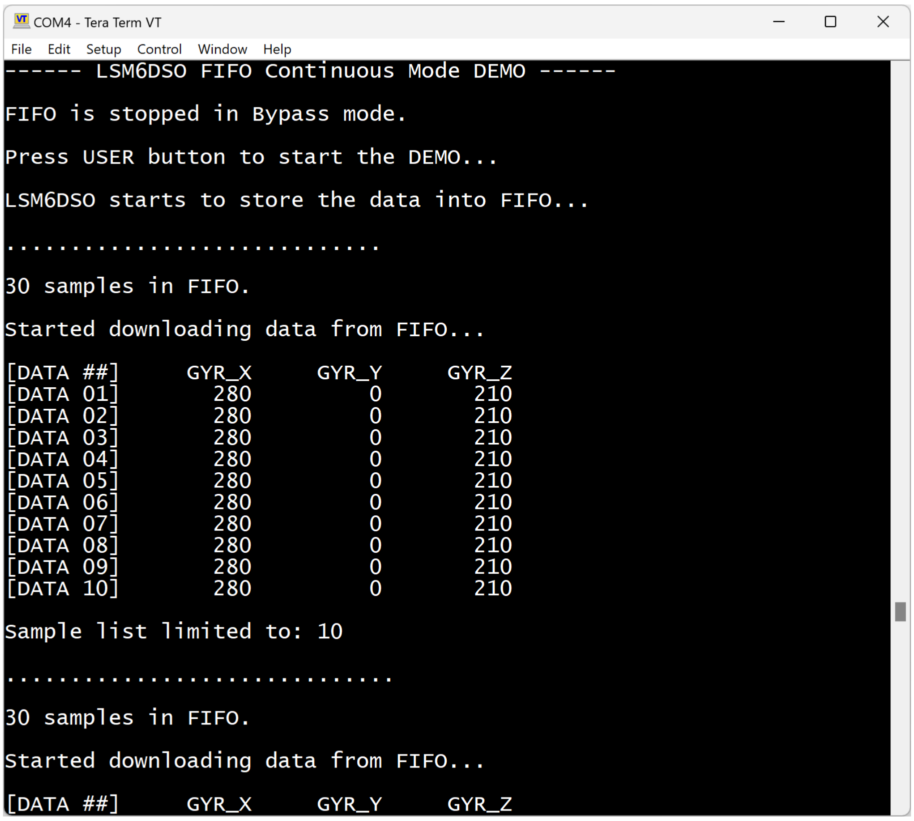
**Hardware:**

- LSM6DSL / LSM6DSO – CUSTOM

**Functionality:** Collect Gyroscope data from a sensor in FIFO continuous mode.

**Usage:** Push the user button to launch the FIFO demo in continuous mode. Every new sample of gyroscope data is indicated by a dot sent to the user. After pressing the button for the second time or when the FIFO is full, all the data stored in the FIFO is read out and sent to the user. The sensor's FIFO mode is then changed to bypass mode. If you press the user button again, the demo restarts.

Figure 36. **FIFO continuous mode example with LSM6DSO**



```
------ LSM6DSO FIFO Continuous Mode DEMO ------

FIFO is stopped in Bypass mode.

Press USER button to start the DEMO...

LSM6DSO starts to store the data into FIFO...

.............................

30 samples in FIFO.

Started downloading data from FIFO...

[DATA ##]     GYR_X     GYR_Y     GYR_Z
[DATA 01]       280         0       210
[DATA 02]       280         0       210
[DATA 03]       280         0       210
[DATA 04]       280         0       210
[DATA 05]       280         0       210
[DATA 06]       280         0       210
[DATA 07]       280         0       210
[DATA 08]       280         0       210
[DATA 09]       280         0       210
[DATA 10]       280         0       210

Sample list limited to: 10

..............................

30 samples in FIFO.

Started downloading data from FIFO...

[DATA ##]     GYR_X     GYR_Y     GYR_Z
```

### 5.3.8 LSM6DSL_FIFOLowPower

**Hardware:**

- LSM6DSL - CUSTOM board

**Functionality:** The sensor autonomously collects data in FIFO mode, allowing the MCU to sleep until the FIFO is full.

**Usage:** The sensor is configured in FIFO mode and wakes the MCU up with a change on the INT pin. When that occurs, that information is sent to the user's PC. If the FIFO is full then all data is read and sent to the user. The user can wake the MCU up by pressing the user button at any time.

### 5.3.9 LSM6DSL_FreeFall / LSM6DSO_FreeFall

**Hardware:**

- LSM6DSL / LSM6DSO – CUSTOM

**Functionality:**Detect the free fall event with the connected sensor.

**Usage:**Imitate fall of the board. When the free fall event is detected, the signal LED will blink. The user button can be used to enable / disable the detection feature. This example sends no data to the user.

### 5.3.10 LSM6DSL_MultiEvent

**Hardware:**

- LSM6DSL - CUSTOM

**Functionality:**Detect various activity events with the LSM6DSL device.

**Usage:** Imitate one of the following activities: Free fall, Single tap, Double tap, Tilt, Wake-up, 6D Orientation. The application will send real-time reports about the activity being performed by the device.

### 5.3.11 LSM6DSL_Pedometer / LSM6DSO_Pedometer

**Hardware:**

- LSM6DSL / LSM6DSO – CUSTOM

**Functionality:**Use the connected device to count steps.

**Usage:** Move the board to simulate steps. Every time a step is detected, the signal LED blinks and the application sends information on how many steps the application has detected. Use the user button to reset the step counter.

### 5.3.12 LSM6DSL_SingleDoubleTap / LSM6DSO_SingleDoubleTap

**Hardware:**

- LSM6DSL / LSM6DSO – CUSTOM

**Functionality:** Detect single and double tap events with the connected sensor.

**Usage:** Tap the board containing the sensor. When the single tap event is detected, the board LED will blink shortly. Press the user button to switch to double tap detection. Now if you double tap the board, the double tap event will be detected and the LED will blink twice. Press the button again to disable the detection features altogether.

### 5.3.13 LSM6DSL_Tilt / LSM6DSO_Tilt

**Hardware:**

- LSM6DSL / LSM6DSO – CUSTOM

**Functionality:** Detect the tilt event using the connected sensor.

**Usage:** Tilt the board containing the detecting sensor. When the tilt event is detected, the configured LED will blink. The user button can be used to enable or disable the tilt detection feature.

### 5.3.14 STTS751_TemperatureLimit

**Hardware:**

- STTS751 – CUSTOM

**Functionality:**Measure temperature and detect excesses of temperature limits.

**Usage:**Application periodically sends information about the current temperature to the user. It also configures a temperature interval. If the measured temperature falls outside the interval, the signal LED will blink.

# 6 Middleware

The X-CUBE-MEMS1 expansion pack contains many libraries pertaining to processing data obtained from MEMS devices. They are developed by ST (apart from EnvAQ which is developed by Sensirion) and provided as middleware in the pack. To learn more about particularities of each library, consult its user manual. The following table gives an overview of all supported libraries and provides high-level description of functionality, supported ARM Cortex® cores and lists applications that demonstrate functionality of each one.

Table 4. **Included Middlewares**

| Library name | Functionality | Supported ARM Cortex® architectures | Library evaluation application |
|---|---|---|---|
| BioVSM | Vital signs monitoring | M0+, M3, M4, M7, M33 | – |
| EnvAQ | Provides a VOC and an NOx Index | All | AirQuality_SGP40 |
| InfraredAL | Presence detection | M0+, M3, M4, M7, M33 | ApproachLeave_STHS34PF80 |
| InfraredPD | Presence and motion detection libraries | M0+, M3, M4, M7, M33 | PresenceDetection_STHS34PF80 |
| MotionAC | Accelerometer calibration | M3, M4, M7, M33 | AccelerometerCalibration |
| MotionAC2 | Realtime accelerometer calibration | M0+, M3, M4, M7, M33 | AccelCalibration2_IIS2ICLX |
| MotionAD | Airplane detection | M3, M4, M7, M33 | AirplaneDetection |
| MotionAR | Activity recognition | M3, M4, M7, M33 | ActivityRecognition |
| MotionAT | Active time | M3, M4, M7, M33 | ActiveTime |
| MotionAW | Activity recognition for wrist | M3, M4, M7, M33 | ActivityRecognitionWrist, PedometerWrist |
| MotionCP | Realtime carry position | M3, M4, M7, M33 | CarryPosition |
| MotionDI | Dynamic inclinometer | M3, M4, M7, M33 | DynamicInclinometer |
| MotionEC | Realtime e-compass | M0+, M3, M4, M7, M33 | Ecompass |
| MotionFA | Fitness activity | M3, M4, M7, M33 | FitnessActivities |
| MotionFD | Realtime fall detection | M3, M4, M7, M33 | FallDetection |
| MotionFX | Sensor fusion | M0+, M3, M4, M7, M33 | DataLogFusion |
| MotionGC | Gyroscope calibration | M0+, M3, M4, M7, M33 | GyroscopeCalibration |
| MotionGR | Realtime gesture recognition | M3, M4, M7, M33 | GestureRecognition |
| MotionGT | Temperature compensated gyroscope calibration parameters | M0+, M3, M4 | – |
| MotionID | Motion intensity detection | M0+, M3, M4, M7, M33 | IntensityDetection |
| MotionMC | Magnetometer calibration | M0+, M3, M4, M7, M33 | Ecompass, MagnetometerCalibration |
| MotionPE | Realtime pose estimation | M3, M4, M7, M33 | PoseEstimation |
| MotionPM | Realtime pedometer library | M3, M4, M7, M33 | Pedometer |
| MotionPW | Realtime pedometer for wrist | M3, M4, M7, M33 | PedometerWrist |
| MotionSD | Standing vs sitting desk detection | M3, M4, M7, M33 | StandingSittingDesk |
| MotionSM | Realtime information about a sleeping user | M3, M4, M7, M33 | – |
| MotionSP | Time and frequency domain vibration monitoring | All | VibrationMonitoring |
| MotionTL | Tilt measurement | M0+, M3, M4, M7, M33 | TiltSensing |
| MotionTL2 | Two-axis tilt sensing | M0+, M3, M4, M7, M33 | TiltSensing2_IIS2ICLX |
| MotionVC | Vertical context | M3, M4, M7, M33 | VerticalContext |
| MotionXLF | High-g and low-g fusion | M0+, M3, M4, M7, M33 | HighGLowGFusion, HighGLowGFusion_LSM6DSV320X |

# 7 Library Evaluation Applications

The predefined Library Evaluation Applications serve to demonstrate the functionality of various Section 6: Middleware supplied as part of the X-CUBE-MEMS1 expansion pack.

Every Library Evaluation application is set up to demonstrate functionality of a particular middleware (see Table 5. Library evaluation applications for an overview of available applications and the middleware they use). The application's firmware is configured to establish communication with the connected sensors. It then collects data from them, uses the enabled middleware to process the data and then sends it over U(S)ART to MEMS-Studio application running on the connected PC. Once the firmware is uploaded to the board and the application started, MEMS-Studio is going to detect the connected board with running application and let you connect to it.

It is strongly recommended that you use the applications in conjunction with the MEMS-Studio PC software, otherwise, the functionality of the applications will be severely limited. Consult Section 3.2: MEMS-Studio to learn more about installation and functionality.

Table 5. **Library evaluation applications**

| Library Eval Application | NUCLEO-L073RZ | NUCLEO-L152RE | NUCLEO-F401RE | NUCLEO-U575ZI-Q | X-NUCLEO-IKS02A1 | X-NUCLEO-IKS4A1 | X-NUCLEO-IKS5A1 | CUSTOM board | Library | Physical quantity |
|---|---|---|---|---|---|---|---|---|---|---|
| AccelCalibration2_IIS2ICLX | ☒ | ☒ | ☒ | ☒ | ☐ | ☐ | ☐ | ☒ | MotionAC2 | Acc |
| AccelerometerCalibration | ☐ | ☒ | ☒ | ☒ | ☒ | ☒ | ☒ | ☒ | MotionAC | Acc |
| ActiveTime | ☐ | ☒ | ☒ | ☒ | ☐ | ☒ | ☐ | ☒ | MotionAT | Acc |
| ActivityRecognition | ☐ | ☒ | ☒ | ☒ | ☐ | ☒ | ☐ | ☒ | MotionAR | Acc |
| ActivityRecognitionWrist | ☐ | ☒ | ☒ | ☒ | ☐ | ☒ | ☐ | ☒ | MotionAW | Acc |
| AirplaneDetection | ☐ | ☒ | ☒ | ☒ | ☐ | ☒ | ☐ | ☒ | MotionAD | Acc, Press, Temp |
| AirQuality_SGP40 | ☒ | ☒ | ☒ | ☒ | ☐ | ☐ | ☐ | ☒ | EnvAQ | Hum, Temp, Press, Gas |
| ApproachLeave_STHS34PF80 | ☒ | ☒ | ☒ | ☒ | ☐ | ☐ | ☐ | ☒ | InfraredAL | Temp |
| CarryPosition | ☐ | ☒ | ☒ | ☒ | ☐ | ☒ | ☐ | ☒ | MotionCP | Acc |
| DataLogFusion | ☒ | ☒ | ☒ | ☒ | ☒ | ☒ | ☒ | ☒ | MotionFX | Acc, Gyr, Mag |
| DynamicInclinometer | ☐ | ☒ | ☒ | ☒ | ☒ | ☒ | ☒ | ☒ | MotionDI | Acc, Gyr |
| Ecompass | ☒ | ☒ | ☒ | ☒ | ☒ | ☒ | ☒ | ☒ | MotionEC,<br>MotionMC | Acc, Mag |
| FallDetection | ☐ | ☒ | ☒ | ☒ | ☐ | ☒ | ☐ | ☒ | MotionFD | Acc |
| FitnessActivities | ☐ | ☒ | ☒ | ☒ | ☐ | ☒ | ☐ | ☒ | MotionFA | Acc, Gyr, Press |
| GestureRecognition | ☐ | ☒ | ☒ | ☒ | ☐ | ☒ | ☐ | ☒ | MotionGR | Acc |
| GyroscopeCalibration | ☒ | ☒ | ☒ | ☒ | ☒ | ☒ | ☒ | ☒ | MotionGC | Acc, Gyr |
| HighGLowGFusion | ☒ | ☒ | ☒ | ☒ | ☐ | ☐ | ☒ | ☐ | MotionXLF | Acc |
| HighGLowGFusion_LSM6DSV320X | ☒ | ☒ | ☒ | ☒ | ☐ | ☐ | ☐ | ☒ | MotionXLF | Acc |
| IntensityDetection | ☒ | ☒ | ☒ | ☒ | ☐ | ☒ | ☐ | ☒ | MotionID | Acc |
| MagnetometerCalibration | ☒ | ☒ | ☒ | ☒ | ☒ | ☒ | ☒ | ☒ | MotionMC | Mag |
| Pedometer | ☐ | ☒ | ☒ | ☒ | ☐ | ☒ | ☐ | ☒ | MotionPM | Acc |

| Library Eval Application | NUCLEO-L073RZ | NUCLEO-L152RE | NUCLEO-F401RE | NUCLEO-U575ZI-Q | X-NUCLEO-IKS02A1 | X-NUCLEO-IKS4A1 | X-NUCLEO-IKS5A1 | CUSTOM board | Library | Physical quantity |
|---|---|---|---|---|---|---|---|---|---|---|
| PedometerWrist | ☐ | ☒ | ☒ | ☒ | ☐ | ☒ | ☐ | ☒ | MotionPW, MotionAW | Acc |
| PoseEstimation | ☐ | ☒ | ☒ | ☒ | ☐ | ☒ | ☐ | ☒ | MotionPE | Acc |
| PresenceDetection_STHS34PF80 | ☒ | ☒ | ☒ | ☒ | ☐ | ☐ | ☐ | ☒ | InfraredPD | Temp |
| StandingSittingDesk | ☐ | ☒ | ☒ | ☒ | ☐ | ☒ | ☐ | ☒ | MotionSD | Acc |
| TiltSensing | ☒ | ☒ | ☒ | ☒ | ☒ | ☒ | ☒ | ☒ | MotionTL | Acc |
| TiltSensing2_IIS2ICLX | ☒ | ☒ | ☒ | ☒ | ☐ | ☐ | ☐ | ☒ | MotionTL2 | Acc |
| VerticalContext | ☐ | ☒ | ☒ | ☒ | ☐ | ☒ | ☐ | ☒ | MotionVC | Acc, Press |
| VibrationMonitoring | ☐ | ☐ | ☒ | ☐ | ☒ | ☐ | ☒ | ☐ | MotionSP | Acc |

## 7.1 Application setup checklist

This chapter provides a checklist to guide you in the Library Evaluation Application configuration process. Additionally, the individual points link to sections that provide detailed instructions on how the configuration can be performed with the help of STM32CubeMX tool.

If you're using a hardware configuration that is indicated as supported inTable 5. Library evaluation applications, you can skip the manual configuration and upload the built binaries directly. See for more information. This does not apply to CUSTOM boards.

*Note:* *Keep in mind that the Section 7.3.28: VibrationMonitoring can't be generated or configured by the user with STM32CubeMX and therefore the following checklist does not apply. The source code is, however, available for you to see and modify manually, if needed. Refer to the application's description to learn more.*

1. Configure MCU peripherals required for library evaluation applications

   TIMER – refer to Section 4.7: TIMER

   SPI / I2C – refer to Section 4.5: I2C / SPI

   U(S)ART – refer to Section 4.8: U(S)ART

   DMA – refer to Section 4.3: DMA

   GPIO output for LED – refer to Section 4.4: GPIO

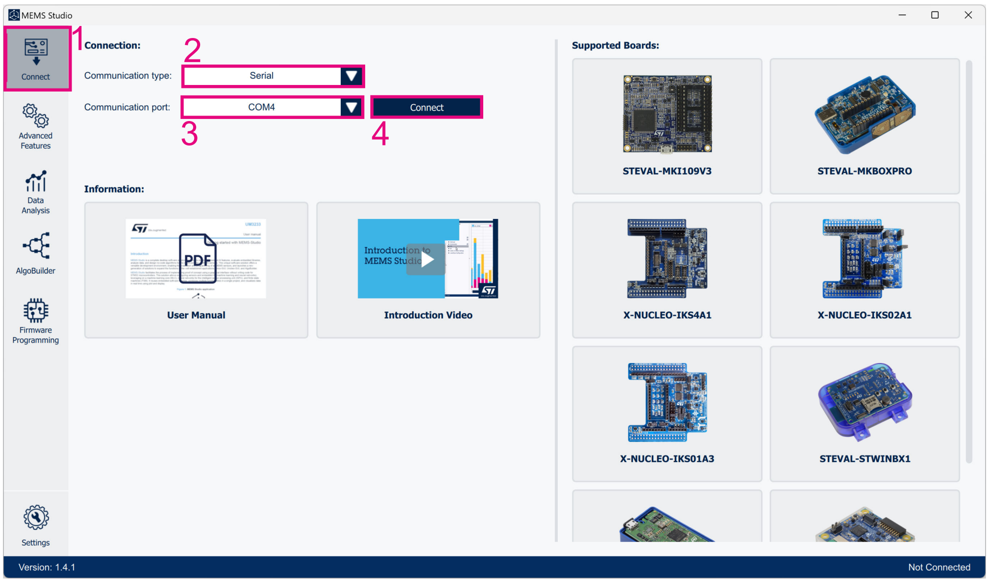
   RTC – refer to Section 4.6: RTC

   CRC – refer to Section 4.2: CRC
2. In `Component Selector` select application, parts, board extensions and middlewares required by the application – depending on your HW configuration refer to Section 4.10: Selecting components for CUSTOM boards or Section 4.9: Selecting components for configurations utilizing X-NUCLEO boards respectively
3. Set `Platform settings` – select what peripherals that are going to be used by the application (peripherals need to be already configured at this point), set application specific `Parameter Settings` (if applicable) – refer to Section 4.11: Platform & Parameter settings
4. Choose IDE for which to create the project and then generate it – refer to
5. Build the project and upload the resulting binary file to your board with an IDE selected in previous step
6. Open the MEMS-Studio software and experiment with the application – refer to Section 7.2: Using MEMS-Studio with library evaluation application

## 7.2 Using MEMS-Studio with library evaluation application

Open the MEMS-Studio application if it's not running already and make sure that board running the correct firmware is connected.

Figure 37. **MEMS-Studio Connection page**



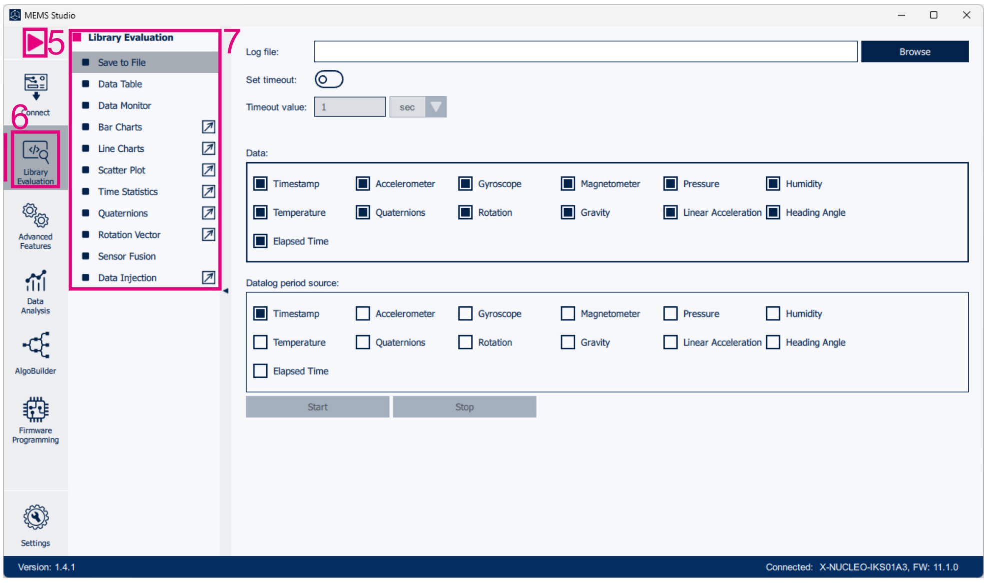
**Step 1.** Click the `Connect` tab

**Step 2.** Set the `Communication type` to `Serial`

**Step 3.** Set the `Communication port` to the one which is used by your board. Refer to Section 3.3: Determining your board's COM port for more information

**Step 4.** Connect to the board by clicking the `Connect` button

Figure 38. **MEMS-Studio Library evaluation**



**Step 5.** Click the start streaming button which will start the data collection and streaming process ▶

**Step 6.** Switch to the `Library Evaluation` tab

**Step 7.** Now you can select how the data is going to be displayed by selecting one of the suggested options. Note that this selection is going to differ based on the application that is running

## 7.3 Detailed description of Applications

### 7.3.1 AccelCalibration2_IIS2ICLX

**Libraries:** MotionAC2

**Hardware:** IIS2ICLK – CUSTOM

**Usage:** The application uses the MotionAC2 library to calibrate the output of IIS2ICLX 2-axis accelerometer. The calibration is facilitated in real-time through offset and scale factor coefficients, which you can adjust from the MEMS Studio PC application.

### 7.3.2 AccelerometerCalibration

**Libraries:** MotionAC

**Hardware:** Accelerometer – Nucleo + X-NUCLEO (IKS02A1 / IKS4A1 / IKS5A1) or CUSTOM

**Usage:** The application uses the MotionAC library to calibrate output of any supported accelerometer. The calibration is facilitated in real-time through offset and scale factor coefficients, which you can adjust from the MEMS Studio PC application.

### 7.3.3 ActiveTime

**Libraries:** MotionAT

**Hardware:** Accelerometer – Nucleo + X-NUCLEO-IKS4A1 or CUSTOM

**Usage:** The application uses data from a connected accelerometer to determine active time of the sensor with MotionAT library. The number of active seconds is displayed in real-time in the MEMS Studio application.

### 7.3.4 ActivityRecognition

**Libraries:** MotionAR

**Hardware:** Accelerometer – Nucleo + X-NUCLEO-IKS4A1 or CUSTOM

**Usage:** The application processes accelerometer data with MotionAR library to recognize the activity being performed. It can distinguish the following: no activity, stationary, walking, fast walking, jogging, biking, and driving. The real-time activity is displayed in MEMS Studio application.

### 7.3.5 ActivityRecognitionWrist

**Libraries:** MotionAW

**Hardware:** Accelerometer – Nucleo + X-NUCLEO-IKS4A1 or CUSTOM

**Usage:** The application collects accelerometer data captured with the device mounted on the user's wrist. The data is processed with the MotionAW library which recognizes what activity is being performed - the following activities are supported: no activity, stationary, standing, sitting, lying, walking, fast walking, jogging, and biking. The real-time data is displayed in the MEMS Studio application.

### 7.3.6 AirplaneDetection

**Libraries:** MotionAD

**Hardware:** Accelerometer + Temperature + Pressure – Nucleo + X-NUCLEO-IKS4A1 or CUSTOM

**Usage:** The application uses the MotionAD library to detect airplane state - on land, take off and landing using accelerometer, pressure and temperature sensors. Real-time data can be displayed in the MEMS Studio application.

### 7.3.7 AirQuality_SGP40

**Libraries:** EnvAQ (Gas Index Algorithm by Sensirion)

**Hardware:** SGP40 + Temperature + Humidity + Pressure – CUSTOM

**Usage:** The application uses the EnvAQ - Gas Index Algorithm library by Sensirion to compute real-time air quality index using SGP40 gas, SHT40AD1B humidity and temperature, and LPS22DF pressure sensors. Data is displayed in the MEMS Studio application.

### 7.3.8 ApproachLeave_STHS34PF80

**Libraries:** InfraredAL

**Hardware:** STHS34PF80 – CUSTOM

**Usage:** The application uses the InfraredAL library to detect human presence using the STHS34PF80 TMOS infrared temperature component. Real-time human presence detection flag and related confidence are displayed in the MEMS Studio application.

### 7.3.9 CarryPosition

**Libraries:** MotionCP

**Hardware:** Accelerometer – Nucleo + X-NUCLEO-IKS4A1 or CUSTOM

**Usage:** The application uses the MotionCP library to determine how the user is carrying the device using data from the connected accelerometer. It can distinguish the following positions: on desk, in hand, near head, shirt pocket, trouser pocket, swinging arm, and jacket pocket which are displayed in real-time in the MEMS Studio application.

### 7.3.10 DataLogFusion

**Libraries:** MotionFX

**Hardware:** Accelerometer + Gyroscope + Magnetometer – Nucleo + X-NUCLEO (IKS02A1 / IKS4A1 / IKS5A1) or CUSTOM

**Usage:** The application uses the MotionFX library for real-time motion data fusion using accelerometer, gyroscope, and magnetometer sensors. It uses the MEMS Studio to illustrate rotation on a teapot model. The PC application is also used to facilitate gyroscope bias and magnetometer hard iron calibration.

### 7.3.11 DynamicInclinometer

**Libraries:** MotionDI

**Hardware:** aaAccelerometer + Gyroscope – Nucleo + X-NUCLEO (IKS02A1 / IKS4A1 / IKS5A1) or CUSTOM

**Usage:** The application uses the MotionDI library to determine real-time inclination from accelerometer and gyroscope data. It uses MEMS Studio application which demonstrates the rotation on a vehicle model and facilitates accelerometer and gyroscope bias calibration.

### 7.3.12 ECompass

**Libraries:** MotionEC, MotionMC

**Hardware:** Accelerometer + Magnetometer – Nucleo + X-NUCLEO (IKS02A1 / IKS4A1 / IKS5A1) or CUSTOM

**Usage:** The application uses the MotionEC library to determine real-time device orientation and movement status using accelerometer and magnetometer sensors. It outputs device orientation (quaternions, Euler angles), device rotation, gravity vector, and linear acceleration to the MEMS Studio application which is used for visualization.

### 7.3.13 FallDetection

**Libraries:** MotionFD

**Hardware:** Accelerometer – Nucleo + X-NUCLEO-IKS4A1 or CUSTOM

**Usage:** The application uses the MotionFD library to detect fall events in real-time using an accelerometer and a pressure sensor. The MEMS Studio application displays real-time information about detected fall events.

### 7.3.14 FitnessActivities

**Libraries:** MotionFA

**Hardware:** Accelerometer + Gyroscope + Pressure – Nucleo + X-NUCLEO-IKS4A1 or CUSTOM

**Usage:** The application uses the MotionFA library to count repetitions of selected fitness activities: biceps curl, squat, and push-up using accelerometer, gyroscope, and pressure components. The real-time counts are displayed in the MEMS Studio application.

### 7.3.15 GestureRecognition

**Libraries:** MotionGR

**Hardware:** Accelerometer – Nucleo + X-NUCLEO-IKS4A1 or CUSTOM

**Usage:** The application uses the MotionGR library to recognize gestures: pick up, glance, and wake up using data from an accelerometer. The real-time gesture information is displayed in the MEMS Studio application.

### 7.3.16 GyroscopeCalibration

**Libraries:** MotionGC

**Hardware:** Accelerometer + Gyroscope – Nucleo + X-NUCLEO (IKS02A1 / IKS4A1 / IKS5A1) or CUSTOM

**Usage:** The application uses the MotionGC library to perform gyroscope calibration through angular zero-rate level coefficients (offset). The calibration is facilitated by the MEMS Studio application.

### 7.3.17 HighGLowGFusion / HighGLowGFusion_LSM6DSV320X

**Libraries:** MotionXLF

**Hardware:**

- HighGLowGFusion – Nucleo + X-NUCLEO-IKS5A1
- HighGLowGFusion_LSM6DSV320X – CUSTOM + LSM6DSV320X

**Usage:** The application utilizes the MotionXLF library to combine the complementary features of the low-g accelerometer core (which offers high resolution but a limited dynamic range) and the high-g accelerometer core (which provides a high dynamic range but limited resolution). The library fuses the data from both cores, and the application displays the resulting acceleration in real time within the MEMS-Studio application.

### 7.3.18 IntensityDetection

**Libraries:** MotionID

**Hardware:** Accelerometer – Nucleo + X-NUCLEO-IKS4A1 or CUSTOM

**Usage:** The application uses the MotionID library to detect motion intensity in a range from 0 to 10 using data from an accelerometer mounted on user's wrist. It can distinguish activities such as on desk, hand on bed/couch pillow, light movements, biking, typing/writing, high intensity typing/slow walking, washing hands/walking, fast walking, fast walking/jogging, fast jogging/brushing teeth, and sprinting. The resulting motion intensity indices and interpretations are displayed in real-time in the MEMS Studio application.

### 7.3.19 MagnetometerCalibration

**Libraries:** MotionMC

**Hardware:** Magnetometer – Nucleo + X-NUCLEO (IKS02A1 / IKS4A1 / IKS5A1) or CUSTOM

**Usage:** The application uses the MotionMC library to perform magnetometer calibration using hard iron (HI) and scale factor coefficients. The calibration is facilitated by the MEMS Studio application.

### 7.3.20 Pedometer

**Libraries:** MotionPM

**Hardware:** Accelerometer – Nucleo + X-NUCLEO-IKS4A1 or CUSTOM

**Usage:** The application uses the MotionPM library and an accelerometer to count the number of steps performed by user. The count is displayed in real-time in the MEMS Studio application.

### 7.3.21 PedometerWrist

**Libraries:** MotionAW, MotionPW

**Hardware:** Accelerometer – Nucleo + X-NUCLEO-IKS4A1 or CUSTOM

**Usage:** The application uses the MotionPW library to count the number of steps performed by the user wearing an accelerometer on their wrist. The count is displayed in real-time in the MEMS Studio application.

### 7.3.22 PoseEstimation

**Libraries:** MotionPE

**Hardware:** Accelerometer – Nucleo + X-NUCLEO-IKS4A1 or CUSTOM

**Usage:** The application uses the MotionPE library to estimate the user's pose (sitting, standing, lying down) in real-time from an accelerometer's data. The current pose is displayed in the MEMS Studio application.

### 7.3.23 PresenceDetection_STHS34PF80

**Libraries:** InfraredPD

**Hardware:** STHS34PF80 – CUSTOM

**Usage:** The application uses the InfraredPD library with the STHS34PF80 TMOS infrared temperature sensor to detect object presence and motion. The ambient temperature, object temperature, compensated object temperature, and the change rate of compensated object temperature in LSB format is displayed in real-time in the MEMS Studio application. Apart from that object motion detection and presence detection flags are displayed.

### 7.3.24 StandingSittingDesk

**Libraries:** MotionSD

**Hardware:** Accelerometer – Nucleo + X-NUCLEO-IKS4A1 or CUSTOM

**Usage:** The application uses the MotionSD library with data from an accelerometer to determine whether the user is working at a sitting desk or standing desk. The MEMS Studio application shows whether the user is at a sitting or a standing desk at the moment.

### 7.3.25 TiltSensing

**Libraries:** MotionTL

**Hardware:** Accelerometer – Nucleo + X-NUCLEO (IKS02A1 / IKS4A1 / IKS5A1) or CUSTOM

**Usage:** The application uses the MotionTL library to provide real-time tilt angle information using the LSM6DSV16X component. It also performs accelerometer 6-position calibration. Both the calibration and the tilt angle visualization are facilitated by the MEMS Studio application.

### 7.3.26 TiltSensing2_IIS2ICLX

**Libraries:** MotionTL2

**Hardware:** IIS2ICLX – CUSTOM

**Usage:** The application uses the MotionTL2 library to provide real-time tilt angle information using the IIS2ICLX 2-axis accelerometer. It also performs 2-axis accelerometer 4-position calibration. Both the calibration and the tilt angle visualization are facilitated by the MEMS Studio application.

### 7.3.27 VerticalContext

**Libraries:** MotionVC

**Hardware:** Accelerometer + Pressure – Nucleo + X-NUCLEO-IKS4A1 or CUSTOM

**Usage:** The application uses the MotionVC library to detect vertical movement and changes in altitude using accelerometer and pressure sensors. It can distinguish between stairs, elevator, and escalator movements. The real-time vertical context information is displayed in the MEMS Studio application.

### 7.3.28 VibrationMonitoring

**Libraries:** MotionSP

**Hardware:** Accelerometer – Nucleo + X-NUCLEO (IKS02A1 / IKS05A1)

**Usage:** The application uses the MotionSP library to analyze data coming from an accelerometer in frequency domain. The resulting FFT is displayed in the MEMS-Studio application, where one figure is dedicated to each exis. You can use the utility to adjust some parameters of the FFT (number of samples, FFT window type, high-pass filter and DC nulling).

*Note:* *This application can't be generated by the user. Projects for supported IDEs are, however, provided so that you can explore the source code and build the application. You can take advantage of Compiled binaries to avoid having to build the application yourself.*

## Revision history

Table 6. **Document revision history**

| Date | Revision | Changes |
|---|---|---|
| 19-Feb-2015 | 1 | Initial release |
| 17-Jun-2015 | 2 | Add support to L1 |
| 30-Sep-2015 | 3 | Removed Middlewares folder and added support for L4 |
| 21-Dec-2015 | 4 | Updated Figure 2: "X-CUBE-MEMS1 software architecture"<br>Updated Section 2.5: "Sample application description" |
| 26-Apr-2016 | 5 | Updated Section 2.1: Overview |
| 04-Nov-2016 | 6 | Text and formatting changes throughout document<br>Added new board compatibility information<br>Added Section 3.1.3: "X-NUCLEO-IKS01A2 expansion board" |
| 22-Mar-2017 | 7 | Updated Section "Introduction", Section 2.1: "Overview", Section 2.2: "Architecture", Section 2.3: "Folder structure", Section 2.5: "DataLog application", Section 2.6: "Unicleo-GUI data logging utility", Section 3.2.2: "Software setup" and Section 3.2.5: "Unicleo-GUI setup".<br>Added Section 2.7: "DataLogExtended application", Section 2.8: "DataLogTerminal application", Section 2.8: "DataLogTerminal application", Section 2.9: "FIFO mode application for pressure sensor", Section 2.10: "6D orientation application for accelerometer sensor", Section 2.11: "FIFO continuous mode application for gyroscope sensor", Section 2.12: "FIFO low power mode application for accelerometer sensor", Section 2.13: "FIFO mode application for gyroscope sensor", Section 2.14: "Free fall detection application for accelerometer sensor", Section 2.15: "Multiple event application for accelerometer sensor", Section 2.16: "Pedometer application for accelerometer sensor", Section 2.17: "Self-test application for accelerometer and gyroscope sensors", Section 2.18: "Single tap and double tap detection for accelerometer sensor", Section 2.19: "Tilt detection for accelerometer sensor", Section 2.20: "Wake up detection for accelerometer sensor" and Section 2.21: "Sample applications for motion libraries". |
| 20-Sep-2017 | 8 | Updated Introduction, Section 2.2 Architecture and Figure 2. X-CUBE-MEMS1 software architecture |
| 10-Jul-2018 | 9 | Updated Introduction, Section 2.1 Overview, Section 2.2 Architecture, Section 2.3 Folder structure and Section 2.5 DataLogExtended application.<br>Removed references to X-NUCLEO-IKS01A1 throughout the document. |
| 22-Feb-2019 | 10 | Updated Introduction, Figure 2. X-CUBE-MEMS1 software architecture, Figure 3. X-CUBE-MEMS1 package folder structure and Section 2.5 DataLogExtended application.<br>Added Section 2.20 Temperature detection and Section 3.1.3 X-NUCLEO-IKS01A3 expansion board.<br>Added X-NUCLEO-IKS01A3 expansion board compatibility information. |
| 05-Jun-2019 | 11 | Updated Introduction, Section 2.1 Overview and Section 2.2 Architecture.<br>Added NUCLEO-L073RZ compatibility information. |
| 25-Nov-2019 | 12 | Updated Introduction, Section 1.1 Overview and Figure 1. X-CUBE-MEMS1 software architecture.<br>Added X-NUCLEO-IKS02A1 expansion board compatibility information.<br>Added Section 1.22 Microphone FFT, Section 1.23 Sound Meter and Section 2.1.4 X-NUCLEOIKS02A1 expansion board. |
| 18-May-2020 | 13 | Updated Introduction, Section 1.1 Overview and Section 1.2 Architecture.<br>Added references to MotionAD and MotionDI software libraries. |
| 23-Jul-2020 | 14 | Updated Introduction and Section 1.1 Overview.<br>Added references to LPS33K MEMS pressure sensor and IIS2ICLX digital inclinometer. |
| 03-Nov-2022 | 15 | Updated *Section 1.1 Overview, Section 1.5 DataLogExtended application*, and *Section 2.3 Hardware setup*.<br>Replaced NUCLEO-L476RG with NUCLEO-U575ZI-Q. |
| 18-Oct-2023 | 16 | Added X-NUCLEO-IKS4A1 expansion board compatibility information. |
| 05-Jul-2024 | 17 | Updated *Section Introduction, Section 1.1: Overview, Section 1.2: Architecture, Section 1.3: Folder* |

| Date | Revision | Changes |
|---|---|---|
| | | *structure*, DataLogExtended application, DataLogExtended application, MEMS-Studio data logging utility, Software setup and *Section 3.2: MEMS-Studio*.<br>Removed Section 1.20 Microphone FFT and Section 1.21 Sound Meter. |
| 16-Dec-2024 | 18 | Complete rework of the whole document.<br>Added details about package use in STM32CubeMX. |
| 03-Mar-2025 | 19 | Added IIS2DULPX and LSM6DSV80X to Supported MEMS devices, updated X-NUCLEO pinout diagram in |
| 05-May-2025 | 20 | Added MotionXLF library and added support for LSM6DSV320X, added<br>*Section 7.3.17: HighGLowGFusion / HighGLowGFusion_LSM6DSV320X application* |
| 11-Sep-2025 | 21 | Added X-NUCLEO-IKS5A1 to supported boards, added ISM6HG256X and ISM330IS to the list of supported MEMS devices in Table 1. Supported MEMS devices, Added X-NUCLEO-IKS5A1 to Table 3. Sensor Evaluation Examples, added X-NUCLEO-IKS5A1 to relevant examples and applications in Section 5: Sensor Evaluation Examples and Section 7: Library Evaluation Applications respectively.<br>Removed X-NUCLEO-IKS01A3 from supported boards, Updated screenshots in Section 4: STM32 configuration, removed the board from `Connection` column in Table 1. Supported MEMS devices, removed from Table 3. Sensor Evaluation Examples, and Section 7: Library Evaluation Applications<br>Added Section 4.3.2: Configuring Linked-List capable DMA with STM32CubeMX. |

## Contents

## List of figures

**IMPORTANT NOTICE – READ CAREFULLY**

STMicroelectronics NV and its subsidiaries (“ST”) reserve the right to make changes, corrections, enhancements, modifications, and improvements to ST products and/or to this document at any time without notice.

In the event of any conflict between the provisions of this document and the provisions of any contractual arrangement in force between the purchasers and ST, the provisions of such contractual arrangement shall prevail.

The purchasers should obtain the latest relevant information on ST products before placing orders. ST products are sold pursuant to ST’s terms and conditions of sale in place at the time of order acknowledgment.

The purchasers are solely responsible for the choice, selection, and use of ST products and ST assumes no liability for application assistance or the design of the purchasers’ products.

No license, express or implied, to any intellectual property right is granted by ST herein.

Resale of ST products with provisions different from the information set forth herein shall void any warranty granted by ST for such product.

If the purchasers identify an ST product that meets their functional and performance requirements but that is not designated for the purchasers' market segment, the purchasers shall contact ST for more information.

ST and the ST logo are trademarks of ST. For additional information about ST trademarks, refer to www.st.com/trademarks. All other product or service names are the property of their respective owners.

Information in this document supersedes and replaces information previously supplied in any prior versions of this document.