Getting started with the STMicroelectronics X-CUBE-EEPRMA1 software package for STM32CubeMX

Introduction

This document provides the guidelines to configure and use the X-CUBE-EEPRMA1 software package for STM32CubeMX (minimum required version V6.3.0). 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-EEPRMA2 and X-NUCLEO-PGEEZ1 expansion board with a Nucleo board or 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 EEPROM components, the X-NUCLEO-EEPRMA2 and X-NUCLEO-PGEEZ1 expansion board and the ST expansion software for EEPROM IC are available on www.st.com.

Contents

List of Figures

1. Acronyms and abbreviations

Table 1: List of acronyms

Acronym	Description
EEPROM	Electrically erasable programmable read-only memory
HAL	Hardware Abstraction Layer
BSP	Board Support Package
CMSIS	Cortex® microcontroller software interface standard
I2C	Inter-Integrated Circuit
SPI	Serial peripheral interface
QSPI	Quad- Serial peripheral interface
RTOS	Real Time Operation System
U(S)ART	Universal (Synchronous) Asynchronous Receiver Transmitter

2. What is STM32Cube?

STM32Cube is a combination of a full set of PC software tools and embedded software blocks running on STM32 microcontrollers and microprocessors:

- STM32CubeMX configuration tool for any STM32 device; it generates initialization C code for Cortex-M cores and the Linux device tree source for Cortex-A cores
- STM32CubeIDE integrated development environment based on open-source solutions like Eclipse or the GNU C/C++ toolchain, including compilation reporting features and advanced debug features
- STM32CubeProgrammer programming tool that provides an easy-to-use and efficient environment for reading, writing, and verifying devices and external memories via a wide variety of available communication media (JTAG, SWD, UART, USB DFU, I2C, SPI, CAN, etc.)
- STM32CubeMonitor family of tools (STM32CubeMonRF, STM32CubeMonUCPD, STM32CubeMonPwr) to help developers customize their applications in real-time
- STM32Cube MCU and MPU packages specific to each STM32 series with drivers (HAL, low-layer, etc.), middleware, and lots of example code used in a wide variety of real-world use cases
- STM32Cube expansion packages for application-oriented solutions

3. License

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

4. Sample Applications and Examples Description

In this section, a short overview of the sample examples included in the X-CUBE-EEPRMA1 pack is provided. For more info kindly refer UM2481 (User Manual of X-CUBE-EEPRMA1)
The sample applications/examples:

- are ready-to-use projects that can be generated through the STM32CubeMX for any Nucleo board and using the X-NUCLEO-EEPRMA2 or X-NUCLEO-PGEEZ1 expansion board.
- show the users how to use the APIs to correctly initialize and use the EEPROMs IC (M24xx/M95xx/M95P32 device).

4.1 EEPRMA2_EEPROMRW

This application shows how to use the X-NUCLEO-EEPRMA2 firmware examples for the developer to start experimenting with the code:

- Standard initialization of the I²C and SPI EEPROM
- Firmware example containing: – block protect, which reads the status after "write enable" and "write disable" for the SPI EEPROM – write protection feature for the I²C EEPROM
- Firmware example that writes the complete memory and reads the information received.

4.2 PGEEZ1_EEPROMRW

This application shows how to use the X-NUCLEO-PGEEZ1 firmware examples for the developer to start experimenting with the code:

- Standard initialization SPI EEPROM
- Firmware example containing: –Write Enable and Write Disable for the QSPI EEPROM, Read Data in SPI, Fast Read in SPI, Dual and Quad Read in QSPI, page erase, sector erase and block erase
- Firmware example also supports AT commands tested with Docklight where the user can test instructions on M95P32

5. Installing the X-CUBE-EEPRMA1 pack in STM32CubeMX

After downloading (from www.st.com), installing and launching the STM32CubeMX (V≥6.3.0), the X-CUBE-EEPRMA1 pack can be installed in a few steps.

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

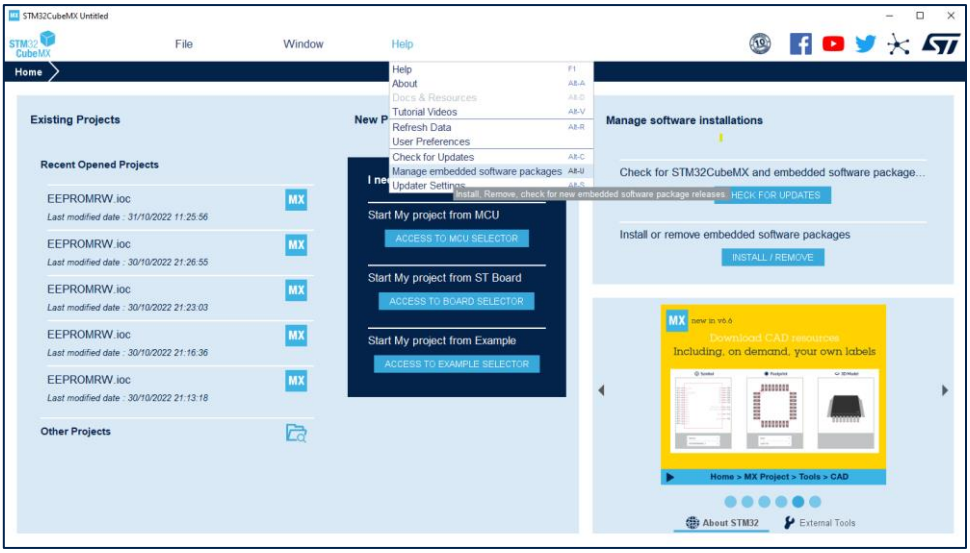

Figure 1: Managing embedded software packs in STM32CubeMX

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-EEPRMA1 pack.

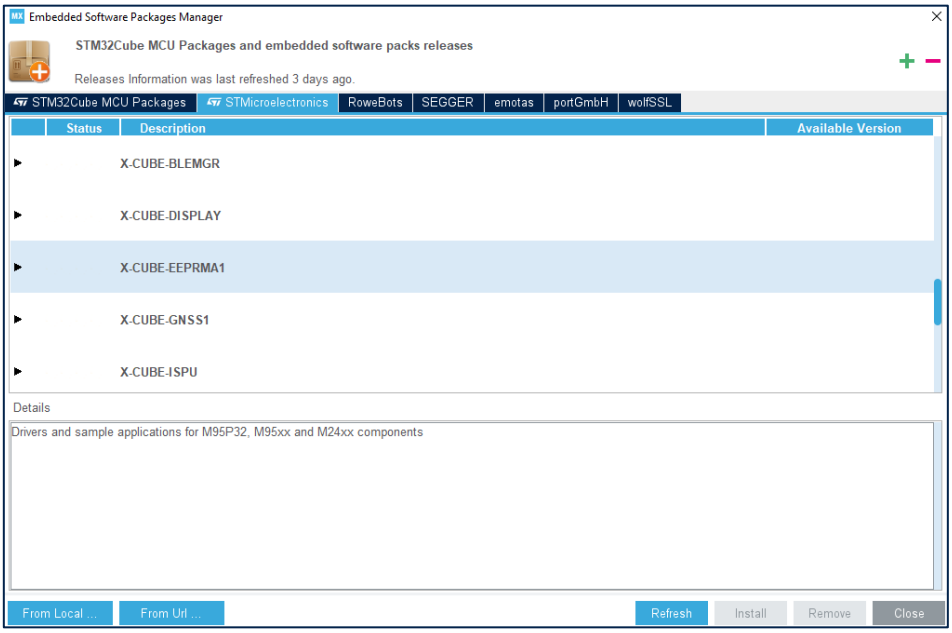

Figure 2: Installing the X-CUBE-EEPRMA1 pack in STM32CubeMX

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.

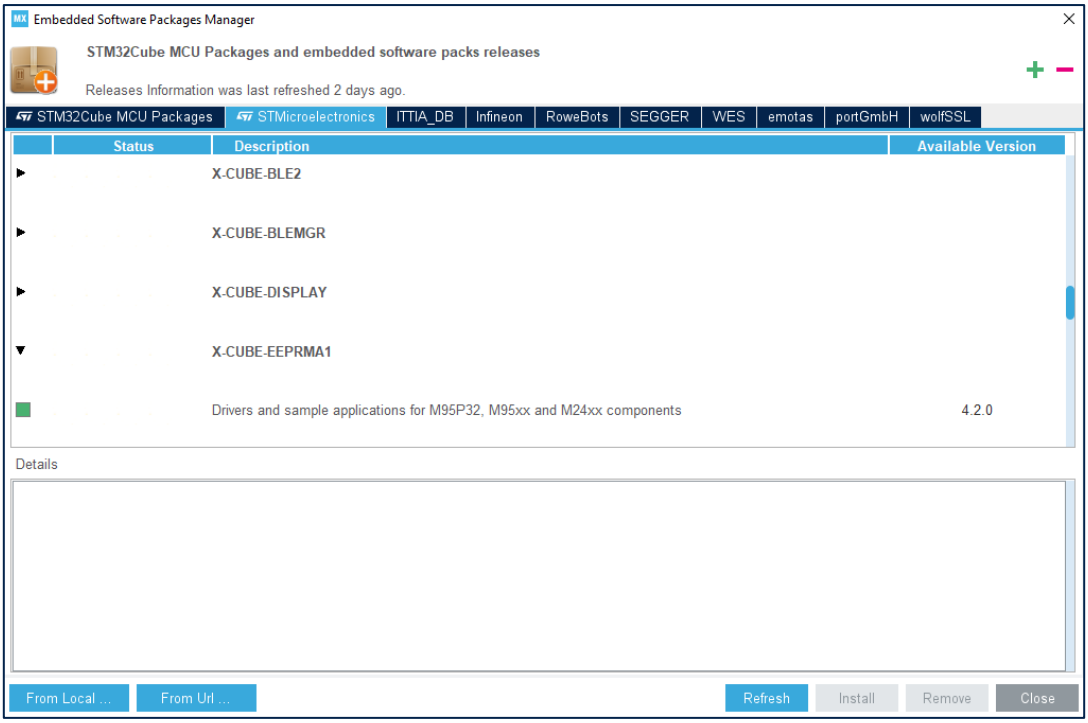

Figure 3: The X-CUBE-EEPRMA1 pack in STM32CubeMX

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.

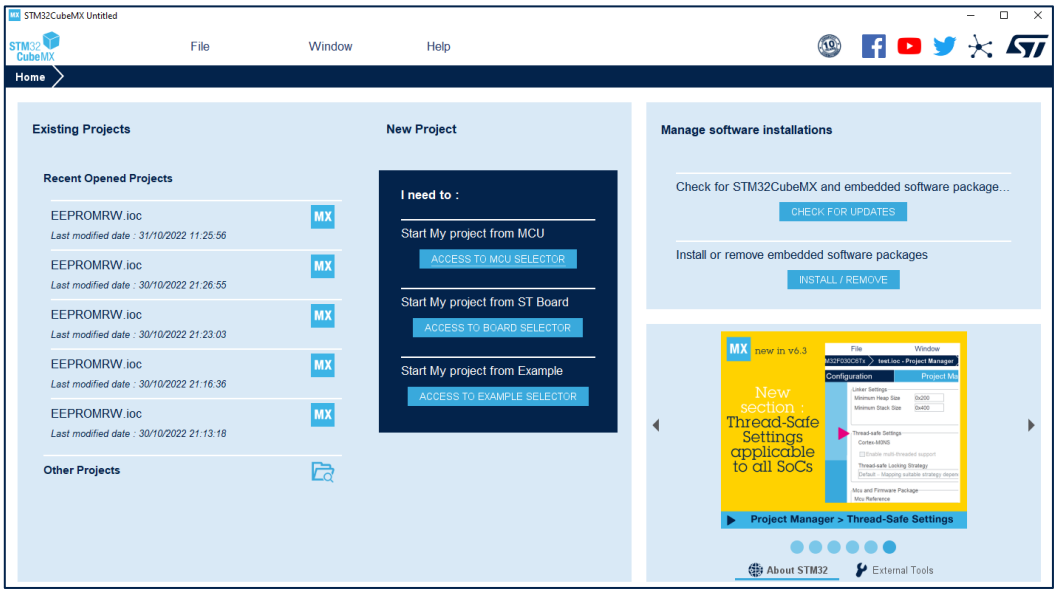

Figure 4: STM32CubeMX main page

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

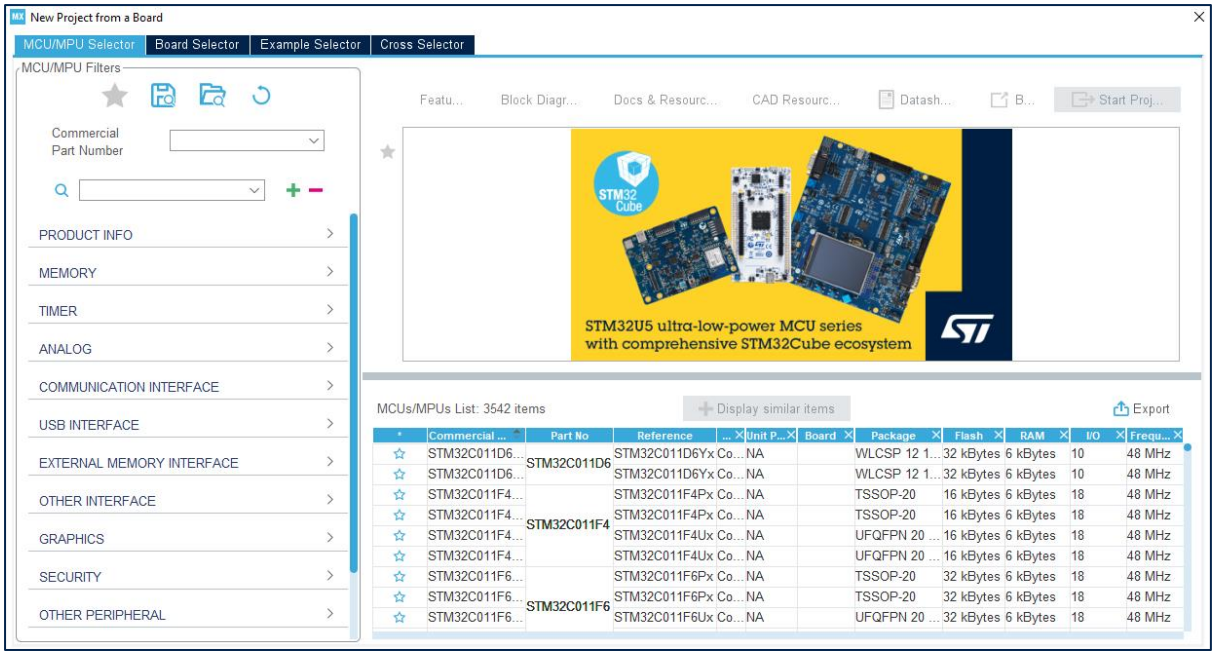

Figure 5: STM32CubeMX MCU/Board Selector/Example Selector

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.

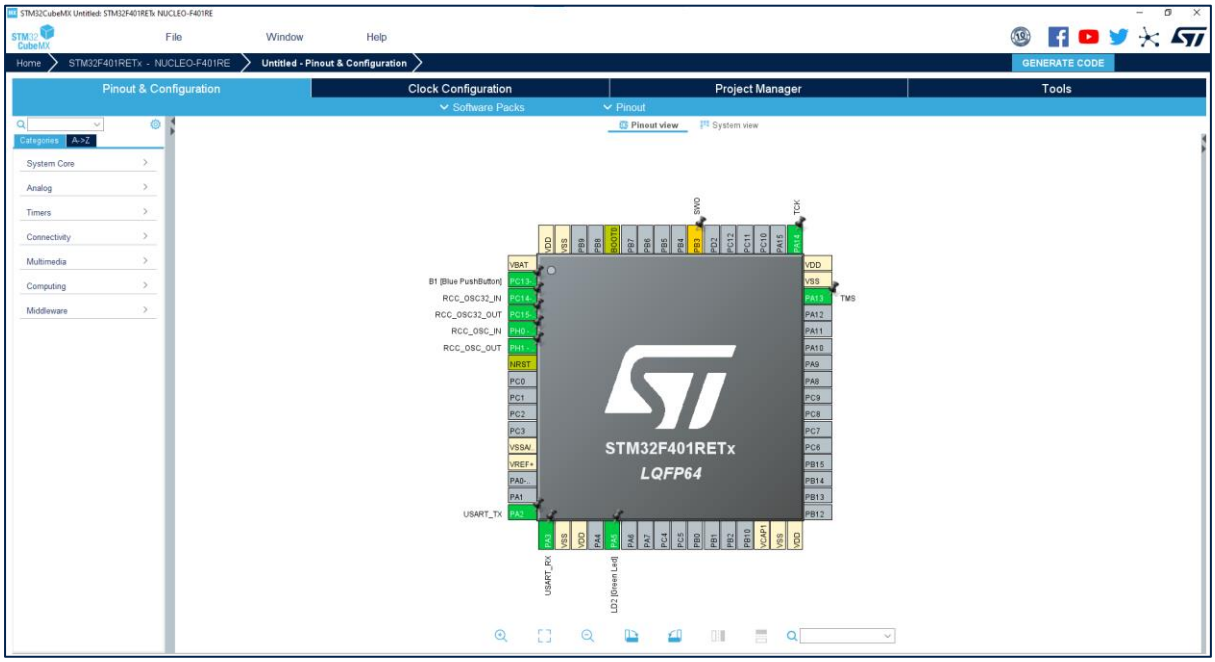

Figure 6: STM32CubeMX Pinout & Configuration window

To add the X-CUBE-EEPRMA1 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.

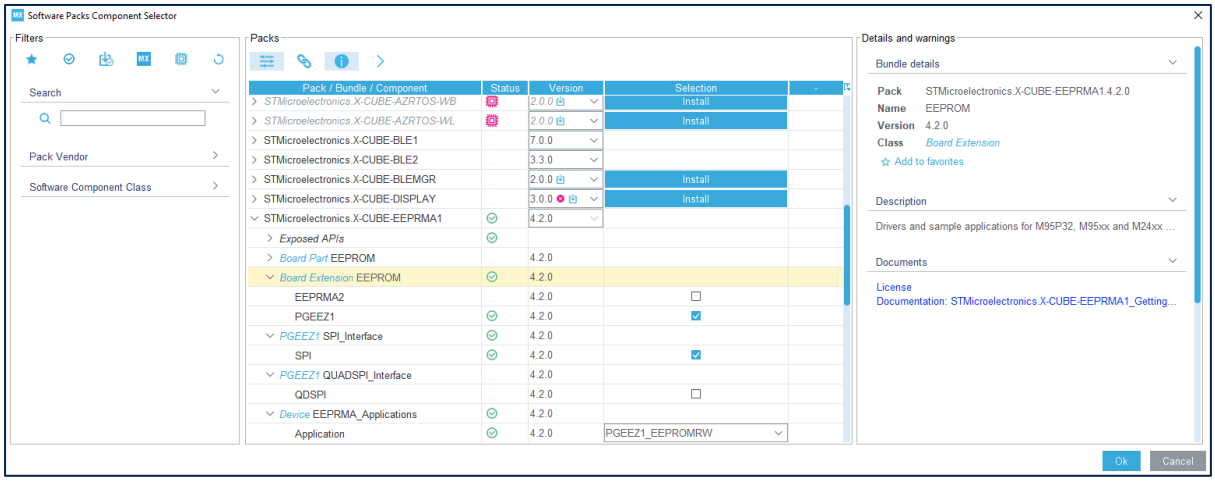

Figure 7: STM32CubeMX Software Packs Component selector window

7. STM32 Configuration Steps

The X-NUCLEO-EEPRMA2 interfaces with the STM32 microcontroller via the SPI and I2C bus. Hence, assuming a user wants to interface the ST X-NUCLEO-EEPRMA2 expansion board with a STM32 Nucleo 64 pins board (e.g. a Nucleo-F401RE) no particular hardware modification must be done.

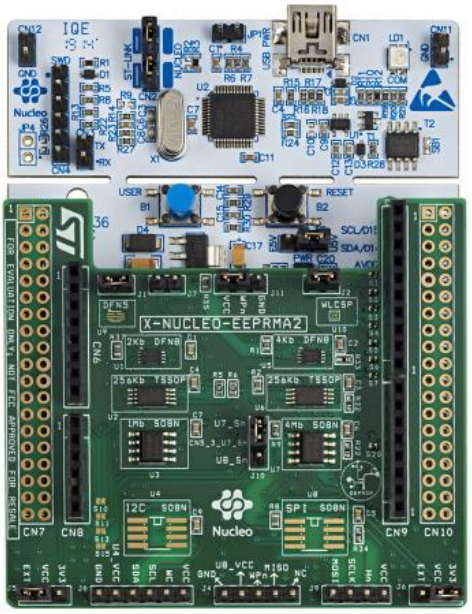

Figure 8: STM32 Nucleo 64 with X-NUCLEO-EEPRMA2

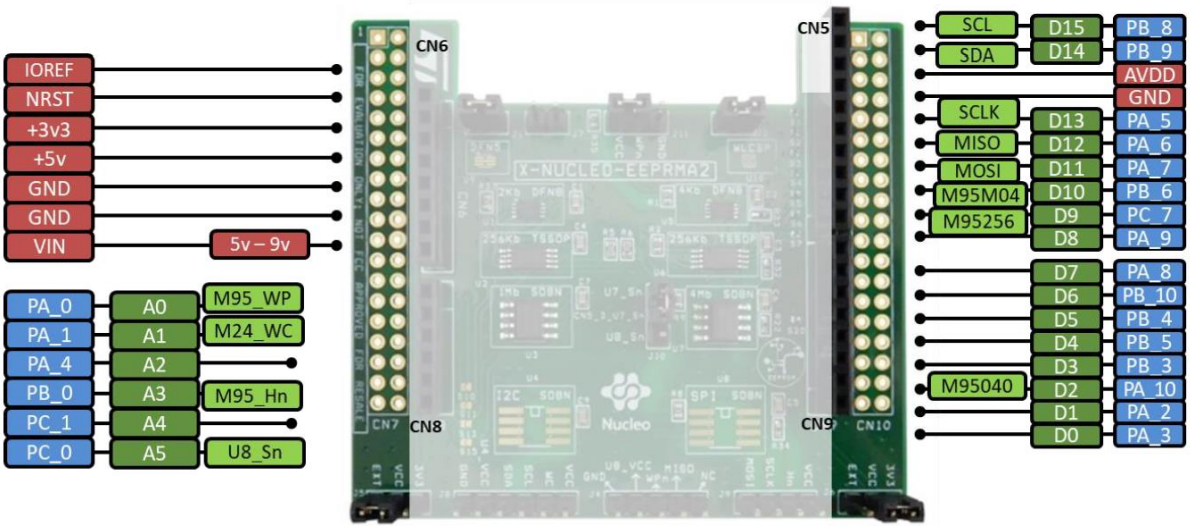

Figure 9: X-NUCLEO-EEPRMA2 pinout

The X-NUCLEO-PGEEZ1 interfaces with the STM32 microcontroller via the QSPI. Hence, assuming a user wants to interface the ST X-NUCLEO-PGEEZ1 expansion board with a STM32 Nucleo 144 pins board (e.g., a Nucleo-H743ZI) no particular hardware modification must be done.

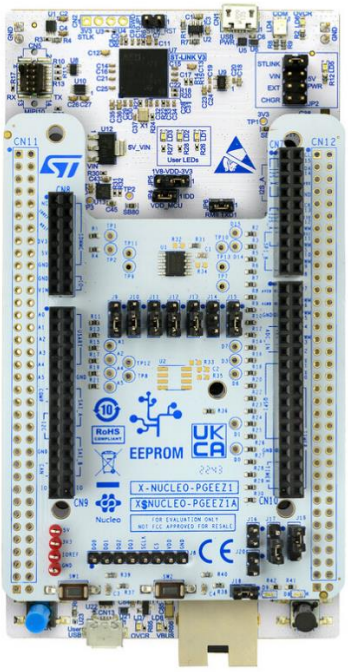

Figure 10: STM32 Nucleo 144 with X-NUCLEO-PGEEZ1

7.1 Use of EEPRMA2 Component with sample example for X-NUCLEO-EEPRMA2

This section outlines how to configure STM32CubeMX with X-NUCLEO-EEPRMA2. To add the X-CUBE-EEPRMA1 additional software to the project, the “Software Pack” and then “Select Components” option is to be chosen. From the “Software Packs Components selector” window, the user must select the example from the “Device” class and “Board Extension” class as shown in the figure below:

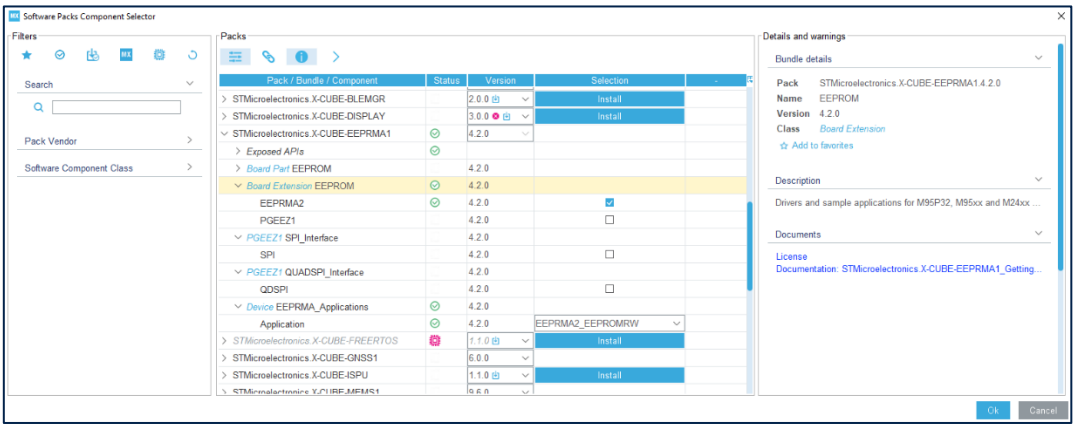

Figure 11: STM32CubeMX EEPRMA2 Software Pack Components selection window

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.
- from the **Pinout** scheme, click on PA6 and set it as SPI1_MISO.
- from the **Pinout** scheme, click on PA7 and set it as SPI1_MOSI.
- from the **Pinout** scheme, click on PA5 and set it as SPI1_SCK.
- enable the SPI1 as SPI from the “Connectivity” category.
- enable the I2C1 as I2C from the “Connectivity” category.
- Configure the I2C1 settings with I2C speed at 400KHz (Fast Mode) from the “Configuration” view; o If STM32L0/STM32L4 MCU families are used, kindly set the Coefficient of Digital Filter to 2 in Parameter settings:

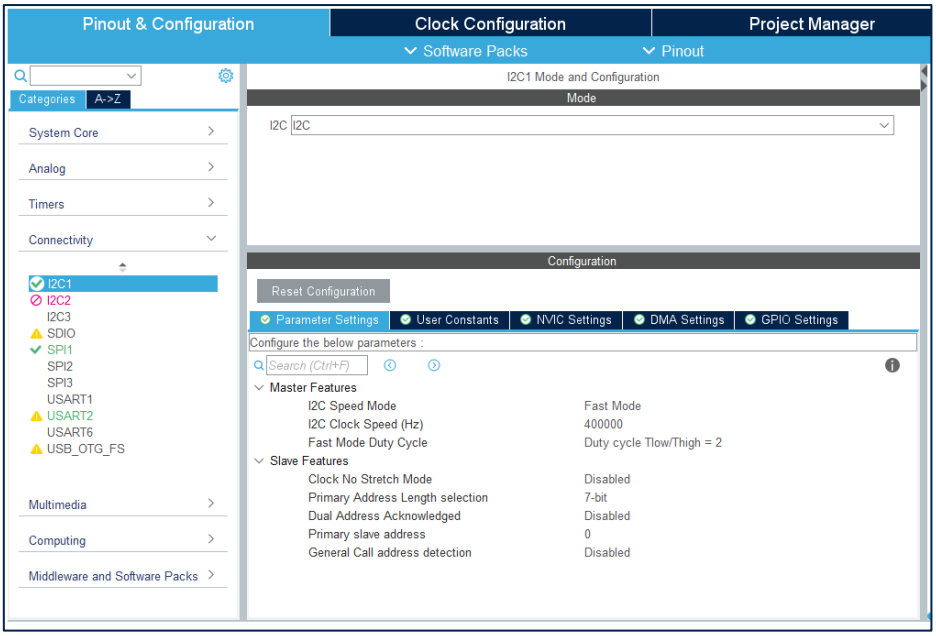

Figure 12: STM32CubeMX X-NUCLEO-EEPRMA2 I2C Configuration

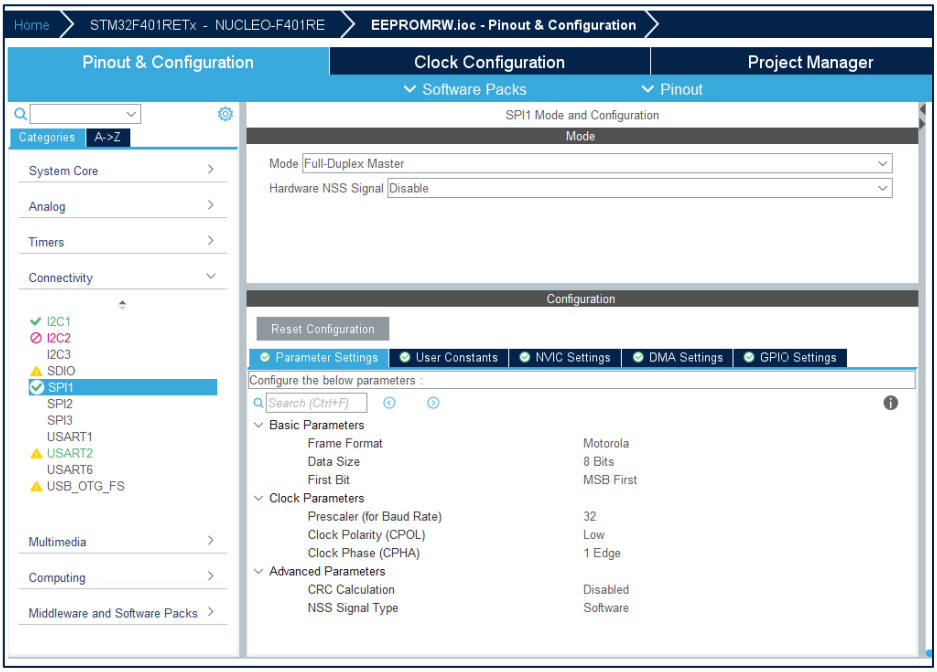

Figure 13: STM32CubeMX X-NUCLEO-EEPRMA2 SPI Configuration

From the **Pinout** scheme set:

Nucleo 64		**Nucleo 144**	
PB8	I2C1_SCL	PB8	I2C1_SCL
PB9	I2C1_SDA	PB9	I2C1_SDA
PA6	SPI1_MISO	PA6	SPI1_MISO
PA7	SPI1_MOSI	PA7	SPI1_MOSI
PA5	SPI1_SCK	PA5	SPI1_SCK

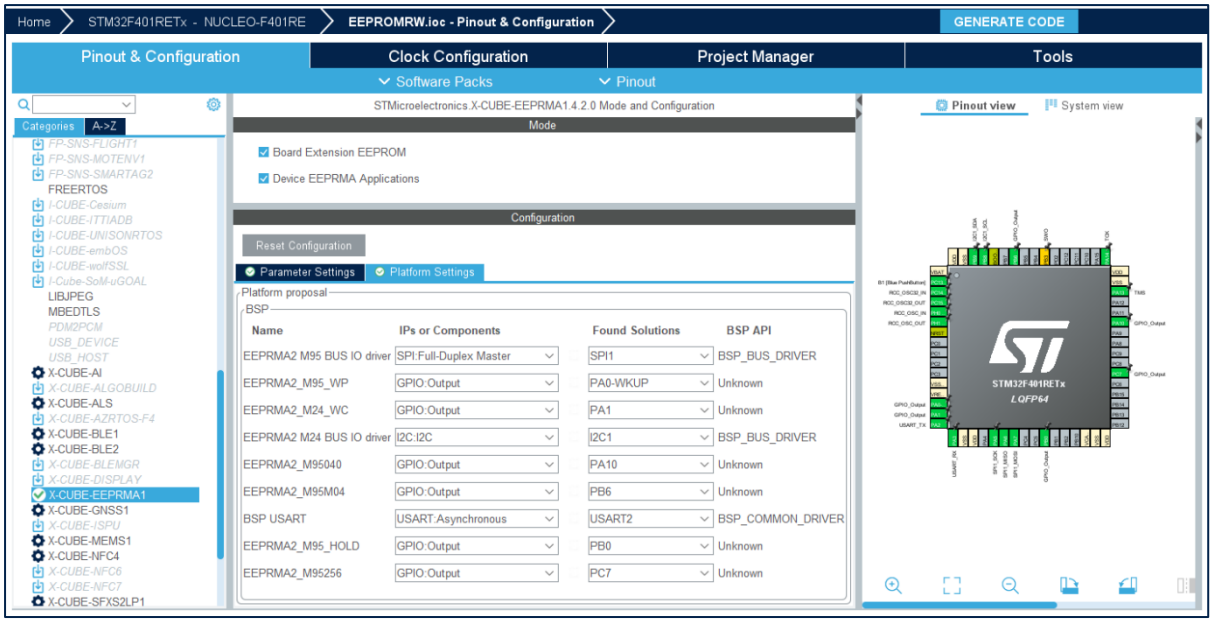

Figure 14: STM32CubeMX Pinout & Configuration tab and I2C and SPI settings

From the **Software Packs** category, press the 'Stmicroelectronics.X-CUBE-EEPRMA1.4.2.0' item, enable the "Board Extension EEPROM" and "Device EEPRMA Applications" checkbox from the "Mode" view and set the following Platform Settings from the "Configuration" view (consider that according to the example chosen some settings can appear or not):

Name	BSP API	**Supported IPs**	**Nucleo 64**	**Nucleo 144**
EEPRMA2 M95 BUS IO driver	BSP_BUS_DRIVER	SPI: Full Duplex-Master	SPI1	SPI1
EEPRMA2 M24 BUS IO driver	BSP_BUS_DRIVER	I2C: I2C	I2C1	I2C1
EEPRMA2_M95M04		GPIO: Output	PB6	PD14
EEPRMA2_M95256		GPIO: Output	PC7	PD15
EEPRMA2_M95040		GPIO: Output	PA10	PF15
EEPRMA2_M95_HOLD		GPIO: Output	PB0	PF3
EEPRMA2_M95_WP		GPIO: Output	PA0	PA3
EEPRMA2_M24_WC		GPIO: Output	PA1	PC0
BSP USART	BSP_COMMON_DRIVER	USART: Asynchronous	USART2	USART3

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

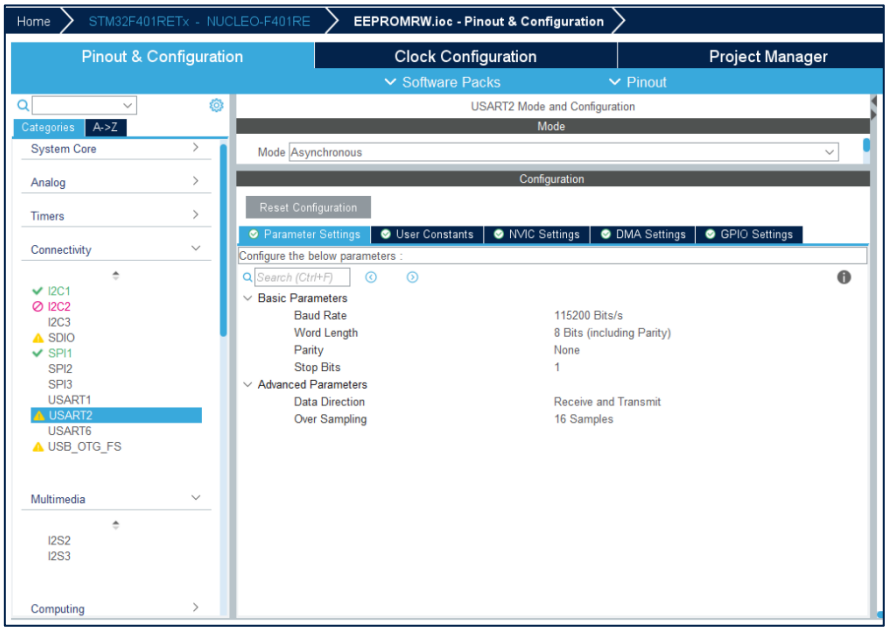

Figure 15: STM32CubeMX USART Configuration

Once all the above-described steps have been performed, the sample applications for X-NUCLEO-EEPRMA2 using the **STMicroelectronics X-CUBE-EEPRMA1** software can be generated clicking the “GENERATE CODE” button.

7.3 Use of PGEEZ1 Component with sample example for X-NUCLEO-PGEEZ1

This section outlines how to configure STM32CubeMX with X-NUCLEO-PGEEZ1.
To add the X-CUBE-EEPRMA1 additional software to the project, the “Software Pack” and then “Select Components” option is to be chosen. From the “Software Packs Components selector” window, the user must select the example from the “Board Extension”, “PGEEZ1” and “Device” class as shown in the figure below:

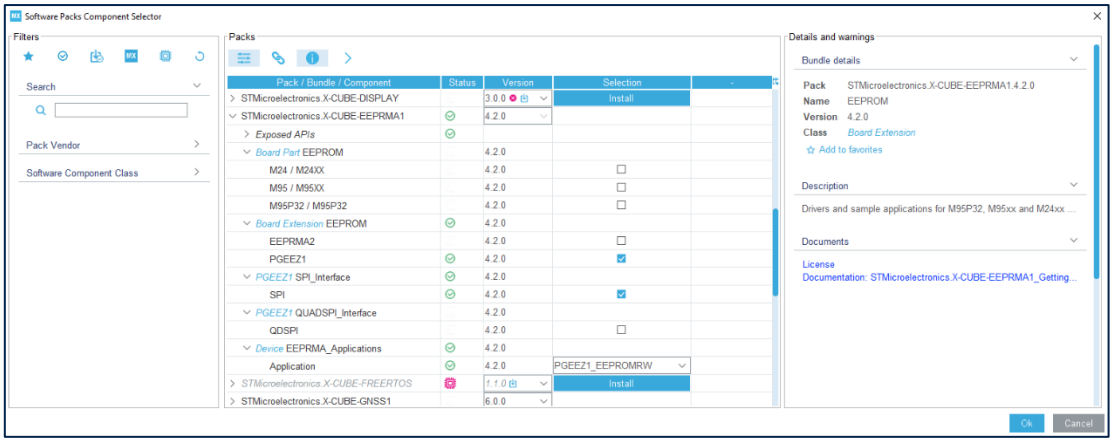

Figure 16: STM32CubeMX PGEEZ1 Software Pack Components selection window

From the **Pinout & Configuration** tab:

If the NUCLEO board is **64-pin**, the software packs support SPI:

- from the **Pinout** scheme, click on PA6 and set it as SPI1_MISO.
- from the **Pinout** scheme, click on PA7 and set it as SPI1_MOSI.
- from the **Pinout** scheme, click on PA5 and set it as SPI1_SCK.
- enable the SPI1 as SPI from the “Connectivity” category

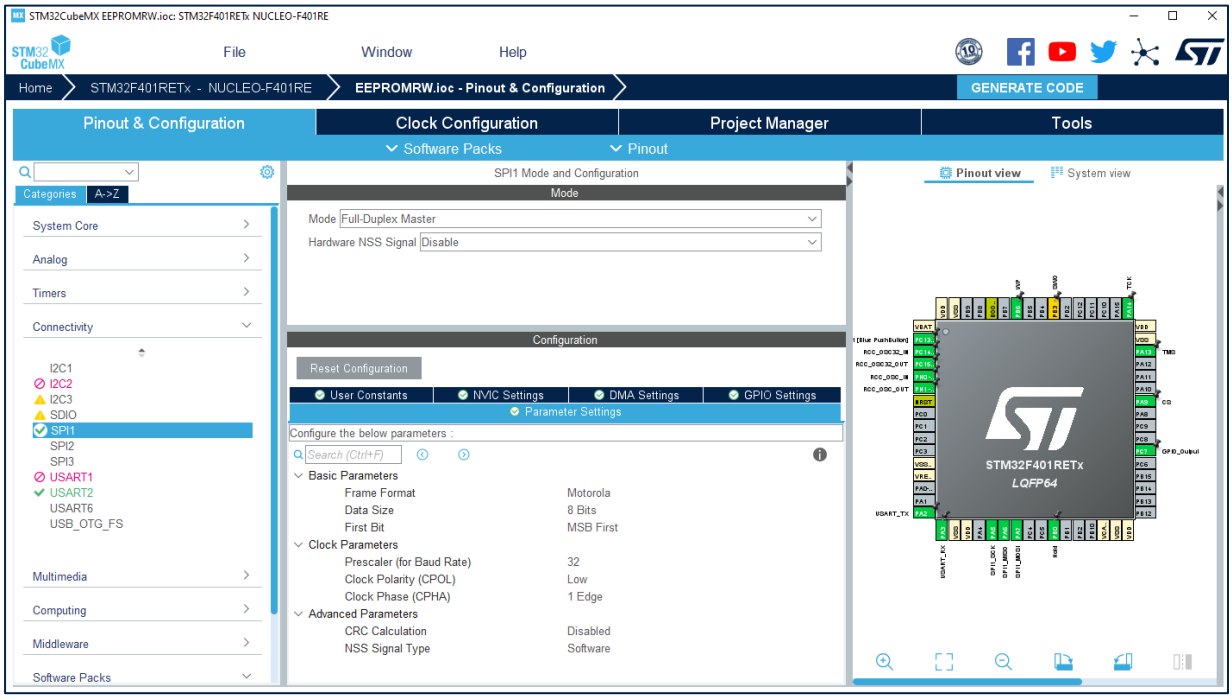

Figure 17: STM32CubeMX X-NUCLEO-PGEEZ1 SPI Configuration

From the **Pinout** scheme set:

Nucleo 64	
PA6	SPI1_MISO
PA7	SPI1_MOSI
PA5	SPI1_SCK

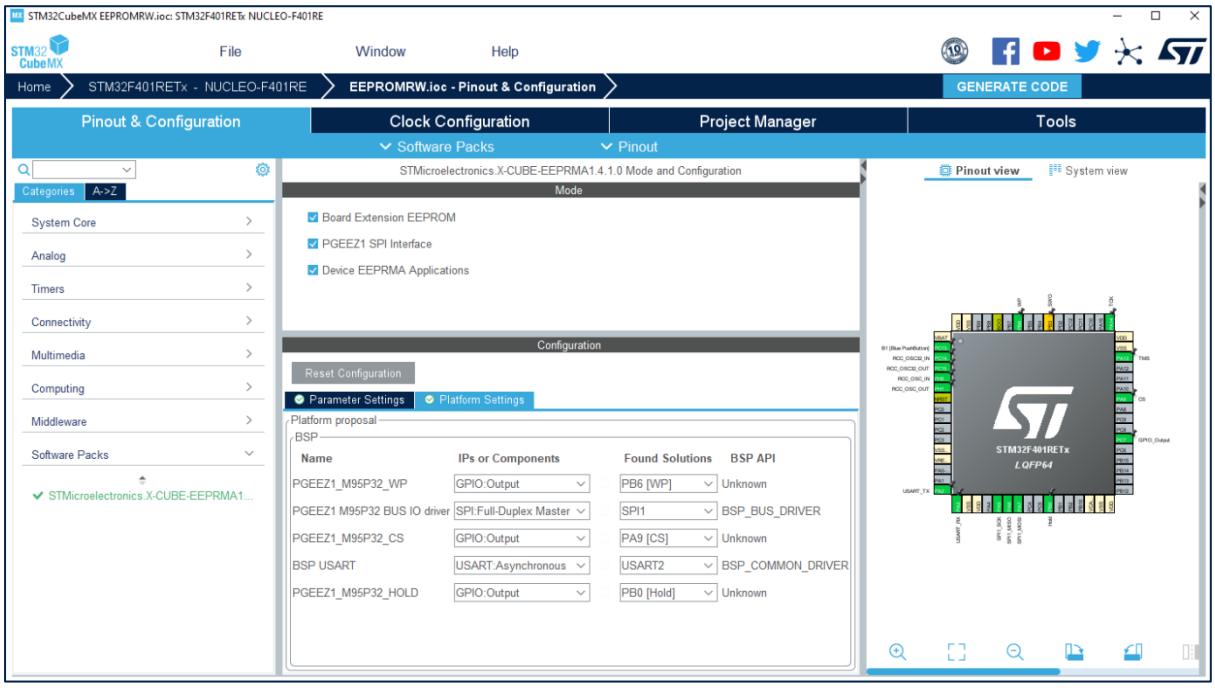

Figure 18: STM32CubeMX Pinout & Configuration tab and SPI settings

From the **Software Packs** category, press the 'Stmicroelectronics.X-CUBE-EEPRMA1.4.2.0' item, enable the "Board Extension EEPROM", "PGEEZ1 SPI Interface" and "Device EEPRMA Applications" checkbox from the "Mode" view and set the following Platform Settings from the "Configuration" view (consider that according to the example chosen some settings can appear or not):

Name	BSP API	**Supported IPs**	**Nucleo 64**
PGEEZ1_M95P32_WP		GPIO: Output	PB6 [WP]
PGEEZ1 M95P32 BUS IO driver	BSP_BUS_DRIVER	SPI:Full-Duplex Master	SPI1
PGEEZ1_M95P32_CS		GPIO: Output	PA9 [CS]
EEPRMA2_M95040		GPIO: Output	PA10
BSP USART	BSP_COMMON_DRIVER	USART:Asynchronous	USART2

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

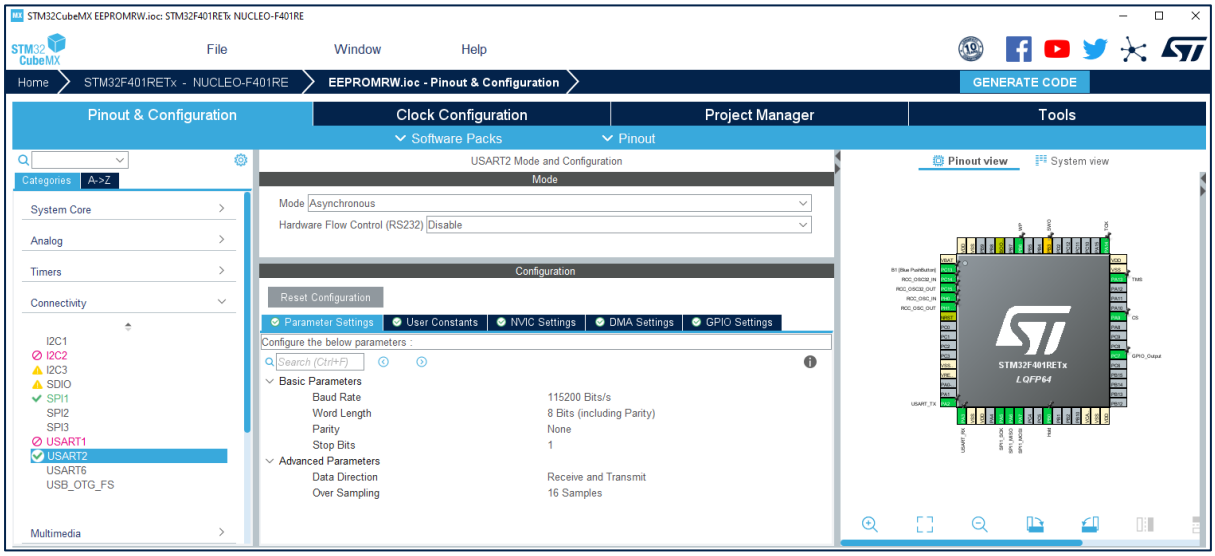

Figure 19: STM32CubeMX USART Configuration

If the NUCLEO board is **144-pin**, the software packs support QSPI:

- from the **Pinout** scheme, click on PB2 and set it as QUADSPI_CLK.
- from the **Pinout** scheme, click on PD11 and set it as QUADSPI_BK1_IO0.
- from the **Pinout** scheme, click on PD12 and set it as QUADSPI_BK1_IO1.
- from the **Pinout** scheme, click on PD13 and set it as QUADSPI_BK1_IO3.
- from the **Pinout** scheme, click on PE2 and set it as QUADSPI_BK1_IO2.
- from the **Pinout** scheme, click on PG6 and set it as QUADSPI_BK1_NCS.
- enable the QUADSPI as SPI from the “Connectivity” category

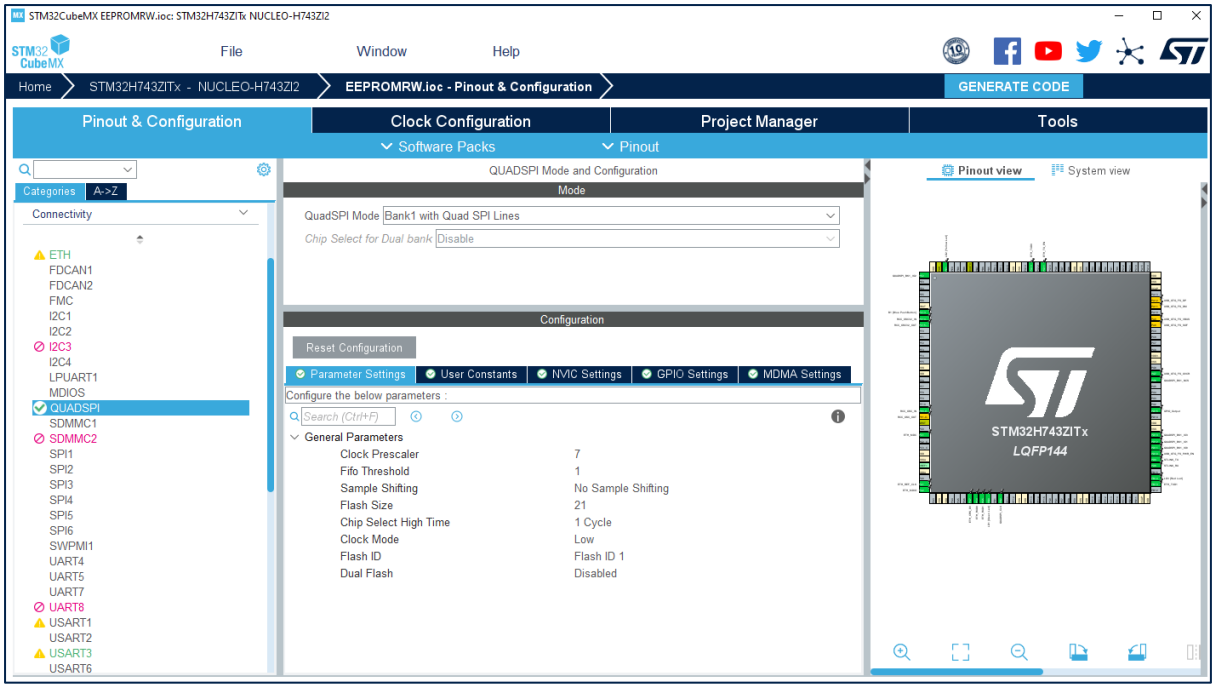

Figure 20: STM32CubeMX X-NUCLEO-PGEEZ1 QUADSPI Configuration

Nucleo 144	
PB2	QUADSPI_CLK
PD11	QUADSPI_BK1_IO0
PD12	QUADSPI_BK1_IO1
PD13	QUADSPI_BK1_IO3
PE2	QUADSPI_BK1_IO2
PG6	QUADSPI_BK1_NCS

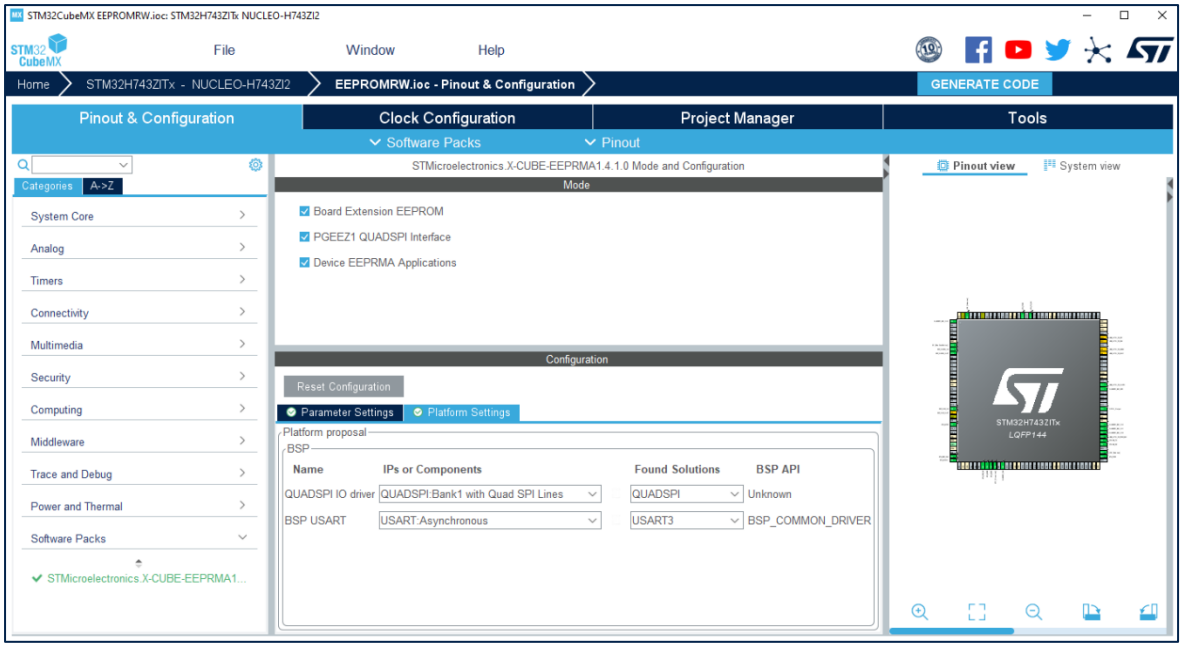

Figure 21: STM32CubeMX Pinout & Configuration tab and QUADSPI settings

From the **Software Packs** category, press the 'Stmicroelectronics.X-CUBE-EEPRMA1.4.2.0' item, enable the "Board Extension EEPROM" and "Device EEPRMA Applications" checkbox from the "Mode" view and set the following Platform Settings from the "Configuration" view (consider that according to the example chosen some settings can appear or not):

Name	BSP API	**Supported IPs**	**Nucleo 144**
QUADSPI IO driver		QUADSPI:Bank1 with Quad SPI Lines	QUADSPI
BSP USART	BSP_COMMON_DRIVER	USART:Asynchronous	USART3

From the **Configuration & Pinout** tab, click on "Connectivity" category and then on USART3 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

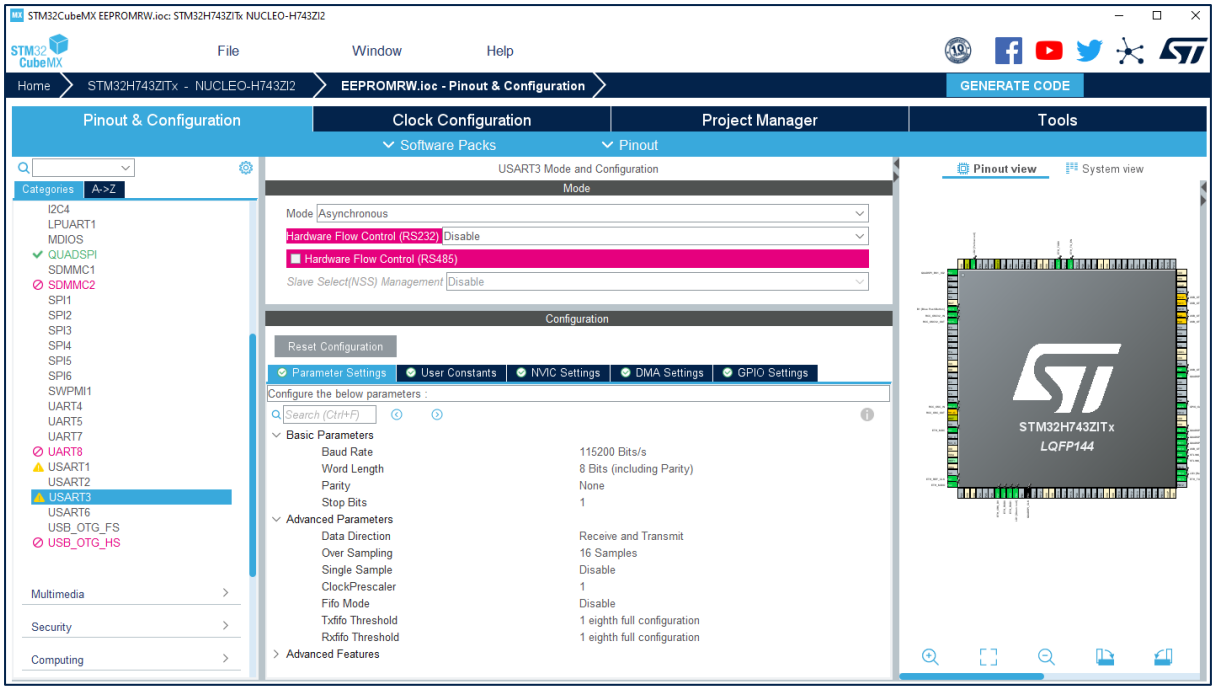

Figure 22: STM32CubeMX USART Configuration

Once all the above-described steps have been performed, the sample applications for X-NUCLEO-EEPRMA2 using the **STMicroelectronics X-CUBE-PGEEZ1** 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-EEPRMA1 for the X-CUBE-EEPRMA1 pack), at the same level as Core and containing inside the App and the Target subfolder.

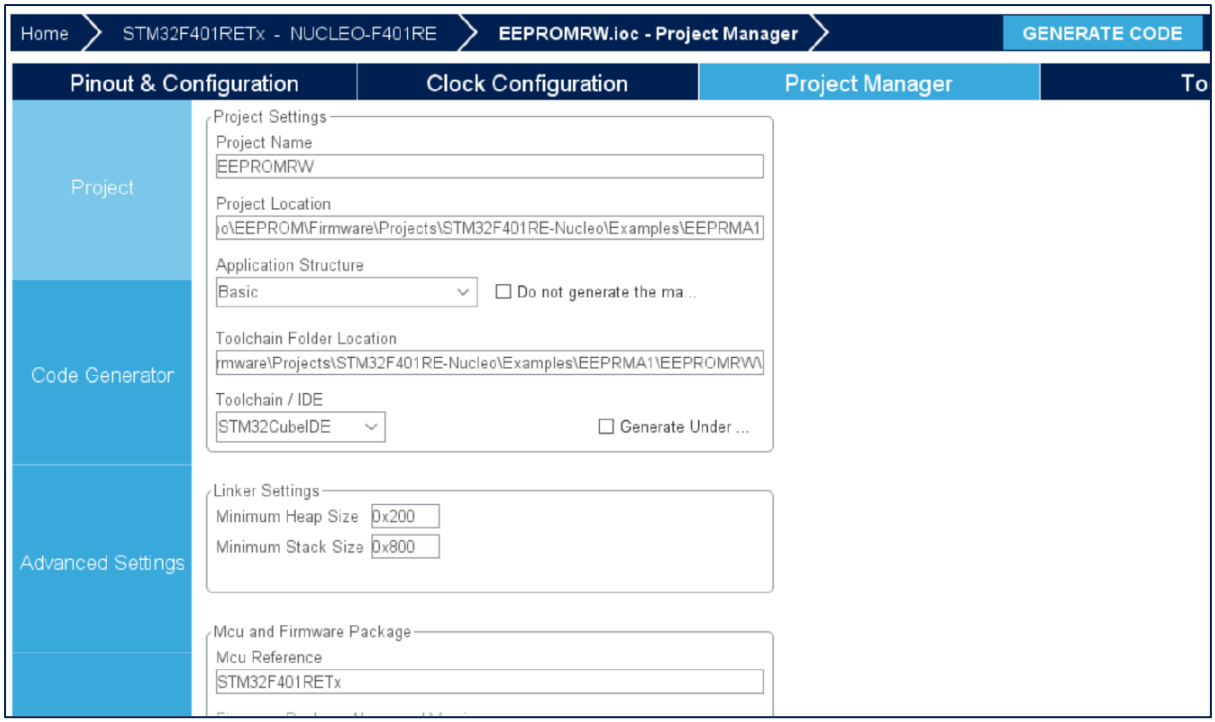

Figure 23: STM32CubeMX Application Structure Configuration

9. Known Limitations and workarounds

STM32CubeMX X-CUBE-EEPRMA1 pack V4.2.0 is fully compatible with STM32CubeMX v6.9.2. It is not fully compatible with the previous version of STM32CubeMX (<=v6.0.0).

10. References

[1] UM2481– User Manual - Getting started with the X-CUBE-EEPRMA1 software expansion for STM32Cube
[2] DB4092– Data Brief - Standard I²C and SPI EEPROM memory expansion board based on M24xx and M95xx series for STM32 Nucleo
[3] UM2665– User Manual - Getting started with the X-NUCLEO-EEPRMA2 standard I²C and SPI EEPROM memory expansion board based on M24xx and M95xx series for STM32 Nucleo
[4] DB4863– Data Brief - Standard SPI page EEPROM memory expansion board based on M95P32 series for STM32 Nucleo
[5] UM3096– User Manual - Getting started with the X-NUCLEO-PGEEZ1 standard SPI page EEPROM memory expansion board based on M95P32 series for STM32 Nucleo

11. Revision History

Date	Version	Changes
16-Sept-2020	1	Initial release
23-June-2021	2	Release version of X-CUBE-EEPRMA1 updated to 3.1.0
31-Oct-2022	3	Release version of X-CUBE-EEPRMA1 updated to 4.0.0
23-Nov-2022	4	Release version of X-CUBE-EEPRMA1 updated to 4.1.0
07-Oct-2023	5	Release version of X-CUBE-EEPRMA1 updated to 4.2.0

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.