



How to develop a STM32Cube Expansion Package

How to develop a STM32Cube Expansion Package



On this page, learn how to create your own STM32Cube Expansion package to challenge the limit of the ecosystem.

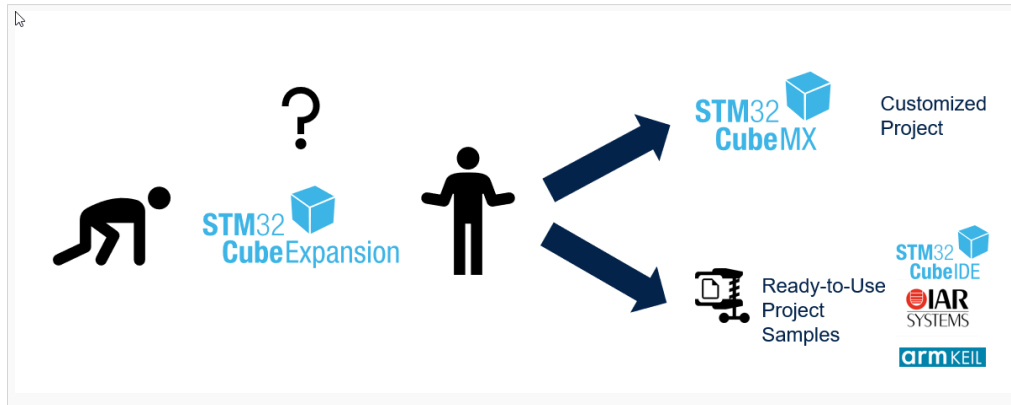
Contents

| | |
|--|----|
| 1 Design of a STM32Cube Expansion enhanced for STM32 Toolset | 3 |
| 2 Three webinars to start up | 4 |
| 3 Auto-assessment | 5 |
| 4 Three steps to design your STM32Cube Expansion enhanced for STM32 Toolset | 7 |
| 4.1 Step1: Design a <i>pack enhanced for STM32CubeMX</i> using <i>STM32PackCreator</i> | 7 |
| 4.2 Step2: Develop few examples | 7 |
| 4.3 Step3: Deliver a STM32Cube Expansion Package (I-CUBE) | 9 |
| 5 STM32Cube Expansion Structure | 10 |
| 6 Going further | 14 |
| 7 Frequently Asked Questions | 16 |



1 Design of a STM32Cube Expansion enhanced for STM32 Toolset

A STM32Cube Expansion enhanced for STM32 Toolset is an evolution of the initial concept of STM32Cube Expansion. The new packages contain now a part readable and configurable into STM32CubeMX (or STM32CubeIDE) on top of Ready-To-Use Project Samples.



It helps the integration of software components into STM32Cube ecosystem. The role is to limit the manual work to integrate a SW component (it limits but it remains some manual operations to be executed by the end user). The major advantage of integrating your software in STM32CubeMX is that you can allow the STM32CubeMX users to integrate your SW component on **any board or MCU from the STM32 Galaxy**.



2 Three webinars to start up

Three webinars have been prepared by our teams to show you a part of the capacity of the STM32Cube Expansion *Do It Yourself*. Each video lasts approximately 1 hour and needs between half a day to one day to reproduce it.

- **Sequencer Sample :**

An utility for multi-tasking without an Operating System, which is HW independent code.

In the example, a LED is managed through a GPIO.

 [Sequencer video](#)

 [Sequencer video \(chinese\)](#)

 [Download the Sequencer Sample code](#)

- **DSPDemo Sample :**

This SW component uses the DSP integrated in the Core to generate an internal signal. This internal signal is then captured and displayed thanks to CubeMonitor. There is no interaction with the external hardware.

 [DSPDemo video](#)

 [DSPDemo video \(chinese\)](#)

 [Download the DSPDemo Sample code](#)

- **STDIO Sample**

The application code frequently uses **standard I/O library functions**, such as *printf()*, *scanf()*, or *fgetc()* to perform input /output operations. The I/O library functions can be redirected to use different channels (peripheral) depending on the hardware and the application needs. The most common channels used for **STDIN** (*scanf*, *getchar*) and **STDOUT** (*printf*) are **UART** and **USB Device Virtual Com Port class**. **STDIN** could retrieve data from a USB keyboard using the USB Host HID class.

 [STDIO video](#)

 [STDIO video \(chinese\)](#)

 [Download the STDIO Sample code](#)



3 Auto-assessment

Any SW component is not adapted to the development of a STM32Cube Expansion enhanced for STM32 Toolset. Indeed, the possibility of configurations is huge and not covered completely by the tools. Some other methods may be feasible but are not considered in the normal flow.



We refer to this list of questions to identify the right candidates. The questions should be taken in the generic understanding of the Industry.

- **Did I already use STM32CubeMX or STM32CubeIDE to generate a project?**
 - **If yes**, move to the next question
 - The concept is based on the usage of STM32CubeMX. We recommend starting only when you have already acted as a standard user of STM32CubeMX. If not, start downloading and opening STM32CubeMX, pick up a STM32 MCU, configure peripherals, generate the project for your preferred IDE and look at the result.
- **Is <My SW component> well structured?**
 - **If yes**, move to the next question
 - My software is structured in functional units/components self-content from an architecture point of view.
- **Does <My SW component> have a standard build system?**
 - **If yes**, move to the next question
 - We exclude cases such as complex cases with different linker scripts, generating more than one binary file, if your software needs to dynamically adapt the run-time configuration. STM32CubeMX generates simple project structure (cf basic or advanced structure).
- **Am I ready to adapt my architecture to STM32Cube rules when interacting STM32Cube components?**
 - **If yes**, move to the next question
 - STMicroelectronics has defined some rules to interact between STM32Cube components to guarantee coherency between SW components. There is no impact on your own SW component architecture but on the structure of the project (location of your SW component into the project itself, location of the HAL drivers, usage of STM32Cube API...). Indeed, the project structure must be as expected by STM32CubeMX. Using STM32PackCreator and STM32CubeMX prevents most of the problems (link to STM32PackCreator Getting Started).
- **Am I ready to add this delivery method on a long run?**
 - **If yes**, move to the next question
 - We value working with partners and customers on the long run. The work related to the STM32CubeExpansion package is valuable if you intend to upgrade it regularly with new versions of your SW component. Therefore, the method could be done one time but needs to be thought on the long run.
- **<My SW component> is HW agnostic**
 - Check the [DSPDEMO STM32CubeExpansion sample training](#)
 - Your software component has no relation with any HW components (no peripherals of STM32Cube Ecosystem offer). This video training and the associated I-Cube describe the creation of STM32CubeExpansion in a simple SW component.
- **<My SW component> has some HW relation with UART, I2C, SPI, GPIO**
 - Check the [Sequencer STM32CubeExpansion sample training](#)
 - This video training and the associated I-Cube describe the creation of STM32CubeExpansion in a simple SW component with the usage of GPIO.
- **<My SW component> has some HW relation with Timers**
 - Take care: refer to *STM32PackCreator limitation*
 - Basic usage of timers is supported. Complex and advanced features are too complex to be implemented in our tools.
- **<My SW component> has some HW relation with other peripherals than the ones mentioned above**
 - Be cautious - check the [STDIO STM32CubeExpansion sample training](#)



4 Three steps to design your STM32Cube Expansion enhanced for STM32 Toolset

The design of a STM32Cube Expansion enhanced for STM32 Toolset is usually done in three steps. In real work, you will have several iterative works between Step1 and Step2 until the right result.

4.1 Step1: Design a *pack enhanced for STM32CubeMX* using *STM32PackCreator*

First and foremost, start STM32PackCreator: Go to

`$YourLocation$\STMicroelectronics\STM32Cube\STM32CubeMX\utilities\STM32PackCreator`

The methods to develop the step1 are explained through the video listed above or in the *Getting started Pack Creator*, accessible in the help menu of the *STM32PackCreator*.

Ensure that you have activated the **PACKAGES STM32Cube RULES COMPLIANT** in the *STM32PackCreator*.

Intermediate steps in this Step1 may be:

- Design a CMSIS Pack
- Design a simple configurable Pack
- Integrate a HW dependency
- Integrate an application

To train yourself on the method, we recommend going first to those [Video & Tutorials](#) and then to read *STM32PackCreator* user manual.

You have obtained a pack enhanced for STM32CubeMX. This pack can be loaded, configured in STM32CubeMX and can generate the associated code.

4.2 Step2: Develop few examples

You are going now to develop two to three examples on one or few STM32 boards. Those examples must of course be generated by the STM32CubeMX tool, using your *pack enhanced for STM32CubeMX*.

This step has two interests:

- Validate your implementation of the pack on different HW and different configurations,
- Prepare some project examples that you will integrate into the STM32CubeExpansion package itself. Your customers may indeed start directly from the example, without using the STM32CubeMX tool.

To execute it, you will act as a user of your pack enhanced for STM32CubeMX and execute the following actions:

- Load your pack “enhanced for STM32CubeMX” inside STM32CubeMX (cf STM32CubeMX user manual for more details).

We recommend the usage of STM32CubeMX. Indeed, even if STM32CubeIDE integrates the STM32CubeMX plugin, it will be not possible to generate the projects for the three IDEs (IAR™ EWARM, Keil® MDK-ARM, and STM32CubeIDE) from STM32CubeIDE. It will be limited to STM32CubeIDE only.

- Configure the peripherals, the SW components, the clock, the project settings to be adequate to your need.
- Copy paste the Drivers (HAL) that you need inside the Drivers directory inside your Package
- Generate a project (check the project settings below to take into account the Drivers from your Package).
- Add some user code in the user section to make it demonstrable.



- Save the STM32CubeMX configuration file (.ioc) for each project. We recommend saving the name of the project like *Name_Of_Application#N*. It will be reflected also on the .ioc file which will be called *Name_Of_Application#N.ioc*.

The project settings must be reflected like that

STM32CubeMX Panel: Project Generator Settings2

and

STM32CubeMX Panel: Project Generator Settings

- Generate your projects for the 3 IDEs (IAR™ EWARM, Keil® MDK-ARM, and STM32CubeIDE) and test them.

i In the special case of STM32CubeIDE, take care to unselect *Generate Under Root* before generating that to have a coherent structure for the three IDE:s.

STM32CubeMX Panel: CubeIDE specific settings



4.3 Step3: Deliver a STM32Cube Expansion Package (I-CUBE)

This is the final step. It integrates the outputs of step1 and step2 together into a package called STM32Cube Expansion Package. These packages are published on [st website](#) with X-CUBE part numbers if they are distributed, maintained and supported by STMicroelectronics. Some packages distributed, supported and maintained by ST Authorized Partners are also exposed on [st website](#) with I-CUBE part numbers. The interests of this step are that you will deliver to your customer a single deliverable. - This .pack deliverable contains all necessary files to import software components and configure them in STM32CubeMX or STM32CubeIDE tool, and generate a project for any STM32-based hardware. - The .pack is in ZIP format and can also be unzipped and users can get started with ready-to-use project examples, pre-configured to compile in the 3 IDEs (IAR™ EWARM, Keil® MDK-ARM, and STM32CubeIDE) and run out-of-the-box on specific ST boards.

First, some cleanup actions are required:

- The line `ProjectManager.FirmwarePackage=XXXX` must be removed from the file `.ioc` included in the examples folder. Indeed, STM32CubeMX versions will continue evolving and to avoid version issues remove this line.
- The STM32Cube Expansion Package PDSC file must be checked with the `PackChk.exe` provided by Arm.
- All non-user comments must be removed from the IPconfig and IPmodes files, when available

Then, some documents must be written

- A user getting started document (chm, rtf, pdf...) located under *Documentation folder* at the root of the *Package* is also expecting. the content must be:
 - How to use your specific *STM32Cube Expansion in STM32CubeMX*
 - The goals and usage of project examples
 - Remaining manual operation on the charge of the end user if any.
- A release note document must be also included, describing the content release, the corrected issues....
- A doc/ folder, containing the file describing APIs and parameters of Middleware elements, must be located at the same level as the Middleware Release_Notes.

Several examples of such documents can be found into [Expansion Packages](#).

The documents must be referenced in the pdsc file using the following options (according to CMSIS-pack standard):

- For component or api documentation, use: `<file category="doc" ...>`
- For bundle documentation, use: `<doc ...>`

You can add some files and example projects in the additional files to the existing package thanks to the STM32PackCreator. You need to use *Additional Files* button.



The **.pack** format is a compressed format, and so can be opened with Winzip or equivalent tool.

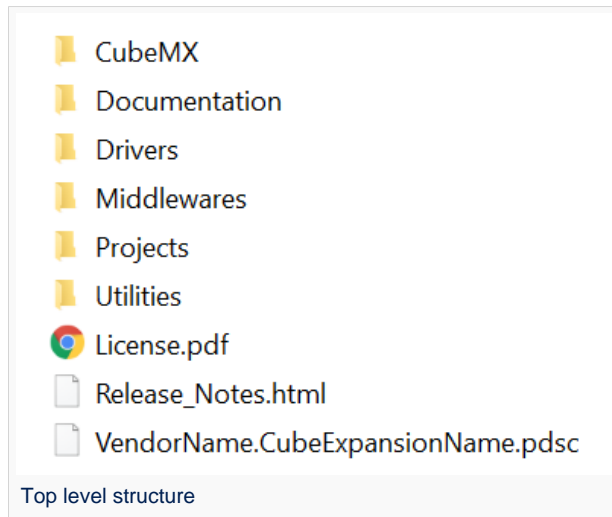
You are done! You can now deliver it as is to your customers, partners.

If you want to go further and make it exposed in the STM32CubeMX tool, you can contact your ST local representative.



5 STM32Cube Expansion Structure

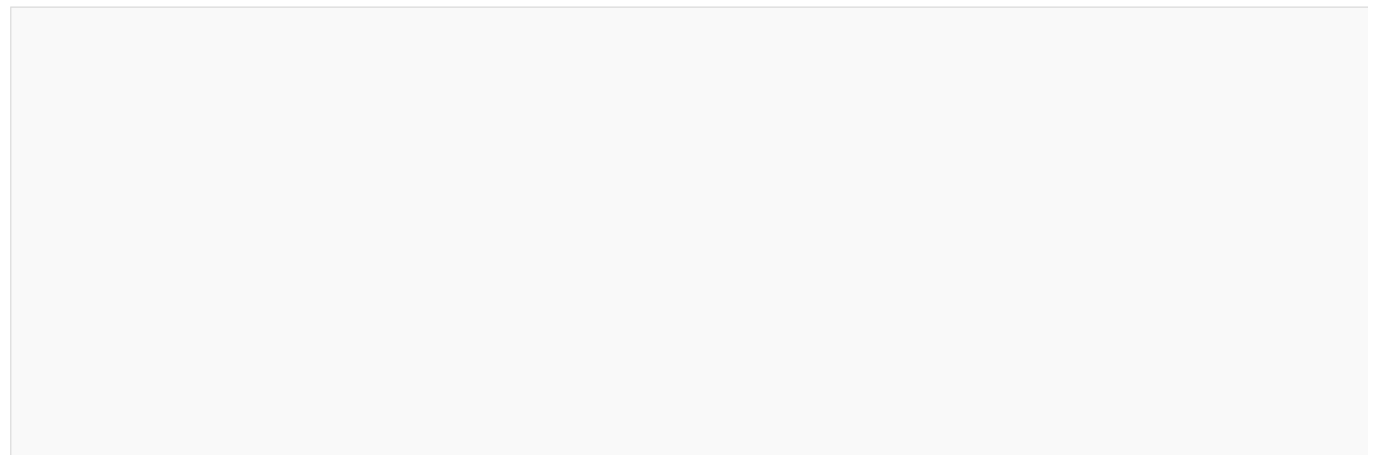
The STM32Cube Expansion Structure top level is the following:

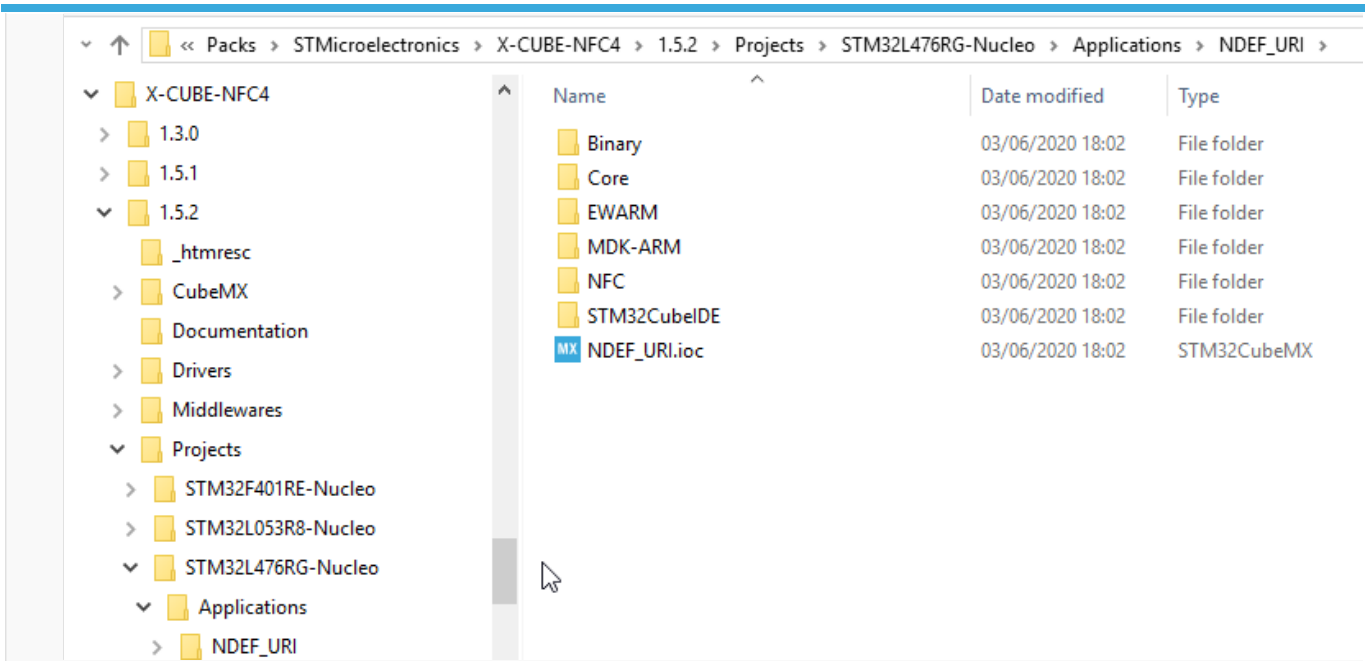


- **STM32CubeMX**: generated by STM32PackCreator (available in STM32CubeMX panel Tools). It contains the files required by STM32CubeMX in order to configure the SW Component.
- **Middlewares**: contains your SW component itself. Under ST subdirectories when coming from ST, under “Third_Party” when coming from your side or from a community.




- **Projects (examples)**: contains the examples of projects (2 or 3). They have been generated by STM32CubeMX for several IDEs and for several boards. Here is an example of advanced structure.

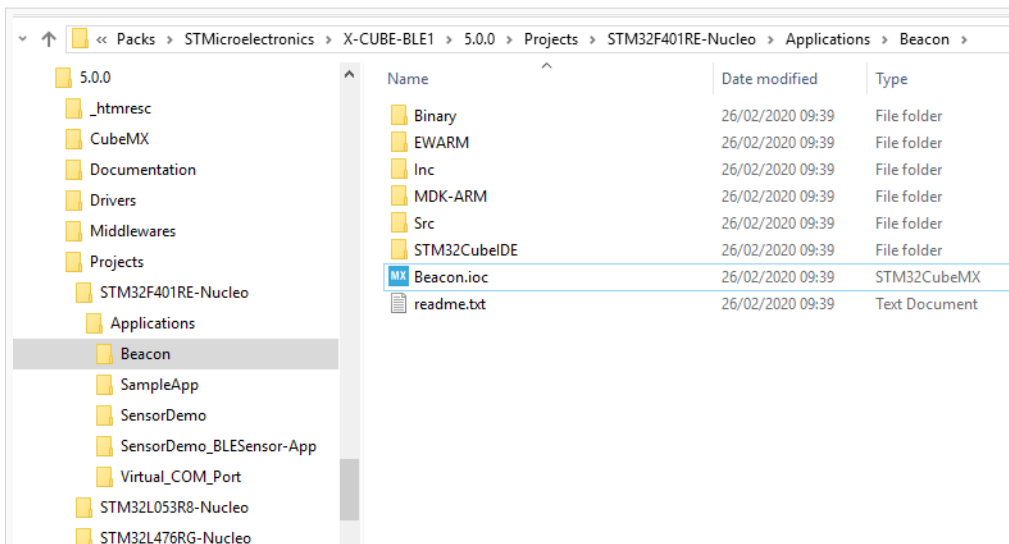




Example of X-Cube-NFC4 package - Advanced Project Structure

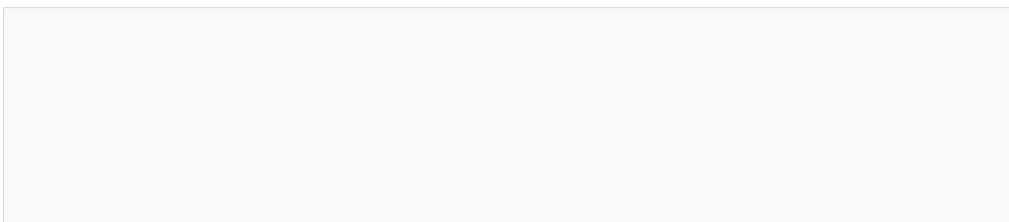
 starting from STM32CubeMX-6.0.0, code generation is in "advanced mode" by default - you can change it on the project generator panel

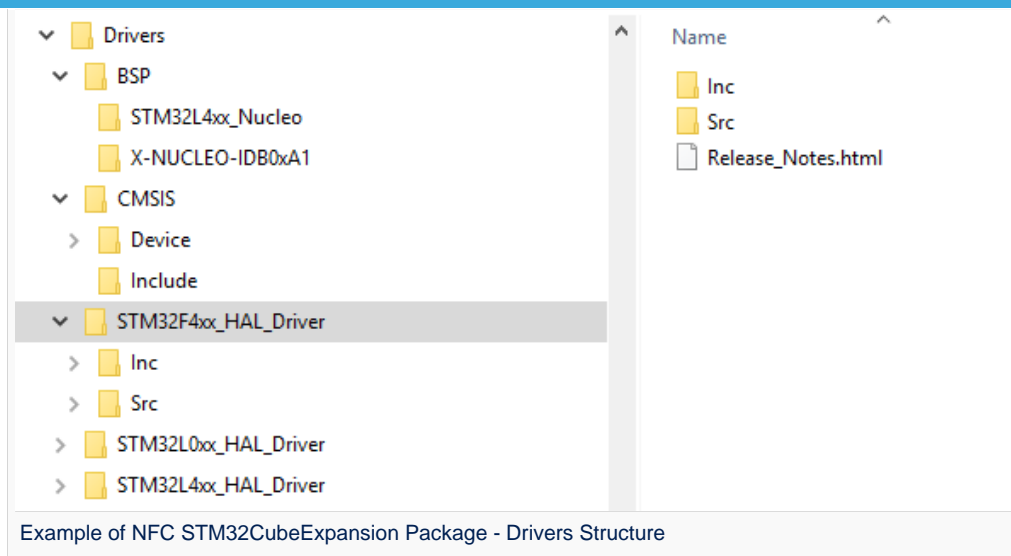
Here is an example of basic structure.



Example of X-Cube-BLE1 package - Basic Project Structure

- **Drivers:** contains the drivers which are required for the projects (examples). Copy paste them from the CubeFW Package of the series that you have taken as examples.





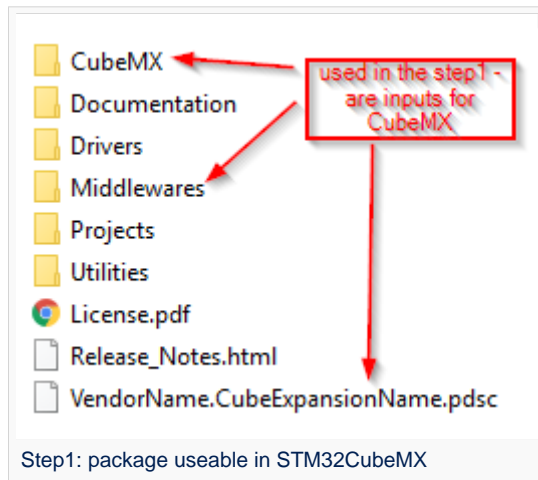
- **Utilities:** contains any PC software which could be useful for your pack.
- **Documentation:** contains the getting started and or any other documentation related to the package except the release note.

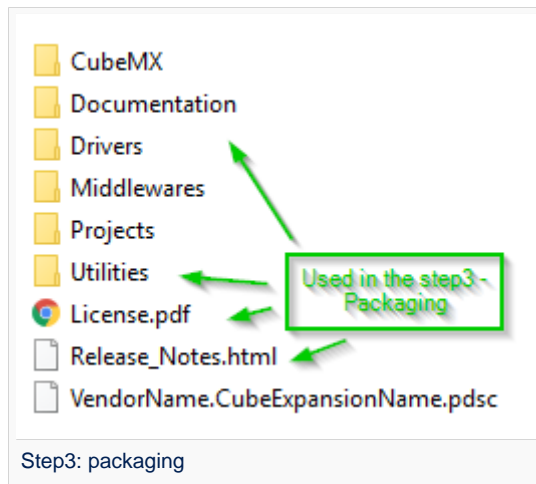
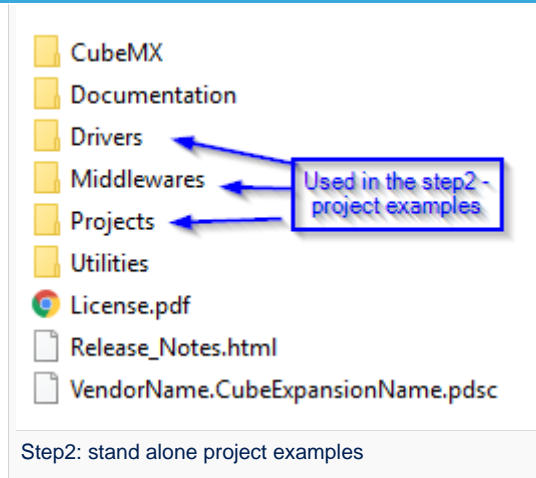
Other documents at the root of the package:

- **License.pdf:** contains the license. It will be displayed during the installation process
- **Release_Notes.html:** contains information on the release such as goals, modified files, corrected bugs and limitation if any.
- **VendorName.CubeExpansionName.pdsc:** the file pdsc contains the description of the file included into the packages.

The STM32CubeExpansion structure follows the standard STM32CubePackage structure. This structure is fully described into the document UM2285 Development guidelines for STM32Cube Expansion Packages.

In conclusion, if you want to map the "step1", "step2" and "step3", it will be like that







6 Going further

- [UM2739 STM32PackCreator user manual - Creating software packs and STM32Cube Expansion packages enhanced for STM32CubeMX, rev1](#)
- [UM2285 Development guidelines for STM32Cube Expansion Packages, rev2](#)
- [UM2388 STM32Cube Firmware Packs Specification, rev1](#)
- [UM2298 STM32Cube BSP drivers development guidelines, rev2](#)
- [UM2312 Development checklist for STM32Cube Expansion Packages, rev3 \(XLS Checklist XLS version of the Development checklist for STM32Cube Expansion Packages, rev4\)](#)





7 Frequently Asked Questions

- There are some existing plug in's for Eclipse to manage CMSIS pack. Could I use them with STM32CubeIDE on top of STM32CubeMX plug-in?

We do not test external eclipse plug in.

- Could I use STM32CubeIDE to test and generate a STM32Cube Expansion Package?

As a partner developing a STM32Cube Expansion package, we recommend using STM32CubeMX. Indeed, the STM32Cube Expansion package should include some examples for the 3 IDEs and this is feasible with STM32CubeMX.

- I have analyzed several ST STM32Cube Expansion (X-CUBE-BLE1, X-CUBE-MEMS) but they cannot be re-opened with STM32PackCreator. Have they been designed with STM32PackCreator?

The first X-CUBE have not been designed with STM32PackCreator. So, you may find some tricks which are not feasible with STM32PackCreator. STMicroelectronics supports only the STM32PackCreator feature set.

- There is a Example keyword into CMSIS pack standard. Could I use it to describe the 2 to 3 examples (step2) ?

So far, this keyword is not used by STM32CubeMX so far.

- There is some beautiful User Interface into STM32CubeMX (for some peripherals). Could I reproduce the same with STM32PackCreator?

STM32CubeMX allows some integration of java code to design complex User Interface. This feature is not expose to external plug in.

- Once the STM32Cube Expansion Package is selected into STM32CubeMX additional software management panel, I would like that some settings from dependent peripherals are automatically selected. Is it possible?

This is not possible and not expected. Indeed, your SW component can be used with different series and so, this is not expected to pre-fill the parameters. Our recommendation is to provide some examples with a defined configuration to your users and also a "getting started" document.

- Could I modify the XML files generated by STM32PackCreator?

STMicroelectronics does not support this case.

- Could I use the Pack ARM CMSIS?

The pack ARM CMSIS can be opened with STM32CubeMX but as some components are also coming from STM32Cube MCU Packages, you may face some incompatibilities. Take care about them.

- What is the difference between a STM32Cube Expansion Package and a CMSIS Pack?

The STM32Cube Expansion Package enhanced for STM32 Toolset is in general configurable. The generated code by STM32CubeMX when using STM32Cube Expansion Package has always a similar structure. It is based on STMicroelectronics HAL and LL drivers. It embeds some ready to use examples with .ioc file.

- Where are my packages stored?



Look at `%HOME%\STM32Cube\Repository\Packs`.