Getting started with the STMicroelectronics X-CUBE-MEMS1 software package for STM32CubeMX

Introduction

This document provides the guidelines to configure and use the X-CUBE-MEMS1 software package V7.1.0 for STM32CubeMX (minimum required version V5.6.0). The document contains a description of the provided sample applications, a description of the steps required to configure a generic project using the X-NUCLEO-IKS01A2 or the X-NUCLEO-IKS01A3 or the X-NUCLEO-IKS02A1 expansion board with a Nucleo board or several MEMS components with custom boards, as well as a description of the steps to configure and use the sample applications provided in the package.

Information and documentation related to the MEMS components, the X-NUCLEO-IKS01A2, the X-NUCLEO-IKS01A3 and the X-NUCLEO-IKS02A1 expansion boards and the ST expansion software for motion MEMS and environmental sensors are available on www.st.com.

Contents

List of figures

1 Acronyms and abbreviations

Table 1: list of acronyms

Acronym	Description
MEMS	Micro-ElectroMechanical Systems
HAL	Hardware Abstraction Layer
I2C	Inter-Integrated Circuit
IOT	Internet Of Things
IP	Internet Protocol
LAN	Local Area Network
NVIC	Nested Vectored Interrupt Controller
ODR	Output Data Rate
PCB	Printed Circuit Board
RTC	Real Time Clock
RTOS	Real Time Operating System
SPI	Serial Peripheral Interface
U(S)ART	Universal (Synchronous) Asynchronous Receiver Transmitter
USB	Universal Serial BUS
TCP	Transmission Control Protocol

2 What is STM32Cube?

STM32Cube™ represents an original initiative by STMicroelectronics to ease developers' life by reducing development effort, time and cost. STM32Cube covers the STM32 portfolio. Version 1.x of STM32Cube includes:

- STM32CubeMX, a graphical software configuration tool that allows the generation of C initialization code using graphical wizards.
- A comprehensive embedded software platform, delivered per series (such as the STM32CubeF4 for STM32F4 series).
    - STM32Cube HAL, an STM32 abstraction layer embedded software, ensuring maximized portability across the STM32 portfolio;
    - a consistent set of middleware components, such as RTOS, USB, TCP/IP, graphics;
    - all embedded software utilities, including a full set of examples.

3 License

The software provided in this package is licensed under Software License Agreement SLA0077.

4 Sample Applications Description

In this section, a short overview of the sample applications included in the X-CUBE-MEMS1 pack is provided.
The sample applications:

- are ready-to-use projects that can be generated through the STM32CubeMX for any Nucleo board and using the X-NUCLEO-IKS01A2 or the X-NUCLEO-IKS01A3 expansion board.
- are ready-to-use projects that can be generated through the STM32CubeMX for any board equipped with an STM32 MCU and using the several supported MEMS components.
- show the users how to use the APIs of the several MEMS components to correctly initialize and use the ST MEMS devices.

4.1 IKS01A3_DataLogTerminal

This application shows how to use the X-NUCLEO-IKS01A3 to send sensor data from a STM32 Nucleo board using UART to a connected PC and display it on generic applications like Tera Term. After connection has been established, the user can view the data from all on-board environment sensors (temperature, humidity and pressure sensors) and all on-board inertial sensors (accelerometer, gyroscope and magnetometer sensors) using a hyper terminal. The application serial settings can be configured by user changing the settings of USART2 in the CubeMX GUI.

4.2 IKS01A3_LIS2DW12_6DOrientation

This application shows how to use the LIS2DW12 component of the X-NUCLEO-IKS01A3 expansion board to find out the 6D orientation, send data from a STM32 Nucleo board using UART to a connected PC and display it on generic applications like Tera Term. After connection has been established, the user can rotate the board to change the 6D orientation and then view the data using a hyper terminal or just push the user button to display the current 6D orientation. The application serial settings can be configured by user changing the settings of USART2 in the CubeMX GUI.

4.3 IKS01A3_LIS2DW12_SelfTest

This application shows how to use the LIS2DW12 component of the X-NUCLEO-IKS01A3 expansion board to test the accelerometer operation mode, send data from a STM32 Nucleo board using UART to a connected PC and display it on generic applications like Tera Term. After connection has been established, the user can push the user button to launch the self-test and then view the data using a hyper terminal. The application serial settings can be configured by user changing the settings of USART2 in the CubeMX GUI.

4.4 IKS01A3_LIS2DW12_WakeUp

This application shows how to detect the wake up event using the LIS2DW12 component of the X-NUCLEO-IKS01A3 expansion board and a STM32 Nucleo board. After application starts, the user can try to touch the STM32 Nucleo board; when the wake up event is detected, the Nucleo board LED is switched on for a while. The Nucleo board user button can be used to enable/disable the wake up detection feature.

4.5 IKS01A3_LIS2MDL_SelfTest

This application shows how to use the LIS2MDL component of the X-NUCLEO-IKS01A3 expansion board to test the magnetometer operation mode, send data from a STM32 Nucleo board using UART to a connected PC and display it on generic applications like Tera Term. After connection has been established, the user can push the user button to launch the self-test and then view the data using a hyper terminal. The application serial settings can be configured by user changing the settings of USART2 in the CubeMX GUI.

4.6 IKS01A3_LPS22HH_FIFOMode

This application shows how to use the LPS22HH component of the X-NUCLEO-IKS01A3 to store pressure and temperature data in FIFO mode, send data from a STM32 Nucleo board using UART to a connected PC and display it on generic applications like Tera Term. After connection has been established, press the user button to store pressure and temperature data in the FIFO mode and then view the data using a hyper terminal. The application serial settings can be configured by user changing the settings of USART2 in the CubeMX GUI.

4.7 IKS01A3_LSM6DSO_6DOrientation

This application shows how to use the LSM6DSO component of the X-NUCLEO-IKS01A3 expansion board to find out the 6D orientation, send data from a STM32 Nucleo board using UART to a connected PC and display it on generic applications like Tera Term. After connection has been established, the user can rotate the board to change the 6D orientation and then view the data using a hyper terminal or just push the user button to display the current 6D orientation. The application serial settings can be configured by user changing the settings of USART2 in the CubeMX GUI.

4.8 IKS01A3_LSM6DSO_FIFOContinuousMode

This application shows how to use the LSM6DSO component of the X-NUCLEO-IKS01A3 expansion board to store gyroscope data in FIFO continuous mode, send data from a STM32 Nucleo board using UART to a connected PC and display it on generic applications like Tera Term. After connection has been established, the user can push the user button to launch the FIFO demo in continuous mode and then view the data using a hyper terminal. By pressing again the STM32 Nucleo board user button, FIFO continuous mode changes into FIFO bypass mode. If you press the user button once again, the FIFO demo restarts in continuous mode and so on. The application serial settings can be configured by user changing the settings of USART2 in the CubeMX GUI.

4.9 IKS01A3_LSM6DSO_FIFOMode

This application shows how to use the LSM6DSO component of the X-NUCLEO-IKS01A3 expansion board to store gyroscope data in FIFO mode, send data from a STM32 Nucleo board using UART to a connected PC and display it on generic applications like Tera Term. After connection has been established, the user can push the user button to launch the FIFO mode demo and then view the data using a hyper terminal; press the user button to launch again the FIFO mode demo. The application serial settings can be configured by user changing the settings of USART2 in the CubeMX GUI.

4.10 IKS01A3_LSM6DSO_FreeFall

This application shows how to detect the free fall event using the LSM6DSO component of the X-NUCLEO-IKS01A3 expansion board and a STM32 Nucleo board. After application starts, the user can try to let the STM32 Nucleo board falling; when the free fall event is detected, the Nucleo board LED is switched on for a while. The Nucleo board user button can be used to

enable/disable the free fall detection feature.

4.11 IKS01A3_LSM6DSO_Pedometer

This application shows how to use the LSM6DSO component of the X-NUCLEO-IKS01A3 expansion board to count steps, send data from a STM32 Nucleo board using UART to a connected PC and display it on generic applications like Tera Term. After connection has been established, the user can shake the board to simulate the steps and then view the data using a hyper terminal or can push the user button to reset the step counter. The application serial settings can be configured by user changing the settings of USART2 in the CubeMX GUI.

4.12 IKS01A3_LSM6DSO_SelfTest

This application shows how to use the LSM6DSO component of the X-NUCLEO-IKS01A3 expansion board to test accelerometer and gyroscope operation mode, send data from a STM32 Nucleo board using UART to a connected PC and display it on generic applications like Tera Term. After connection has been established, the user can push the user button to launch the self-test and then view the data using a hyper terminal. The application serial settings can be configured by user changing the settings of USART2 in the CubeMX GUI.

4.13 IKS01A3_LSM6DSO_SingleDoubleTap

This application shows how to detect the single and double tap events using the LSM6DSO component of the X-NUCLEO-IKS01A3 expansion board and a STM32 Nucleo board. After application starts, the user can try to tap the STM32 Nucleo board; when the single tap event is detected, the Nucleo board LED is switched on for a while. The user can press the user button to pass from the single tap detection to the double tap detection feature; when the double tap event is detected, the LED is switched on twice for a while. The user can press again the Nucleo board user button to disable the single/double tap detection feature and so on.

4.14 IKS01A3_LSM6DSO_Tilt

This application shows how to detect the tilt event using the LSM6DSO component of the X-NUCLEO-IKS01A3 expansion board and a STM32 Nucleo board. After application starts, the user can try to tilt the STM32 Nucleo board; when the tilt event is detected, the Nucleo board LED is switched on for a while. The Nucleo board user button can be used to enable/disable the tilt detection feature.

4.15 IKS01A3_LSM6DSO_WakeUp

This application shows how to detect the wake up event using the LSM6DSO component of the X-NUCLEO-IKS01A3 expansion board and a STM32 Nucleo board. After application starts, the user can try to touch the STM32 Nucleo board; when the wake up event is detected, the Nucleo board LED is switched on for a while. The Nucleo board user button can be used to enable/disable the wake up detection feature.

4.16 IKS01A3_STTS751_TemperatureLimit

This application shows how to measure temperature and detect exceeding of temperature limits using the STTS751 component of the X-NUCLEO-IKS01A3 expansion board and a STM32 Nucleo board. After application starts, the user can heat up or cool down the board and view the data using a hyper terminal. The application serial settings can be configured by user changing the settings of USART2 in the CubeMX GUI.

4.17 IKS01A3_IntensityDetection

This application shows how to use the MotionID middleware library developed by STMicroelectronics with the LSM6DSO component of the X-NUCLEO-IKS01A3 expansion board and a STM32 Nucleo board. After application starts, the user can view the data using a

Unicleo-GUI application also developed by STMicroelectronics. The library is designed to be used on STM32 microcontrollers based on the ARM Cortex-M0+, ARM Cortex-M3, ARM Cortex-M4 and ARM Cortex-M7 architecture and is intended for wrist-based devices. The application is able to distinguish motion intensity in a range from 0 to 10 (according to the library indexes) corresponding to the following activities: on desk, hand on bed/couch/cushion, light movements, biking, typing/writing, high intensity typing/slow walking, washing hands/walking, fast walking/jogging, running/brushing teeth, sprinting.

4.18 IKS01A3_DataLogFusion

This application shows how to use the MotionFX middleware library developed by STMicroelectronics with the LSM6DSO and LIS2MDL components of the X-NUCLEO-IKS01A3 expansion board and a STM32 Nucleo board. After application starts, the user can view the data using a Unicleo-GUI application also developed by STMicroelectronics. The library is designed to be used on STM32 microcontrollers based on the ARM Cortex-M0+, ARM Cortex-M3, ARM Cortex-M4 and ARM Cortex-M7 architecture. The application provides real-time motion-sensor data fusion demonstrated on the teapot rotation. It also performs gyroscope bias and magnetometer hard iron calibration.

4.19 IKS01A3_GyroscopeCalibration

This application shows how to use the MotionGC middleware library developed by STMicroelectronics with the LSM6DSO component of the X-NUCLEO-IKS01A3 expansion board and a STM32 Nucleo board. After application starts, the user can view the data using a Unicleo-GUI application also developed by STMicroelectronics. The library is designed to be used on STM32 microcontrollers based on the ARM Cortex-M0+, ARM Cortex-M3, ARM Cortex-M4 and ARM Cortex-M7 architecture. The application provides real-time gyroscope calibration through angular zero-rate level coefficients (offset) used to correct gyroscope data.

4.20 IKS01A3_AccelerometerCalibration

This application shows how to use the MotionAC middleware library developed by STMicroelectronics with the LSM6DSO component of the X-NUCLEO-IKS01A3 expansion board and a STM32 Nucleo board. After application starts, the user can view the data using a Unicleo-GUI application also developed by STMicroelectronics. The library is designed to be used on STM32 microcontrollers based on the ARM Cortex-M3, ARM Cortex-M4 and ARM Cortex-M7 architecture. The application provides real-time accelerometer calibration through offset and scale factor coefficients used to correct accelerometer data.

4.21 IKS01A3_MagnetometerCalibration

This application shows how to use the MotionMC middleware library developed by STMicroelectronics with the LIS2MDL component of the X-NUCLEO-IKS01A3 expansion board and a STM32 Nucleo board. After application starts, the user can view the data using a Unicleo-GUI application also developed by STMicroelectronics. The library is designed to be used on STM32 microcontrollers based on the ARM Cortex-M0+, ARM Cortex-M3, ARM Cortex-M4 and ARM Cortex-M7 architecture. The application provides real-time magnetometer calibration using hard iron (HI) and scale factor coefficients to correct magnetometer data.

4.22 IKS01A3_TiltSensing

This application shows how to use the MotionTL middleware library developed by STMicroelectronics with the LSM6DSO component of the X-NUCLEO-IKS01A3 expansion board and a STM32 Nucleo board. After application starts, the user can view the data using a Unicleo-GUI application also developed by STMicroelectronics. The library is designed to be used on STM32 microcontrollers based on the ARM Cortex-M0+, ARM Cortex-M3, ARM Cortex-M4 and ARM Cortex-M7 architecture. The application provides real-time information about the tilt angles of the user device, i.e. cell phone. It is also able to perform accelerometer

6-position calibration.

4.23 IKS01A3_VerticalContext

This application shows how to use the MotionVC middleware library developed by STMicroelectronics with the LSM6DSO and LPS22HH components of the X-NUCLEO-IKS01A3 expansion board and a STM32 Nucleo board. After application starts, the user can view the data using a Unicleo-GUI application also developed by STMicroelectronics. The library is designed to be used on STM32 microcontrollers based on the ARM Cortex-M3, ARM Cortex-M4 and ARM Cortex-M7 architecture. The application provides real-time information about vertical movement. It is able to detect a change of altitude and distinguish the type of vertical movement: stairs, elevator, and escalator.

4.24 IKS01A3_CarryPosition

This application shows how to use the MotionCP middleware library developed by STMicroelectronics with the LSM6DSO component of the X-NUCLEO-IKS01A3 expansion board and a STM32 Nucleo board. After application starts, the user can view the data using a Unicleo-GUI application also developed by STMicroelectronics. The library is designed to be used on STM32 microcontrollers based on the ARM Cortex-M3, ARM Cortex-M4 and ARM Cortex-M7 architecture. The application provides real-time information about how the user is carrying a device (i.e. cell phone). It is able to distinguish the following positions: on desk, in hand, near head, shirt pocket, trouser pocket, swinging arm and jacket pocket.

4.25 IKS01A3_ECompass

This application shows how to use the MotionEC middleware library developed by STMicroelectronics with the LSM6DSO and LIS2MDL components of the X-NUCLEO-IKS01A3 expansion board and a STM32 Nucleo board. After application starts, the user can view the data using a Unicleo-GUI application also developed by STMicroelectronics. The library is designed to be used on STM32 microcontrollers based on the ARM Cortex-M0+, ARM Cortex-M3, ARM Cortex-M4 and ARM Cortex-M7 architecture. The application provides real-time information about the device orientation and movement status based on data from a device. It provides the following outputs: device orientation (quaternions, Euler angles), device rotation (virtual gyroscope functionality), gravity vector and linear acceleration.

4.26 IKS01A3_GestureRecognition

This application shows how to use the MotionGR middleware library developed by STMicroelectronics with the LSM6DSO component of the X-NUCLEO-IKS01A3 expansion board and a STM32 Nucleo board. After application starts, the user can view the data using a Unicleo-GUI application also developed by STMicroelectronics. The library is designed to be used on STM32 microcontrollers based on the ARM Cortex-M3, ARM Cortex-M4 and ARM Cortex-M7 architecture. The application provides real-time information about the gesture just performed by the user with the device, such as a cell phone. It is able to distinguish the following gestures: pick up, glance, wake up.

4.27 IKS01A2_DataLogTerminal

This application shows how to use the X-NUCLEO-IKS01A2 to send sensor data from a STM32 Nucleo board using UART to a connected PC and display it on generic applications like Tera Term. After connection has been established, the user can view the data from all on-board environment sensors (temperature, humidity and pressure sensors) and all on-board inertial sensors (accelerometer, gyroscope and magnetometer sensors) using a hyper terminal. The application serial settings can be configured by user changing the settings of USART2 in the CubeMX GUI.

4.28 IKS01A2_LPS22HB_FIFOMode

This application shows how to use the LPS22HB component of the X-NUCLEO-IKS01A2 to store pressure and temperature data in FIFO mode, send data from a STM32 Nucleo board using UART to a connected PC and display it on generic applications like Tera Term. After connection has been established, press the user button to store pressure and temperature data in the FIFO mode and then view the data using a hyper terminal. The application serial settings can be configured by user changing the settings of USART2 in the CubeMX GUI.

4.29 IKS01A2_LSM6DSL_6DOrientation

This application shows how to use the LSM6DSL component of the X-NUCLEO-IKS01A2 expansion board to find out the 6D orientation, send data from a STM32 Nucleo board using UART to a connected PC and display it on generic applications like Tera Term. After connection has been established, the user can rotate the board to change the 6D orientation and then view the data using a hyper terminal or just push the user button to display the current 6D orientation. The application serial settings can be configured by user changing the settings of USART2 in the CubeMX GUI.

4.30 IKS01A2_LSM6DSL_FIFOContinuousMode

This application shows how to use the LSM6DSL component of the X-NUCLEO-IKS01A2 expansion board to store gyroscope data in FIFO continuous mode, send data from a STM32 Nucleo board using UART to a connected PC and display it on generic applications like Tera Term. After connection has been established, the user can push the user button to launch the FIFO demo in continuous mode and then view the data using a hyper terminal. By pressing again the STM32 Nucleo board user button, FIFO continuous mode changes into FIFO bypass mode. If you press the user button once again, the FIFO demo restarts in continuous mode and so on. The application serial settings can be configured by user changing the settings of USART2 in the CubeMX GUI.

4.31 IKS01A2_LSM6DSL_FIFOLowPower

This application shows how to use the LSM6DSL component of the X-NUCLEO-IKS01A2 expansion board to store accelerometer data in FIFO continuous mode, send data from a STM32 Nucleo board using UART to a connected PC and display it on generic applications like Tera Term. After connection has been established, the user can push the user button to launch the FIFO low power demo and then view the data using a hyper terminal; afterwards, the component enters sleep mode. The user can press the user button to launch again the FIFO low power demo. The application serial settings can be configured by user changing the settings of USART2 in the CubeMX GUI.

4.32 IKS01A2_LSM6DSL_FIFOMode

This application shows how to use the LSM6DSL component of the X-NUCLEO-IKS01A2 expansion board to store gyroscope data in FIFO mode, send data from a STM32 Nucleo board using UART to a connected PC and display it on generic applications like Tera Term. After connection has been established, the user can push the user button to launch the FIFO mode demo and then view the data using a hyper terminal; press the user button to launch again the FIFO mode demo. The application serial settings can be configured by user changing the settings of USART2 in the CubeMX GUI.

4.33 IKS01A2_LSM6DSL_FreeFall

This application shows how to detect the free fall event using the LSM6DSL component of the X-NUCLEO-IKS01A2 expansion board and a STM32 Nucleo board. After application starts, the user can try to let the STM32 Nucleo board falling; when the free fall event is detected, the

Nucleo board LED is switched on for a while. The Nucleo board user button can be used to enable/disable the free fall detection feature.

4.34 IKS01A2_LSM6DSL_MultiEvent

This application shows how to use the LSM6DSL component of the X-NUCLEO-IKS01A2 expansion board to detect free fall, tap, double tap, tilt, wake up, 6D Orientation and step events, send data from a STM32 Nucleo board using UART to a connected PC and display it on generic applications like Tera Term. After connection has been established, the user can simulate all the events and then view the data using a hyper terminal or can push the user button to enable/disable all hardware features. The application serial settings can be configured by user changing the settings of USART2 in the CubeMX GUI.

4.35 IKS01A2_LSM6DSL_Pedometer

This application shows how to use the LSM6DSL component of the X-NUCLEO-IKS01A2 expansion board to count steps, send data from a STM32 Nucleo board using UART to a connected PC and display it on generic applications like Tera Term. After connection has been established, the user can shake the board to simulate the steps and then view the data using a hyper terminal or can push the user button to reset the step counter. The application serial settings can be configured by user changing the settings of USART2 in the CubeMX GUI.

4.36 IKS01A2_LSM6DSL_SelfTest

This application shows how to use the LSM6DSL component of the X-NUCLEO-IKS01A2 expansion board to test accelerometer and gyroscope operation mode, send data from a STM32 Nucleo board using UART to a connected PC and display it on generic applications like Tera Term. After connection has been established, the user can push the user button to launch the self-test and then view the data using a hyper terminal. The application serial settings can be configured by user changing the settings of USART2 in the CubeMX GUI.

4.37 IKS01A2_LSM6DSL_SingleDoubleTap

This application shows how to detect the single and double tap events using the LSM6DSL component of the X-NUCLEO-IKS01A2 expansion board and a STM32 Nucleo board. After application starts, the user can try to tap the STM32 Nucleo board; when the single tap event is detected, the Nucleo board LED is switched on for a while. The user can press the user button to pass from the single tap detection to the double tap detection feature; when the double tap event is detected, the LED is switched on twice for a while. The user can press again the Nucleo board user button to disable the single/double tap detection feature and so on.

4.38 IKS01A2_LSM6DSL_Tilt

This application shows how to detect the tilt event using the LSM6DSL component of the X-NUCLEO-IKS01A2 expansion board and a STM32 Nucleo board. After application starts, the user can try to tilt the STM32 Nucleo board; when the tilt event is detected, the Nucleo board LED is switched on for a while. The Nucleo board user button can be used to enable/disable the tilt detection feature.

4.39 IKS01A2_LSM6DSL_WakeUp

This application shows how to detect the wake up event using the LSM6DSL component of the X-NUCLEO-IKS01A2 expansion board and a STM32 Nucleo board. After application starts, the user can try to touch the STM32 Nucleo board; when the wake up event is detected, the Nucleo board LED is switched on for a while. The Nucleo board user button can be used to enable/disable the wake up detection feature.

4.40 IKS01A2_IntensityDetection

This application shows how to use the MotionID middleware library developed by STMicroelectronics with the LSM6DSL component of the X-NUCLEO-IKS01A2 expansion board and a STM32 Nucleo board. After application starts, the user can view the data using a Unicleo-GUI application also developed by STMicroelectronics. The library is designed to be used on STM32 microcontrollers based on the ARM Cortex-M0+, ARM Cortex-M3, ARM Cortex-M4 and ARM Cortex-M7 architecture and is intended for wrist-based devices. The application is able to distinguish motion intensity in a range from 0 to 10 (according to the library indexes) corresponding to the following activities: on desk, hand on bed/couch/cushion, light movements, biking, typing/writing, high intensity typing/slow walking, washing hands/walking, fast walking/jogging, running/brushing teeth, sprinting.

4.41 IKS01A2_DataLogFusion

This application shows how to use the MotionFX middleware library developed by STMicroelectronics with the LSM6DSL and LSM303AGR components of the X-NUCLEO-IKS01A2 expansion board and a STM32 Nucleo board. After application starts, the user can view the data using a Unicleo-GUI application also developed by STMicroelectronics. The library is designed to be used on STM32 microcontrollers based on the ARM Cortex-M0+, ARM Cortex-M3, ARM Cortex-M4 and ARM Cortex-M7 architecture. The application provides real-time motion-sensor data fusion demonstrated on the teapot rotation. It also performs gyroscope bias and magnetometer hard iron calibration.

4.42 IKS01A2_GyroscopeCalibration

This application shows how to use the MotionGC middleware library developed by STMicroelectronics with the LSM6DSL component of the X-NUCLEO-IKS01A2 expansion board and a STM32 Nucleo board. After application starts, the user can view the data using a Unicleo-GUI application also developed by STMicroelectronics. The library is designed to be used on STM32 microcontrollers based on the ARM Cortex-M0+, ARM Cortex-M3, ARM Cortex-M4 and ARM Cortex-M7 architecture. The application provides real-time gyroscope calibration through angular zero-rate level coefficients (offset) used to correct gyroscope data.

4.43 IKS01A2_AccelerometerCalibration

This application shows how to use the MotionAC middleware library developed by STMicroelectronics with the LSM6DSL component of the X-NUCLEO-IKS01A2 expansion board and a STM32 Nucleo board. After application starts, the user can view the data using a Unicleo-GUI application also developed by STMicroelectronics. The library is designed to be used on STM32 microcontrollers based on the ARM Cortex-M3, ARM Cortex-M4 and ARM Cortex-M7 architecture. The application provides real-time accelerometer calibration through offset and scale factor coefficients used to correct accelerometer data.

4.44 IKS01A2_MagnetometerCalibration

This application shows how to use the MotionMC middleware library developed by STMicroelectronics with the LSM303AGR component of the X-NUCLEO-IKS01A2 expansion board and a STM32 Nucleo board. After application starts, the user can view the data using a Unicleo-GUI application also developed by STMicroelectronics. The library is designed to be used on STM32 microcontrollers based on the ARM Cortex-M0+, ARM Cortex-M3, ARM Cortex-M4 and ARM Cortex-M7 architecture. The application provides real-time magnetometer calibration using hard iron (HI) and scale factor coefficients to correct magnetometer data.

4.45 IKS01A2_TiltSensing

This application shows how to use the MotionTL middleware library developed by STMicroelectronics with the LSM6DSL component of the X-NUCLEO-IKS01A2 expansion

board and a STM32 Nucleo board. After application starts, the user can view the data using a Unicleo-GUI application also developed by STMicroelectronics. The library is designed to be used on STM32 microcontrollers based on the ARM Cortex-M0+, ARM Cortex-M3, ARM Cortex-M4 and ARM Cortex-M7 architecture. The application provides real-time information about the tilt angles of the user device, i.e. cell phone. It is also able to perform accelerometer 6-position calibration.

4.46 IKS01A2_VerticalContext

This application shows how to use the MotionVC middleware library developed by STMicroelectronics with the LSM6DSL and LPS22HB components of the X-NUCLEO-IKS01A2 expansion board and a STM32 Nucleo board. After application starts, the user can view the data using a Unicleo-GUI application also developed by STMicroelectronics. The library is designed to be used on STM32 microcontrollers based on the ARM Cortex-M3, ARM Cortex-M4 and ARM Cortex-M7 architecture. The application provides real-time information about vertical movement. It is able to detect a change of altitude and distinguish the type of vertical movement: stairs, elevator, and escalator.

4.47 IKS01A2_CarryPosition

This application shows how to use the MotionCP middleware library developed by STMicroelectronics with the LSM6DSL component of the X-NUCLEO-IKS01A2 expansion board and a STM32 Nucleo board. After application starts, the user can view the data using a Unicleo-GUI application also developed by STMicroelectronics. The library is designed to be used on STM32 microcontrollers based on the ARM Cortex-M3, ARM Cortex-M4 and ARM Cortex-M7 architecture. The application provides real-time information about how the user is carrying a device (i.e. cell phone). It is able to distinguish the following positions: on desk, in hand, near head, shirt pocket, trouser pocket, swinging arm and jacket pocket.

4.48 IKS01A2_ECompass

This application shows how to use the MotionEC middleware library developed by STMicroelectronics with the LSM6DSL and LIS2MDL components of the X-NUCLEO-IKS01A2 expansion board and a STM32 Nucleo board. After application starts, the user can view the data using a Unicleo-GUI application also developed by STMicroelectronics. The library is designed to be used on STM32 microcontrollers based on the ARM Cortex-M0+, ARM Cortex-M3, ARM Cortex-M4 and ARM Cortex-M7 architecture. The application provides real-time information about the device orientation and movement status based on data from a device. It provides the following outputs: device orientation (quaternions, Euler angles), device rotation (virtual gyroscope functionality), gravity vector and linear acceleration.

4.49 IKS01A2_GestureRecognition

This application shows how to use the MotionGR middleware library developed by STMicroelectronics with the LSM6DSL component of the X-NUCLEO-IKS01A2 expansion board and a STM32 Nucleo board. After application starts, the user can view the data using a Unicleo-GUI application also developed by STMicroelectronics. The library is designed to be used on STM32 microcontrollers based on the ARM Cortex-M3, ARM Cortex-M4 and ARM Cortex-M7 architecture. The application provides real-time information about the gesture just performed by the user with the device, such as a cell phone. It is able to distinguish the following gestures: pick up, glance, wake up.

4.50 IKS02A1_DataLogTerminal

This application shows how to use the X-NUCLEO-IKS02A1 to send sensor data from a STM32 Nucleo board using UART to a connected PC and display it on generic applications like Tera Term. After connection has been established, the user can view the data from all on-board

inertial sensors (accelerometer, gyroscope and magnetometer sensors) using a hyper terminal. The application serial settings can be configured by user changing the settings of USART2 in the CubeMX GUI.

4.51 DataLogTerminal

This application shows how to use the several MEMS components supported in the CubeMX MEMS Library to send sensor data from a generic custom board based on a STM32 MCU using UART to a connected PC and display it on generic applications like Tera Term. After connection has been established, the user can view the data from all selected environment sensors (temperature, humidity and pressure sensors) and all selected inertial sensors (accelerometer, gyroscope and magnetometer sensors) using a hyper terminal. Every MEMS component can be configured via I2C or via SPI 4-Wires (if it is supported). The application requires to configure in the custom board a GPIO pin for a LED, a GPIO pin for a User Button and a GPIO pin for a UART/USART TX. The application serial settings can be configured by user changing the settings of UART/USART chosen in the CubeMX GUI.

4.52 LIS2DW12_6DOrientation

This application shows how to use the LIS2DW12 component via I2C or via SPI 4-Wires interface to find out the 6D orientation, send data from a generic custom board based on a STM32 MCU using UART to a connected PC and display it on generic applications like Tera Term. After connection has been established, the user can rotate the board to change the 6D orientation and then view the data using a hyper terminal or just push the user button to display the current 6D orientation. The application requires configuring in the custom board a GPIO pin for a LED, a GPIO pin for a User Button and a GPIO pin for a UART/USART TX. The application serial settings can be configured by user changing the settings of UART/USART chosen in the CubeMX GUI.

4.53 LIS2DW12_SelfTest

This application shows how to use the LIS2DW12 component via I2C or via SPI 4-Wires interface to test the accelerometer operation mode, send data a generic custom board based on a STM32 MCU using UART to a connected PC and display it on generic applications like Tera Term. After connection has been established, the user can push the user button to launch the self-test and then view the data using a hyper terminal. The application requires configuring in the custom board a GPIO pin for a LED, a GPIO pin for a User Button and a GPIO pin for a UART/USART TX. The application serial settings can be configured by user changing the settings of UART/USART chosen in the CubeMX GUI.

4.54 LIS2DW12_WakeUp

This application shows how to detect the wake up event using the LIS2DW12 component via I2C or via SPI 4-Wires interface on a generic custom board based on a STM32 MCU. After application starts, the user can try to touch the STM32 Nucleo board; when the wake up event is detected, the board LED is switched on for a while. The board user button can be used to enable/disable the wake up detection feature. The application requires configuring in the custom board a GPIO pin for a LED, a GPIO pin for a User Button and a GPIO pin for a UART/USART TX. The application serial settings can be configured by user changing the settings of UART/USART chosen in the CubeMX GUI.

4.55 LIS2MDL_SelfTest

This application shows how to use the LIS2MDL component via I2C or via SPI 4-Wires interface

to test the magnetometer operation mode, send data from a generic custom board based on a STM32 MCU using UART to a connected PC and display it on generic applications like Tera Term. After connection has been established, the user can push the user button to launch the self-test and then view the data using a hyper terminal. The application requires configuring in the custom board a GPIO pin for a LED, a GPIO pin for a User Button and a GPIO pin for a UART/USART TX. The application serial settings can be configured by user changing the settings of UART/USART chosen in the CubeMX GUI.

4.56 LPS22HB_FIFOMode

This application shows how to use the LPS22HB component via I2C or via SPI 4-Wires interface to store pressure and temperature data in FIFO mode, send data from a generic custom board based on a STM32 MCU using UART to a connected PC and display it on generic applications like Tera Term. After connection has been established, press the user button to store pressure and temperature data in the FIFO mode and then view the data using a hyper terminal. The application requires configuring in the custom board a GPIO pin for a LED, a GPIO pin for a User Button and a GPIO pin for a UART/USART TX. The application serial settings can be configured by user changing the settings of UART/USART chosen in the CubeMX GUI.

4.57 LPS22HH_FIFOMode

This application shows how to use the LPS22HH component via I2C or via SPI 4-Wires interface to store pressure and temperature data in FIFO mode, send data from a generic custom board based on a STM32 MCU using UART to a connected PC and display it on generic applications like Tera Term. After connection has been established, press the user button to store pressure and temperature data in the FIFO mode and then view the data using a hyper terminal. The application requires configuring in the custom board a GPIO pin for a LED, a GPIO pin for a User Button and a GPIO pin for a UART/USART TX. The application serial settings can be configured by user changing the settings of UART/USART chosen in the CubeMX GUI.

4.58 LSM6DSL_6DOrientation

This application shows how to use the LSM6DSL component via I2C or via SPI 4-Wires interface to find out the 6D orientation, send data from a generic custom board based on a STM32 MCU using UART to a connected PC and display it on generic applications like Tera Term. After connection has been established, the user can rotate the board to change the 6D orientation and then view the data using a hyper terminal or just push the user button to display the current 6D orientation. The application requires configuring in the custom board a GPIO pin for a LED, a GPIO pin for a User Button and a GPIO pin for a UART/USART TX. The application serial settings can be configured by user changing the settings of UART/USART chosen in the CubeMX GUI.

4.59 LSM6DSL_FIFOContinuousMode

This application shows how to use the LSM6DSL component via I2C or via SPI 4-Wires interface to store gyroscope data in FIFO continuous mode, send data from a generic custom board based on a STM32 MCU using UART to a connected PC and display it on generic applications like Tera Term. After connection has been established, the user can push the user button to launch the FIFO demo in continuous mode and then view the data using a hyper terminal. By pressing again the user button, FIFO continuous mode changes into FIFO bypass mode. If you press the user button once again, the FIFO demo restarts in continuous mode and so on. The application requires configuring in the custom board a GPIO pin for a LED, a GPIO pin for a User Button and a GPIO pin for a UART/USART TX. The application serial settings can be configured by user changing the settings of UART/USART chosen in the CubeMX GUI.

4.60 LSM6DSL_FIFOLowPower

This application shows how to use the LSM6DSL component via I2C or via SPI 4-Wires interface to store accelerometer data in FIFO continuous mode, send data from a generic custom board based on a STM32 MCU using UART to a connected PC and display it on generic applications like Tera Term. After connection has been established, the user can push the user button to launch the FIFO low power demo and then view the data using a hyper terminal; afterwards, the component enters sleep mode. The user can press the user button to launch again the FIFO low power demo. The application requires configuring in the custom board a GPIO pin for a LED, a GPIO pin for a User Button and a GPIO pin for a UART/USART TX. The application serial settings can be configured by user changing the settings of UART/USART chosen in the CubeMX GUI.

4.61 LSM6DSL_FIFOMode

This application shows how to use the LSM6DSL component via I2C or via SPI 4-Wires interface to store gyroscope data in FIFO mode, send data from a generic custom board based on a STM32 MCU using UART to a connected PC and display it on generic applications like Tera Term. After connection has been established, the user can push the user button to launch the FIFO mode demo and then view the data using a hyper terminal; press the user button to launch again the FIFO mode demo. The application requires configuring in the custom board a GPIO pin for a LED, a GPIO pin for a User Button and a GPIO pin for a UART/USART TX. The application serial settings can be configured by user changing the settings of UART/USART chosen in the CubeMX GUI.

4.62 LSM6DSL_FreeFall

This application shows how to detect the free fall event using the LSM6DSL component via I2C or via SPI 4-Wires interface on a generic custom board based on a STM32 MCU. After application starts, the user can try to let the board falling; when the free fall event is detected, the board LED is switched on for a while. The user button can be used to enable/disable the free fall detection feature. The application requires configuring in the custom board a GPIO pin for a LED, a GPIO pin for a User Button and a GPIO pin for a UART/USART TX. The application serial settings can be configured by user changing the settings of UART/USART chosen in the CubeMX GUI.

4.63 LSM6DSL_MultiEvent

This application shows how to use the LSM6DSL component via I2C or via SPI 4-Wires interface to detect free fall, tap, double tap, tilt, wake up, 6D Orientation and step events, send data from a generic custom board based on a STM32 MCU using UART to a connected PC and display it on generic applications like Tera Term. After connection has been established, the user can simulate all the events and then view the data using a hyper terminal or can push the user button to enable/disable all hardware features. The application requires configuring in the custom board a GPIO pin for a LED, a GPIO pin for a User Button and a GPIO pin for a UART/USART TX. The application serial settings can be configured by user changing the settings of UART/USART chosen in the CubeMX GUI.

4.64 LSM6DSL_Pedometer

This application shows how to use the LSM6DSL component via I2C or via SPI 4-Wires interface to count steps, send data from a generic custom board based on a STM32 MCU using UART to a connected PC and display it on generic applications like Tera Term. After connection has been established, the user can shake the board to simulate the steps and then view the data using a hyper terminal or can push the user button to reset the step counter. The application requires configuring in the custom board a GPIO pin for a LED, a GPIO pin for a User Button and a GPIO pin for a UART/USART TX. The application serial settings can be

configured by user changing the settings of UART/USART chosen in the CubeMX GUI.

4.65 LSM6DSL_SelfTest

This application shows how to use the LSM6DSL component via I2C or via SPI 4-Wires interface to test accelerometer and gyroscope operation mode, send data from a generic custom board based on a STM32 MCU using UART to a connected PC and display it on generic applications like Tera Term. After connection has been established, the user can push the user button to launch the self-test and then view the data using a hyper terminal. The application requires configuring in the custom board a GPIO pin for a LED, a GPIO pin for a User Button and a GPIO pin for a UART/USART TX. The application serial settings can be configured by user changing the settings of UART/USART chosen in the CubeMX GUI.

4.66 LSM6DSL_SingleDoubleTap

This application shows how to detect the single and double tap events using the LSM6DSL component via I2C or via SPI 4-Wires interface on a generic custom board based on a STM32 MCU. After application starts, the user can try to tap the board; when the single tap event is detected, the board LED is switched on for a while. The user can press the user button to pass from the single tap detection to the double tap detection feature; when the double tap event is detected, the LED is switched on twice for a while. The user can press again the user button to disable the single/double tap detection feature and so on. The application requires configuring in the custom board a GPIO pin for a LED, a GPIO pin for a User Button and a GPIO pin for a UART/USART TX. The application serial settings can be configured by user changing the settings of UART/USART chosen in the CubeMX GUI.

4.67 LSM6DSL_Tilt

This application shows how to detect the tilt event using the LSM6DSL component via I2C or via SPI 4-Wires interface on a generic custom board based on a STM32 MCU. After application starts, the user can try to tilt the board; when the tilt event is detected, the board LED is switched on for a while. The user button can be used to enable/disable the tilt detection feature. The application requires configuring in the custom board a GPIO pin for a LED, a GPIO pin for a User Button and a GPIO pin for a UART/USART TX. The application serial settings can be configured by user changing the settings of UART/USART chosen in the CubeMX GUI.

4.68 LSM6DSL_WakeUp

This application shows how to detect the wake up event using the LSM6DSL component via I2C or via SPI 4-Wires interface on a generic custom board based on a STM32 MCU. After application starts, the user can try to touch the board; when the wake up event is detected, the board LED is switched on for a while. The board user button can be used to enable/disable the wake up detection feature. The application requires configuring in the custom board a GPIO pin for a LED, a GPIO pin for a User Button and a GPIO pin for a UART/USART TX. The application serial settings can be configured by user changing the settings of UART/USART chosen in the CubeMX GUI.

4.69 LSM6DSO_6DOrientation

This application shows how to use the LSM6DSO component via I2C or via SPI 4-Wires interface to find out the 6D orientation, send data from a generic custom board based on a STM32 MCU using UART to a connected PC and display it on generic applications like Tera Term. After connection has been established, the user can rotate the board to change the 6D orientation and then view the data using a hyper terminal or just push the user button to display the current 6D orientation. The application requires configuring in the custom board a GPIO pin for a LED, a GPIO pin for a User Button and a GPIO pin for a UART/USART TX. The application serial settings can be configured by user changing the settings of UART/USART chosen in the

CubeMX GUI.

4.70 LSM6DSO_FIFOContinuousMode

This application shows how to use the LSM6DSO component via I2C or via SPI 4-Wires interface to store gyroscope data in FIFO continuous mode, send data from a generic custom board based on a STM32 MCU using UART to a connected PC and display it on generic applications like Tera Term. After connection has been established, the user can push the user button to launch the FIFO demo in continuous mode and then view the data using a hyper terminal. By pressing again the user button, FIFO continuous mode changes into FIFO bypass mode. If you press the user button once again, the FIFO demo restarts in continuous mode and so on. The application requires configuring in the custom board a GPIO pin for a LED, a GPIO pin for a User Button and a GPIO pin for a UART/USART TX. The application serial settings can be configured by user changing the settings of UART/USART chosen in the CubeMX GUI.

4.71 LSM6DSO_FIFOMode

This application shows how to use the LSM6DSO component via I2C or via SPI 4-Wires interface to store gyroscope data in FIFO mode, send data from a generic custom board based on a STM32 MCU using UART to a connected PC and display it on generic applications like Tera Term. After connection has been established, the user can push the user button to launch the FIFO mode demo and then view the data using a hyper terminal; press the user button to launch again the FIFO mode demo. The application requires configuring in the custom board a GPIO pin for a LED, a GPIO pin for a User Button and a GPIO pin for a UART/USART TX. The application serial settings can be configured by user changing the settings of UART/USART chosen in the CubeMX GUI.

4.72 LSM6DSO_FreeFall

This application shows how to detect the free fall event using the LSM6DSO component via I2C or via SPI 4-Wires interface on a generic custom board based on a STM32 MCU. After application starts, the user can try to let the board falling; when the free fall event is detected, the board LED is switched on for a while. The user button can be used to enable/disable the free fall detection feature. The application requires configuring in the custom board a GPIO pin for a LED, a GPIO pin for a User Button and a GPIO pin for a UART/USART TX. The application serial settings can be configured by user changing the settings of UART/USART chosen in the CubeMX GUI.

4.73 LSM6DSO_Pedometer

This application shows how to use the LSM6DSO component via I2C or via SPI 4-Wires interface to count steps, send data from a generic custom board based on a STM32 MCU using UART to a connected PC and display it on generic applications like Tera Term. After connection has been established, the user can shake the board to simulate the steps and then view the data using a hyper terminal or can push the user button to reset the step counter. The application requires configuring in the custom board a GPIO pin for a LED, a GPIO pin for a User Button and a GPIO pin for a UART/USART TX. The application serial settings can be configured by user changing the settings of UART/USART chosen in the CubeMX GUI.

4.74 LSM6DSO_SelfTest

This application shows how to use the LSM6DSO component via I2C or via SPI 4-Wires interface to test accelerometer and gyroscope operation mode, send data from a generic custom board based on a STM32 MCU using UART to a connected PC and display it on generic applications like Tera Term. After connection has been established, the user can push the user button to launch the self-test and then view the data using a hyper terminal. The application requires configuring in the custom board a GPIO pin for a LED, a GPIO pin for a User Button

and a GPIO pin for a UART/USART TX. The application serial settings can be configured by user changing the settings of UART/USART chosen in the CubeMX GUI.

4.75 LSM6DSO_SingleDoubleTap

This application shows how to detect the single and double tap events using the LSM6DSO component via I2C or via SPI 4-Wires interface on a generic custom board based on a STM32 MCU. After application starts, the user can try to tap the board; when the single tap event is detected, the board LED is switched on for a while. The user can press the user button to pass from the single tap detection to the double tap detection feature; when the double tap event is detected, the LED is switched on twice for a while. The user can press again the user button to disable the single/double tap detection feature and so on. The application requires configuring in the custom board a GPIO pin for a LED, a GPIO pin for a User Button and a GPIO pin for a UART/USART TX. The application serial settings can be configured by user changing the settings of UART/USART chosen in the CubeMX GUI.

4.76 LSM6DSO_Tilt

This application shows how to detect the tilt event using the LSM6DSO component via I2C or via SPI 4-Wires interface on a generic custom board based on a STM32 MCU. After application starts, the user can try to tilt the board; when the tilt event is detected, the board LED is switched on for a while. The user button can be used to enable/disable the tilt detection feature. The application requires configuring in the custom board a GPIO pin for a LED, a GPIO pin for a User Button and a GPIO pin for a UART/USART TX. The application serial settings can be configured by user changing the settings of UART/USART chosen in the CubeMX GUI.

4.77 LSM6DSO_WakeUp

This application shows how to detect the wake up event using the LSM6DSO component via I2C or via SPI 4-Wires interface on a generic custom board based on a STM32 MCU. After application starts, the user can try to touch the board; when the wake up event is detected, the board LED is switched on for a while. The board user button can be used to enable/disable the wake up detection feature. The application requires configuring in the custom board a GPIO pin for a LED, a GPIO pin for a User Button and a GPIO pin for a UART/USART TX. The application serial settings can be configured by user changing the settings of UART/USART chosen in the CubeMX GUI.

4.78 STTS751_TemperatureLimit

This application shows how to measure temperature and detect exceeding of temperature limits using the STTS751 component via I2C or via SPI 4-Wires interface on a generic custom board based on a STM32 MCU. After application starts, the user can heat up or cool down the board and view the data using a hyper terminal. The application requires configuring in the custom board a GPIO pin for a LED, a GPIO pin for a User Button and a GPIO pin for a UART/USART TX. The application serial settings can be configured by user changing the settings of UART/USART chosen in the CubeMX GUI.

4.79 CUSTOM_IntensityDetection

This application shows how to use the MotionID middleware library developed by STMicroelectronics with one of the several MEMS accelerometer components supported in the CubeMX MEMS Library on a generic custom board and a STM32 Nucleo board. Every MEMS component can be configured via I2C or via SPI 4-Wires (if it is supported). The application requires to configure in the custom board a GPIO pin for a LED, a GPIO pin for a User Button, a GPIO pin for a UART/USART TX and a GPIO input pin for a I3C SW disable feature. The application serial settings can be configured by user changing the settings of UART/USART chosen in the CubeMX GUI. After application starts, the user can view the data using a Unicleo-GUI application also developed by STMicroelectronics. The library is designed to be used on

STM32 microcontrollers based on the ARM Cortex-M0+, ARM Cortex-M3, ARM Cortex-M4 and ARM Cortex-M7 architecture and is intended for wrist-based devices. The application is able to distinguish motion intensity in a range from 0 to 10 (according to the library indexes) corresponding to the following activities: on desk, hand on bed/couch/cushion, light movements, biking, typing/writing, high intensity typing/slow walking, washing hands/walking, fast walking/jogging, running/brushing teeth, sprinting.

4.80 CUSTOM_DataLogFusion

This application shows how to use the MotionFX middleware library developed by STMicroelectronics with one of the several MEMS accelerometer, gyroscope and magnetometer components supported in the CubeMX MEMS Library on a generic custom board and a STM32 Nucleo board. Every MEMS component can be configured via I2C or via SPI 4-Wires (if it is supported). The application requires to configure in the custom board a GPIO pin for a LED, a GPIO pin for a User Button, a GPIO pin for a UART/USART TX and a GPIO input pin for a I3C SW disable feature. The application serial settings can be configured by user changing the settings of UART/USART chosen in the CubeMX GUI. After application starts, the user can view the data using a Unicleo-GUI application also developed by STMicroelectronics. The library is designed to be used on STM32 microcontrollers based on the ARM Cortex-M0+, ARM Cortex-M3, ARM Cortex-M4 and ARM Cortex-M7 architecture. The application provides real-time motion-sensor data fusion demonstrated on the teapot rotation. It also performs gyroscope bias and magnetometer hard iron calibration.

4.81 CUSTOM_GyroscopeCalibration

This application shows how to use the MotionGC middleware library developed by STMicroelectronics with one of the several MEMS gyroscope components supported in the CubeMX MEMS Library on a generic custom board and a STM32 Nucleo board. Every MEMS component can be configured via I2C or via SPI 4-Wires (if it is supported). The application requires to configure in the custom board a GPIO pin for a LED, a GPIO pin for a User Button, a GPIO pin for a UART/USART TX and a GPIO input pin for a I3C SW disable feature. The application serial settings can be configured by user changing the settings of UART/USART chosen in the CubeMX GUI. After application starts, the user can view the data using a Unicleo-GUI application also developed by STMicroelectronics. The library is designed to be used on STM32 microcontrollers based on the ARM Cortex-M0+, ARM Cortex-M3, ARM Cortex-M4 and ARM Cortex-M7 architecture. The application provides real-time gyroscope calibration through angular zero-rate level coefficients (offset) used to correct gyroscope data.

4.82 CUSTOM_AccelerometerCalibration

This application shows how to use the MotionAC middleware library developed by STMicroelectronics with one of the several MEMS accelerometer components supported in the CubeMX MEMS Library on a generic custom board and a STM32 Nucleo board. Every MEMS component can be configured via I2C or via SPI 4-Wires (if it is supported). The application requires to configure in the custom board a GPIO pin for a LED, a GPIO pin for a User Button, a GPIO pin for a UART/USART TX and a GPIO input pin for a I3C SW disable feature. The application serial settings can be configured by user changing the settings of UART/USART chosen in the CubeMX GUI. After application starts, the user can view the data using a Unicleo-GUI application also developed by STMicroelectronics. The library is designed to be used on STM32 microcontrollers based on the ARM Cortex-M3, ARM Cortex-M4 and ARM Cortex-M7 architecture. The application provides real-time accelerometer calibration through offset and scale factor coefficients used to correct accelerometer data.

4.83 CUSTOM_MagnetometerCalibration

This application shows how to use the MotionMC middleware library developed by STMicroelectronics with one of the several MEMS magnetometer components supported in the

CubeMX MEMS Library on a generic custom board and a STM32 Nucleo board. Every MEMS component can be configured via I2C or via SPI 4-Wires (if it is supported). The application requires to configure in the custom board a GPIO pin for a LED, a GPIO pin for a User Button, a GPIO pin for a UART/USART TX and a GPIO input pin for a I3C SW disable feature. The application serial settings can be configured by user changing the settings of UART/USART chosen in the CubeMX GUI. After application starts, the user can view the data using a Unicleo-GUI application also developed by STMicroelectronics. The library is designed to be used on STM32 microcontrollers based on the ARM Cortex-M0+, ARM Cortex-M3, ARM Cortex-M4 and ARM Cortex-M7 architecture. The application provides real-time magnetometer calibration using hard iron (HI) and scale factor coefficients to correct magnetometer data.

4.84 CUSTOM_TiltSensing

This application shows how to use the MotionTL middleware library developed by STMicroelectronics with one of the several MEMS accelerometer components supported in the CubeMX MEMS Library on a generic custom board and a STM32 Nucleo board. Every MEMS component can be configured via I2C or via SPI 4-Wires (if it is supported). The application requires to configure in the custom board a GPIO pin for a LED, a GPIO pin for a User Button, a GPIO pin for a UART/USART TX and a GPIO input pin for a I3C SW disable feature. The application serial settings can be configured by user changing the settings of UART/USART chosen in the CubeMX GUI. After application starts, the user can view the data using a Unicleo-GUI application also developed by STMicroelectronics. The library is designed to be used on STM32 microcontrollers based on the ARM Cortex-M0+, ARM Cortex-M3, ARM Cortex-M4 and ARM Cortex-M7 architecture. The application provides real-time information about the tilt angles of the user device, i.e. cell phone. It is also able to perform accelerometer 6-position calibration.

4.85 CUSTOM_VerticalContext

This application shows how to use the MotionVC middleware library developed by STMicroelectronics with one of the several MEMS accelerometer and pressure components supported in the CubeMX MEMS Library on a generic custom board and a STM32 Nucleo board. Every MEMS component can be configured via I2C or via SPI 4-Wires (if it is supported). The application requires to configure in the custom board a GPIO pin for a LED, a GPIO pin for a User Button, a GPIO pin for a UART/USART TX and a GPIO input pin for a I3C SW disable feature. The application serial settings can be configured by user changing the settings of UART/USART chosen in the CubeMX GUI. After application starts, the user can view the data using a Unicleo-GUI application also developed by STMicroelectronics. The library is designed to be used on STM32 microcontrollers based on the ARM Cortex-M3, ARM Cortex-M4 and ARM Cortex-M7 architecture. The application provides real-time information about vertical movement. It is able to detect a change of altitude and distinguish the type of vertical movement: stairs, elevator, and escalator.

4.86 CUSTOM_CarryPosition

This application shows how to use the MotionCP middleware library developed by STMicroelectronics with one of the several MEMS accelerometer components supported in the CubeMX MEMS Library on a generic custom board and a STM32 Nucleo board. Every MEMS component can be configured via I2C or via SPI 4-Wires (if it is supported). The application requires to configure in the custom board a GPIO pin for a LED, a GPIO pin for a User Button, a GPIO pin for a UART/USART TX and a GPIO input pin for a I3C SW disable feature. The application serial settings can be configured by user changing the settings of UART/USART chosen in the CubeMX GUI. After application starts, the user can view the data using a Unicleo-GUI application also developed by STMicroelectronics. The library is designed to be used on STM32 microcontrollers based on the ARM Cortex-M3, ARM Cortex-M4 and ARM Cortex-M7 architecture. The application provides real-time information about how the user is carrying a device (i.e. cell phone). It is able to distinguish the following positions: on desk, in hand, near

head, shirt pocket, trouser pocket, swinging arm and jacket pocket.

4.87 CUSTOM_ECompass

This application shows how to use the MotionEC middleware library developed by STMicroelectronics with one of the several MEMS accelerometer and magnetometer components supported in the CubeMX MEMS Library on a generic custom board and a STM32 Nucleo board. Every MEMS component can be configured via I2C or via SPI 4-Wires (if it is supported). The application requires to configure in the custom board a GPIO pin for a LED, a GPIO pin for a User Button, a GPIO pin for a UART/USART TX and a GPIO input pin for a I3C SW disable feature. The application serial settings can be configured by user changing the settings of UART/USART chosen in the CubeMX GUI. After application starts, the user can view the data using a Unicleo-GUI application also developed by STMicroelectronics. The library is designed to be used on STM32 microcontrollers based on the ARM Cortex-M0+, ARM Cortex-M3, ARM Cortex-M4 and ARM Cortex-M7 architecture. The application provides real-time information about the device orientation and movement status based on data from a device. It provides the following outputs: device orientation (quaternions, Euler angles), device rotation (virtual gyroscope functionality), gravity vector and linear acceleration.

4.88 CUSTOM_GestureRecognition

This application shows how to use the MotionGR middleware library developed by STMicroelectronics with one of the several MEMS accelerometer components supported in the CubeMX MEMS Library on a generic custom board and a STM32 Nucleo board. Every MEMS component can be configured via I2C or via SPI 4-Wires (if it is supported). The application requires to configure in the custom board a GPIO pin for a LED, a GPIO pin for a User Button, a GPIO pin for a UART/USART TX and a GPIO input pin for a I3C SW disable feature. The application serial settings can be configured by user changing the settings of UART/USART chosen in the CubeMX GUI. After application starts, the user can view the data using a Unicleo-GUI application also developed by STMicroelectronics. The library is designed to be used on STM32 microcontrollers based on the ARM Cortex-M3, ARM Cortex-M4 and ARM Cortex-M7 architecture. The application provides real-time information about the gesture just performed by the user with the device, such as a cell phone. It is able to distinguish the following gestures: pick up, glance, wake up.

5 Installing the X-CUBE-MEMS1 pack in STM32CubeMX

After downloading (from www.st.com), installing and launching the STM32CubeMX (V≥5.6.0), the X-CUBE-MEMS1 pack can be installed in few steps.

1. From the menu, select Help > Manage embedded software packages

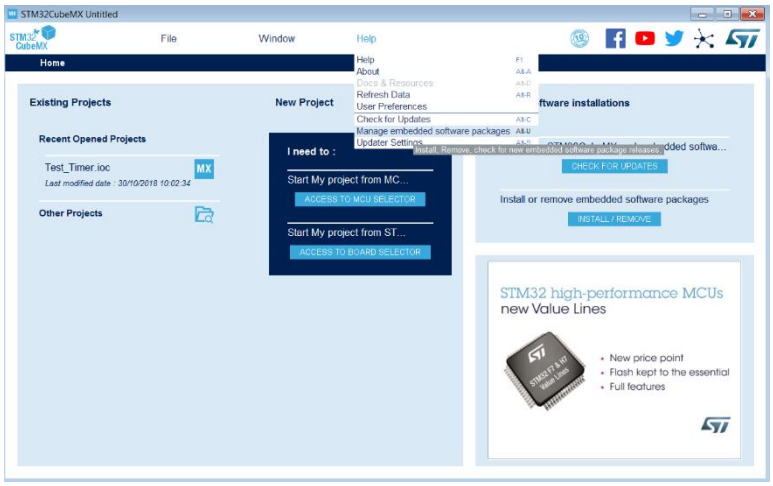

Figure 1 *Managing embedded software packs in STM32CubeMX*

2. From the Embedded Software Packages Manager window, press the 'Refresh' button to get an updated list of the add-on packs. Go to the 'STMicroelectronics' tab to find the X-CUBE-MEMS1 pack.

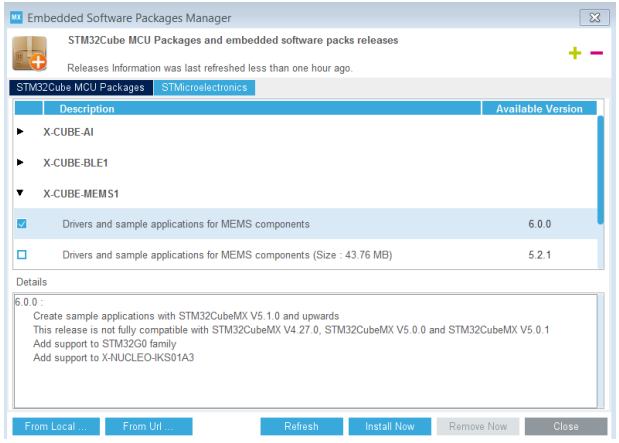

Figure 2 *Installing the X-CUBE-MEMS1 pack in STM32CubeMX*

3. Select it checking the corresponding box and install it pressing the 'Install Now' button. Once the installation is completed, the corresponding box will become green, the 'Close' button can be pressed and the configuration of a new project can start.

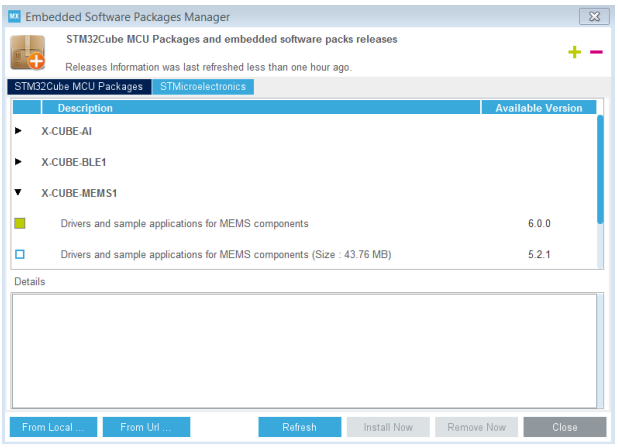

***Figure 3** The X-CUBE-MEMS1 pack in STM32CubeMX*

6 Starting a new project

After launching the STM32CubeMX, you can choose if starting a New Project from the MCU Selector or from the Board Selector.

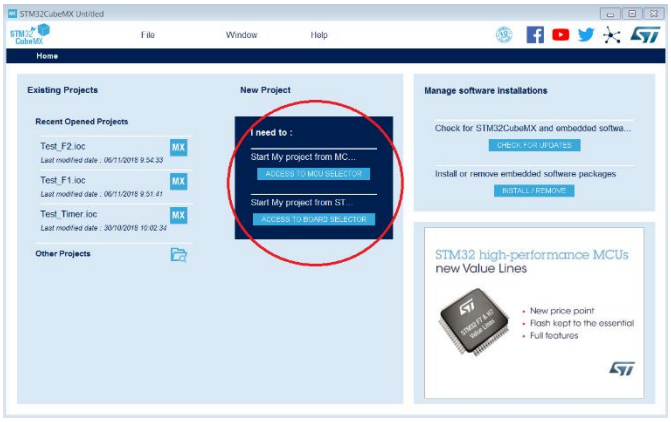

***Figure 4** STM32CubeMX main page*

The **MCU/Board selector** window will pop up. From this window, the STM32 MCU or platform can be selected.

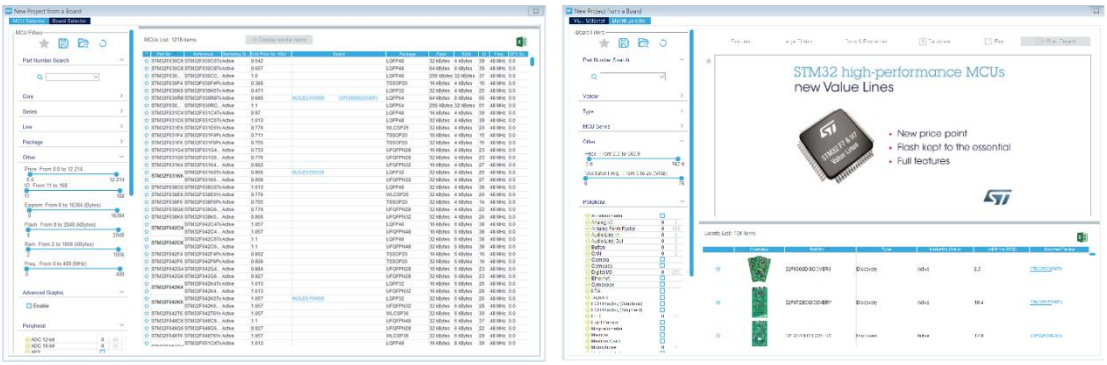

***Figure 5** STM32CubeMX **MCU/Board Selector** windows*

After selecting the MCU or the Board, the selected STM32 pinout will appear. From this window the user can set up the project, by adding one or more Additional Software and peripherals and configuring the clock.

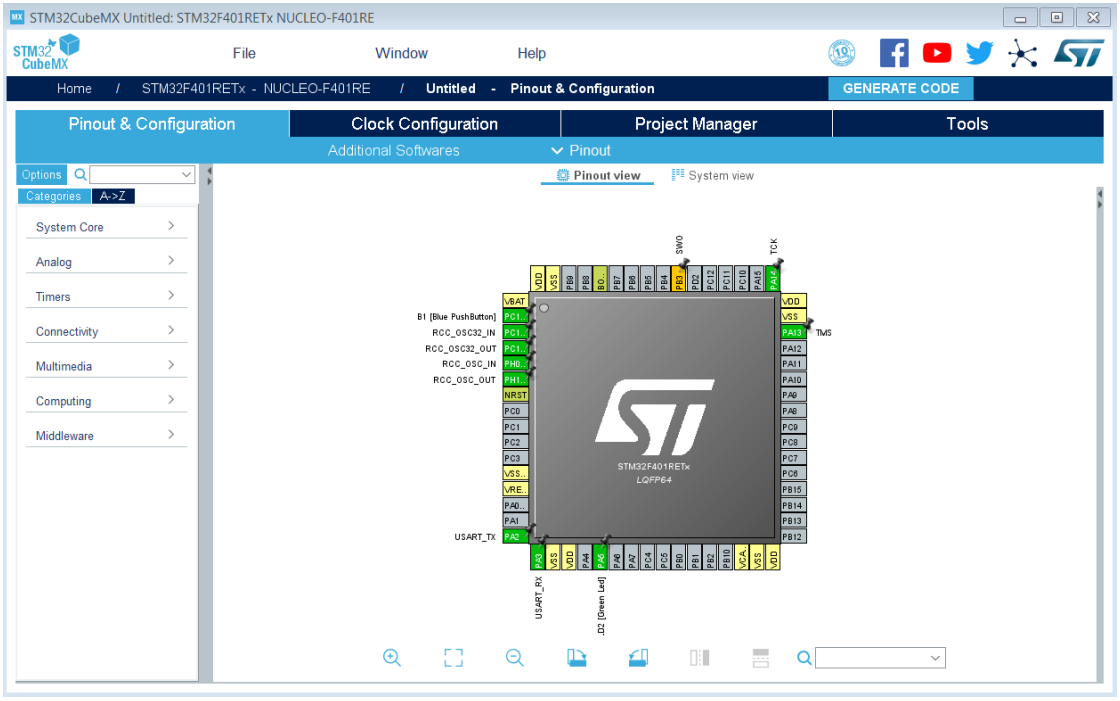

***Figure 6** STM32CubeMX **Pinout & Configuration** window*

To add the X-CUBE-MEMS1 additional software to the project, the “Additional Softwares” button must be clicked.
From the Additional Software Component Selection window, the user can either choose to generate, for the selected MCU/Board, one of the enclosed sample applications or a new project. In this latter case, the user must just implement the main application logic without bothering with the pinout and peripherals configuration code that will be automatically generated by STM32CubeMX.

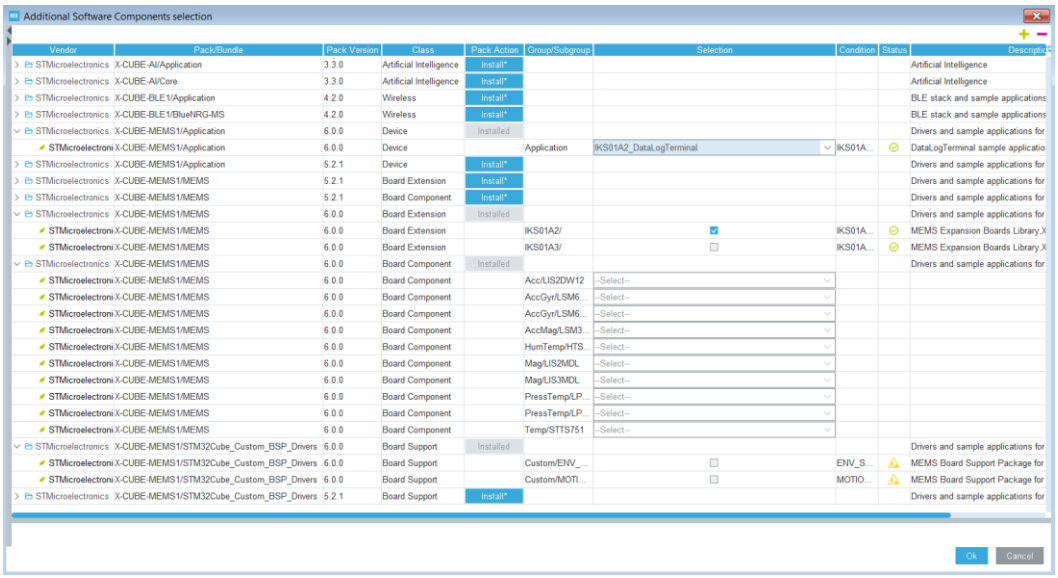

***Figure 7** STM32CubeMX **Additional Software Components selection** window*

7 STM32 Configuration Steps

The X-NUCLEO-IKS01A2 or the X-NUCLEO-IKS01A3 interfaces with the STM32 microcontroller via the I2C bus. Hence, assuming a user wants to interface the X-NUCLEO-IKS01A2 or X-NUCLEO-IKS01A3 expansion board with a STM32 Nucleo 64 pins board (e.g. a Nucleo-F401RETx) or a STM32 Nucleo 144 pins board (e.g. a Nucleo F429ZITx), no particular hardware modification must be done.

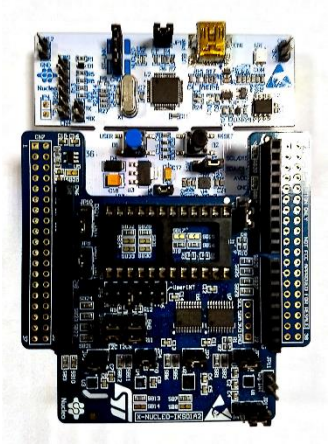

***Figure 8** STM32 Nucleo 64 pins and X-NUCLEO-IKS01A2*

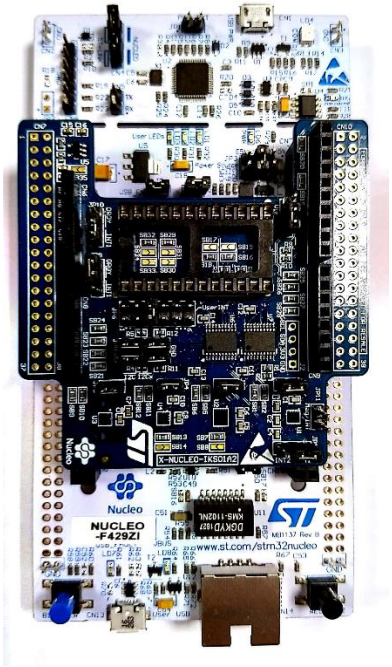

Figure 9 *STM32 Nucleo 144 pins and X-NUCLEO-IKS01A2*

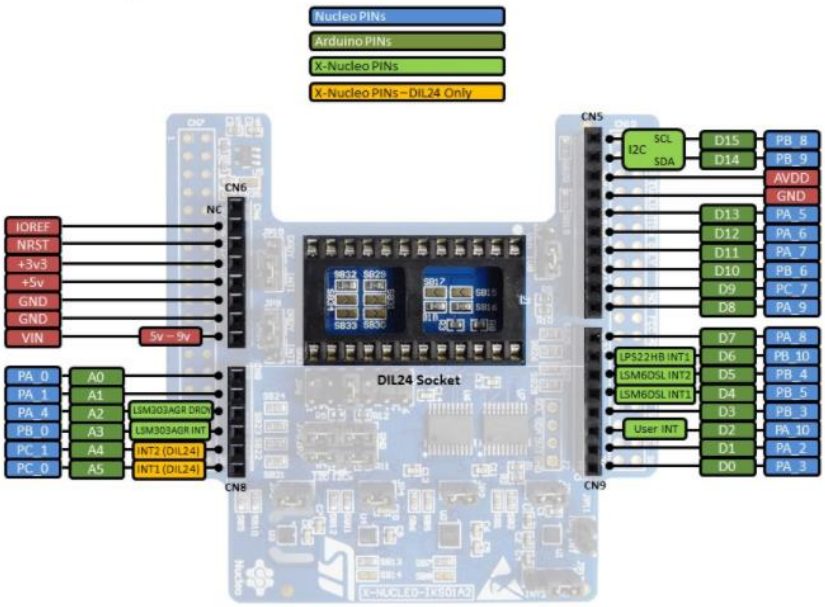

Figure 10 *X-NUCLEO-IKS01A2 pinout*

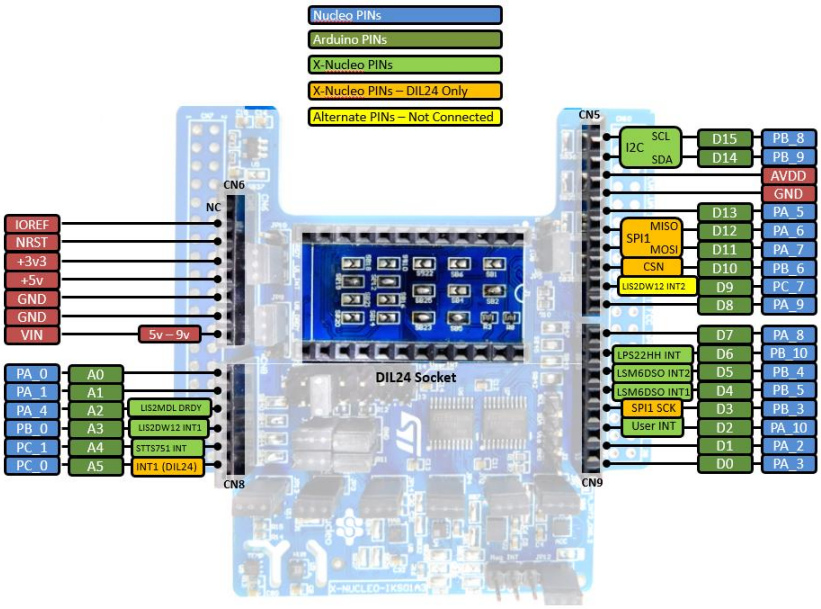

Figure 11 *X-NUCLEO-IKS01A3 pinout*

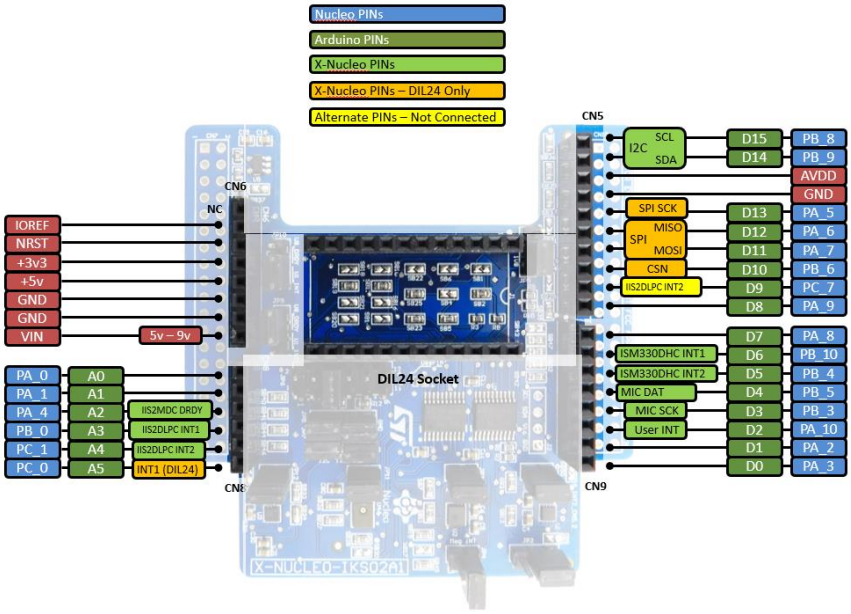

***Figure 12** X-NUCLEO-IKS02A1 pinout*

7.1 Use of MEMS Library without sample applications for X-NUCLEO-IKS01A2 or X-NUCLEO-IKS01A3 or X-NUCLEO-IKS02A1

This section outlines how to configure STM32CubeMX with X-NUCLEO-IKS01A2 or X-NUCLEO-IKS01A3 or X-NUCLEO-IKS02A1 when the use of the sample applications is not required. With such setup, only driver layers will be configured. This setup is useful when the user does not intend to leverage the sample application provided in the package.

To add the X-CUBE-MEMS1 additional software to the project, the "Additional Softwares" button must be clicked. From the "Additional Software Components selection" window, the user has to select just the "Board Extension" class as shown in the figure below.

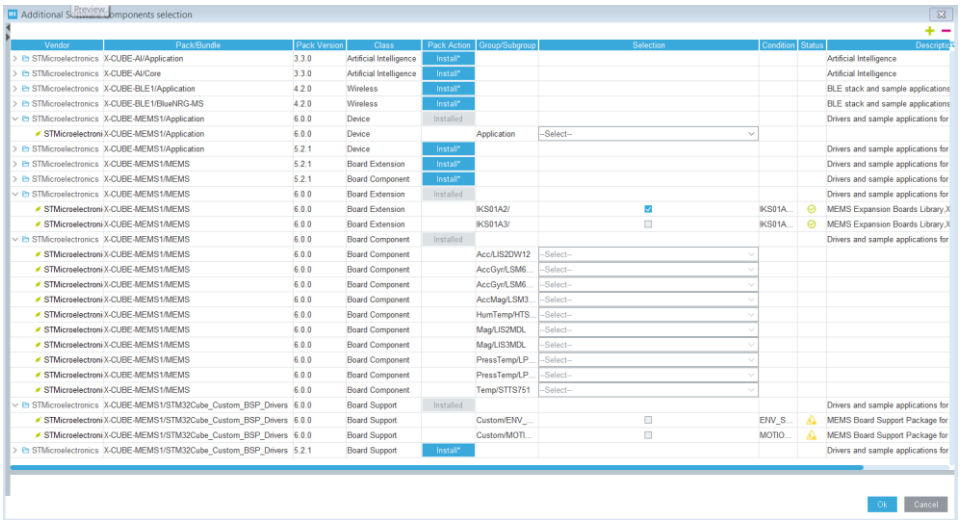

***Figure 13** STM32CubeMX **Additional Software Components selection** window*

From the **Pinout & Configuration** tab:

- from the **Pinout** scheme, click on PB8 and set it as I2C1_SCL;
- from the **Pinout** scheme, click on PB9 and set it as I2C1_SDA;
- enable the I2C1 as I2C from the “Connectivity” category;
- Configure the I2C1 settings with I2C speed at 400KHz (Fast Mode) from the “Configuration” view;

From the **Pinout** scheme set:

Nucleo 64		**Nucleo 144**	
PB8	I2C1_SCL	PB8	I2C1_SCL
PB9	I2C1_SDA	PB9	I2C1_SDA

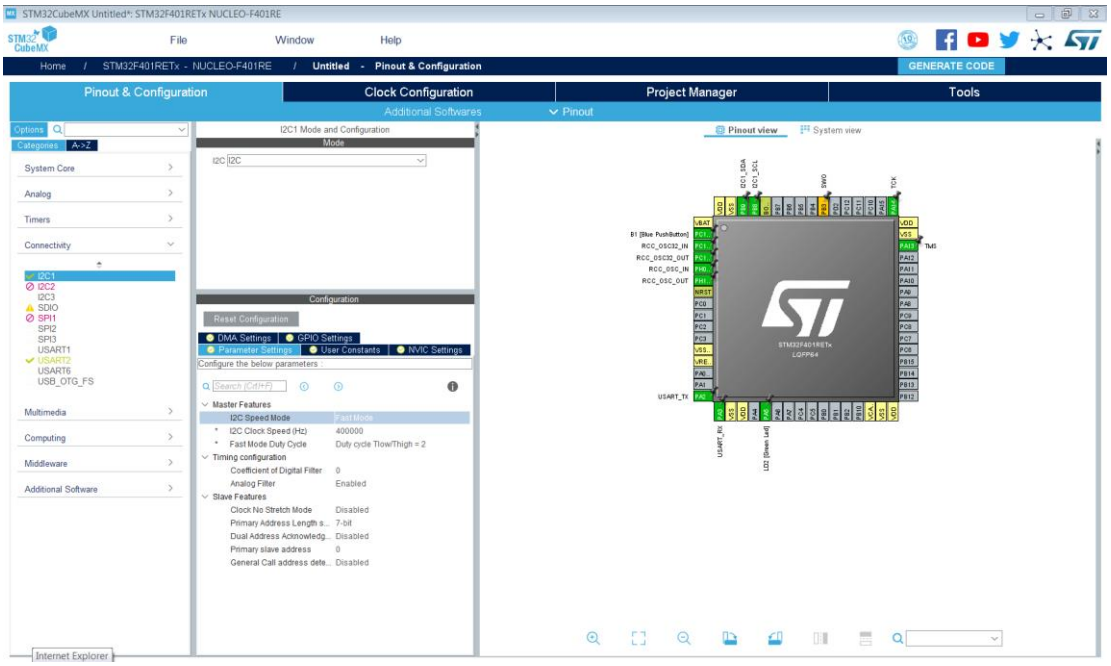

***Figure 14** STM32CubeMX **Pinout & Configuration** tab and I2C settings*

From the **Additional Software** category, press the 'Stmicroelectronics.X-CUBE-MEMS1.7.1.0' item, enable the "Board Extension MEMS" checkbox from the "Mode" view and set the following Platform Settings from the "Configuration" view:

Name	BSP_Api	Supported IPs	Nucleo 64	Nucleo 144
IKS01A2 BUS IO driver	BSP_BUS_DRIVER	I2C:I2C	I2C1	I2C1

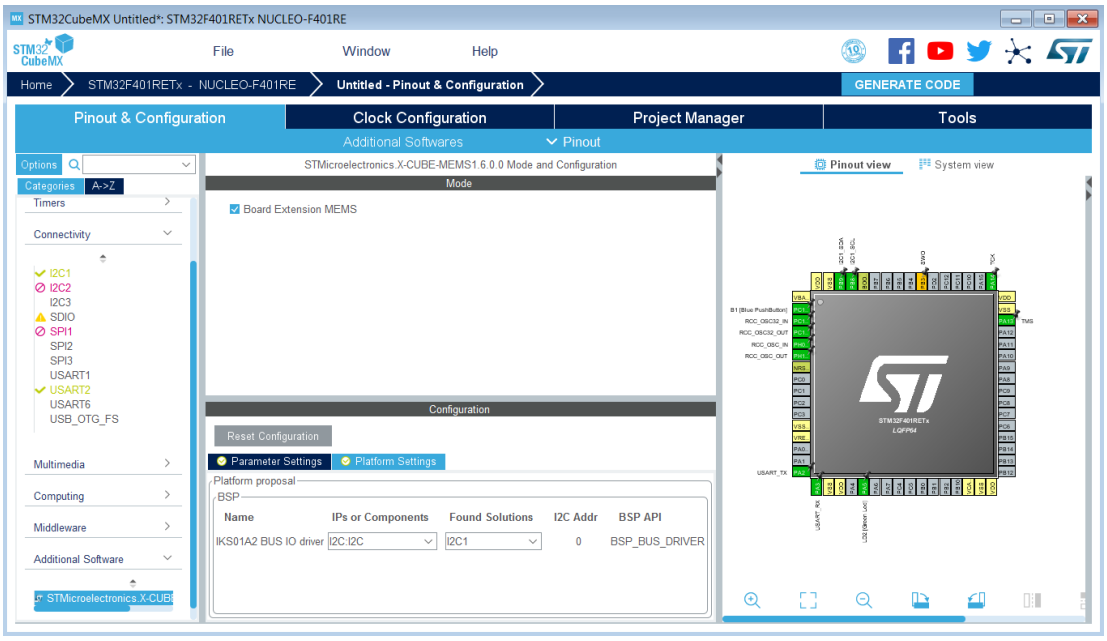

***Figure 15** STM32CubeMX **Pinout & Configuration** tab and Additional Software settings*

Once all the above described steps have been performed, the source code of the project using the **STMicroelectronics X-CUBE-MEMS1** software can be generated clicking the "GENERATE CODE" button.

7.2 Use of MEMS Library with sample applications for X-NUCLEO-IKS01A2 or X-NUCLEO-IKS01A3 or X-NUCLEO-IKS02A1

This section outlines how to configure STM32CubeMX with X-NUCLEO-IKS01A2 or X-NUCLEO-IKS01A3 or X-NUCLEO-IKS02A1 when the use of the sample applications is required. With such setup, all the components of the expansion software package, including applications, will be properly configured.

To add the X-CUBE-MEMS1 additional software to the project, the "Additional Softwares" button must be clicked. From the "Additional Software Components selection" window, the user has to select just the "Board Extension" class and an application from the "Device" class as shown in the figure below.

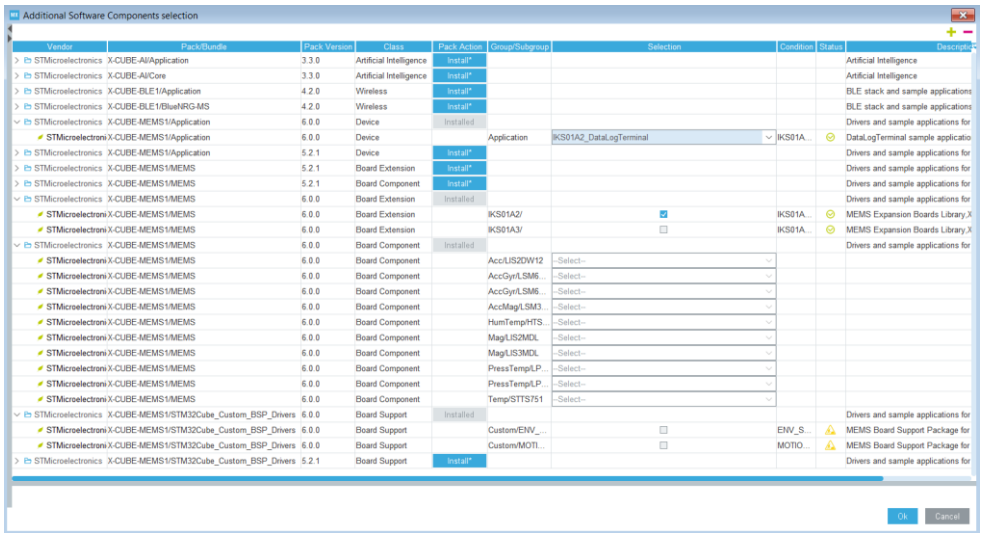

***Figure 16** STM32CubeMX **Additional Software Components selection** window example for the IKS01A2*

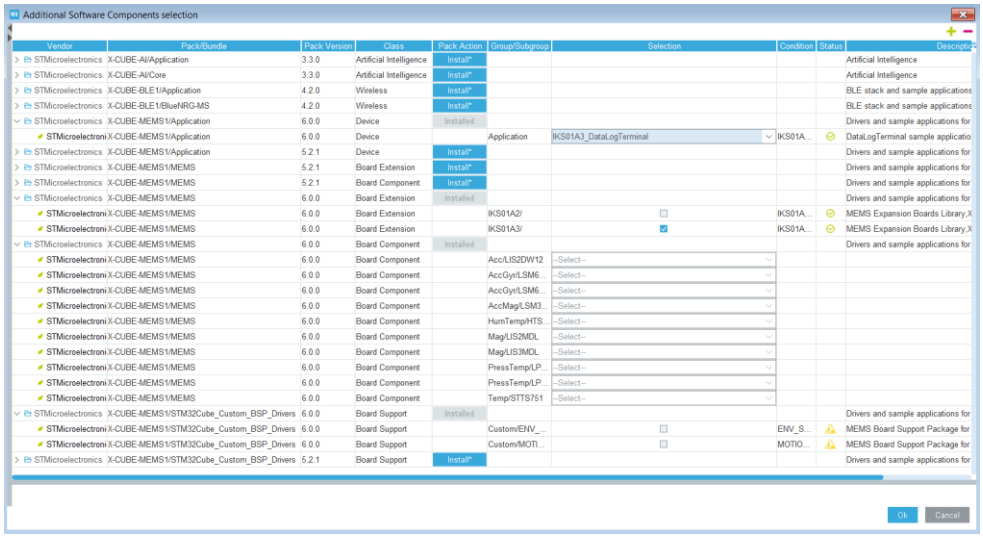

***Figure 17** STM32CubeMX **Additional Software Components selection** window example for the IKS01A3*

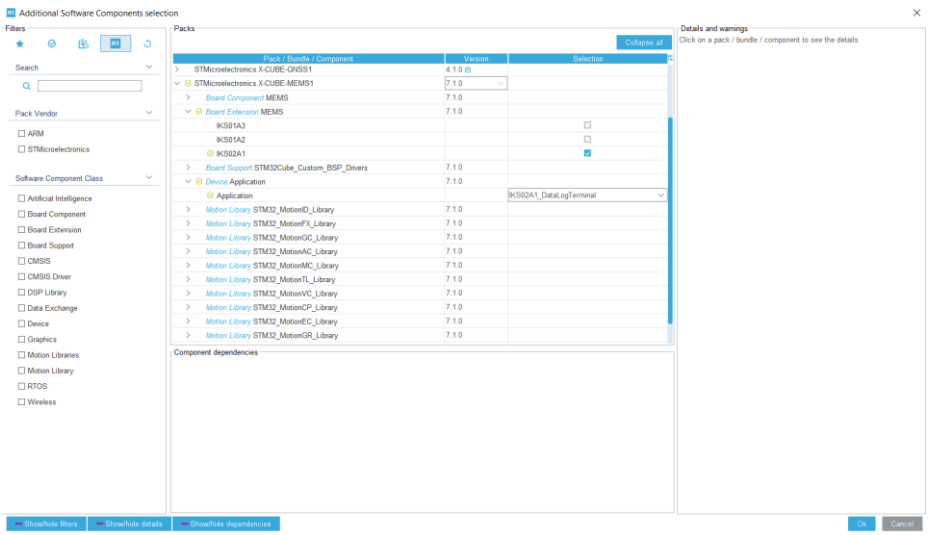

***Figure 18** STM32CubeMX **Additional Software Components selection** window example for the IKS02A1*

From the **Pinout & Configuration** tab:

- from the **Pinout** scheme, click on PB8 and set it as I2C1_SCL;
- from the **Pinout** scheme, click on PB9 and set it as I2C1_SDA;
- enable the I2C1 as I2C from the "Connectivity" category;
- if not enabled yet:
    a. enable the USART2 in Asynchronous mode (for Nucleo 64) from the "Connectivity" category
    b. enable the USART3 in Asynchronous mode (for Nucleo 144) from the "Connectivity" category

From the **Pinout** scheme, if not already set, if you are using the X-NUCLEO-IKS01A2 set:

Nucleo 64			**Nucleo 144**		
PIN	*Mode*	*Label*	*PIN*	*Mode*	*Label*
PB5	GPIO_EXTI5		PF14	GPIO_EXTI14	
PB4	GPIO_EXTI4		PE11	GPIO_EXTI11	
PB10	GPIO_EXTI10		PE9	GPIO_EXTI9	
PA2	USART2_TX	USART_TX	PD8	USART3_TX	USART_TX
PA3	USART2_RX	USART_RX	PD9	USART3_RX	USART_RX
PA5	GPIO_Output	LD2 [Green Led]	PB7	GPIO_Output	LD2[Blue]
PC13	GPIO_EXTI13	B1 [Blue PushButton]	PC13	GPIO_EXTI13	USER_Btn[B1]

From the **Pinout** scheme, if not already set, if you are using the X-NUCLEO-IKS01A3 set:

Nucleo 64			Nucleo 144		
PIN	*Mode*	*Label*	*PIN*	*Mode*	*Label*
PB5	GPIO_EXTI5		PF14	GPIO_EXTI14	
PB4	GPIO_EXTI4		PE11	GPIO_EXTI11	
PB10	GPIO_EXTI10		PE9	GPIO_EXTI9	
PB0	GPIO_EXTI0		PF3	GPIO_EXTI3	
PC1	GPIO_EXTI1		PF5	GPIO_EXTI5	
PA2	USART2_TX	USART_TX	PD8	USART3_TX	USART_TX
PA3	USART2_RX	USART_RX	PD9	USART3_RX	USART_RX
PA5	GPIO_Output	LD2 [Green Led]	PB7	GPIO_Output	LD2[Blue]
PC13	GPIO_EXTI13	B1 [Blue PushButton]	PC13	GPIO_EXTI13	USER_Btn[B1]

From the **Pinout** scheme, if not already set, if you are using the X-NUCLEO-IKS02A1 set:

Nucleo 64			Nucleo 144		
PIN	*Mode*	*Label*	*PIN*	*Mode*	*Label*
PA2	USART2_TX	USART_TX	PD8	USART3_TX	USART_TX
PA3	USART2_RX	USART_RX	PD9	USART3_RX	USART_RX
PA5	GPIO_Output	LD2 [Green Led]	PB7	GPIO_Output	LD2[Blue]
PC13	GPIO_EXTI13	B1 [Blue PushButton]	PC13	GPIO_EXTI13	USER_Btn[B1]

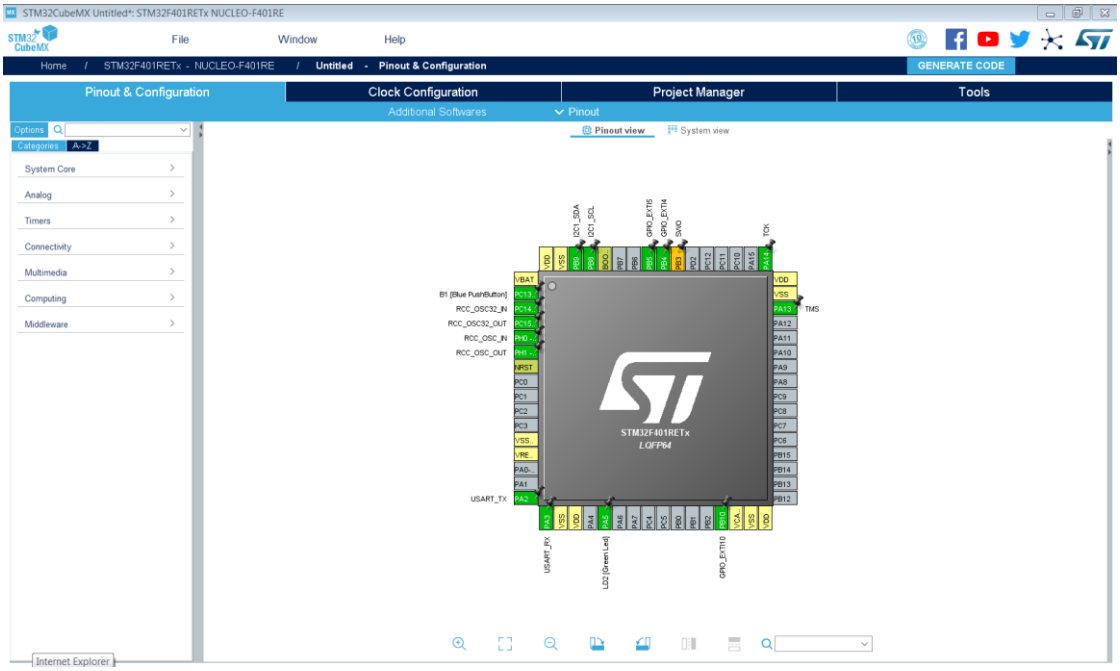

***Figure 19** STM32CubeMX **Pinout & Configuration** tab for X-NUCLEO-IKS01A2*

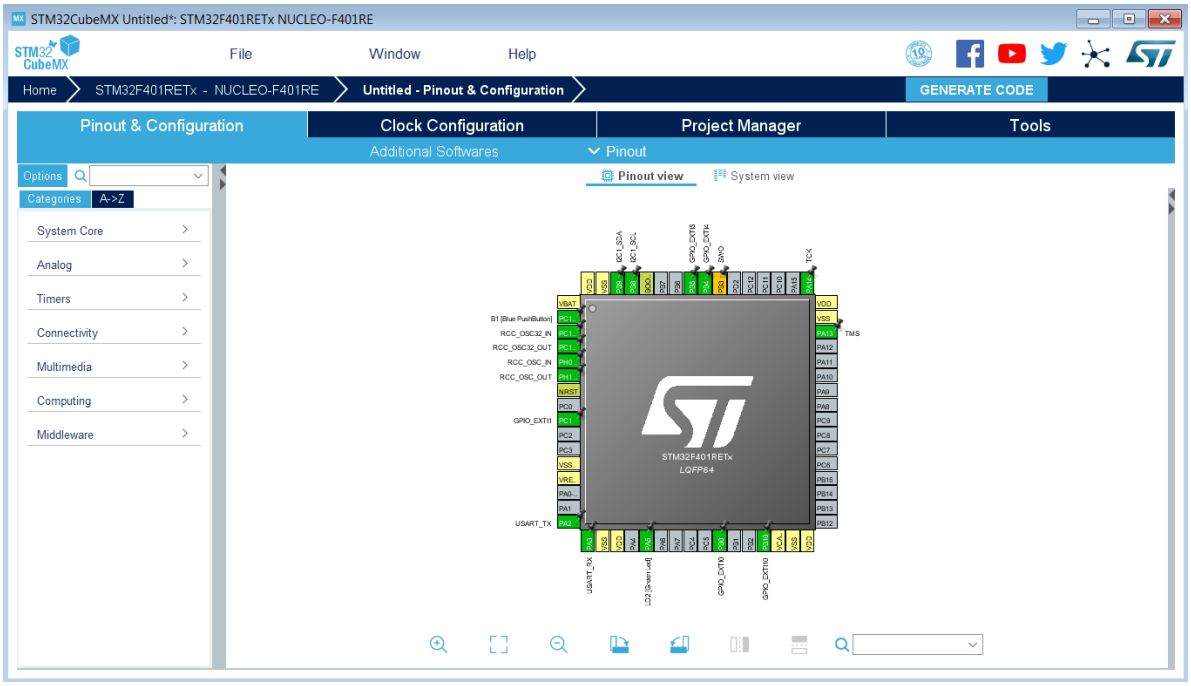

***Figure 20** STM32CubeMX **Pinout & Configuration** tab for X-NUCLEO-IKS01A3*

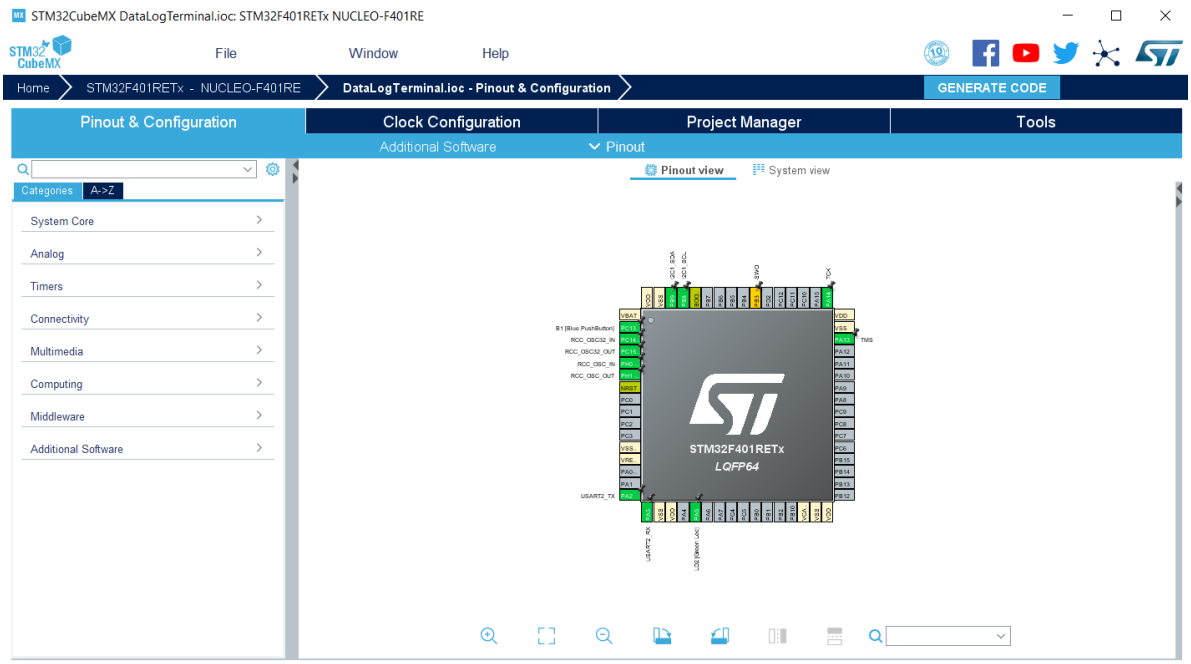

***Figure 21** STM32CubeMX **Pinout & Configuration** tab for X-NUCLEO-IKS02A1*

From the **Additional Software** category, press the 'Stmicroelectronics.X-CUBE-MEMS1.7.1.0' item, enable the "Board Extension MEMS" and the "Device Application" checkboxes from the

“Mode” view.
If you are using the X-NUCLEO-IKS01A2, you need to set the following Platform Settings from the “Configuration” view (take into account that according the example chosen some settings can appear or not):

Name	BSP_Api	Supported IPs	Nucleo 64	Nucleo 144
IKS01A2 BUS IO driver	BSP_BUS_DRIVER	I2C:I2C	I2C1	I2C1
MEMS_INT_PIN_A (All IKS01A2 applications except to IKS01A2_LPS22HB_FIFOMode)	HAL_EXTI_DRIVER	GPIO:EXTI	PB5	PF14
MEMS_INT_PIN_A (IKS01A2_LPS22HB_FIFOMode application)	HAL_EXTI_DRIVER	GPIO:EXTI	PB10	PE9
MEMS_INT_PIN_B	HAL_EXTI_DRIVER	GPIO:EXTI	PB4	PE11
BSP LED	BSP_COMMON_DRIVER	GPIO:Output	PA5	PB7
BSP BUTTON	BSP_COMMON_DRIVER	GPIO:EXTI	PC13	PC13
BSP USART	BSP_COMMON_DRIVER	USART:Asynchronous	USART2	USART3

If you are using the X-NUCLEO-IKS01A3, you need to set the following Platform Settings from the “Configuration” view (take into account that according the example chosen some settings can appear or not):

Name	BSP_Api	Supported IPs	Nucleo 64	Nucleo 144
IKS01A3 BUS IO driver	BSP_BUS_DRIVER	I2C:I2C	I2C1	I2C1
MEMS_INT_PIN_A (All IKS01A3 applications based on LSM6DSO component)	HAL_EXTI_DRIVER	GPIO:EXTI	PB5	PF14
MEMS_INT_PIN_A (IKS01A3 application based on LPS22HH component)	HAL_EXTI_DRIVER	GPIO:EXTI	PB10	PE9
MEMS_INT_PIN_A (IKS01A3 applications based on LIS2DW12 component)	HAL_EXTI_DRIVER	GPIO:EXTI	PB0	PF3

Name	BSP_Api	Supported IPs	Nucleo 64	Nucleo 144
MEMS_INT_PIN_A (IKS01A3 application based on STTS751 component)	HAL_EXTI_DRIVER	GPIO:EXTI	PC1	PF5
BSP LED	BSP_COMMON_DRIVER	GPIO:Output	PA5	PB7
BSP BUTTON	BSP_COMMON_DRIVER	GPIO:EXTI	PC13	PC13
BSP USART	BSP_COMMON_DRIVER	USART:Asynchronous	USART2	USART3

If you are using the X-NUCLEO-IKS02A1, you need to set the following Platform Settings from the “Configuration” view:

Name	BSP_Api	Supported IPs	Nucleo 64	Nucleo 144
IKS01A2 BUS IO driver	BSP_BUS_DRIVER	I2C:I2C	I2C1	I2C1
BSP LED	BSP_COMMON_DRIVER	GPIO:Output	PA5	PB7
BSP BUTTON	BSP_COMMON_DRIVER	GPIO:EXTI	PC13	PC13
BSP USART	BSP_COMMON_DRIVER	USART:Asynchronous	USART2	USART3

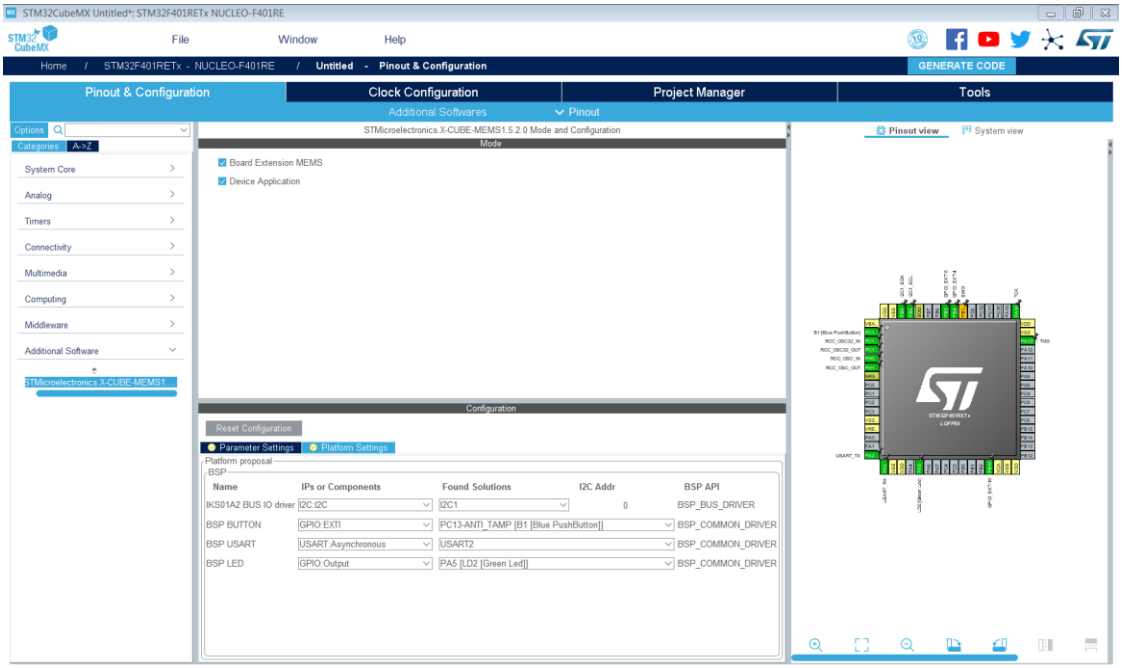

***Figure 22** STM32CubeMX **Pinout & Configuration** tab and Additional Software settings for IKS01A2_DataLogTerminal and IKS01A2_LSM6DSL_SelfTest applications*

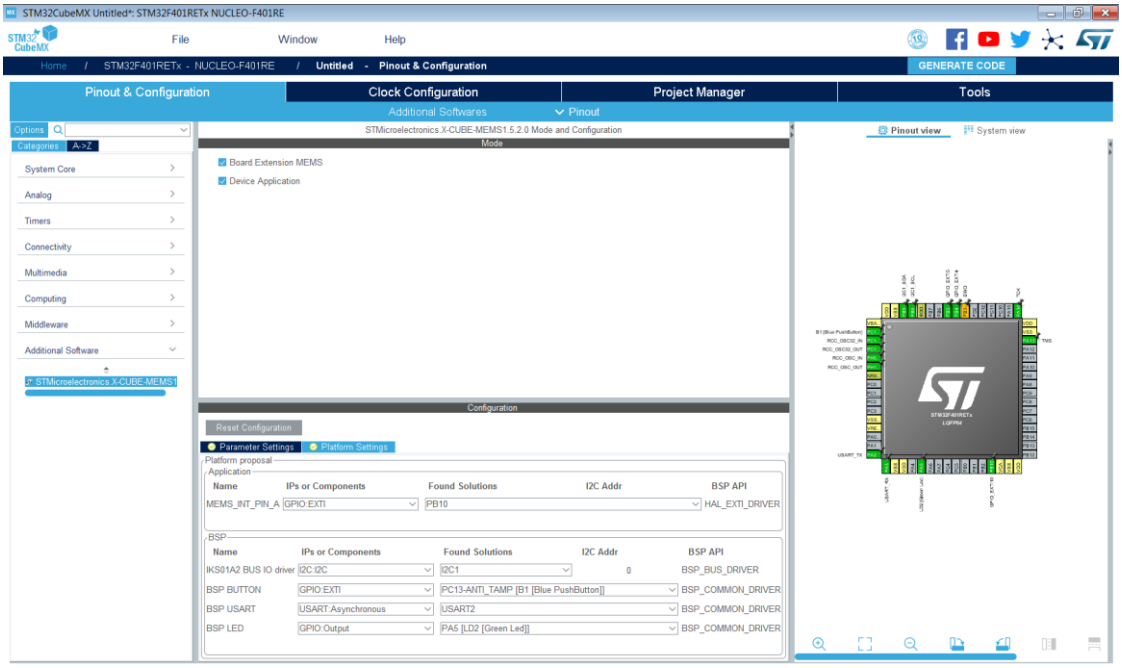

***Figure 23** STM32CubeMX **Pinout & Configuration** tab and Additional Software settings for IKS01A2_LPS22HB_FIFOMode application*

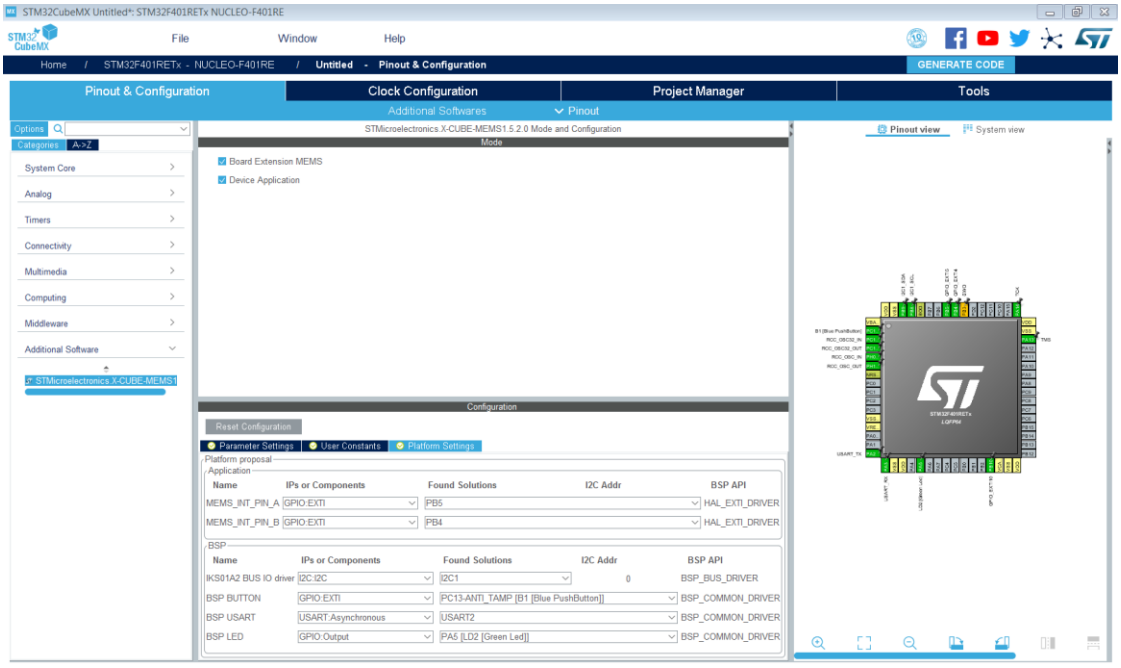

***Figure 24** STM32CubeMX **Pinout & Configuration** tab and Additional Software settings for IKS01A2_LSM6DSL_MultiEvent application*

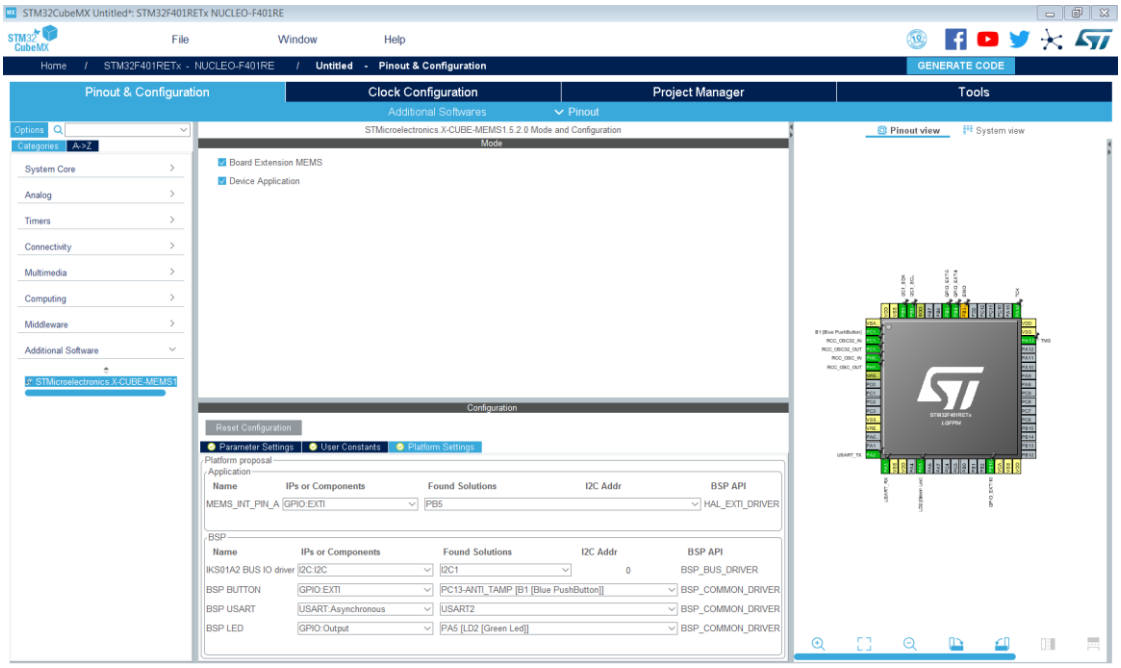

***Figure 25** STM32CubeMX **Pinout & Configuration** tab and Additional Software settings for all other IKS01A2 applications*

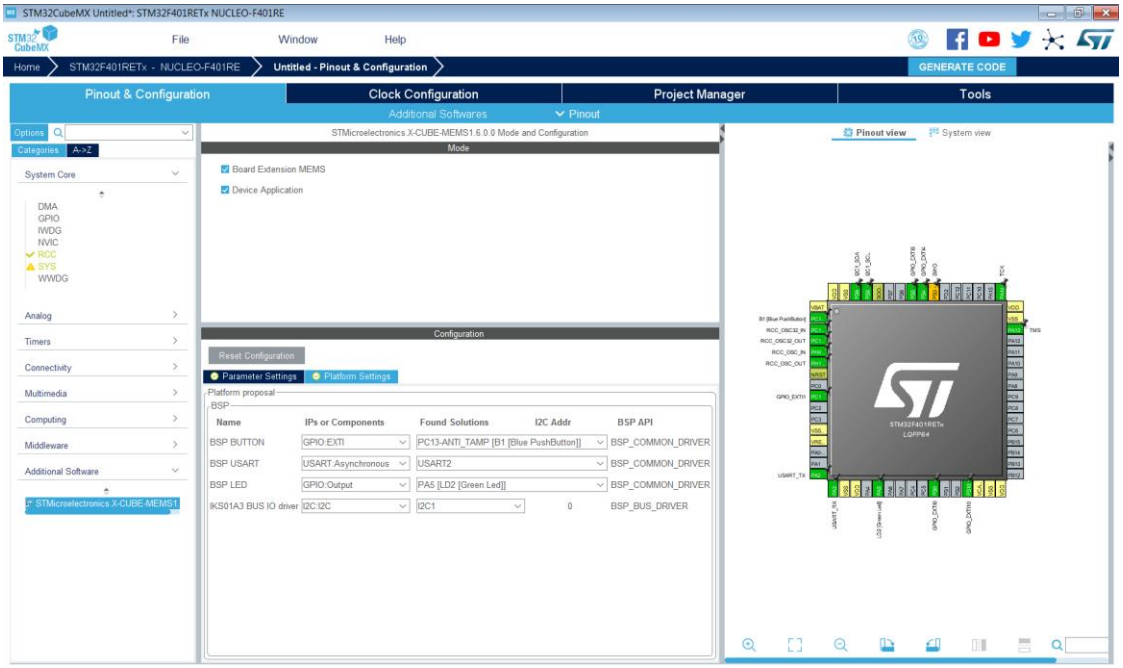

***Figure 26** STM32CubeMX **Pinout & Configuration** tab and Additional Software settings for IKS01A3_DataLogTerminal, IKS01A3_LIS2DW12_SelfTest, IKS01A3_LIS2MDL_SelfTest and IKS01A3_LSM6DSO_SelfTest applications*

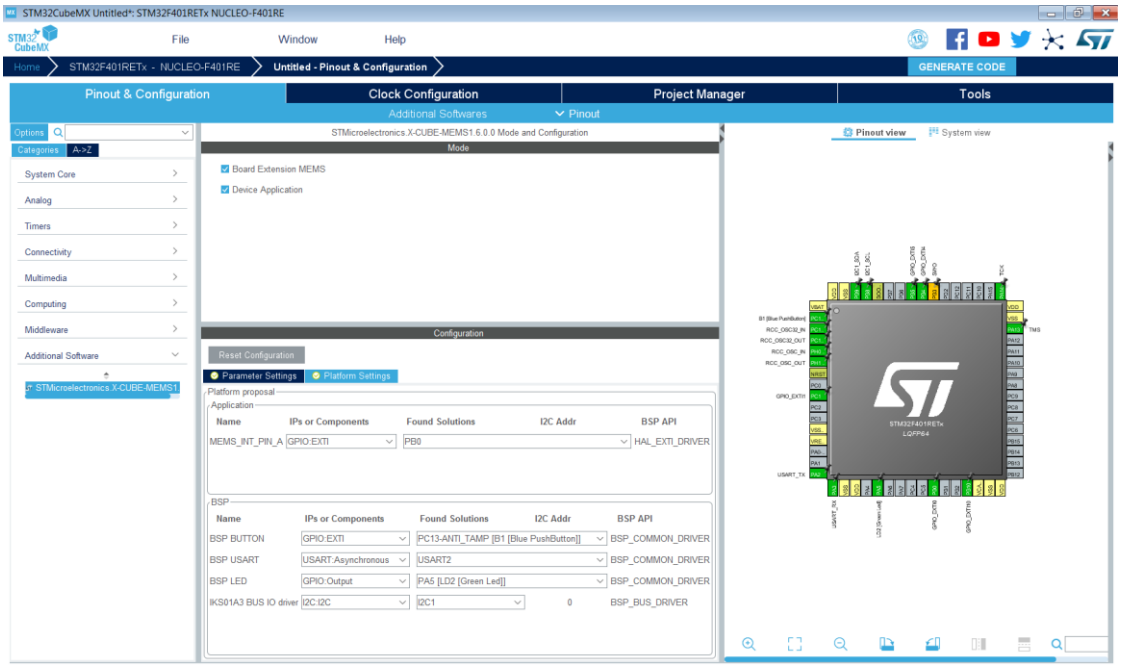

***Figure 27** STM32CubeMX **Pinout & Configuration** tab and Additional Software settings for IKS01A3_LIS2DW12_6DOrientation and IKS01A3_LIS2DW12_WakeUp applications*

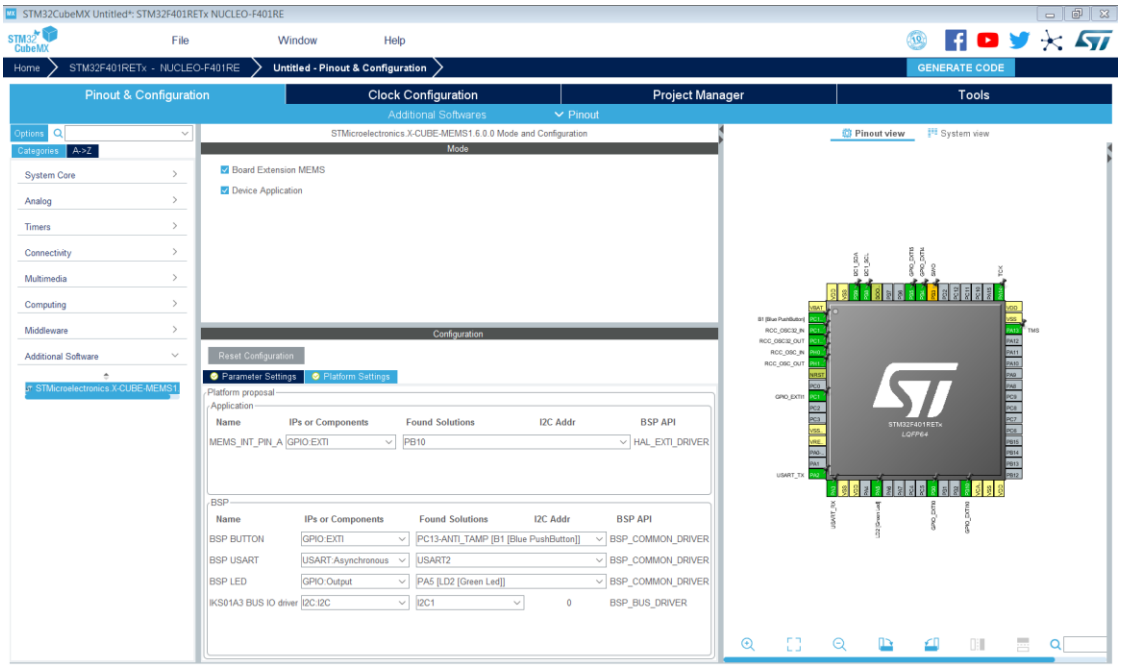

***Figure 28** STM32CubeMX **Pinout & Configuration** tab and Additional Software settings for IKS01A3_LPS22HH_FIFOMode application*

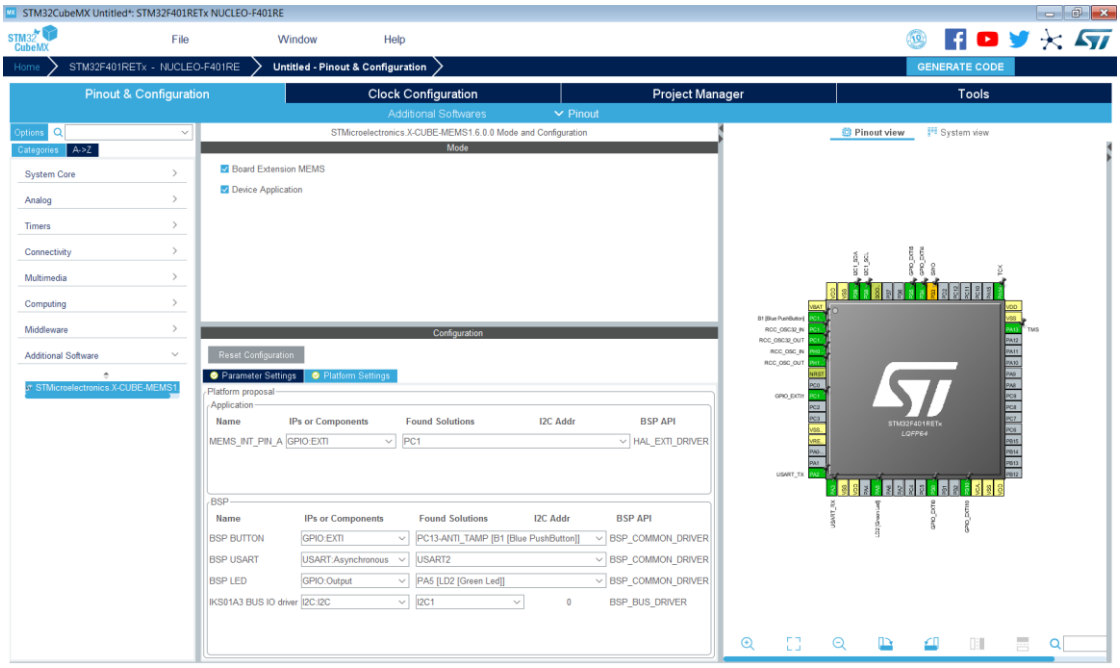

***Figure 29** STM32CubeMX **Pinout & Configuration** tab and Additional Software settings for IKS01A3_STTS751_TemperatureLimit application*

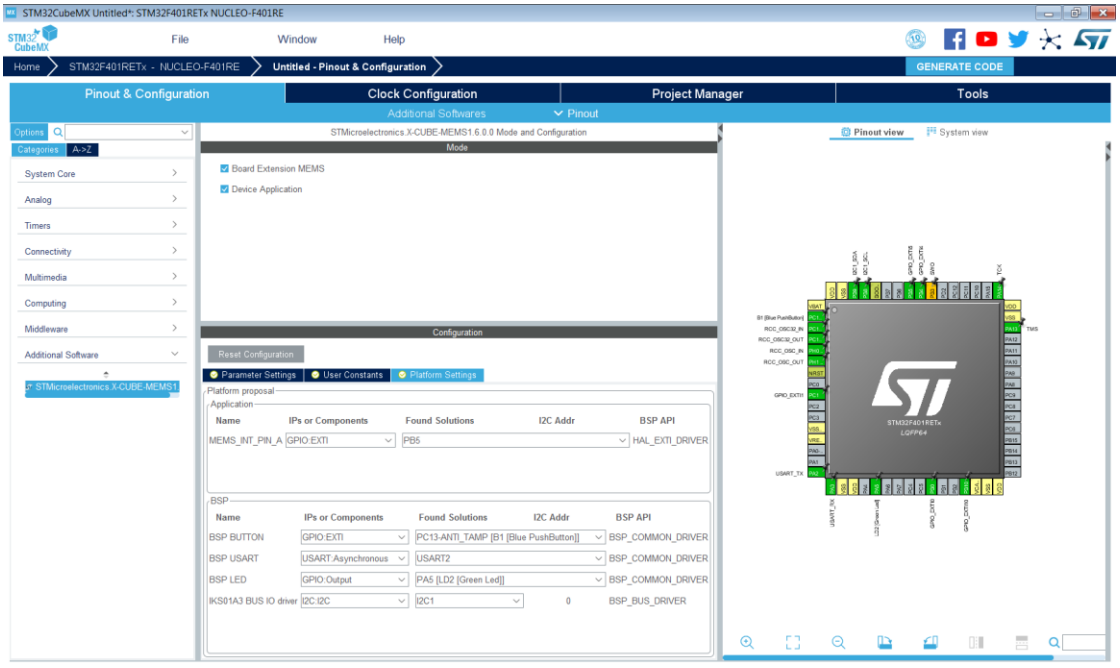

***Figure 30** STM32CubeMX **Pinout & Configuration** tab and Additional Software settings for all other IKS01A3 applications*

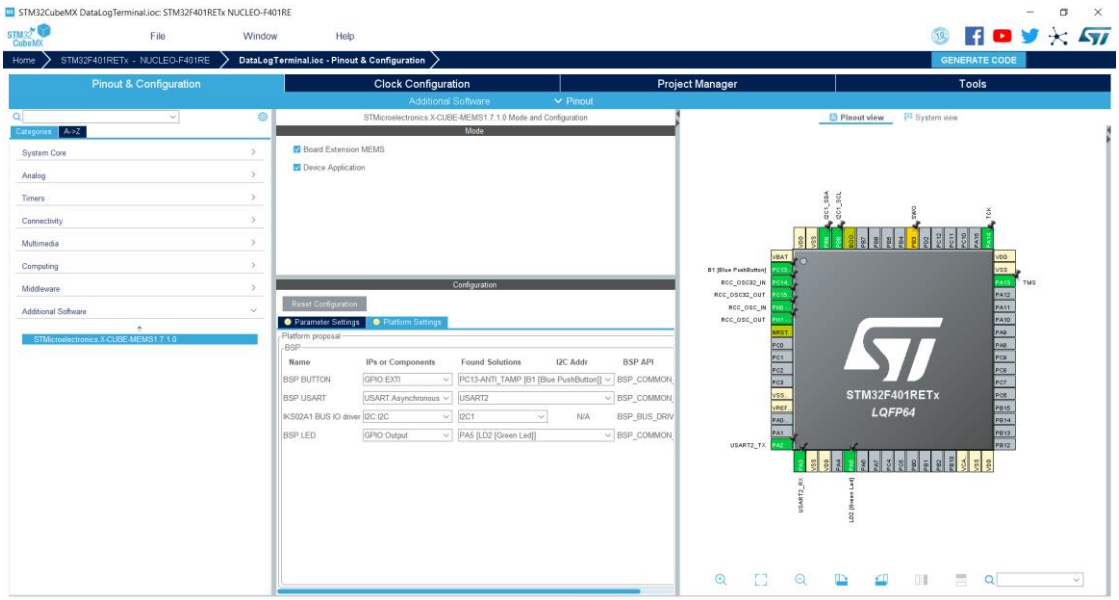

***Figure 31** STM32CubeMX **Pinout & Configuration** tab and Additional Software settings for IKS02A1_DataLogTerminal application*

From the **Parameter Settings** tab, some parameters for the routing of the interrupt signals can be changed.
For all the IKS01A2 and IKS01A3 sample applications, the default parameters can be used.

From the **Configuration & Pinout** tab, click on “System Core” category and then on NVIC item to enable the EXTI line interrupts when the IKS01A2 is used:

Nucleo 64	Nucleo 144
EXTI line 4 interrupt	EXTI line 11 interrupt
EXTI line 5 interrupt	EXTI line 14 interrupt
EXTI line 10 interrupt	EXTI line 9 interrupt
EXTI line 13 interrupt	EXTI line 13 interrupt

From the **Configuration & Pinout** tab, click on “System Core” category and then on NVIC item to enable the EXTI line interrupts when the IKS01A3 is used:

Nucleo 64	Nucleo 144
EXTI line 0 interrupt	EXTI line 3 interrupt
EXTI line 1 interrupt	EXTI line 5 interrupt
EXTI line 4 interrupt	EXTI line 11 interrupt
EXTI line 5 interrupt	EXTI line 14 interrupt
EXTI line 10 interrupt	EXTI line 9 interrupt
EXTI line 13 interrupt	EXTI line 13 interrupt

From the **Configuration & Pinout** tab, click on “System Core” category and then on NVIC item to enable the EXTI line interrupts when the IKS02A1 is used:

Nucleo 64	Nucleo 144
EXTI line 13 interrupt	EXTI line 13 interrupt

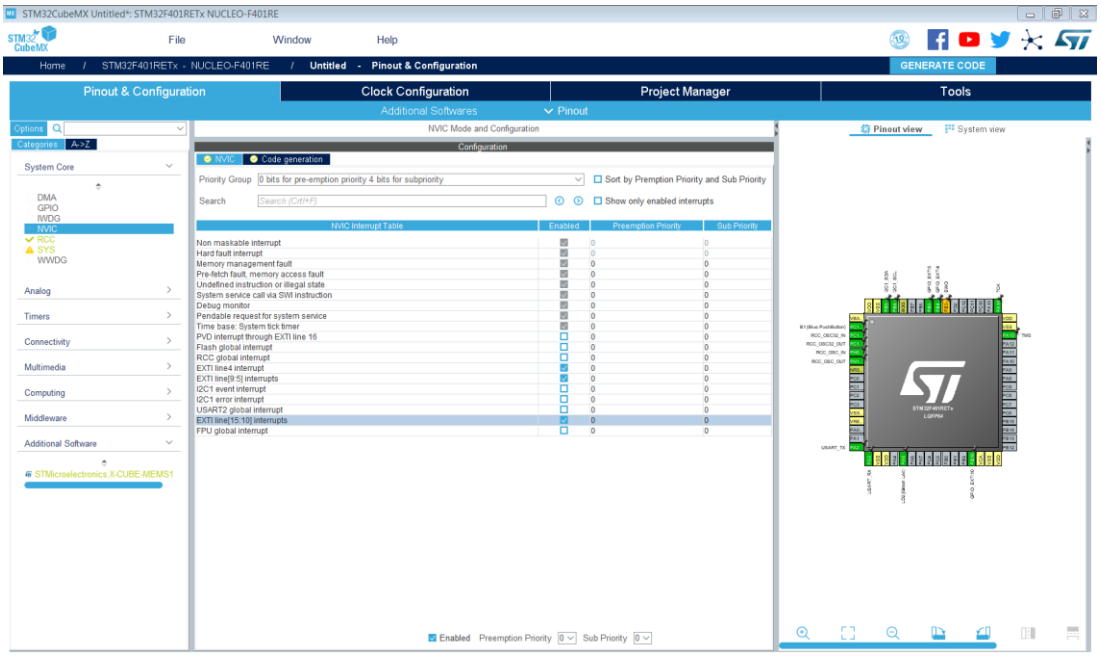

Figure 32 *STM32CubeMX NVIC Configuration for IKS01A2*

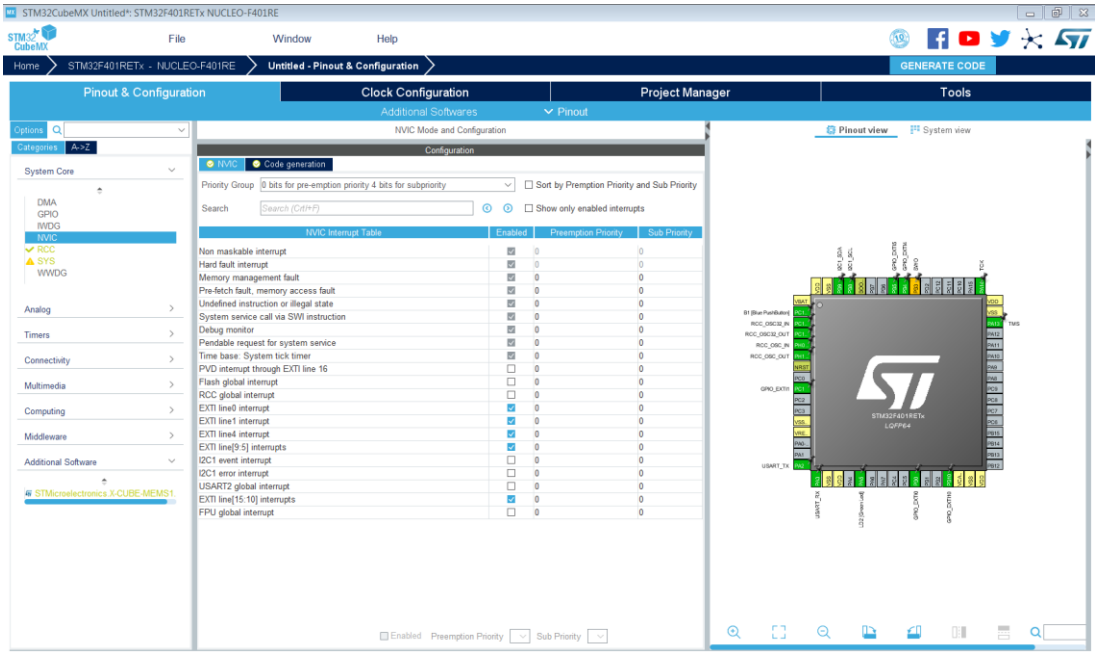

Figure 33 *STM32CubeMX NVIC Configuration for IKS01A3*

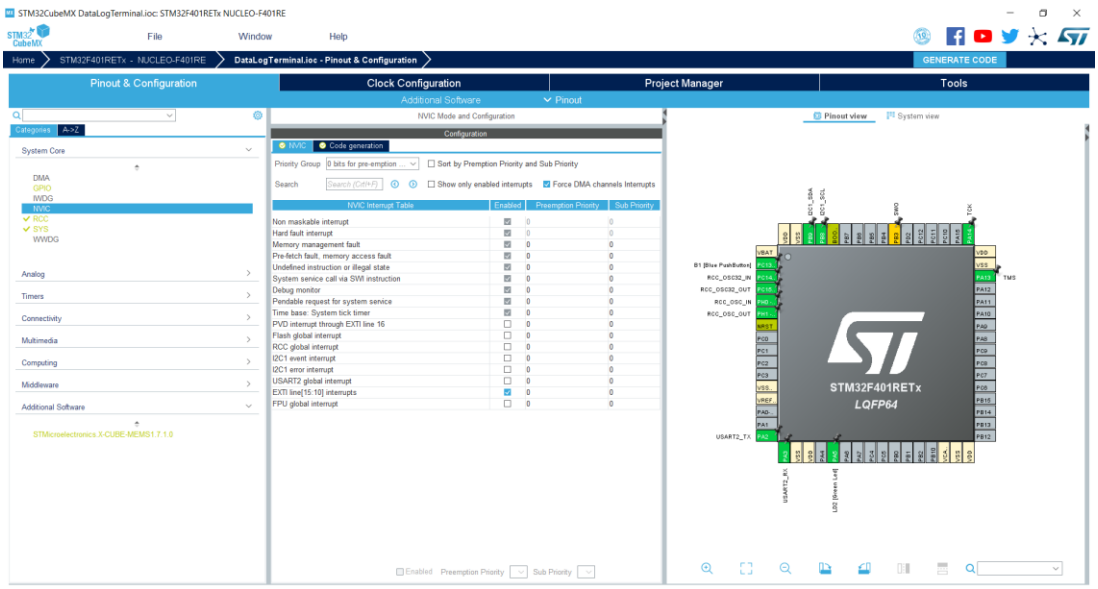

Figure 34 *STM32CubeMX NVIC Configuration for IKS02A1*

Regarding the X-NUCLEO-IKS01A3, the GPIO pin that manages the interrupt of the STTS751 component must be configured in Falling mode; instead, for the interrupts of all the other components we need to use the Rising mode.

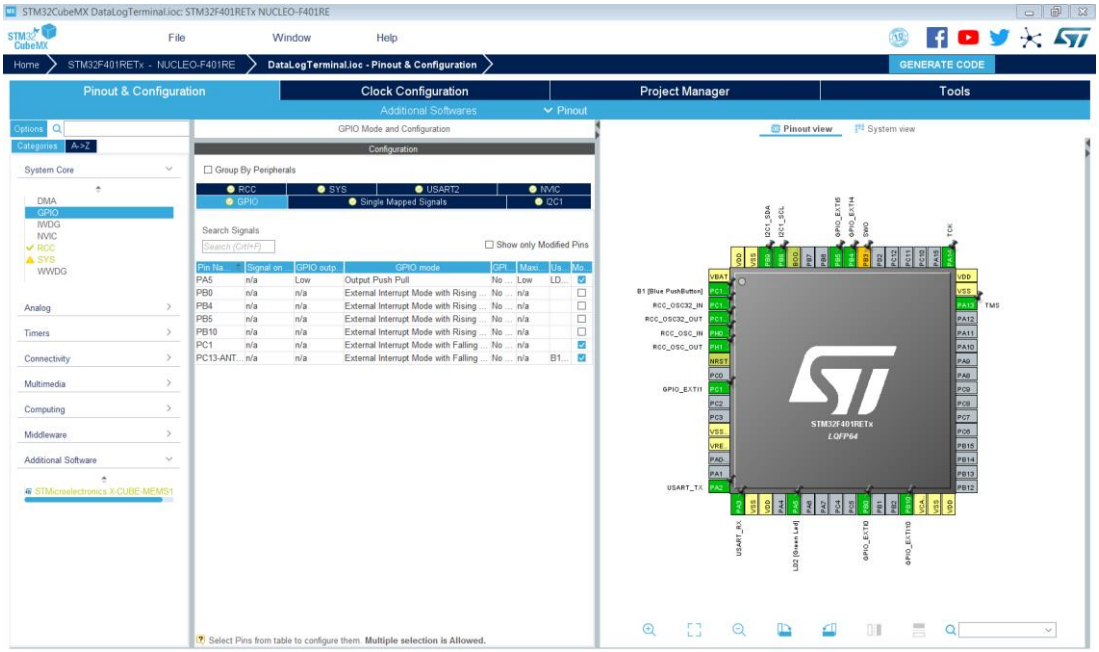

Figure 35 *STM32CubeMX GPIO Configuration for IKS01A3*

From the **Configuration & Pinout** tab, click on “Connectivity” category and then on I2C1 item to set the I2C speed at 400KHz (Fast Mode):

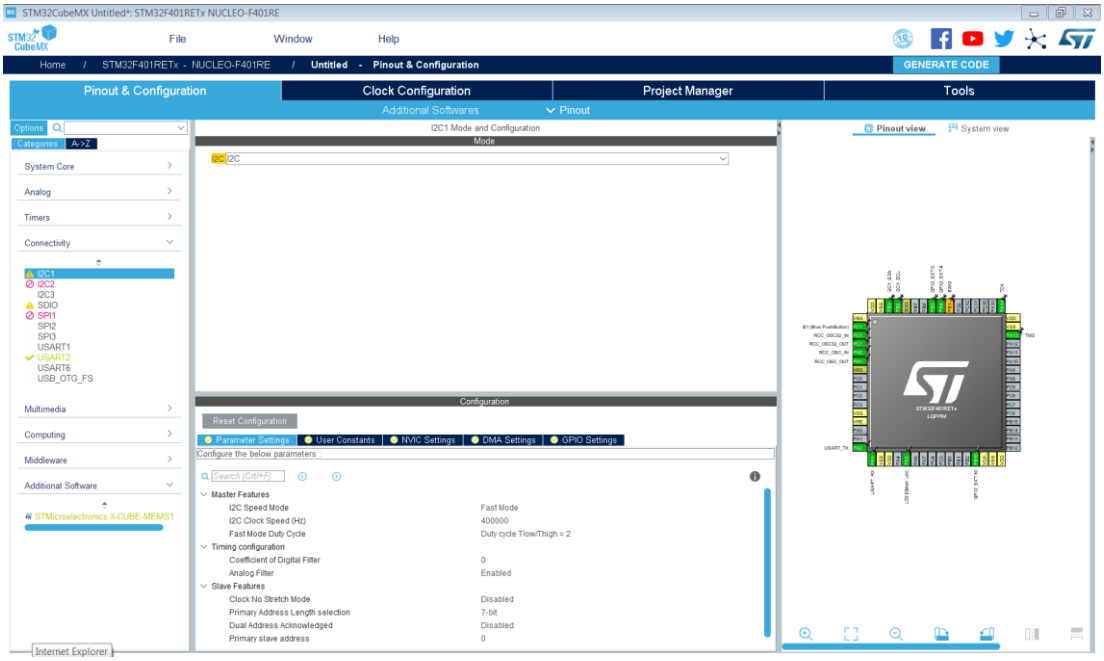

Figure 36 *STM32CubeMX I2C Configuration*

From the **Configuration & Pinout** tab, click on “Connectivity” category and then on USART2 item and check that the following configuration is set:

Baud Rate	115200 Bits/s
Word Length	8 Bits (including Parity)
Parity	None
Stop Bits	1

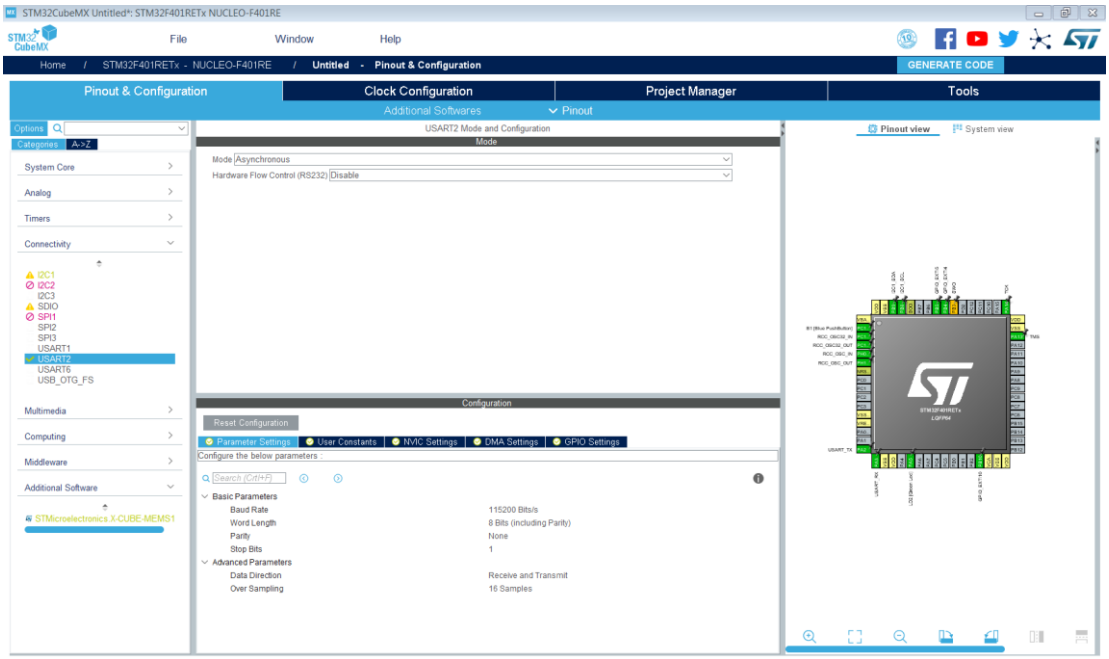

Figure 37 *STM32CubeMX USART Configuration*

Once all the above described steps have been performed, the sample applications for IKS01A2 or IKS01A3 or IKS02A1 using the **STMicroelectronics X-CUBE-MEMS1** software can be generated clicking the “GENERATE CODE” button.

7.3 Use of MEMS Library with sample applications for custom boards

This section outlines how to configure STM32CubeMX with a custom board that mounts MEMS devices when the use of the sample applications is required. In this case you can configure the MEMS device in order to be used via I2C bus or via SPI bus (only 4-Wires).

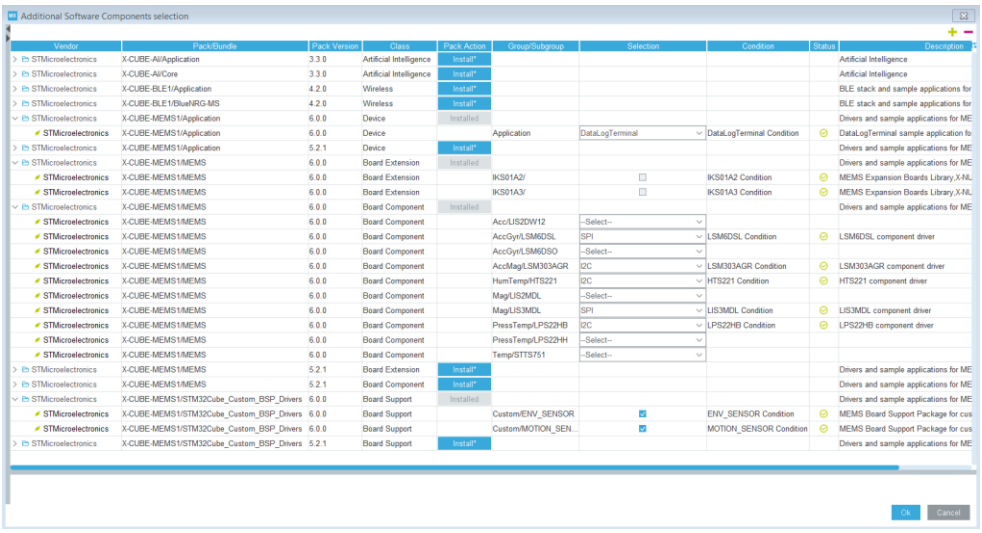

***Figure 38** STM32CubeMX **Additional Software Components selection** window*

You can see below an example of the **Pinout** scheme. In this example, we have configured SPI1 and I2C1, the Chip Select for LSM6DSL (PB6) and the Chip Select for LIS3MDL (PA8) as GPIO Output pins; we have configured USART2 connected to the hyperterminal; we have configured PA5 as LED pin; finally we have configured PC13 as GPIO Input pin (User Button). Eventually, you need to configure the EXTI pins for the applications that use interrupts like the ones based on Hardware Events of the accelerometers.

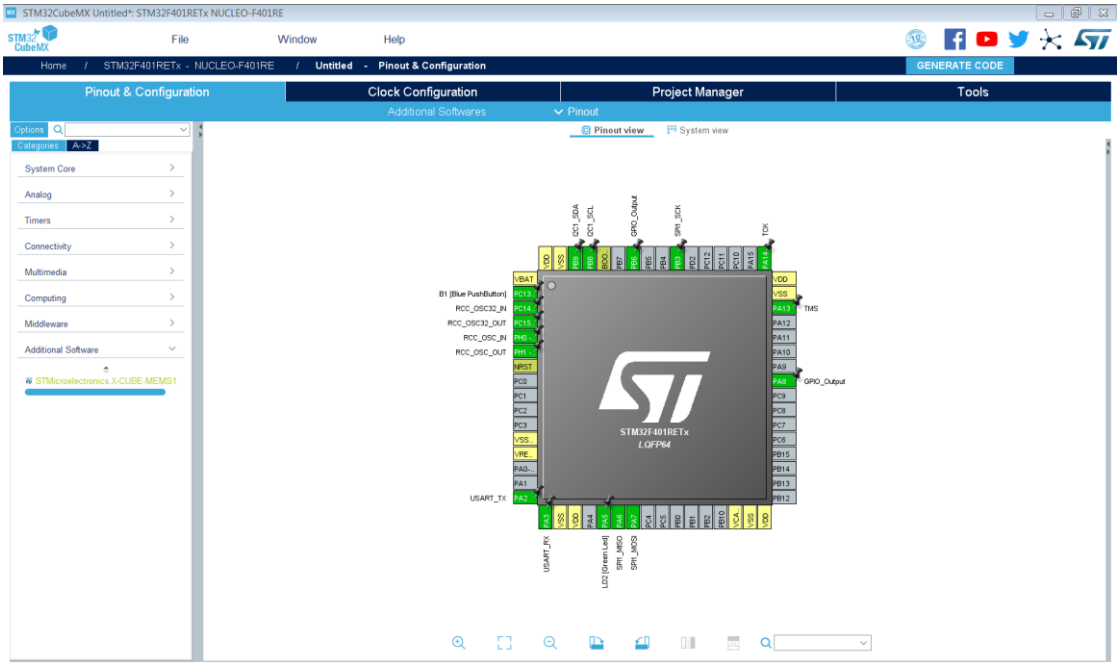

***Figure 39** STM32CubeMX **Pinout & Configuration** tab*

From the **Pinout and Configuration** tab, click on the **Additional Software** category, press the 'Stmicroelectronics.X-CUBE-MEMS1.7.1.0' item, enable the "Board Component MEMS", the "Board Support STM32Cube Custom BSP Drivers" and the "Device Application" checkboxes

from the “Mode” view and set the following Platform Settings from the “Configuration” view according your custom board (take into account that according the example chosen some settings can appear or not). A sample of configuration is shown below.

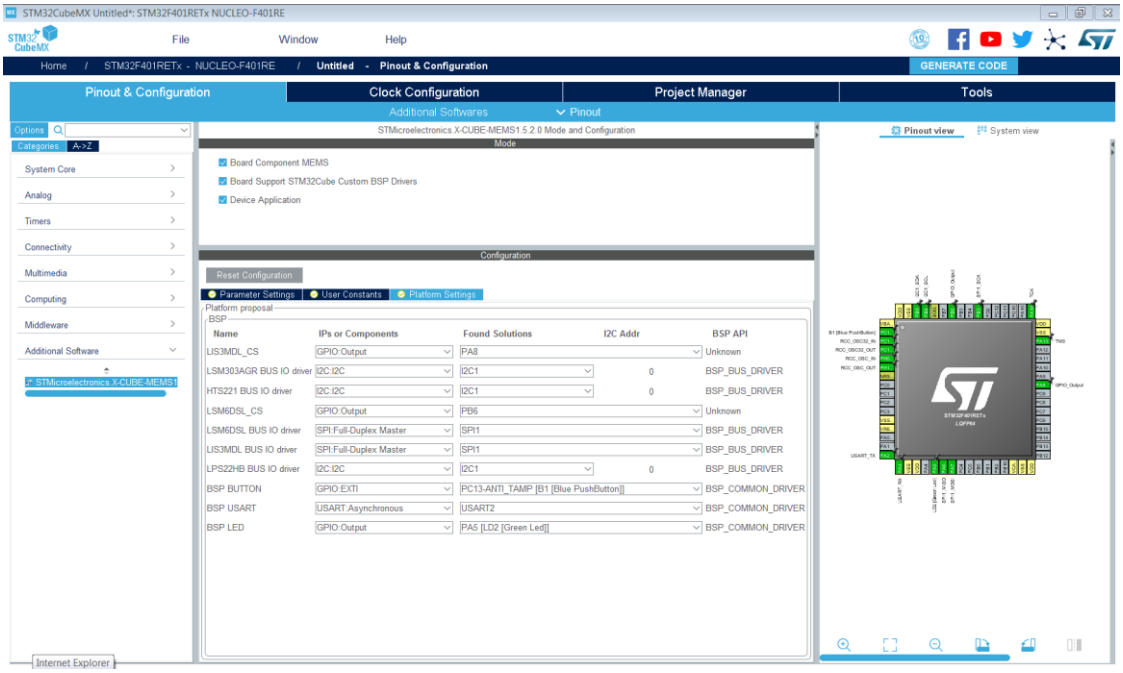

***Figure 40** STM32CubeMX **Pinout & Configuration** tab and Additional Software settings for DataLogTerminal application of a custom board*

From the **Parameter Settings** tab, some parameters of the MEMS devices can be changed. For example, the user can decide if the SA0 pin is connected to VDD or GND in order to associate the correct I2C address to the MEMS device; the Output Data Rate and the Full Scale of the MEMS sensors can be selected (only for DataLogTerminal application); some parameters for the routing of the interrupt signals can be changed. You can see an example of configuration below.

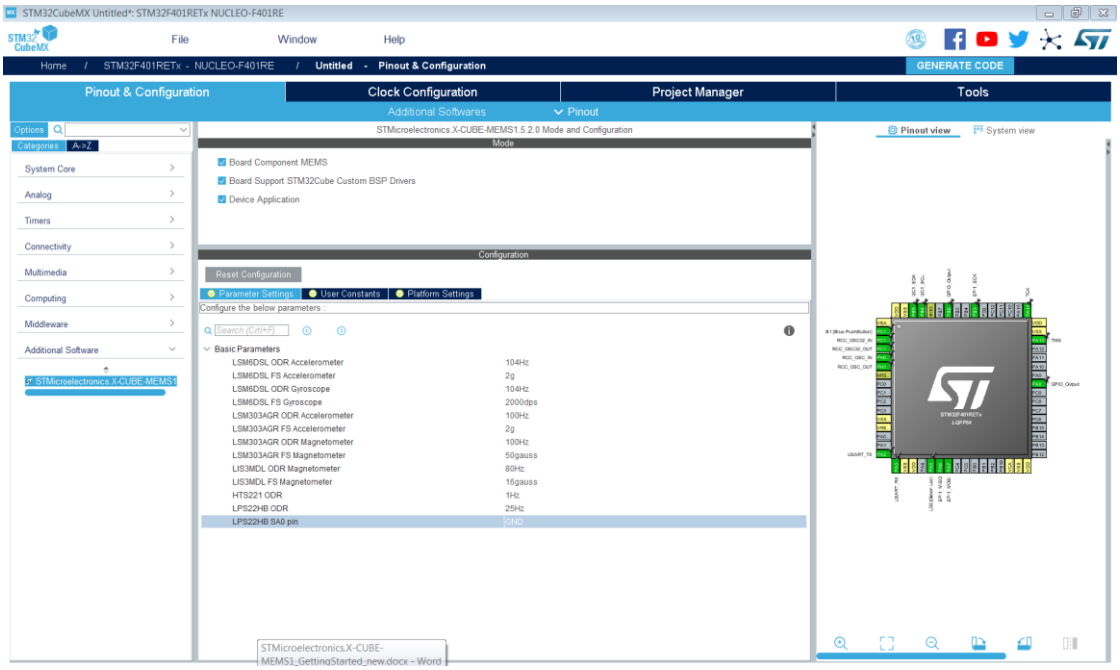

***Figure 41** STM32CubeMX **Parameter Settings** for DataLogTerminal application of a custom board*

From the **Pinout & Configuration** tab, click on “System Core” category and then on GPIO item and check that the Chip Select pins are initialized as HIGH level.

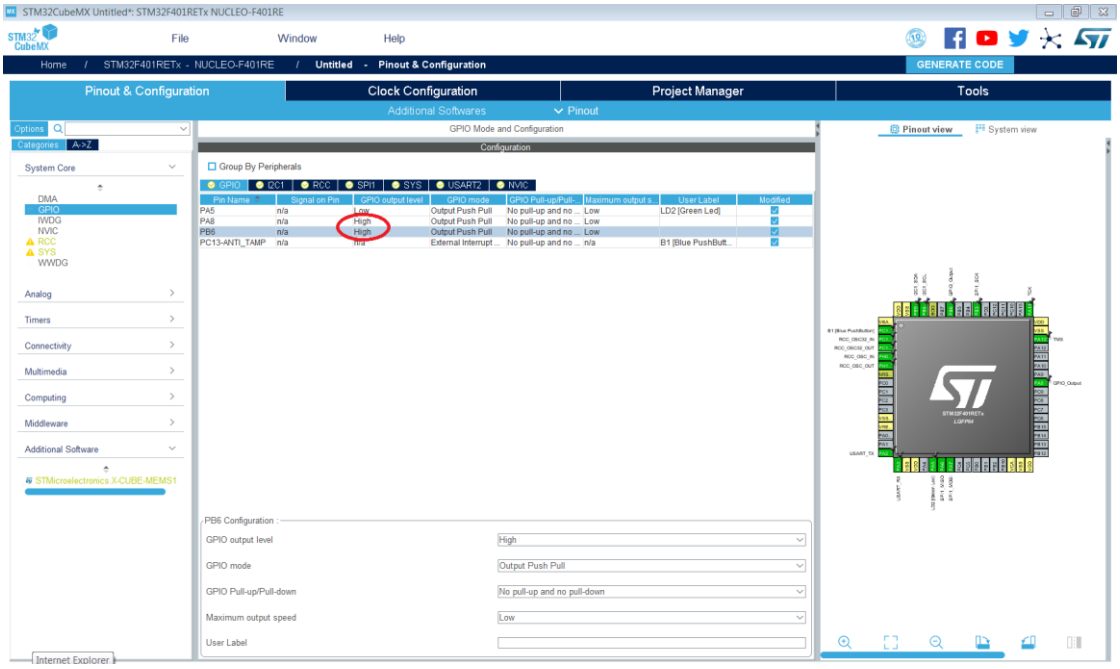

***Figure 42** STM32CubeMX GPIO Configuration*

From the **Configuration & Pinout** tab, click on “System Core” category and then on NVIC item to enable the EXTI line interrupts.

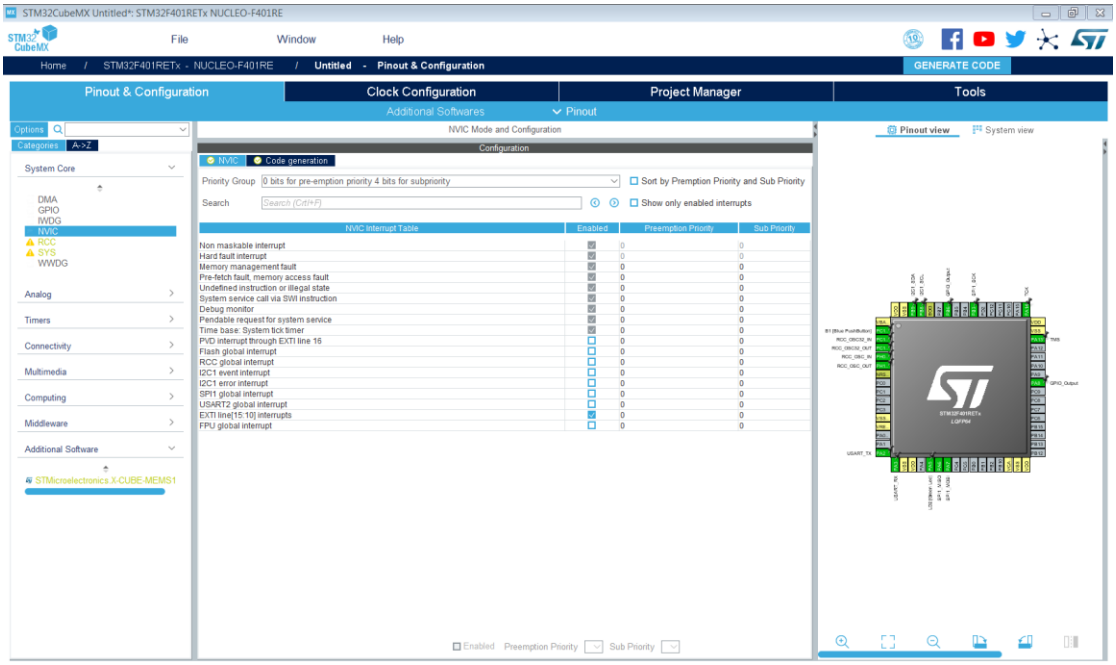

***Figure 43** STM32CubeMX NVIC Configuration*

From the **Configuration & Pinout** tab, click on “Connectivity” category and then on SPI1 item and:

- check that the Data Size is 8 Bits
- set the Prescaler (for Baud Rate) to a value so that HCLK/Prescaler is less or equal to 10MHz (the maximum supported SPI speed by MEMS devices)
- Set the Clock Polarity to “HIGH” and the Clock Phase to “2 Edge”
- Set the GPIO pin of the SPI clock with a Pull-Up state

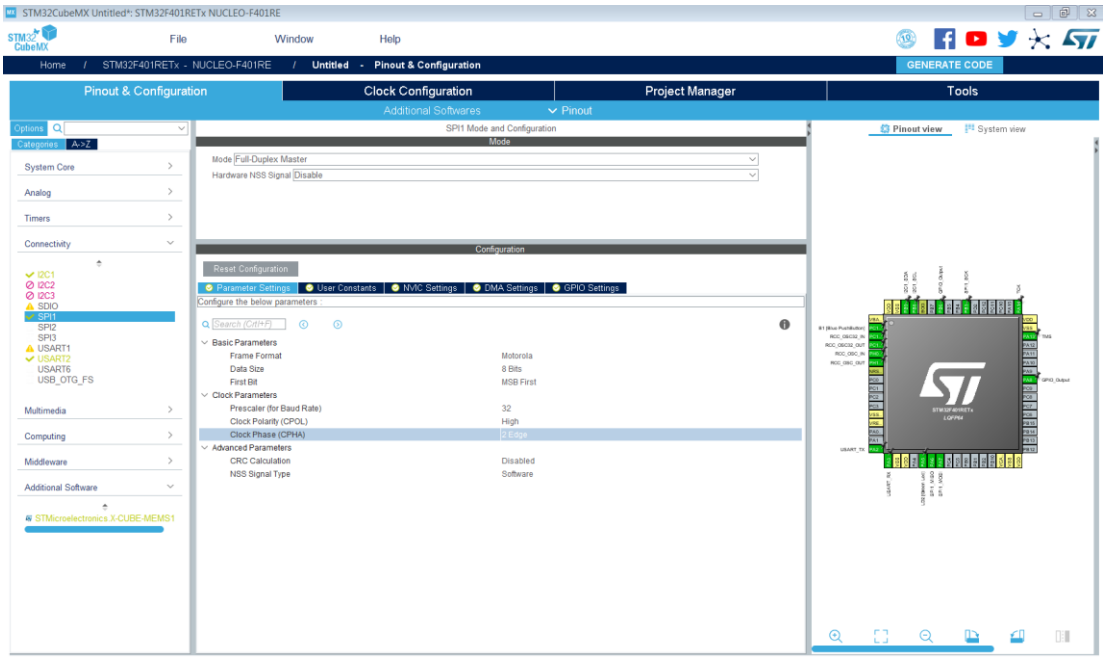

Figure 44 *STM32CubeMX SPI Configuration*

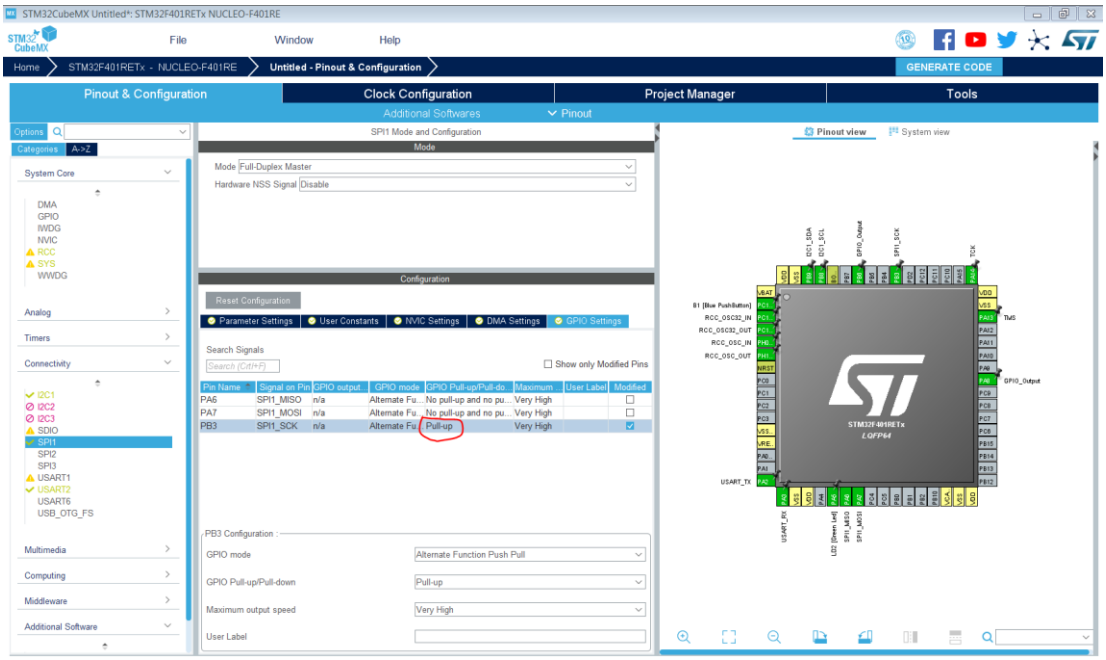

Figure 45 *STM32CubeMX SPI Clock GPIO*

From the **Configuration & Pinout** tab, click on “Connectivity” category and then on I2C1 item to set the I2C speed at 400KHz (Fast Mode):

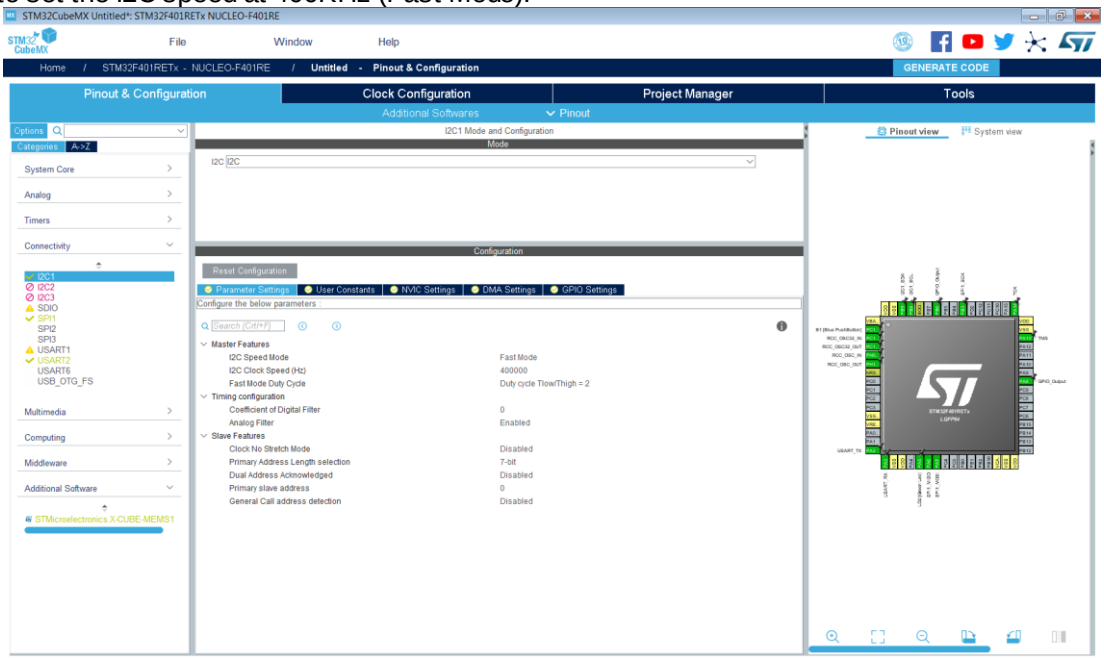

Figure 46 *STM32CubeMX I2C Configuration*

From the **Configuration & Pinout** tab, click on “Connectivity” category and then on USART2 button and check the following configuration is set:

Baud Rate	115200 Bits/s
Word Length	8 Bits (including Parity)
Parity	None
Stop Bits	1

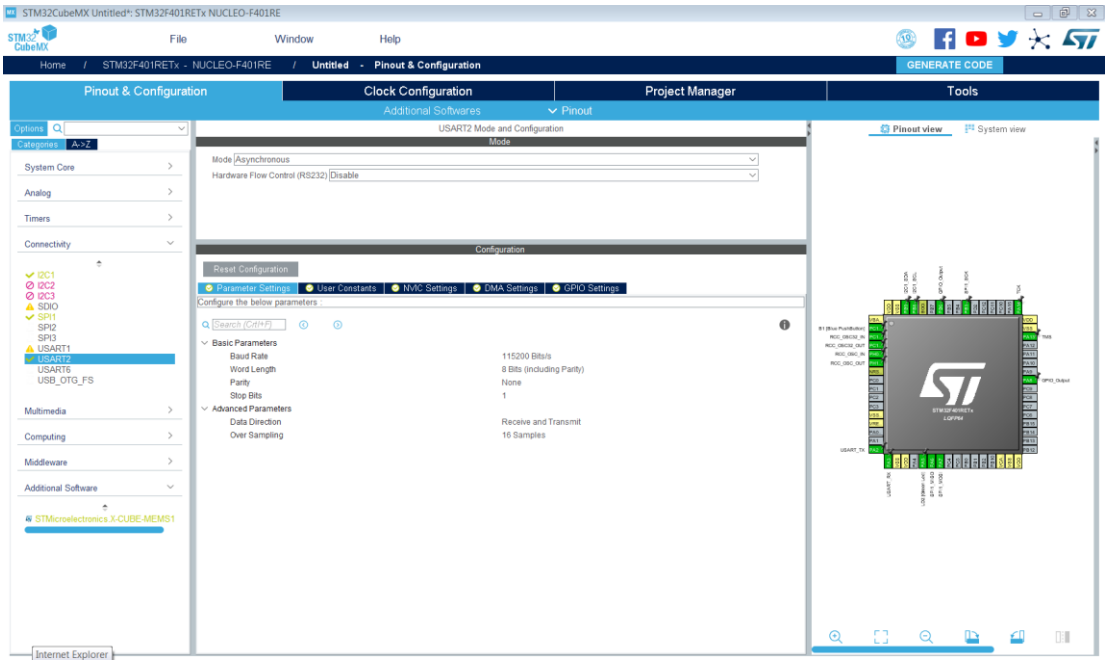

***Figure 47** STM32CubeMX USART Configuration*

Once all the above described steps have been performed, the sample applications for a custom board using the **STMicroelectronics X-CUBE-MEMS1** software can be generated clicking the "GENERATE CODE" button.

7.4 Use of MEMS Library with middleware applications for X-NUCLEO-IKS01A2 or X-NUCLEO-IKS01A3

This section outlines how to configure STM32CubeMX with X-NUCLEO-IKS01A2 or X-NUCLEO-IKS01A3 when the use of the middleware applications is required. With such setup, all the components of the expansion software package, including applications, will be properly configured.

To add the X-CUBE-MEMS1 additional software to the project, the "Additional Softwares" button must be clicked. From the "Additional Software Components selection" window, the user has to select the "Board Extension" class, an application from the "Device" class and "Motion Library" class as shown in the figure below.

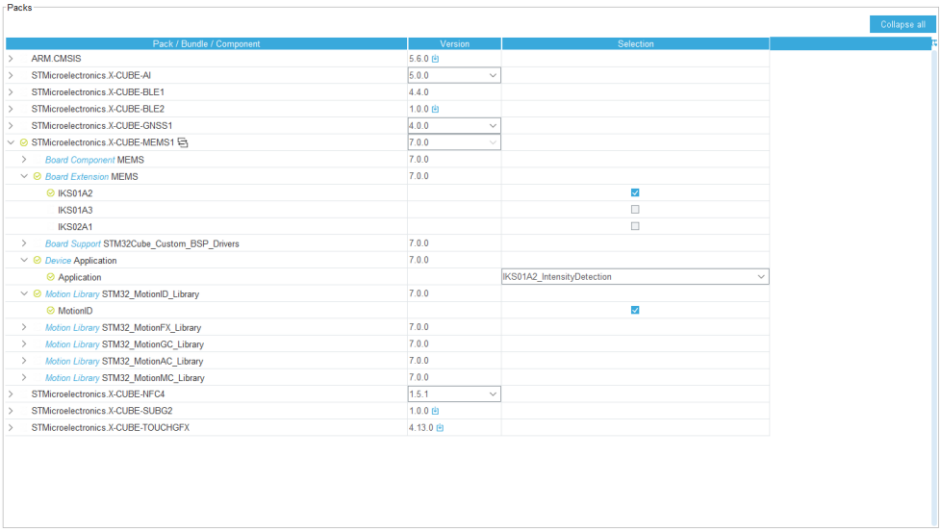

***Figure 48** STM32CubeMX **Additional Software Components selection** window example for the IKS01A2*

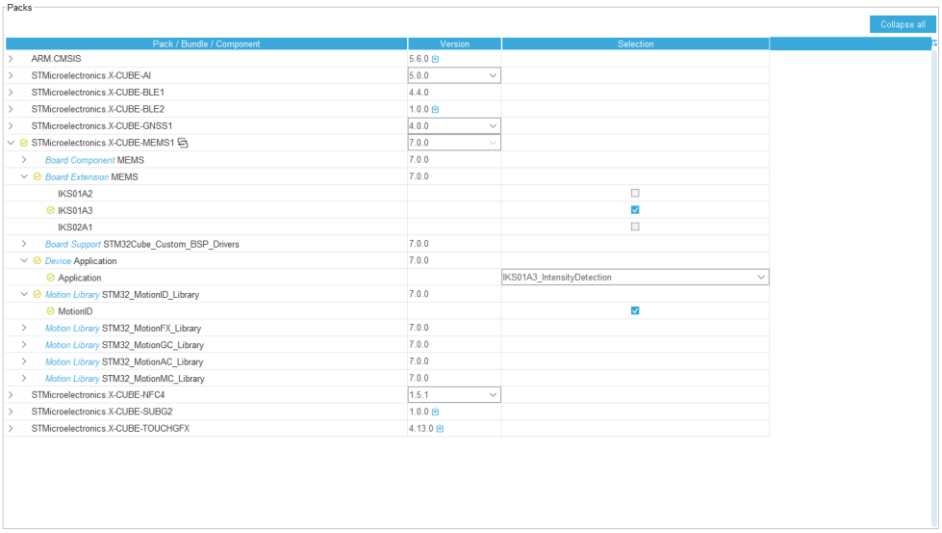

***Figure 49** STM32CubeMX **Additional Software Components selection** window example for the IKS01A3*

From the **Pinout & Configuration** tab:

- from the **Pinout** scheme, click on PB8 and set it as I2C1_SCL;
- from the **Pinout** scheme, click on PB9 and set it as I2C1_SDA;
- enable the I2C1 as I2C from the “Connectivity” category;
- if not enabled yet:
    a. enable the USART2 in Asynchronous mode (for Nucleo 64) from the “Connectivity” category
    b. enable the USART3 in Asynchronous mode (for Nucleo 144) from the “Connectivity” category

From the **Pinout** scheme, if not already set:

Nucleo 64			Nucleo 144		
PIN	*Mode*	*Label*	*PIN*	*Mode*	*Label*
PA2	USART2_TX	USART_TX	PD8	USART3_TX	STLK_RX
PA3	USART2_RX	USART_RX	PD9	USART3_RX	STLK_TX
PA5	GPIO_Output	LD2 [Green Led]	PB7	GPIO_Output	LD2[Blue]
PC13	GPIO_EXTI13	B1 [Blue PushButton]	PC13	GPIO_EXTI13	USER_Btn [B1]

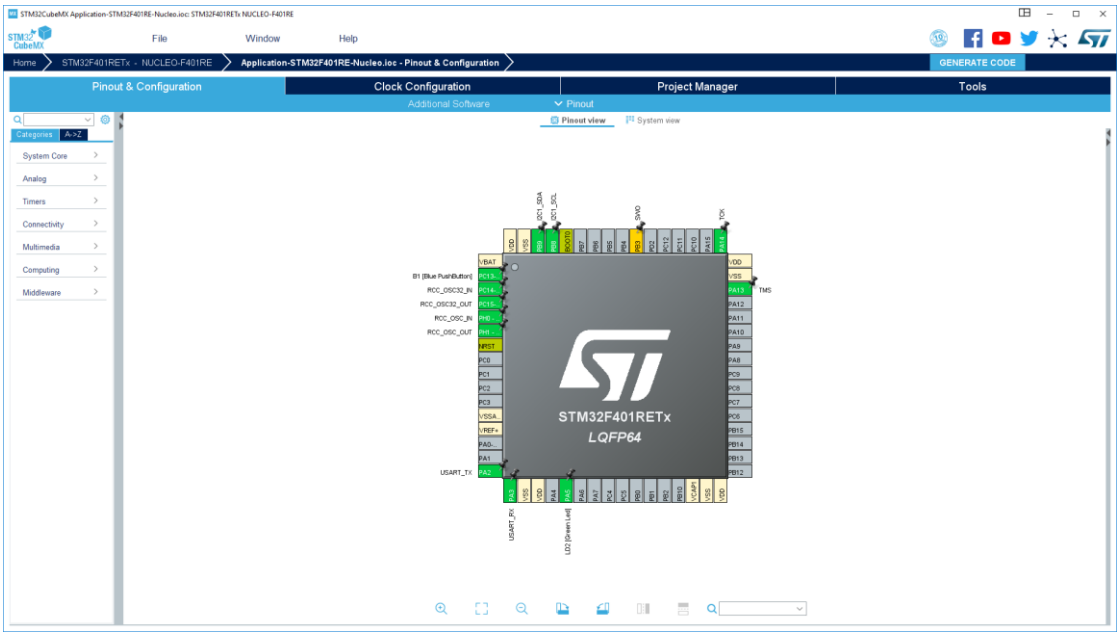

Figure 50 *STM32CubeMX* ***Pinout & Configuration*** *tab for Nucleo 64*

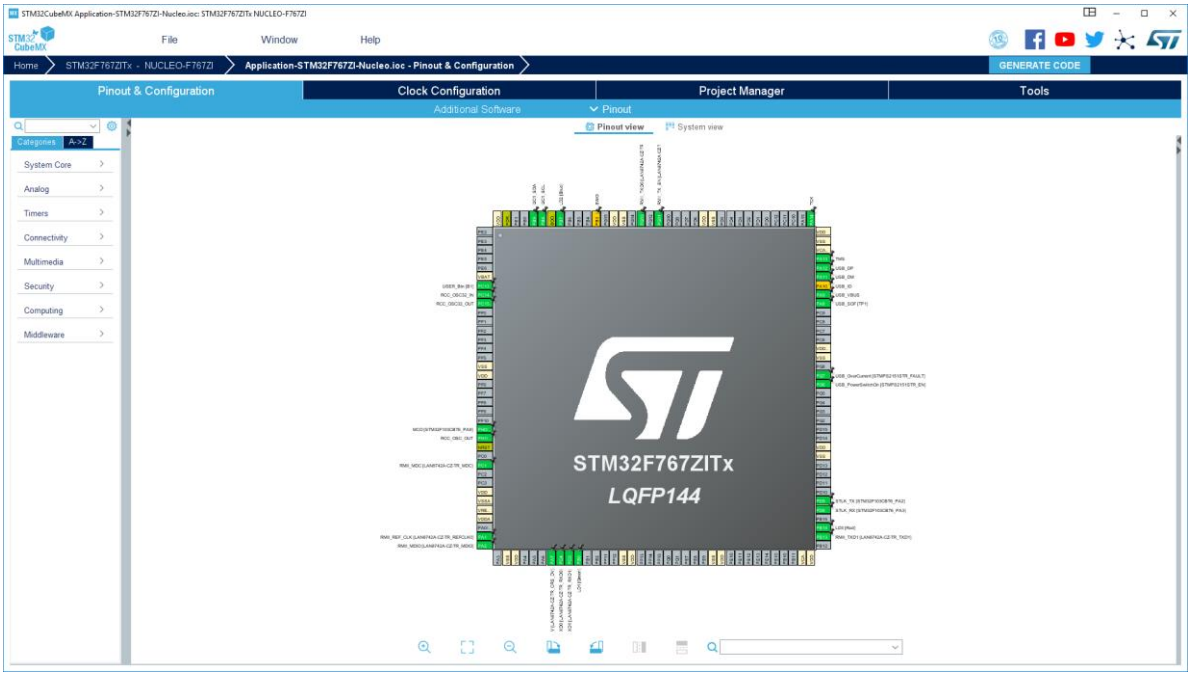

Figure 51 *STM32CubeMX* ***Pinout & Configuration*** *tab for Nucleo 144*

From the **Additional Software** category, press the 'STMicroelectronics.X-CUBE-MEMS1.7.1.0' item, enable the "Board Extension MEMS", the "Device Application" and all the "Motion Library STM32 MotionXX Library" checkboxes from the "Mode" view.

Set the following "Platform Settings" from the "Configuration" view for Nucleo 64:

Name	IPs or Components	Found Solutions	BSP API
TIMER	TIM1_8:Internal Clock	TIM3	Unknown
BSP USART	USART:Asynchronous	USART2	BSP_COMMON_DRIVER
IKS01AX BUS IO driver	I2C:I2C	I2C1	BSP_BUS_DRIVER
BSP LED	GPIO:Output	PA5	BSP_COMMON_DRIVER

Set the following "Platform Settings" from the "Configuration" view for Nucleo 144:

Name	IPs or Components	Found Solutions	BSP API
TIMER	TIM1_8F77:Internal Clock	TIM3	Unknown
BSP USART	USART:Asynchronous	USART3	BSP_COMMON_DRIVER
IKS01AX BUS IO driver	I2C:I2C	I2C1	BSP_BUS_DRIVER
BSP LED	GPIO:Output	PB7	BSP_COMMON_DRIVER

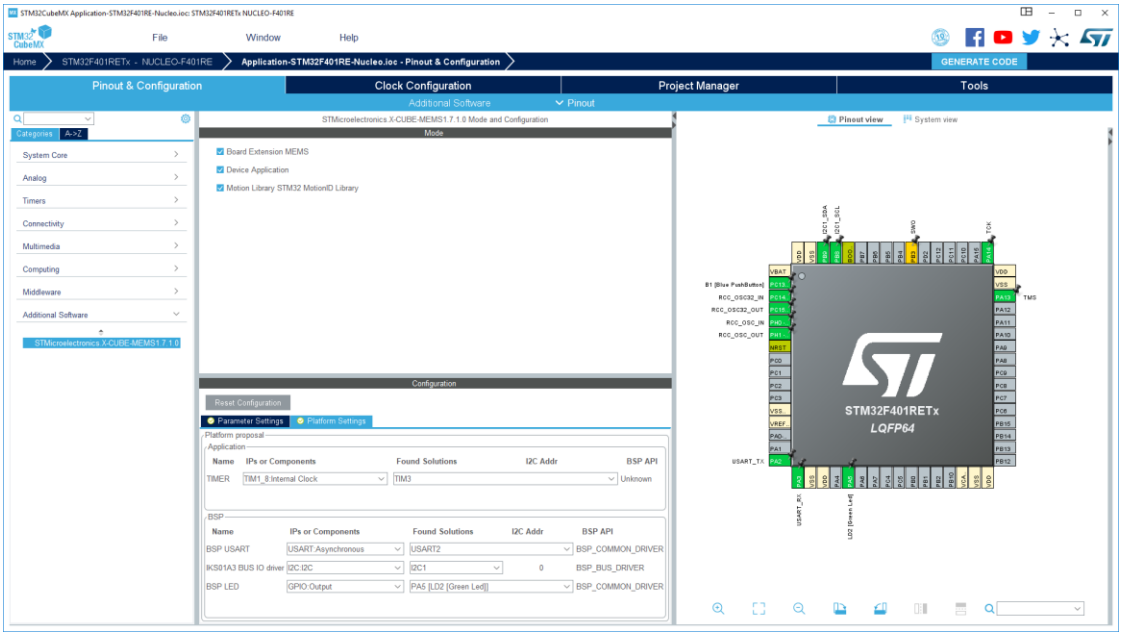

***Figure 52** STM32CubeMX **Pinout & Configuration** tab and Additional Software settings for Nucleo 64*

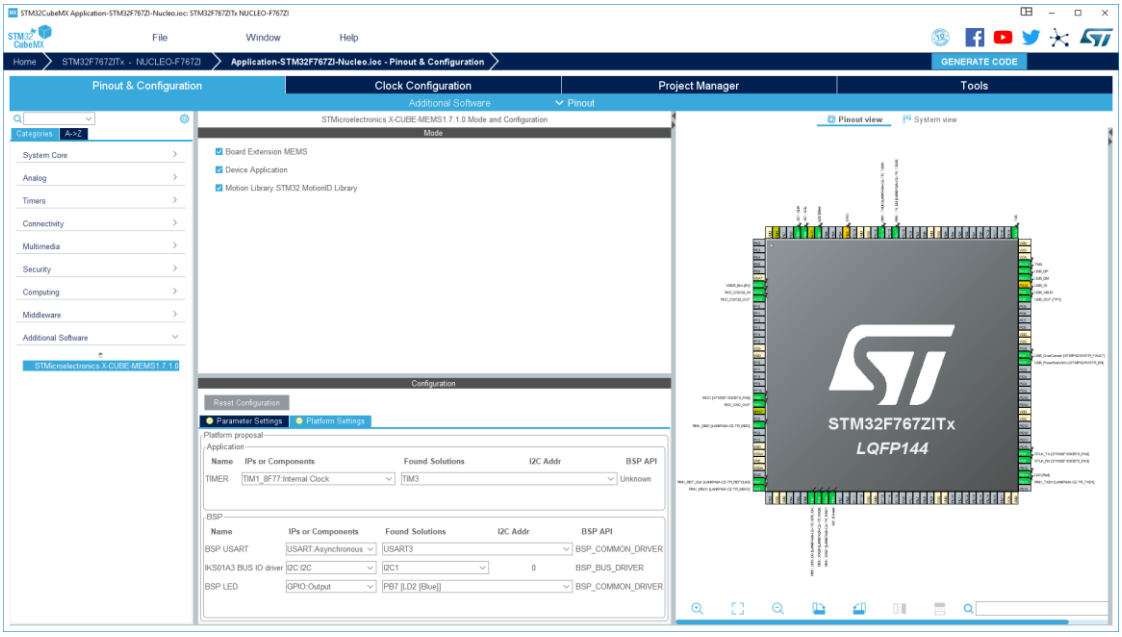

***Figure 53** STM32CubeMX **Pinout & Configuration** tab and Additional Software settings for Nucleo 144*

Setup RTC to use real time clock and date stamp in data stream sent from application to Unicleo-GUI. From the **Pinout & Configuration** tab, click on "Timers" category and then on RTC item:

- Check "Activate Clock Source"
- Check "Activate Calendar"

No need to fill the initial time and date since the FW will receive this information from Unicleo-GUI during startup connection.

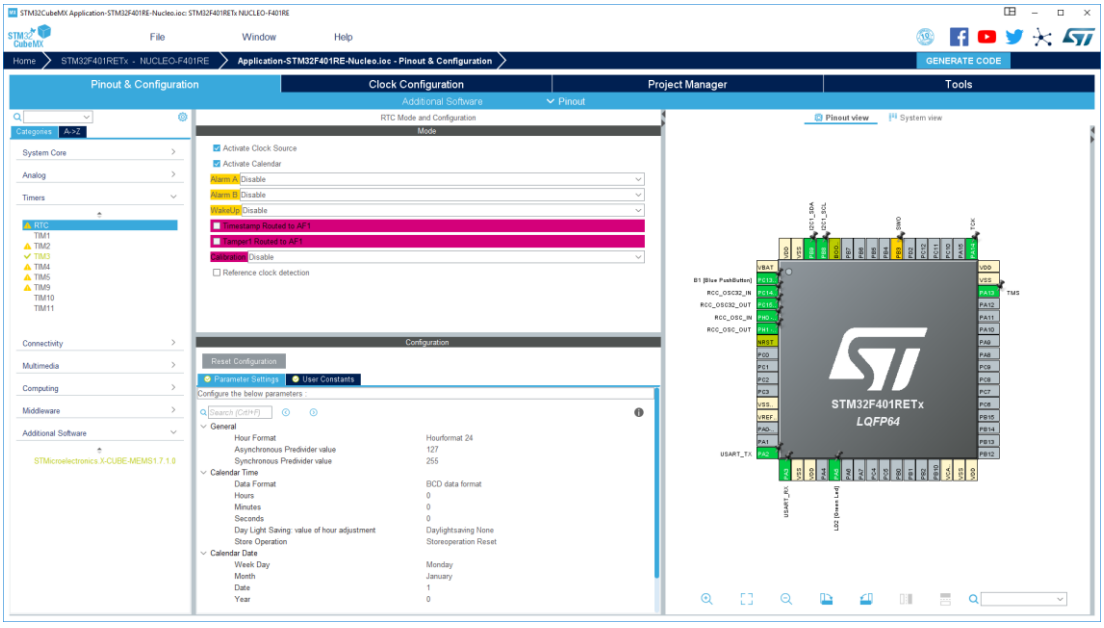

***Figure 54** STM32CubeMX RTC Configuration for Nucleo 64*

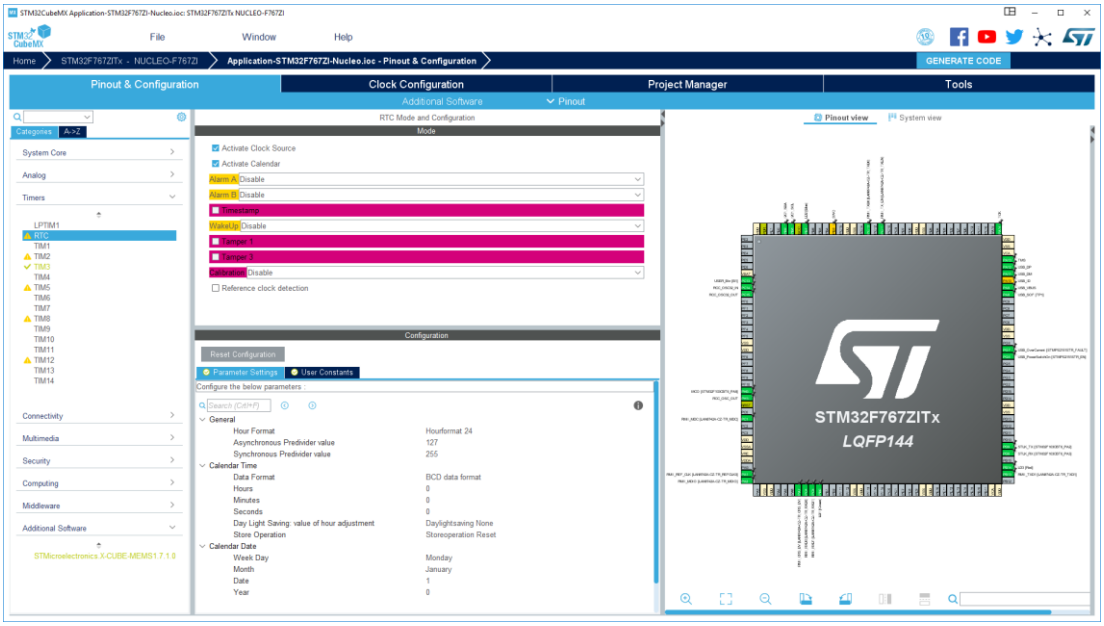

Figure 55 *STM32CubeMX RTC Configuration for Nucleo 144*

Setup Timer for data acquisition precise timing. The selected Timer instance has to be able to raise the interrupt after counting to the desired value. From the **Pinout & Configuration** tab, click on "Timers" category and then on TIM3 item:

- Set "Clock Source" to "Internal Clock"
- In "NVIC Settings" tab check "TIM3 global interrupt"

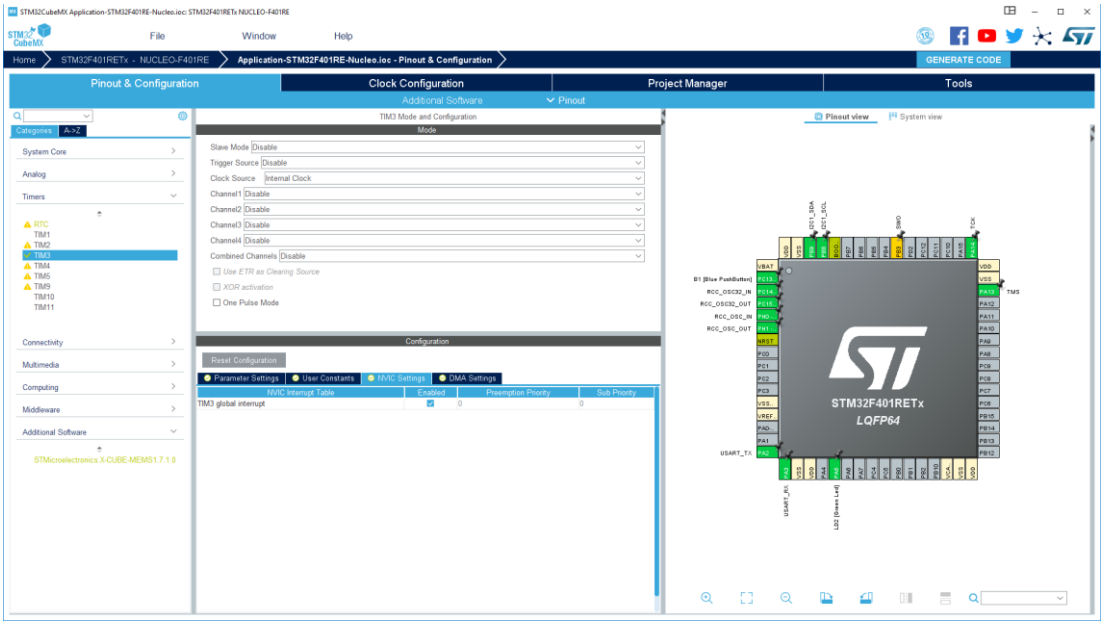

Figure 56 *STM32CubeMX TIM3 Configuration for Nucleo 64*

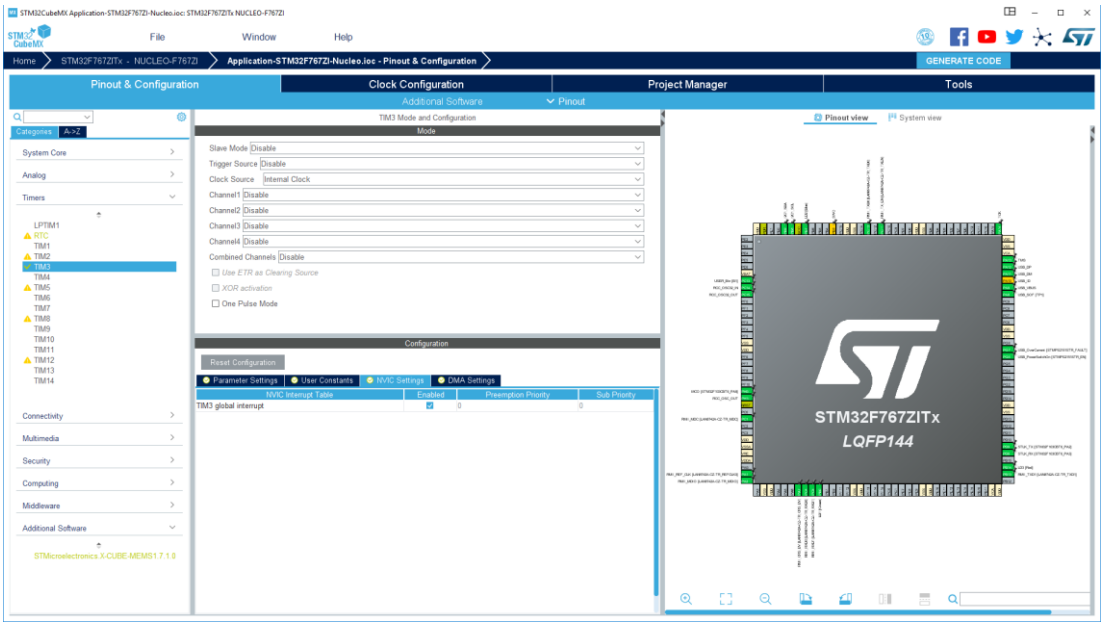

Figure 57 *STM32CubeMX TIM3 Configuration for Nucleo 144*

Setup I2C for communication between sensor and MCU. From the **Pinout & Configuration** tab, click on "Connectivity" category and then on I2C1 item:

- Set I2C to I2C
- The rest is filled automatically

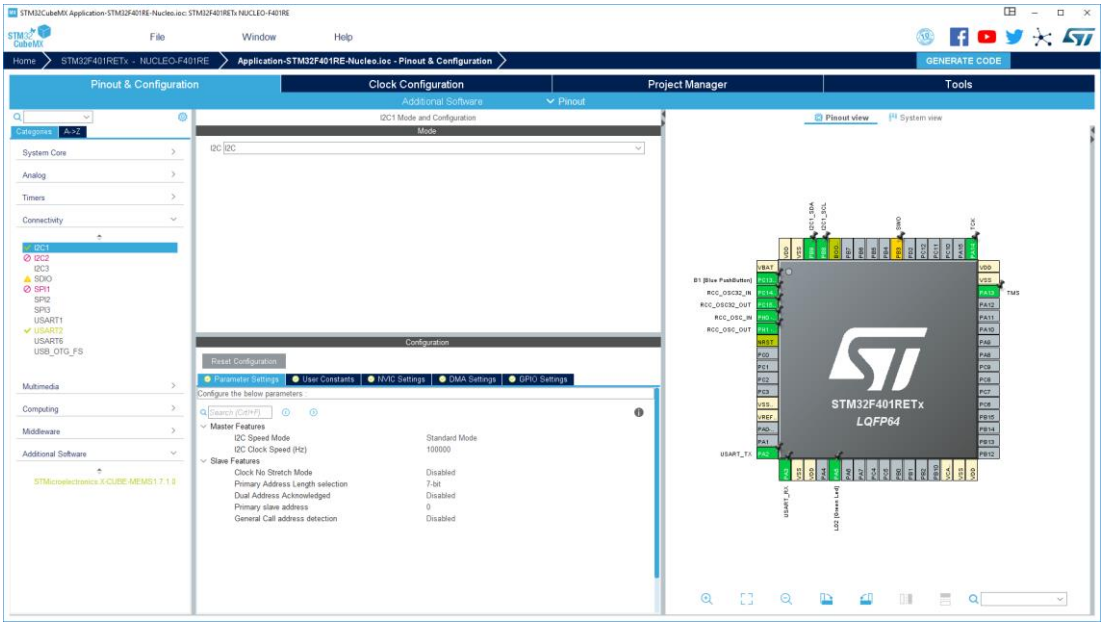

Figure 58 *STM32CubeMX I2C1 Configuration for Nucleo 64*

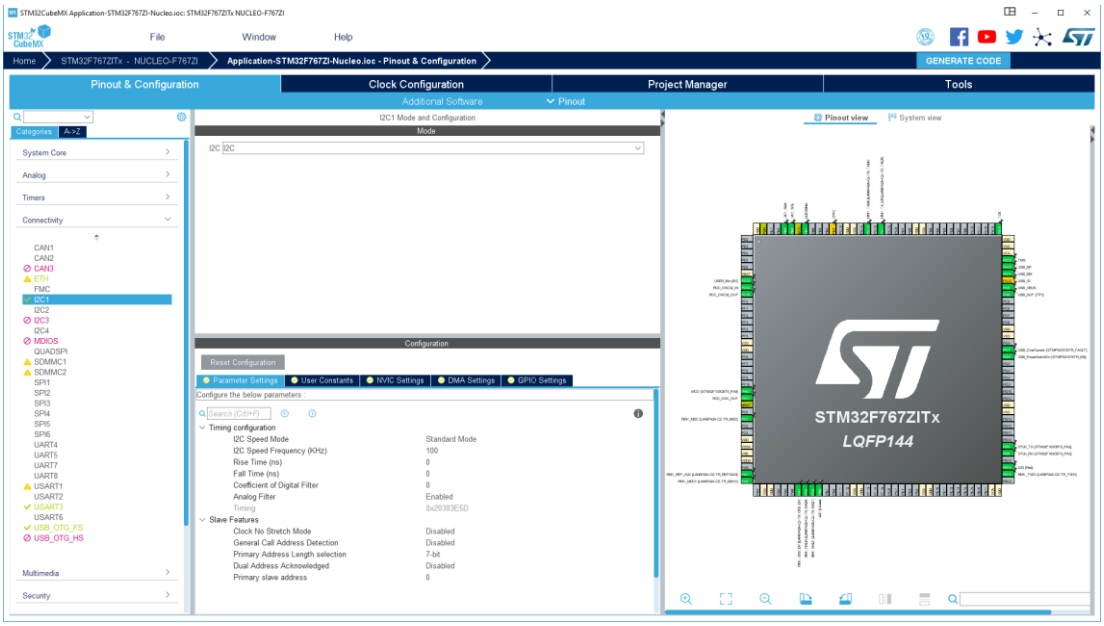

Figure 59 *STM32CubeMX I2C1 Configuration for Nucleo 144*

Setup USART for communication between MCU (FW) and PC (Unicleo-GUI). From the **Pinout & Configuration** tab, click on “Connectivity” category and then on USART2 item (Nucleo 64) or USART3 item (Nucleo 144):

- Select Asynchronous Mode

At “Parameter Settings” tab:

- Set Baud Rate to 921600 Bits/s
- Set Word Length to 8 Bits (including Parity)
- Set Parity to None
- Set Stop Bits to 1
- The rest is filled automatically

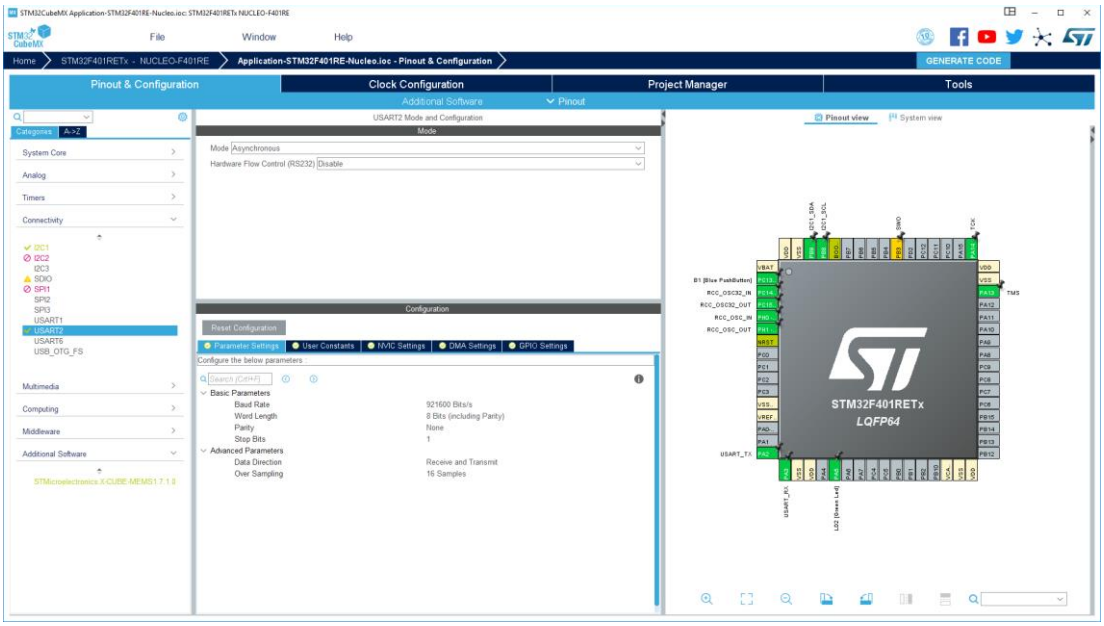

Figure 60 *STM32CubeMX USART2 Configuration for Nucleo 64*

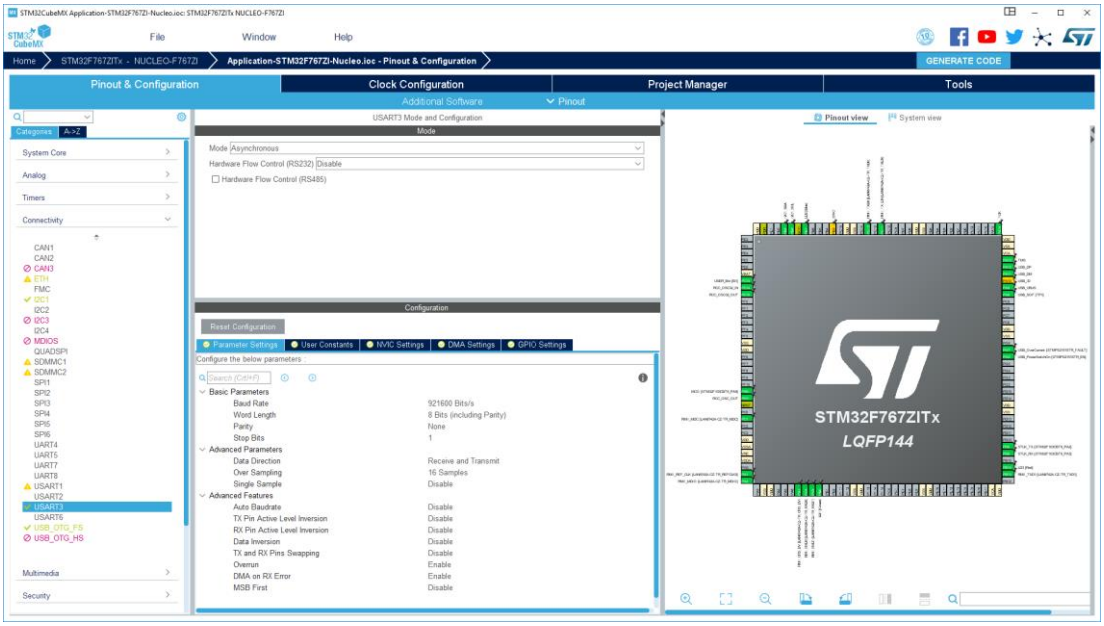

Figure 61 *STM32CubeMX USART3 Configuration for Nucleo 144*

Setup USART DMA for receiving data from Unicleo-GUI. At “DMA Settings” tab:

- Click Add button and select USART2_RX for Nucleo 64 or USART3_RX for Nucleo 144
- Select Circular Mode
- Check Memory Increment Address
- The rest is filled automatically

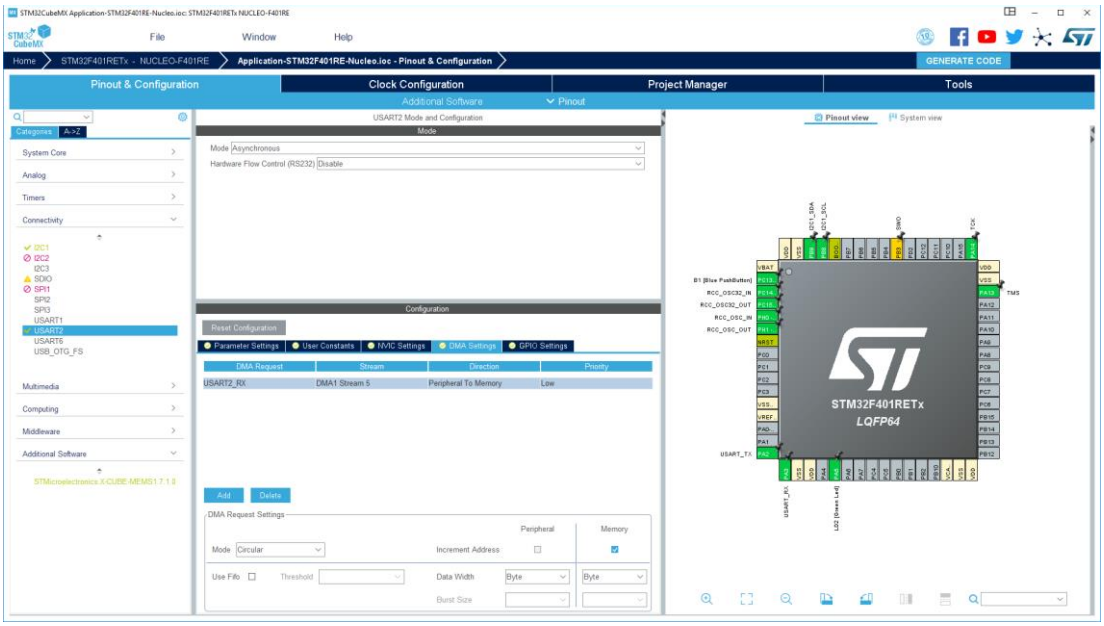

Figure 62 *STM32CubeMX USART2_RX DMA Configuration for Nucleo 64*

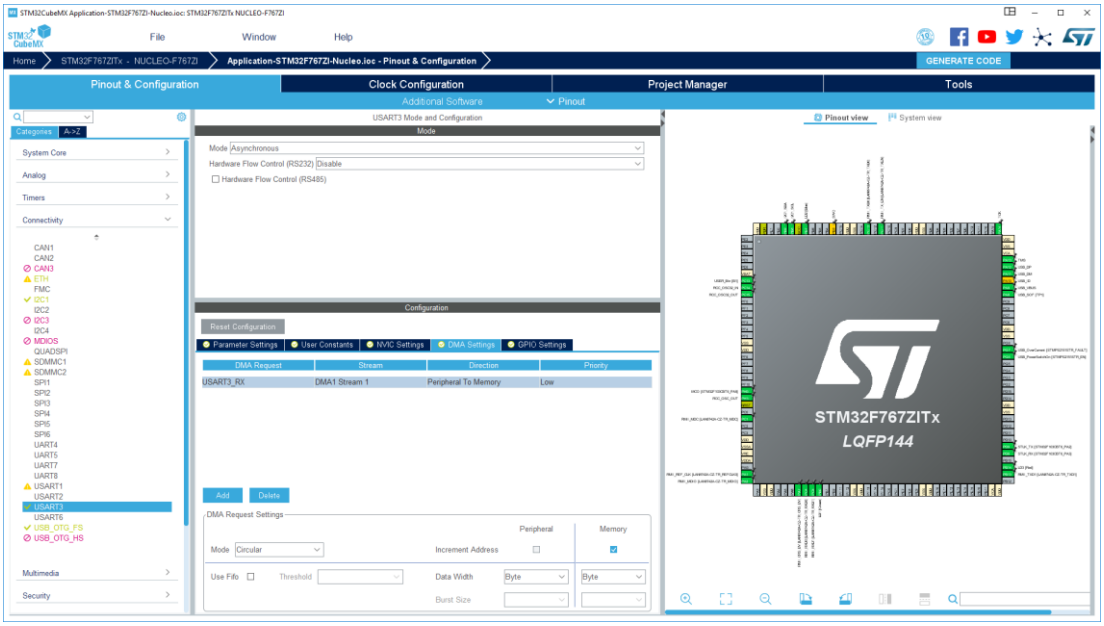

Figure 63 *STM32CubeMX USART3_RX DMA Configuration for Nucleo 144*

Setup CRC for Cyclic Redundancy Check used for library integrity verification. From the **Pinout & Configuration** tab, click on “Computing” category and then on CRC item:

- Check Activated Mode

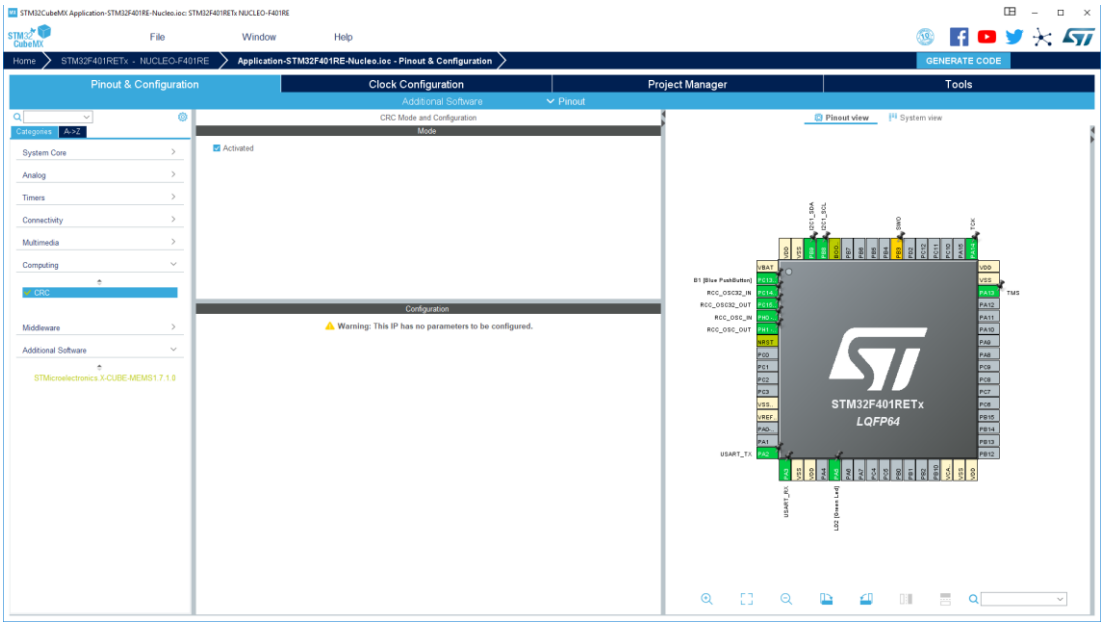

Figure 64 *STM32CubeMX CRC Configuration for Nucleo 64*

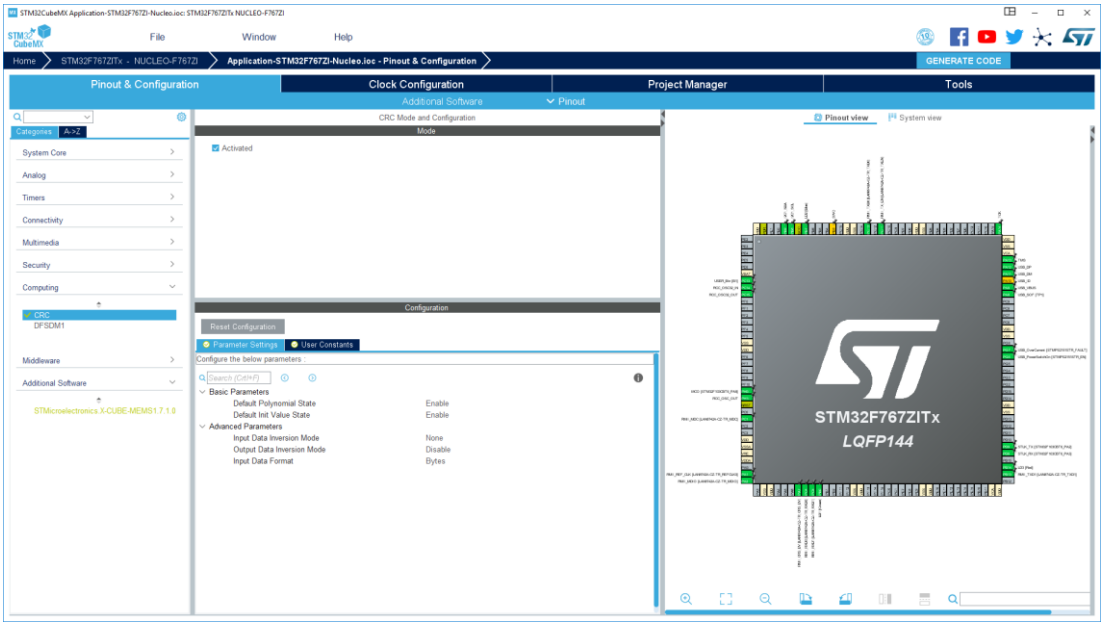

Figure 65 *STM32CubeMX CRC Configuration for Nucleo 144*

Once all the above described steps have been performed, the middleware applications for IKS01A2 or IKS01A3 using the **STMicroelectronics X-CUBE-MEMS1** software can be generated clicking the “GENERATE CODE” button.

7.5 Use of MEMS Library with middleware applications for custom boards

This section outlines how to configure STM32CubeMX with a custom board that mounts MEMS devices when the use of the middleware applications is required. In this case you can configure the MEMS device in order to be used via I2C bus or via SPI bus (only 4-Wires).

To add the X-CUBE-MEMS1 additional software to the project, the “Additional Softwares” button must be clicked. From the “Additional Software Components selection” window, the user has to select the “Board Support” class, an application from the “Device” class and “Motion Library” class as shown in the figure below.

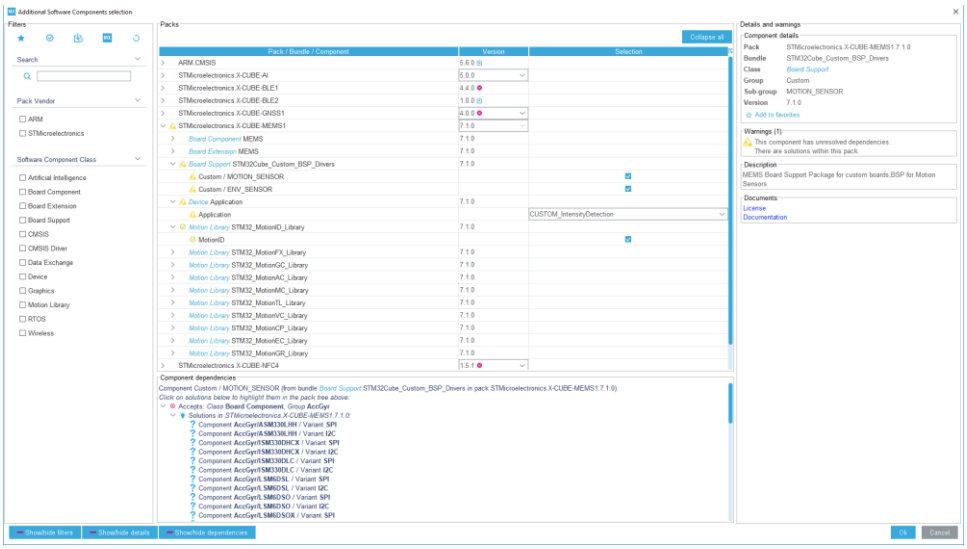

***Figure 66** STM32CubeMX **Additional Software Components selection** window example for a custom board - Step 1*

Then user has to select the “Board Component” class and add all sensors according to real custom board configuration.

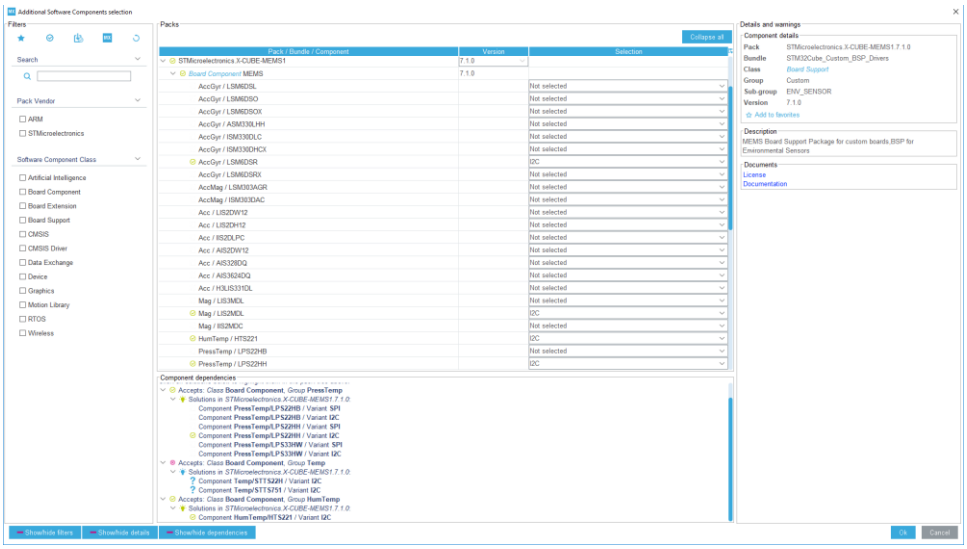

***Figure 67** STM32CubeMX **Additional Software Components selection** window example for a custom board – Step 2*

From the **Pinout & Configuration** tab:

- from the **Pinout** scheme, click on PB8 and set it as I2C1_SCL;
- from the **Pinout** scheme, click on PB9 and set it as I2C1_SDA;
- enable the I2C1 as I2C from the "Connectivity" category;
- if not enabled yet:
    a. enable the USART2 in Asynchronous mode (for Nucleo 64) from the "Connectivity" category
    b. enable the USART3 in Asynchronous mode (for Nucleo 144) from the "Connectivity" category

From the **Pinout** scheme, if not already set:

Nucleo 64			**Nucleo 144**		
PIN	*Mode*	*Label*	*PIN*	*Mode*	*Label*
PC0	GPIO_Input		PF10 [1)]	GPIO_Input	
PA2	USART2_TX	USART_TX	PD8	USART3_TX	STLK_RX
PA3	USART2_RX	USART_RX	PD9	USART3_RX	STLK_TX
PA5	GPIO_Output	LD2 [Green Led]	PB7	GPIO_Output	LD2[Blue]
PC13	GPIO_EXTI13	B1 [Blue PushButton]	PC13	GPIO_EXTI13	USER_Btn [B1]

1) Might differ for various Nucleo 144 boards – e.g.: PD13, PC5. Check with datasheet.

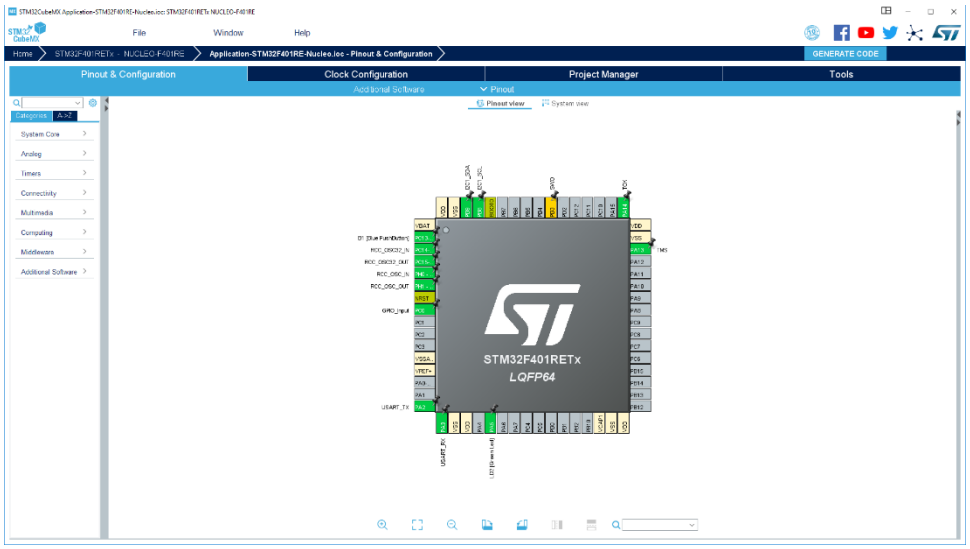

***Figure 68** STM32CubeMX **Pinout & Configuration** tab for Nucleo 64*

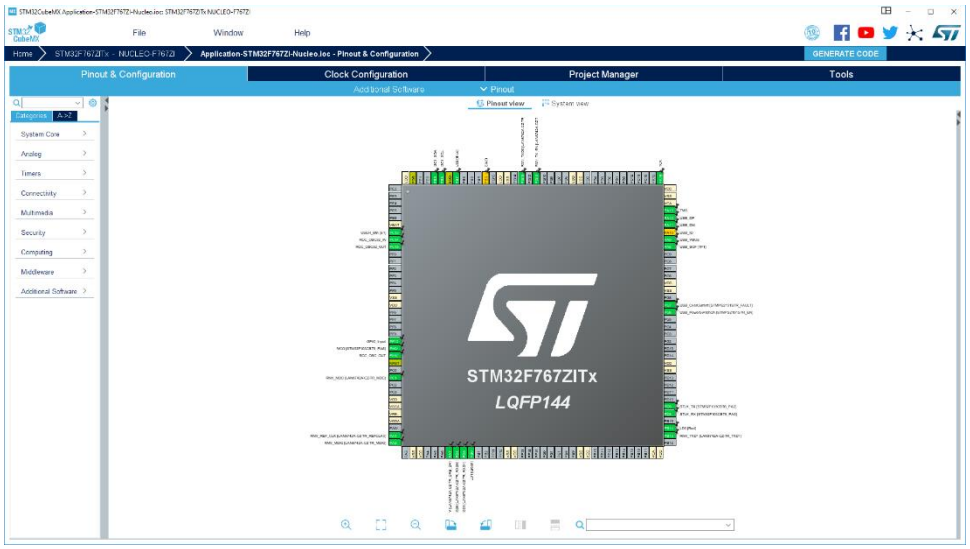

***Figure 69** STM32CubeMX **Pinout & Configuration** tab for Nucleo 144*

From the **Additional Software** category, press the 'STMicroelectronics.X-CUBE-MEMS1.7.1.0' item, enable the "Board Components MEMS", the "Board Support STM32Cube Custom BSP Drivers", the "Device Application" and all the "Motion Library STM32 MotionXX Library" checkboxes from the "Mode" view.

Set the "Platform Settings" and the "Parameter Settings" from the "Configuration" view according your custom board (take into account that according the example chosen some settings can appear or not). A sample of configuration for Nucleo 64 is shown below:

Name	IPs or Components	Found Solutions	BSP API
TIMER	TIM1_8:Internal Clock	TIM3	Unknown
MEMS INT1	GPIO:Input	PC0	Unknown
LIS2MDL BUS IO driver	I2C:I2C	I2C1	BSP_BUS_DRIVER
HTS221 BUS IO driver	I2C:I2C	I2C1	BSP_BUS_DRIVER
LPS22HH BUS IO driver	I2C:I2C	I2C1	BSP_BUS_DRIVER
LSM6DSR BUS IO driver	I2C:I2C	I2C1	BSP_BUS_DRIVER
BSP USART	USART:Asynchronous	USART2	BSP_COMMON_DRIVER
BSP LED	GPIO:Output	PA5	BSP_COMMON_DRIVER

Set the “Platform Settings” and the “Parameter Settings” from the “Configuration” view according your custom board (take into account that according the example chosen some settings can appear or not). A sample of configuration for Nucleo 144 is shown below:

Name	IPs or Components	Found Solutions	BSP API
TIMER	TIM1_8F77:Internal Clock	TIM3	Unknown
MEMS INT1	GPIO:Input	PF10	Unknown
LIS2MDL BUS IO driver	I2C:I2C	I2C1	BSP_BUS_DRIVER
HTS221 BUS IO driver	I2C:I2C	I2C1	BSP_BUS_DRIVER
LPS22HH BUS IO driver	I2C:I2C	I2C1	BSP_BUS_DRIVER
LSM6DSR BUS IO driver	I2C:I2C	I2C1	BSP_BUS_DRIVER
BSP USART	USART:Asynchronous	USART3	BSP_COMMON_DRIVER
BSP LED	GPIO:Output	PB7	BSP_COMMON_DRIVER

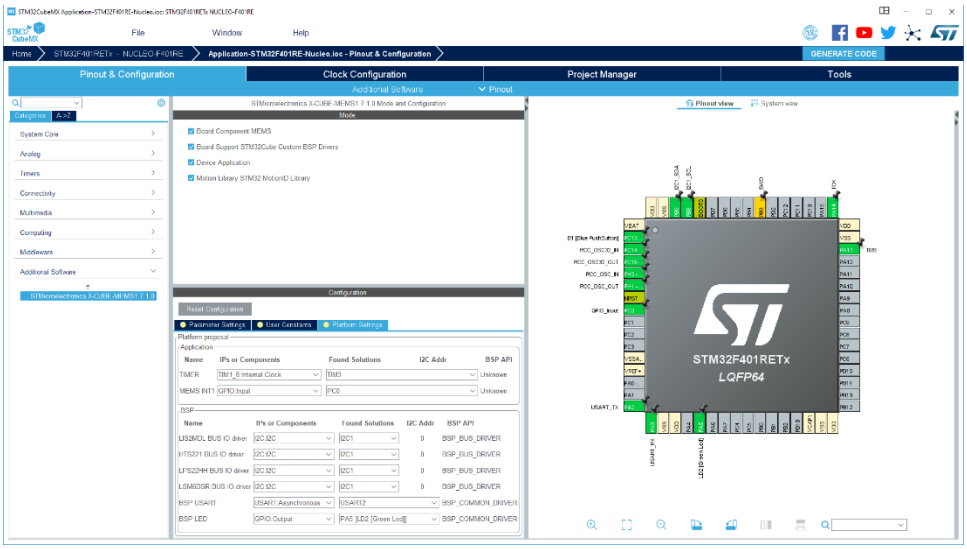

***Figure 70** STM32CubeMX **Pinout & Configuration** tab and "Additional Software - Platform Settings" for Nucleo 64*

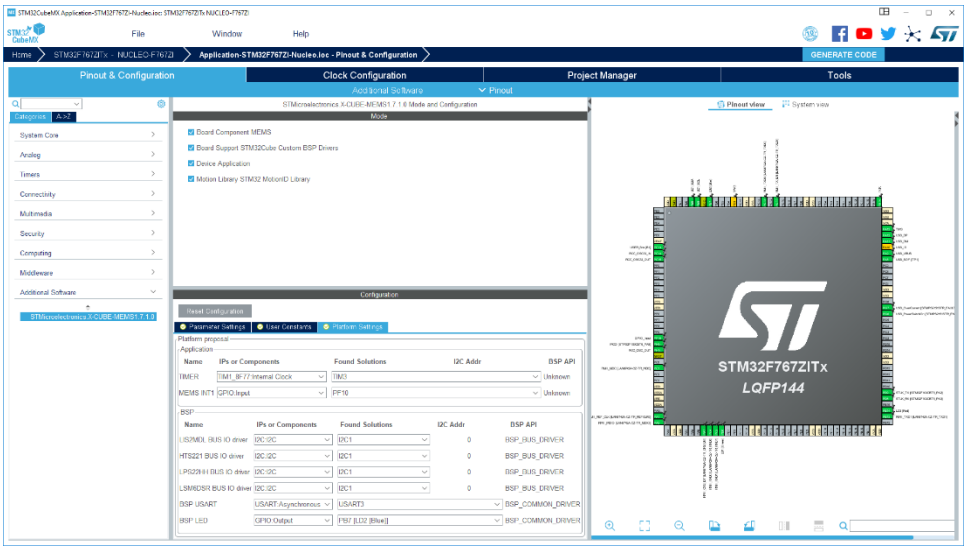

***Figure 71** STM32CubeMX **Pinout & Configuration** tab and "Additional Software - Platform Settings" for Nucleo 144*

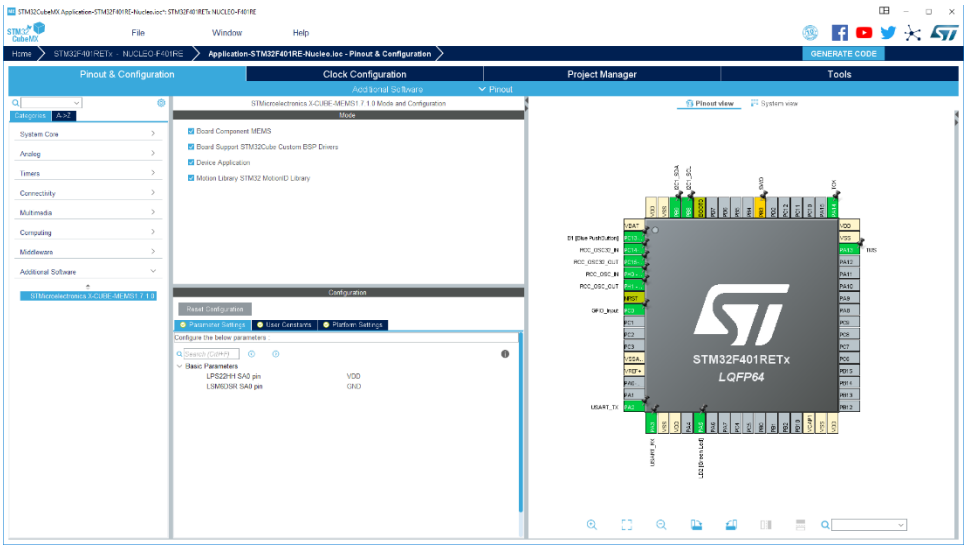

***Figure 72** STM32CubeMX **Pinout & Configuration** tab and "Additional Software - Parameter Settings" for Nucleo 64*

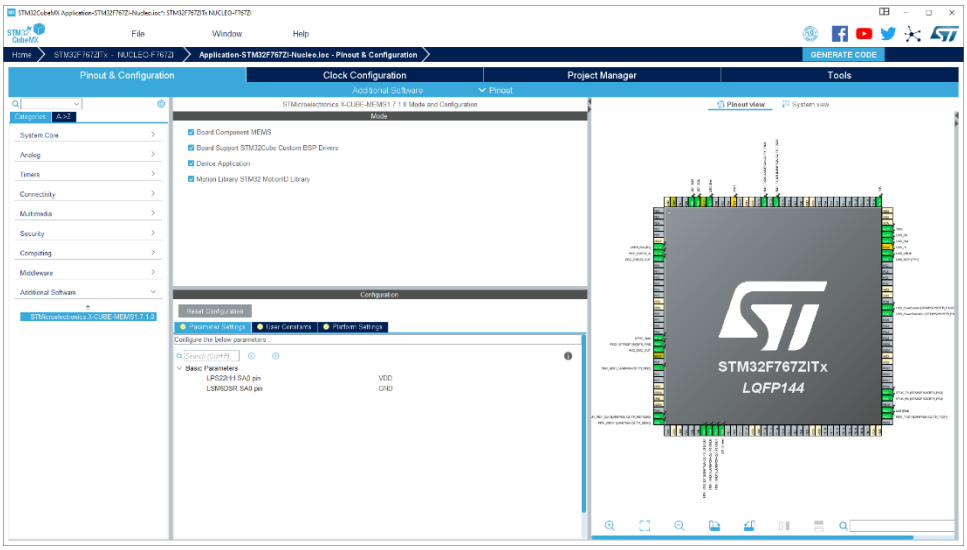

***Figure 73** STM32CubeMX **Pinout & Configuration** tab and "Additional Software - Parameter Settings" for Nucleo 144*

Setup RTC to use real time clock and date stamp in data stream sent from application to Unicleo-GUI. From the **Pinout & Configuration** tab, click on "Timers" category and then on RTC item:

- Check "Activate Clock Source"
- Check "Activate Calendar"

No need to fill the initial time and date since the FW will receive this information from Unicleo-GUI during startup connection.

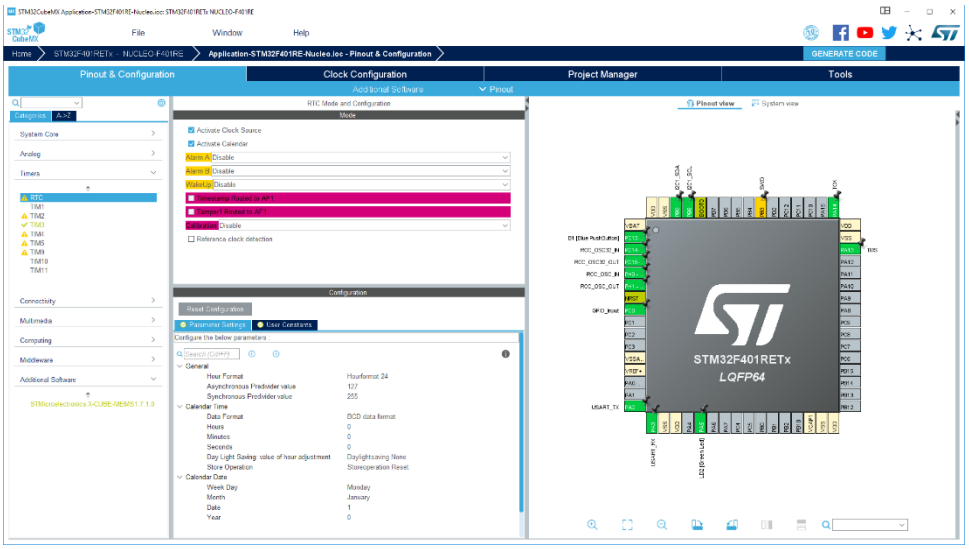

Figure 74 *STM32CubeMX RTC Configuration for Nucleo 64*

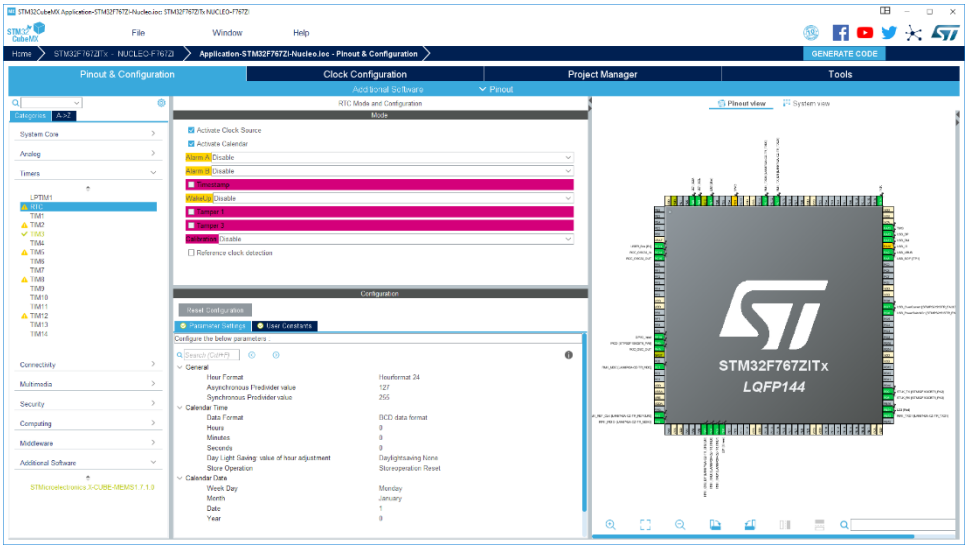

Figure 75 *STM32CubeMX RTC Configuration for Nucleo 144*

Setup Timer for data acquisition precise timing. The selected Timer instance has to be able to raise the interrupt after counting to the desired value. From the **Pinout & Configuration** tab, click on "Timers" category and then on TIM3 item:

- Set "Clock Source" to "Internal Clock"
- In "NVIC Settings" tab check "TIM3 global interrupt"

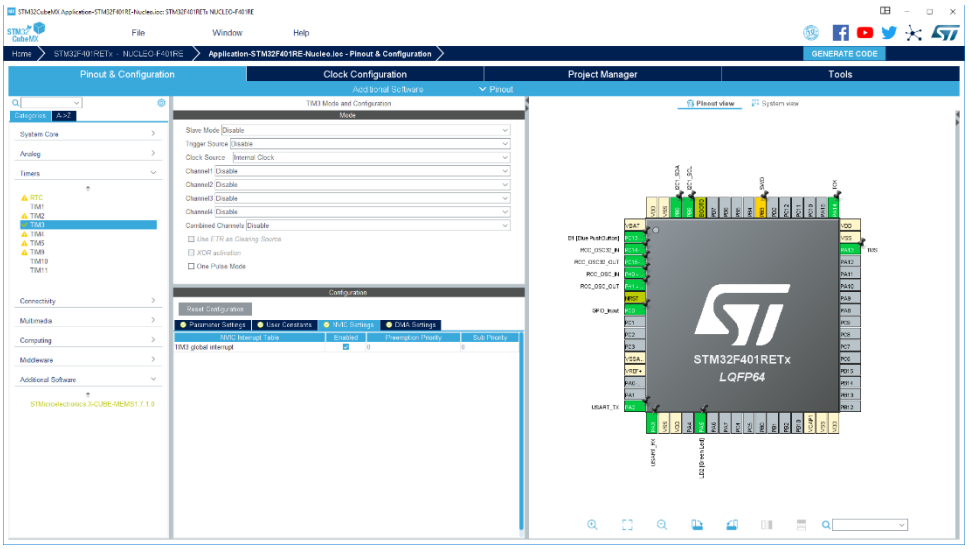

Figure 76 *STM32CubeMX TIM3 Configuration for Nucleo 64*

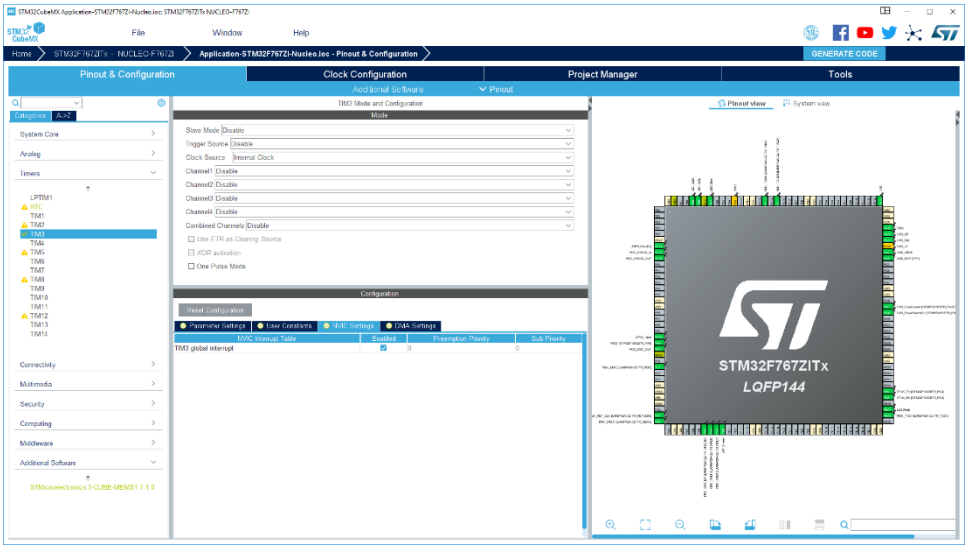

Figure 77 *STM32CubeMX TIM3 Configuration for Nucleo 144*

Setup I2C for communication between sensor and MCU. From the **Pinout & Configuration** tab, click on “Connectivity” category and then on I2C1 item:

- Set I2C to I2C
- The rest is filled automatically

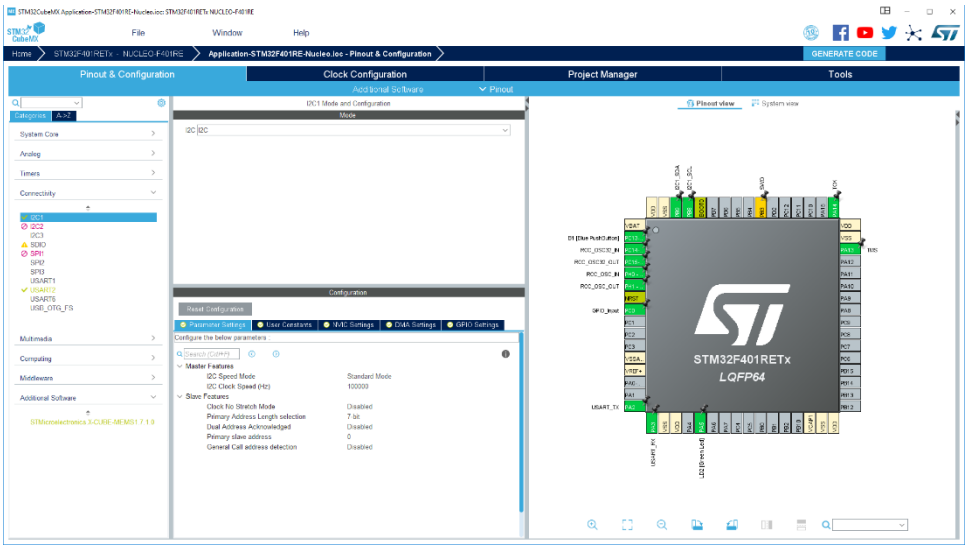

Figure 78 *STM32CubeMX I2C1 Configuration for Nucleo 64*

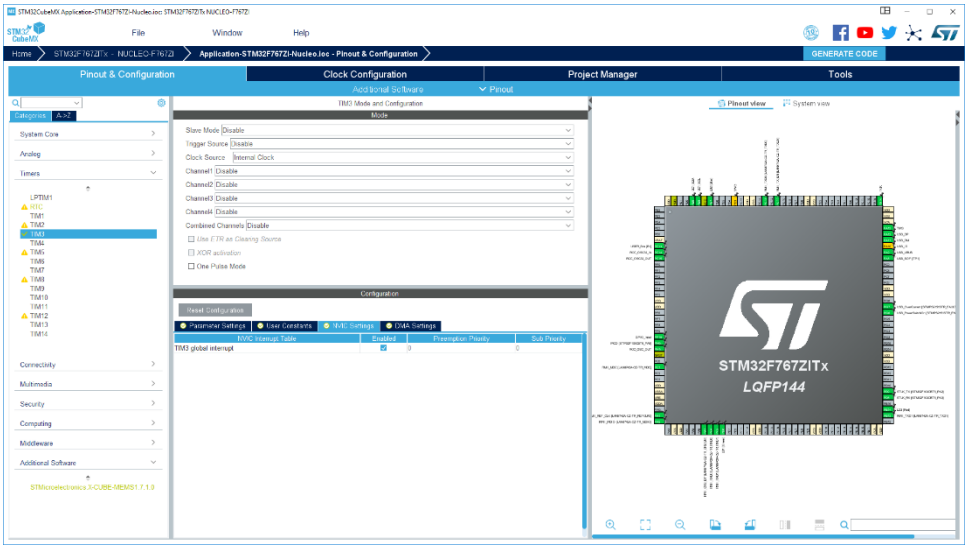

Figure 79 *STM32CubeMX I2C1 Configuration for Nucleo 144*

Setup USART for communication between MCU (FW) and PC (Unicleo-GUI). From the **Pinout & Configuration** tab, click on “Connectivity” category and then on USART2 item (Nucleo 64) or USART3 item (Nucleo 144):

- Select Asynchronous Mode

At “Parameter Settings” tab:

- Set Baud Rate to 921600 Bits/s
- Set Word Length to 8 Bits (including Parity)
- Set Parity to None
- Set Stop Bits to 1
- The rest is filled automatically

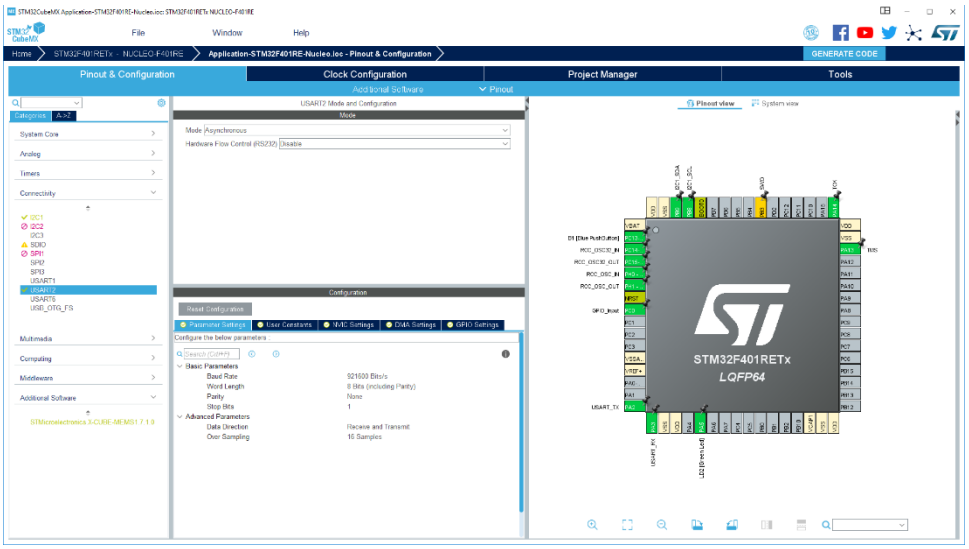

Figure 80 *STM32CubeMX USART2 Configuration for Nucleo 64*

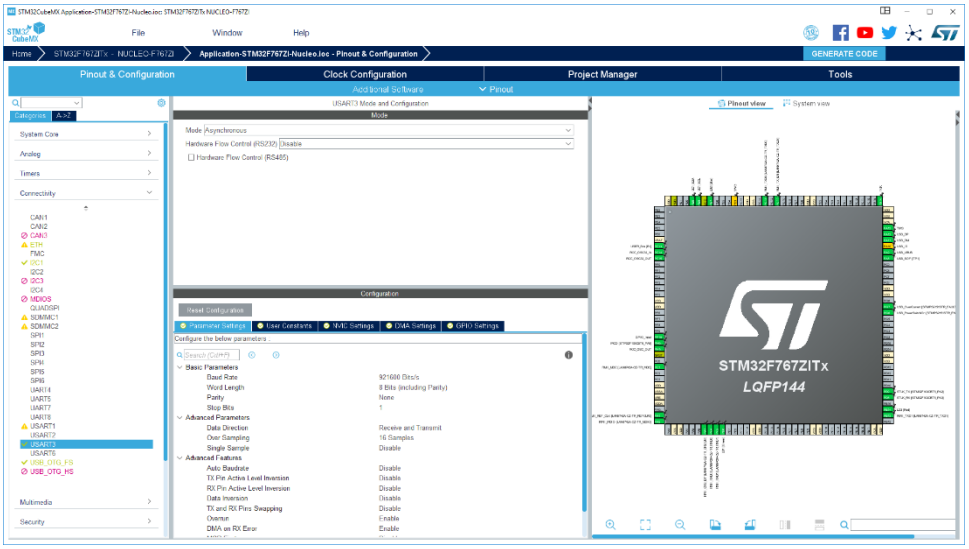

Figure 81 *STM32CubeMX USART3 Configuration for Nucleo 144*

Setup USART DMA for receiving data from Unicleo-GUI. At “DMA Settings” tab:

- Click Add button and select USART2_RX for Nucleo 64 or USART3_RX for Nucleo 144
- Select Circular Mode
- Check Memory Increment Address
- The rest is filled automatically

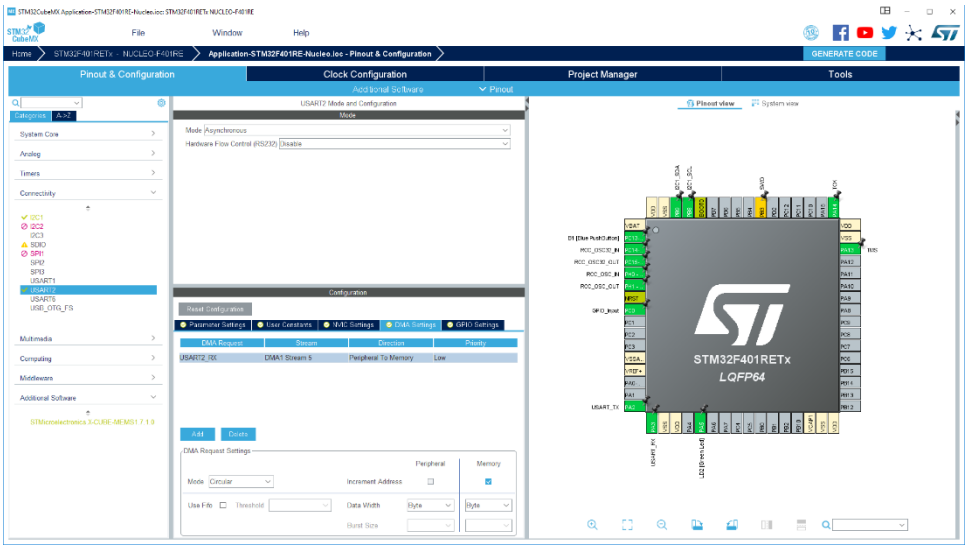

Figure 82 *STM32CubeMX USART2_RX DMA Configuration for Nucleo 64*

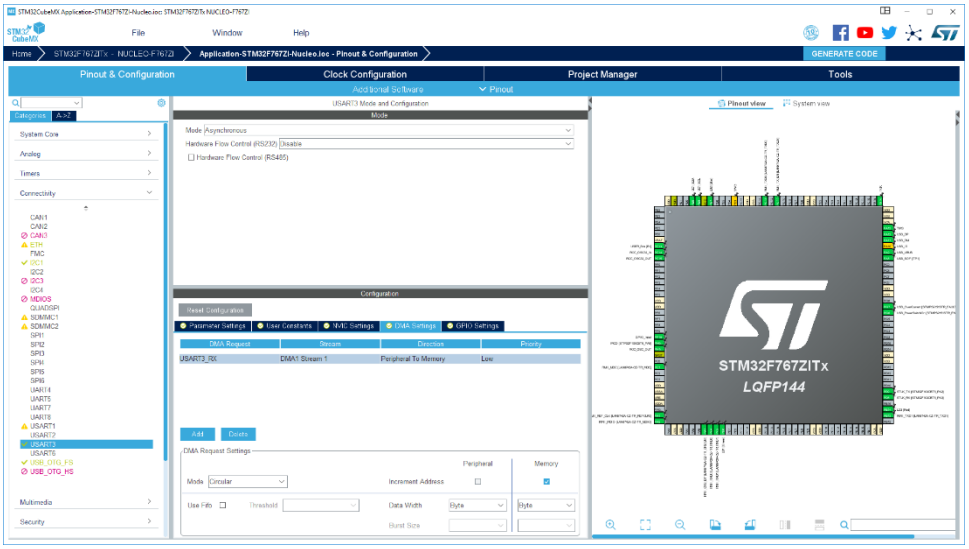

Figure 83 *STM32CubeMX USART3_RX DMA Configuration for Nucleo 144*

Setup CRC for Cyclic Redundancy Check used for library integrity verification. From the **Pinout & Configuration** tab, click on “Computing” category and then on CRC item:

- Check Activated Mode

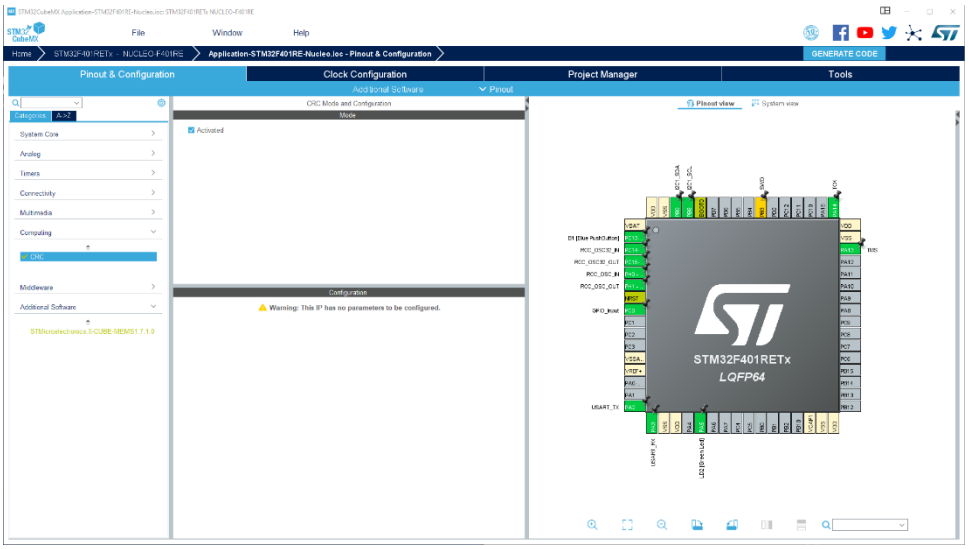

Figure 84 *STM32CubeMX CRC Configuration for Nucleo 64*

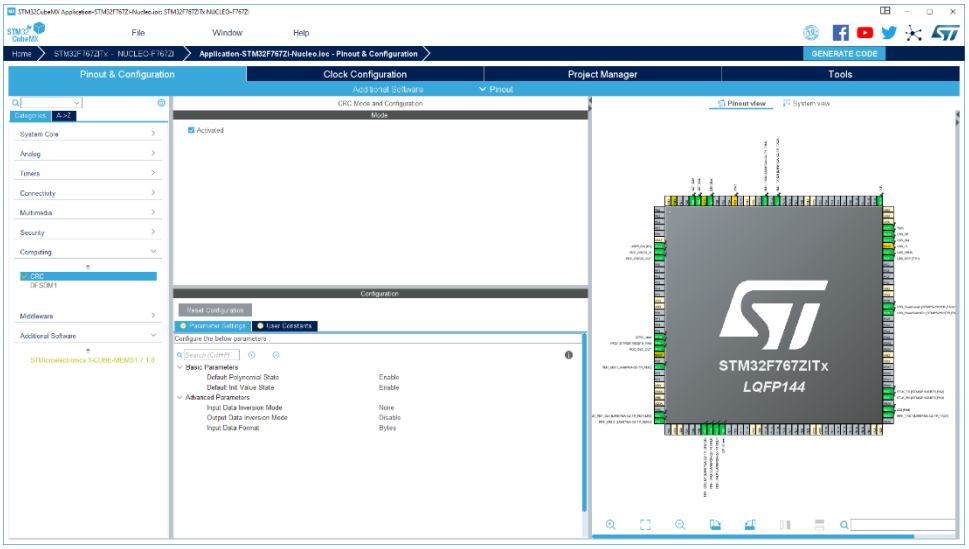

Figure 85 *STM32CubeMX CRC Configuration for Nucleo 144*

Once all the above described steps have been performed, the middleware applications for custom boards using the **STMicroelectronics X-CUBE-MEMS1** software can be generated clicking the "GENERATE CODE" button.

8 Generated Folders Structure

When generating a project, two models of folders structure can be adopted when using a high level firmware component (i.e. a middleware in the STM32Cube MCU package):

- **Basic Structure**: the basic structure is often used with HAL examples and single

package projects. This structure consists of having the IDE configuration folder in the same level as the sources (organized in *Inc* and *Src* subfolders).

- **Advanced Structure**: the advanced structure provides a more efficient and organized folders model that allows ease middleware applications integration when several packages are used.
  In the Advanced mode *Src* and *Inc* are generated under folder *Core*.
  For each package, the list of the generated files is under <*Package_Name*> (*X-CUBE-MEMS1* for the X-CUBE-MEMS1 pack), at the same level as *Core* and containing inside the *App* and the *Target* subfolder.

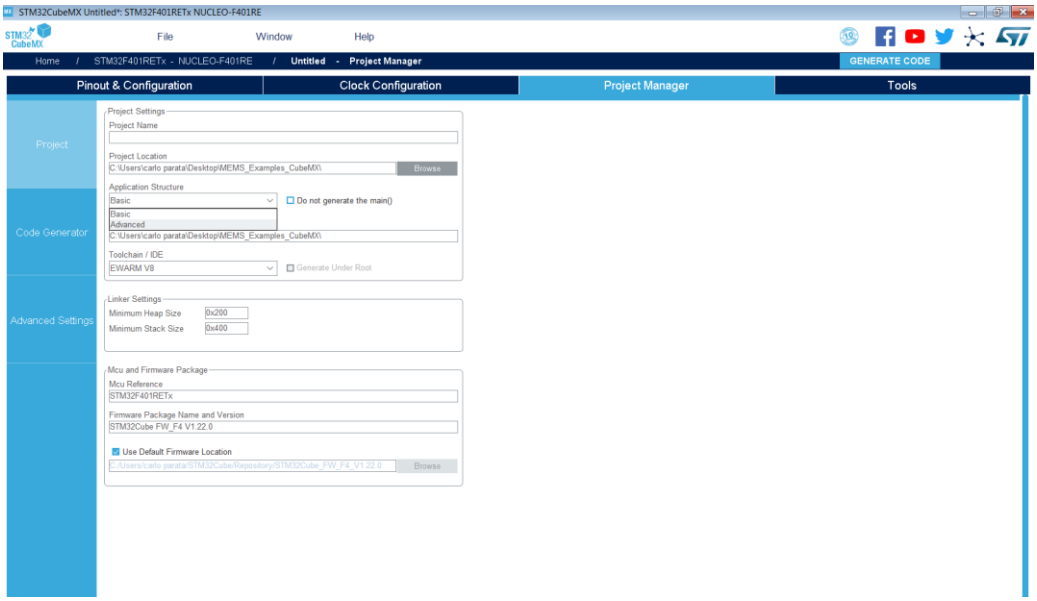

***Figure 86** STM32CubeMX **Application Structure** Configuration*

9 Known Limitations and workarounds

CubeMX MEMS pack V7.1.0 is fully compatible with CubeMX V5.6.0 and upwards. It is not fully compatible with previous versions of CubeMX.

When you generate the source code of a Middleware Application for STM32CubeIDE using as target a platform with hardware FPU, the source code does not compile because the binary libraries are compiled with “softfp” option and STM32CubeIDE by default sets FPU as “hard”. In order to compile the source code, it needs to edit the project options and replace the FPU “hard” with “softfp”.

Motion Libraries and the relative Applications that do not support Cortex-M0/Cortex-M0+ and/or Cortex-M33 platforms are displayed by CubeMX v5.6.0 when you select a Cortex-M0/Cortex-M0+ or a Cortex-M33 platform. In this case the generated source code could not compile and work properly. None of the Motion Libraries supports Cortex-M33 platforms and the only Motion Libraries that support Cortex-M0/Cortex-M0+ platforms are: MotionID, MotionFX, MotionGC, MotionMC, MotionTL and MotionEC.

9 References

[1] UM1859 – User Manual - *Getting started with the X-CUBE-MEMS1 Motion MEMS and environmental sensor software expansion for STM32Cube*

10 Revision history

Table 2: Document revision history

Date	Version	Changes
11-Sep-2018	1	Initial release
07-Nov-2018	2	Update the manual to CubeMX v5.0.0 GUI
22-Nov-2018	3	Update pack version number
14-Feb-2019	4	Add IKS01A3 support
10-Feb-2020	5	Add 10 Middleware applications description

IMPORTANT NOTICE – PLEASE 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. 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 acknowledgement.

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 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.

ST and the ST logo are trademarks of ST. 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.