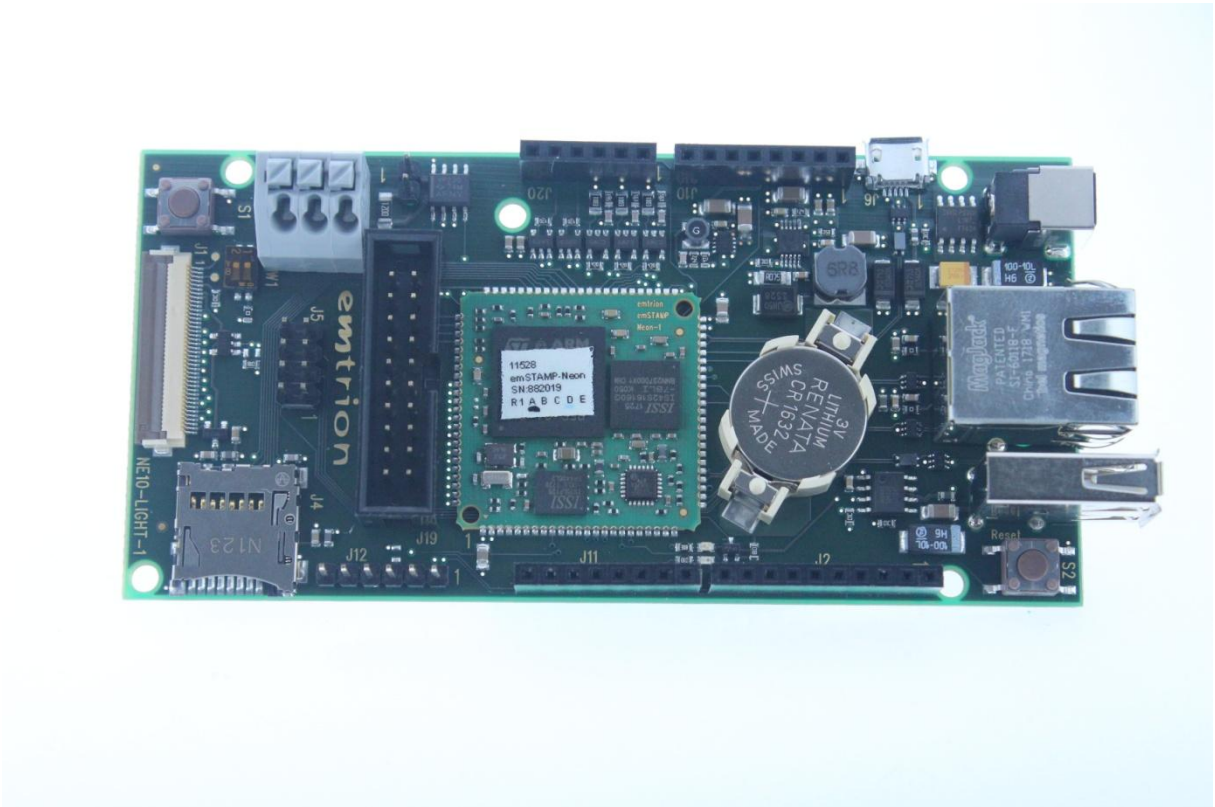
emSTAMP Neon Developer Kit

Software Manual

Rev02 / 27.09.2018



emtrion GmbH

emtrion
embedded systems

Revision: **02 / 27.09.2018**

Rev	Date/Signature	Changes
1	01.08.2018/Ha	Initial release
2	27.09.2018/Ha	Minor change, fix typo

1 Table of contents

Revision: **01 / 01.08.2018** 2
2 Introduction 4
3 Hardware requirement 4
3.1 JTAG debugger/programmer 4
4 Workstation Software installation 5
4.1 STM32CubeMX initialization code generator 5
4.2 STM32 ST-LINK utility 5
4.3 System Workbench for STM32 5
5 Generating initialization code with STM32CubeMX 5
6 Testing your in-circuit Debugger/programmer and your emStamp-Neon 7
7 Working with SW4STM32 to develop your application 8
8 Using the additional software package for peripheral initialization 12
8.1 How to use the snippet code 12

2 Introduction

Welcome to emSTAMP-Neon documentation. This manual will give you a startup software guideline of our developer kit. It will describe how to use the different free software to program your developer kit.

It is assumed that users of emtrion developer kits are already familiar with software development. Programming knowledge are out of the scope of this document. emtrion will gladly assist you in acquiring this knowledge. If you are interested in training courses or getting support, please contact the emtrion sales department.

The examples in this manual are demonstrated on specific hardware but if not mentioned otherwise they all work on all supported emtrion devices.

Please refer to the "Hardware Description" of to emSTAMP-Neon available on the emtrion support website (http://support.emtrion.de) for more detailed info of the capability of the product.

3 Hardware requirement

3.1 JTAG debugger/programmer

In this manual, the in-circuit debugger/programmer used is the ST-LINK/V2 (https://www.st.com/en/development-tools/st-link-v2.html).

4 Workstation Software installation

Before starting you need to prepare your workstation. In order to build an image that runs on the target, you need to install the following set of free software available online on the ST microelectronics, the microcontroller manufacturer (www.st.com).

4.1 STM32CubeMX initialization code generator

STM32CubeMX is a graphical tool that allows a very easy configuration of STM32 microcontrollers and the generation of the corresponding initialization C code through a step-by-step process. (https://www.st.com/en/development-tools/stm32cubemx.html)

Emtrion is providing a project file that is compliant with the developer kit and let you generate the MCU peripheral (GPIO, USART, Pin MUX …) and the middleware (USB, TCP/IP, FreeRTOS…). The output will be a bundle of initialization file in C code and all project files that allow you to create your main project.

4.2 STM32 ST-LINK utility

STM32 ST-LINK Utility is a full-featured software interface for programming STM32 microcontrollers (https://www.st.com/en/development-tools/stsw-link004.html)

The tool offers a wide range of features to program STM32 internal memories (Flash, RAM, OTP and others), external memories, to verify the programming content (checksum, verify during and after programming, compare with file) and to automate STM32 programming.STM32 ST-LINK Utility is delivered as a graphical user interface (GUI) with a command line interface (CLI).

4.3 System Workbench for STM32

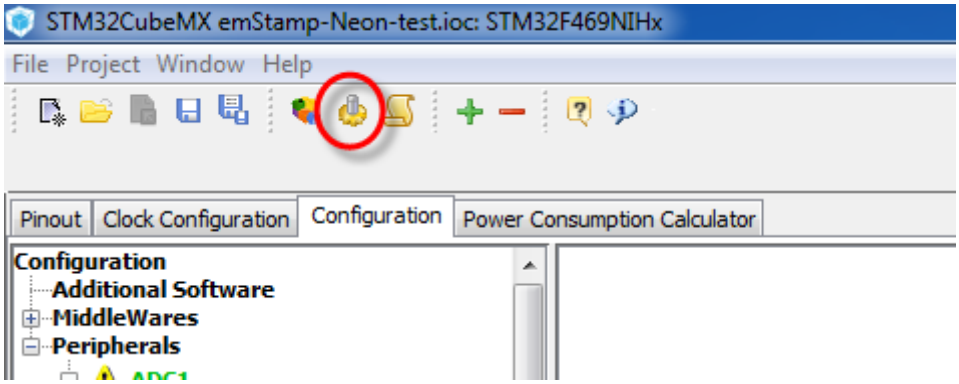
The System Workbench toolchain, called SW4STM32, is a free multi-OS software development environment based on Eclipse, which supports the full range of STM32 microcontrollers and associated boards (https://www.st.com/en/development-tools/sw4stm32.html)

The SW4STM32 toolchain may be obtained from the website (www.openstm32.org), which includes forums, blogs, and trainings for technical support. Once registered to this site, users will get installation instructions at the Documentation > System Workbench page to proceed with the download of the free toolchain.

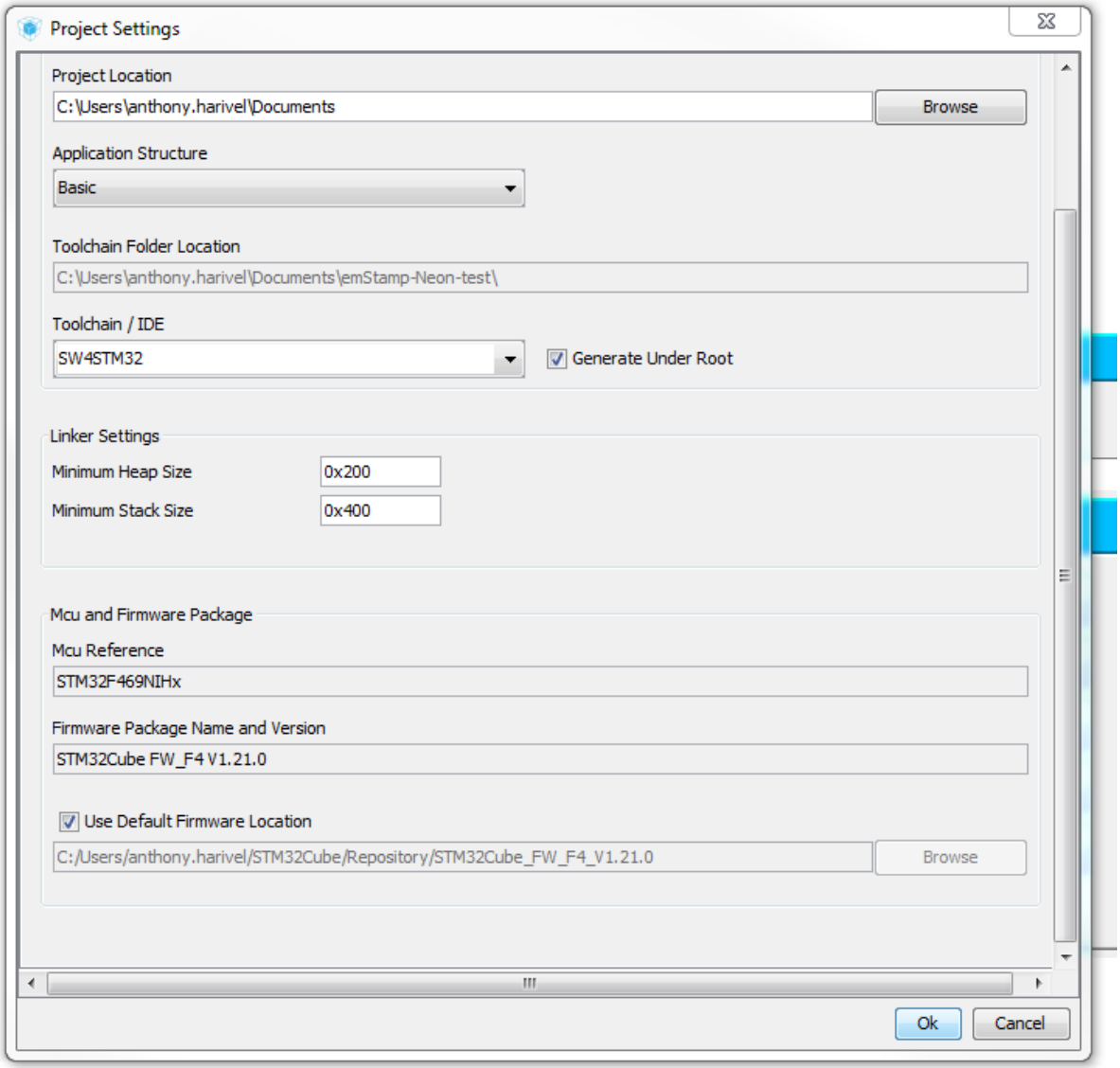
5 Generating initialization code with STM32CubeMX

Emtrion is providing the configuration file that gives you the possibility to load the entire pin muxing, clock configuration and middleware of the MCU STM32F469NIHx used in the emStamp-Neon. To proceed to the generation, load the project file called STM32-Stamp-Pinmux-V06-Release-emSTAMP-Neon-edi5937.ioc

Then click on the generation button here:



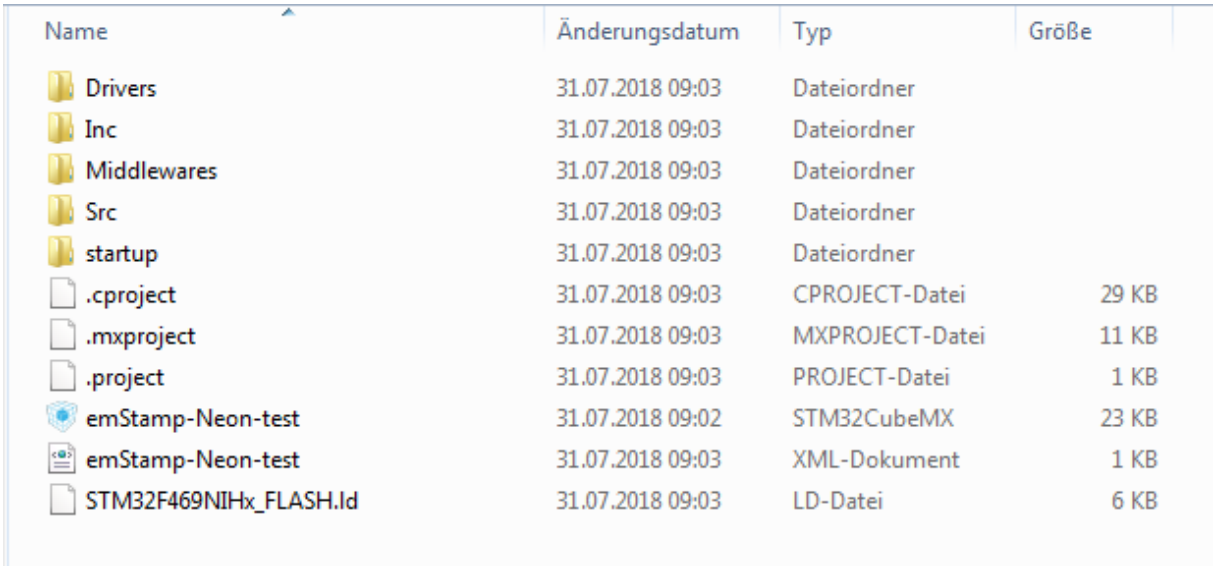
Then file the project setting:



The most important part is the Toolchain/IDE; choose SW4STM32 to be compliant with the rest of this manual. Click “Ok”.

Then wait for a moment while the code is generated.

Once finished you should get the source code framework of your project:

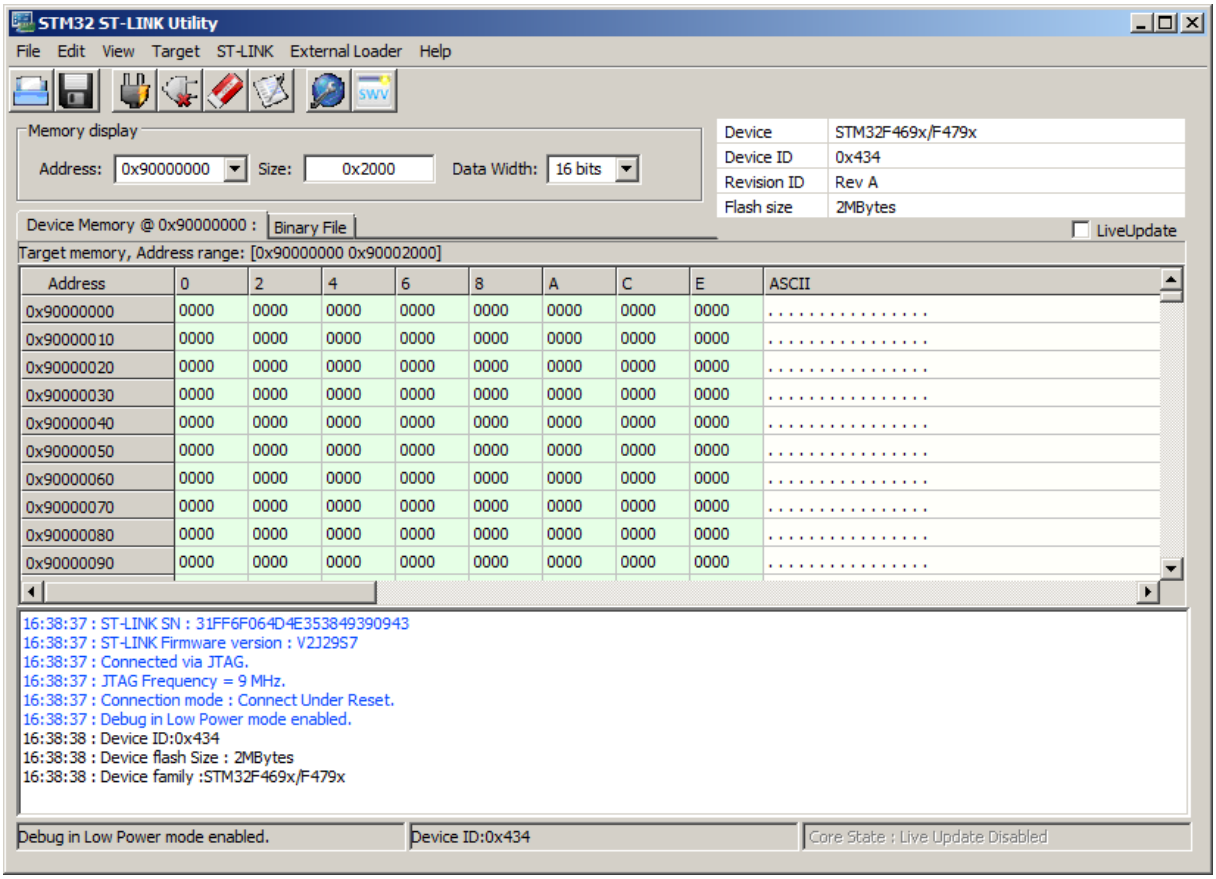


Name	Änderungsdatum	Typ	Größe
Drivers	31.07.2018 09:03	Dateiordner	
Inc	31.07.2018 09:03	Dateiordner	
Middlewares	31.07.2018 09:03	Dateiordner	
Src	31.07.2018 09:03	Dateiordner	
startup	31.07.2018 09:03	Dateiordner	
.cproject	31.07.2018 09:03	CPROJECT-Datei	29 KB
.mxproject	31.07.2018 09:03	MXPROJECT-Datei	11 KB
.project	31.07.2018 09:03	PROJECT-Datei	1 KB
emStamp-Neon-test	31.07.2018 09:02	STM32CubeMX	23 KB
emStamp-Neon-test	31.07.2018 09:03	XML-Dokument	1 KB
STM32F469NIHx_FLASH.ld	31.07.2018 09:03	LD-Datei	6 KB

6 Testing your in-circuit Debugger/programmer and your emStamp-Neon

It is strongly advice to test the communication between your workstation and your target using the ST-LINK utility tool before starting any debugging and/or programming with SW4STM32.

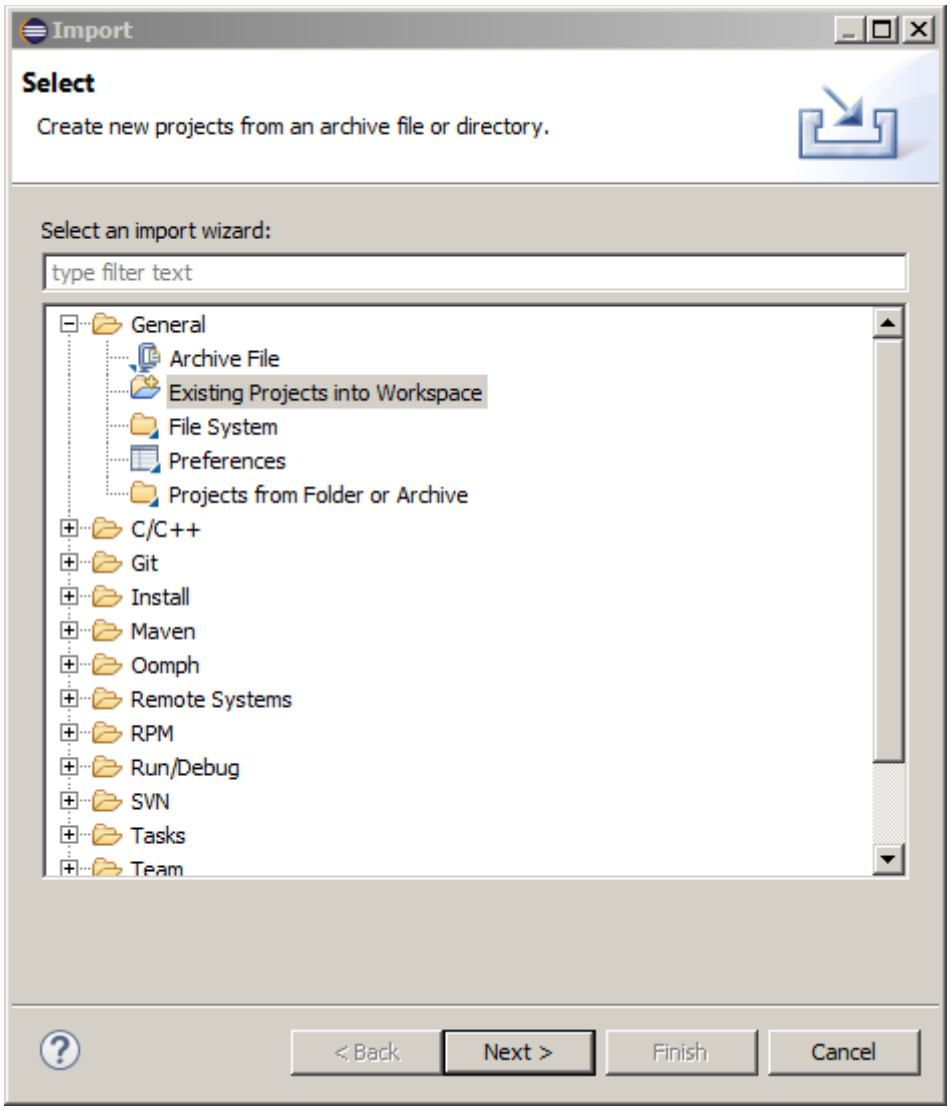
The output of a good communication should look like this:

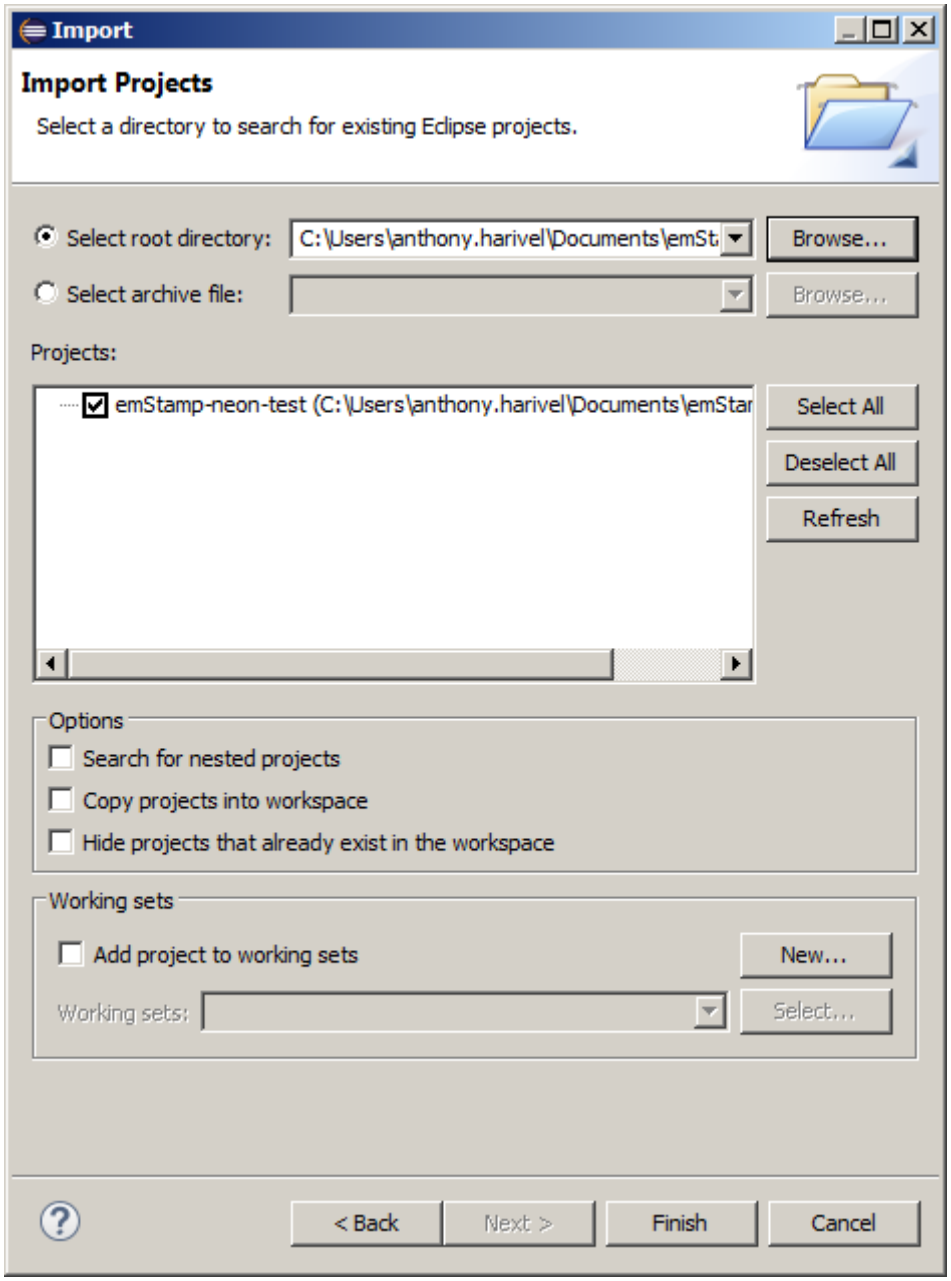


7 Working with SW4STM32 to develop your application

Once your code is generated, you can import your project into System Workbench for STM32 (SW4STM32).

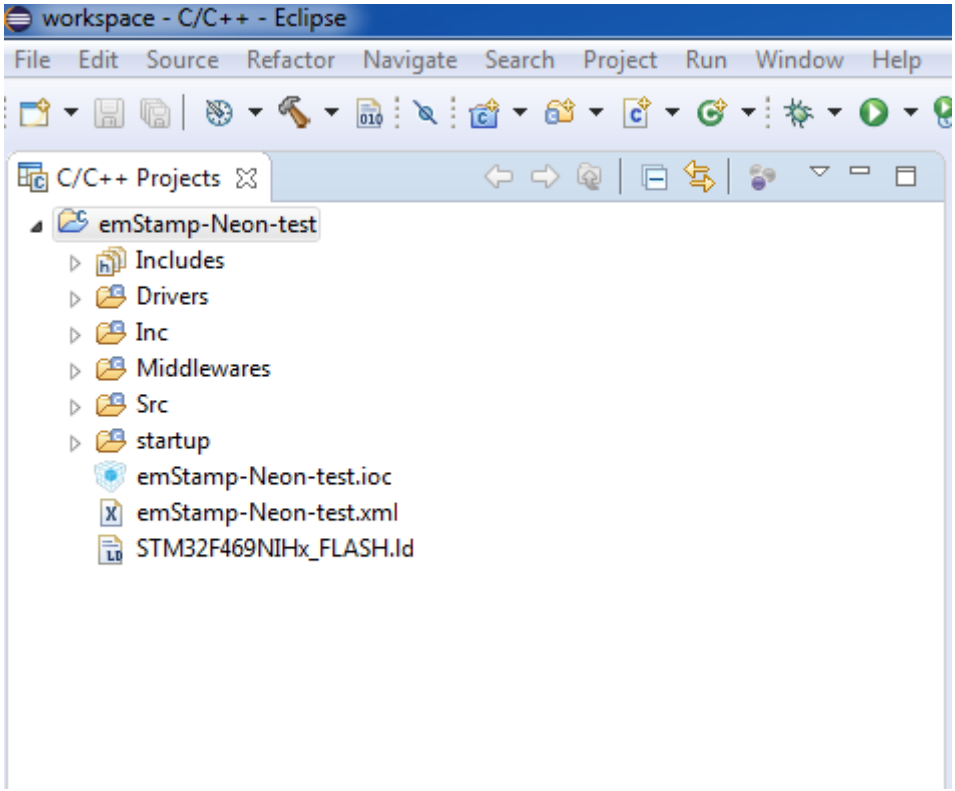
To import a project, choose the following setting:



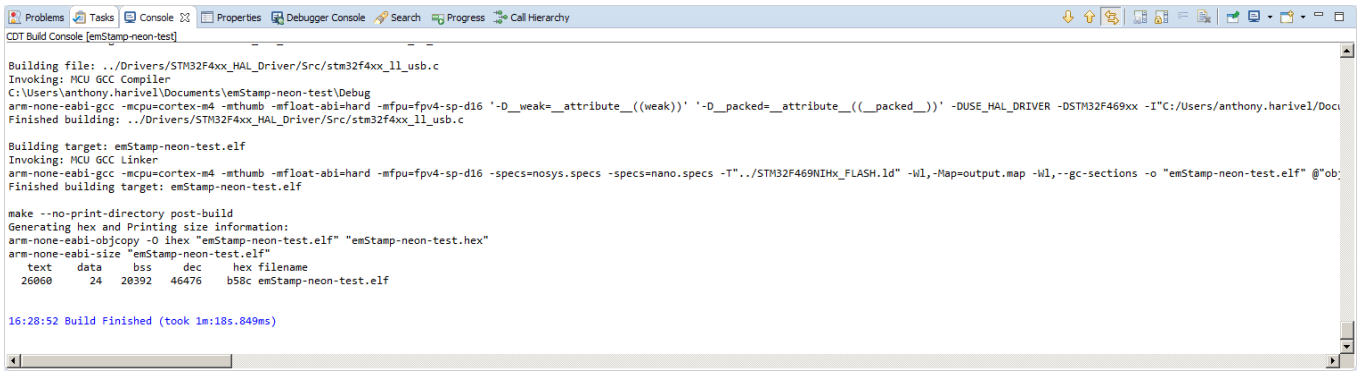


Click on “Finish” and let the IDE prepare your workspace.

Once imported you should see this source code framework into the IDE like this:



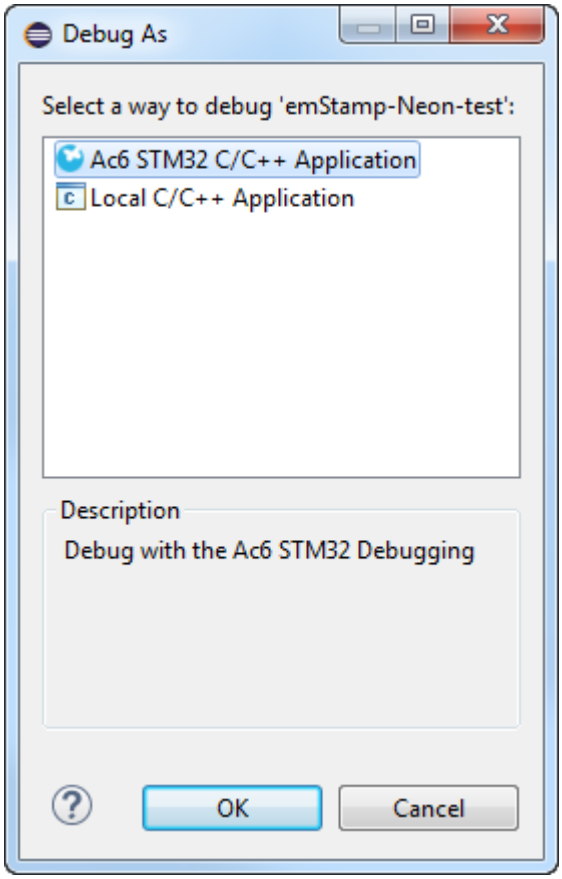
Once you are ready to compile, click on the hammer icon to build the “debug” version of your project. After a while, you should see this successful build with no error like this:



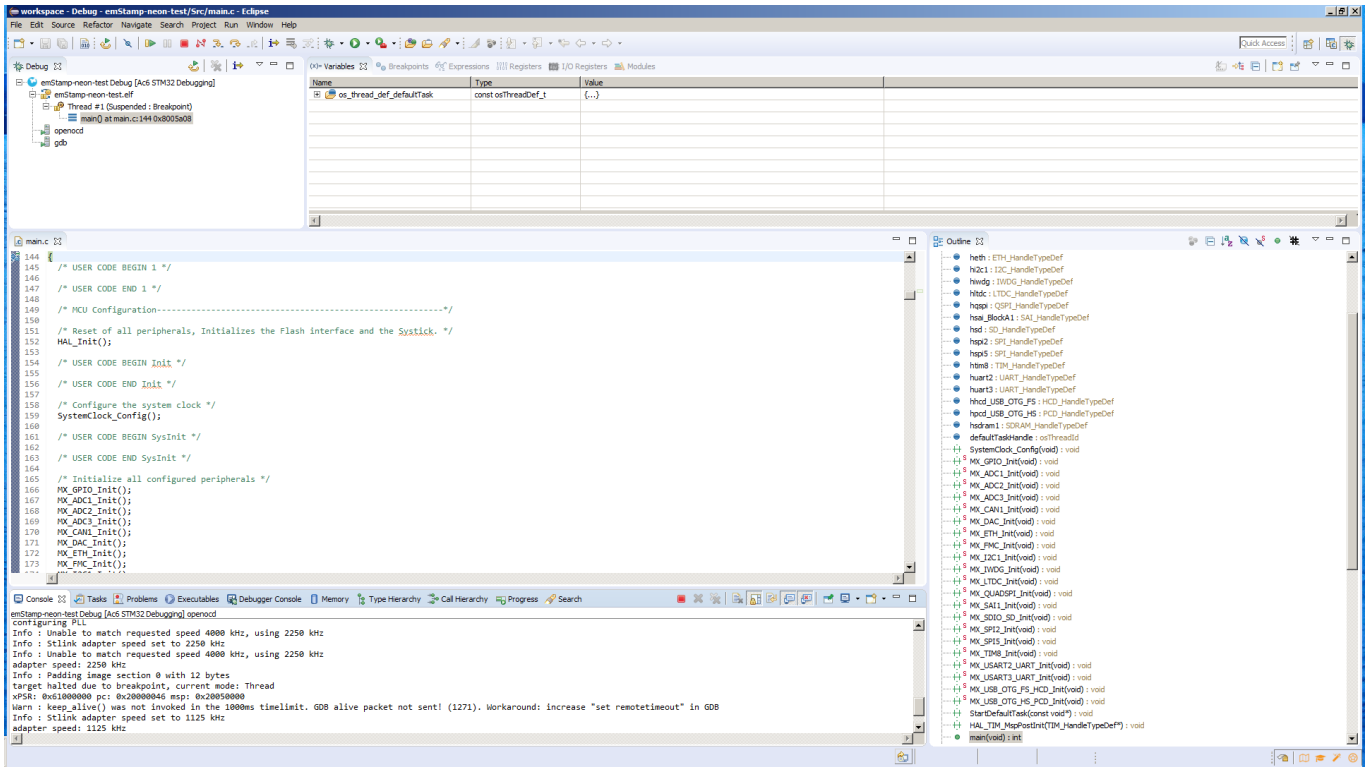
You are now ready to debug your application.

Make sure your target is power up and that the ST-LINK is connected properly and hit the bug icon to debug your project.

Choose the following configuration:



Once done you should get the IDE ready to debug the code on the target like this:



8 Using the additional software package for peripheral initialization

In order to put some peripheral like the RAM, the QSPI Flash, the LCD … in operation, some additional pieces of software need to be added in your project.

Emtrion is providing a software package with snippet code you can add in your code.

The package is divided into several folders corresponding to the peripheral you need to initialize.

The software configuration used into the snippet code is compatible only with the emStamp-Neon module peripheral.

8.1 How to use the snippet code

Once you have the previous steps done and you are able to debug your emStamp-Neon developer kit, you can customize your code.

The "main.c" file is the file where you can find your application entry point. There is a set of init functions that initializes the MCU and the peripheral.

For example, if you need to the external RAM to be in operation and you have selected it in STM32CubeMX, you will find in the "main.c" file the following function:

```
/* FMC initialization function */
static void MX_FMC_Init(void)
{
        …
```

In the software package, you will find a file "snippet.c" with the same function.

Replace the content of the snippet code into your init function and add the extra functions as well.

```
static void BSP_SDRAM_Initialization_Sequence(SDRAM_HandleTypeDef *hsdram,
FMC_SDRAM_CommandTypeDef *Command)
{
…
```

Now your project has the initialized the external RAM!

For some peripheral like the QSPI, a full driver has to be added into your project. Please follow each README.txt on the respectively folder that guides you to the installation.