MicroEJ Platform Reference Implementation

Developer's Guide

STM32F412GDISCO 1.1.0

Reference:	TLT-0795-DGI-PlatformReferenceImplementation-STM32F412GDISCO
Version:	1.1.0
Revision:	1.1.0

Confidentiality & Intellectual Property

Revision History		
Revision 1.1.0	July 13th 2017	
MicroEJ 4.1 compliant		
Revision 1.0.3	February 2th 2017	
Front Panel chapter		
Revision 1.0.2	December 14th 2016	
BSP changes		
Revision 1.0.1	July 19th 2016	
BSP changes		
Revision 1.0.0	July 4th 2016	
First release		

Table of Contents

List of Figures

List of Tables

Chapter 1. Introduction

1.1. Intended Audience

The intended audience for this document are developers who wish to develop their first MicroEJ plaform with MicroEJ SDK and deploy a MicroEJ standalone application onto. Notes:

- This document is for the STMICRO STM32F412G-DISCO board.
- This document is not a user guide for the C development environment used for the final application link. Please consult the supplier of the C development environment for more information.
- Please visit the website https://developer.microej.com for more information about STM32F412G-DISCO products (platforms, videos, examples, application notes, etc.).

1.2. Scope

This document describes, step by step, how to start your development with MicroEJ SDK

- Create a MicroEJ platform for STM32F412G-DISCO board.
- Run a MicroEJ standalone application on the MicroEJ simulator.
- Run a MicroEJ standalone application on the MicroEJ platform and deploy it on the STM32F412G-DISCO board.

Note

This platform is delivered as a developer preview. It shall only be used for early development stage and not for production. This platform development, documentation and validation is not complete. Its known limitations are described in the associated errata document. There is no guaranty that the APIs or features will be maintained.

1.3. Prerequisites

- PC with Windows 7 or later.
- The MicroEJ SDK environment must be installed.
- STM32F412G-DISCO board.
- The STM32 ST-LINK utility (minimal version 3.9.0).
- Keil MDK-ARM µVision 5.18.0.0 or higher. The Keil µVision evaluation version is 32KB code size limited. To get a Keil µVision evaluation license for MicroEJ SDK, please consult the chapter “Install Keil MDK-ARM”.

Chapter 2. Create and Use Your First MicroEJ Platform

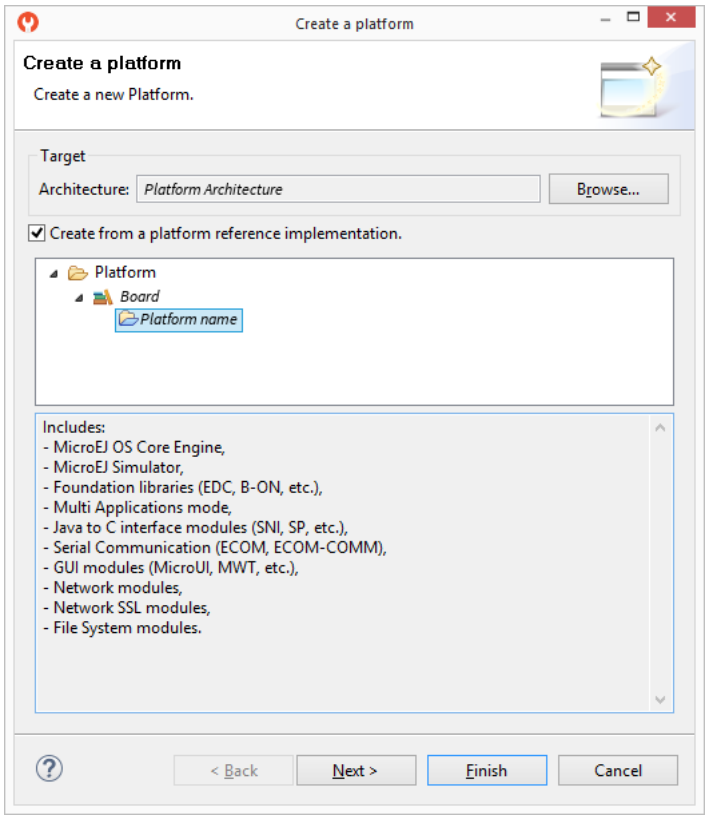
2.1. Create a MicroEJ Platform

The aim of this chapter is to create a MicroEJ platform from a MicroEJ architecture. The platform will then be used to run a MicroEJ standalone application in subsequent chapters.

Although it is possible to use MicroEJ SDK to create every aspect of a MicroEJ platform in accordance with specific requirements, in this chapter we will use a pre-packaged example of a MicroEJ platform that is already configured for the STM32F412G-DISCO.

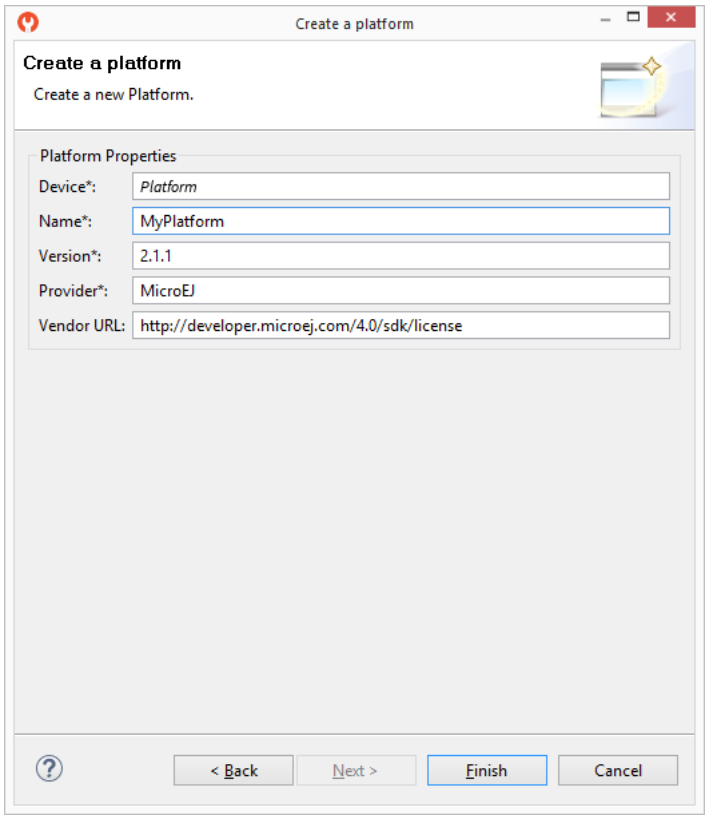
- Open MicroEJ SDK.
- Open the MicroEJ platform wizard: `File > New > Platform`.
- Select the MicroEJ architecture ARM Cortex-M4 ARMCC from the combo box. A MicroEJ Platform Reference Implementation is available:

Figure 2.1. MicroEJ Platform Reference Implementation Selection



- Select the MicroEJ platform SingleApp for the STM32F412G-DISCO from the combo box.
- Click on Next. Give a name which be used as prefix for all MicroEJ platform projects. For instance: `MyPlatform`.

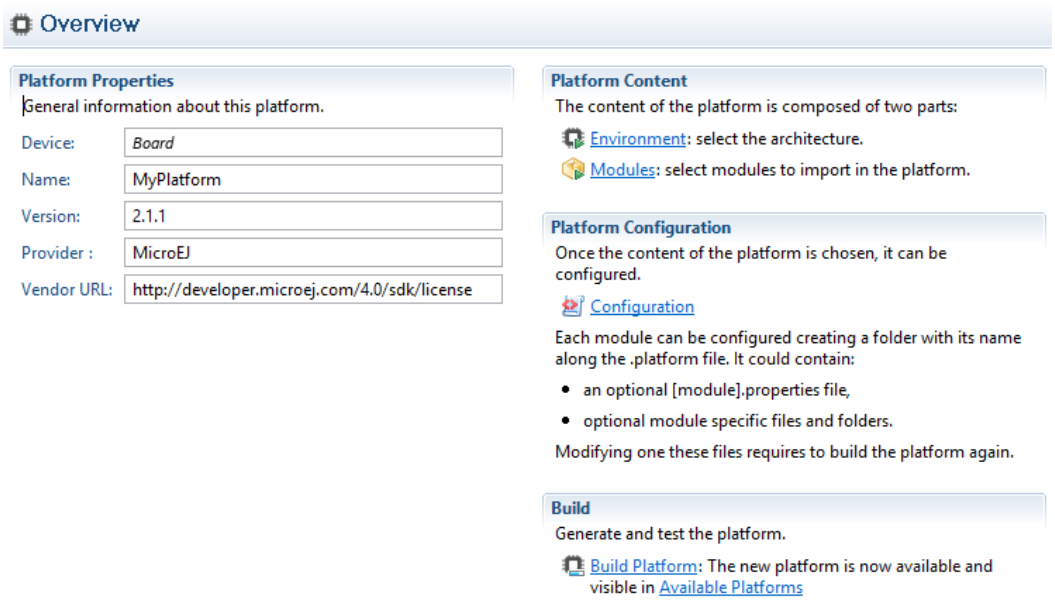
Figure 2.2. New MicroEJ Platform Naming



- Click on `Finish`. The selected example is imported as several projects prefixed by the given name:
  - STM32F412GDISCO-MyPlatform-CM4hardfp_ARMCC5-configuration: Contains the platform reference implementation configuration description. Some modules are described in a specific sub-folder / with some optional configuration files (`.properties` and / or `.xml`).
  - STM32F412GDISCO-MyPlatform-CM4hardfp_ARMCC5-bsp: Contains a ready-to-use BSP software project for the STM32F412G-DISCO board, including a Keil uVision project, an implementation of MicroEJ core engine (and extensions) port on FreeRTOS RTOS and the STM32F412G-DISCO board support package.
  - STM32F412GDISCO-MyPlatform-CM4hardfp_ARMCC5-fp: Contains the board description and images for the MicroEJ simulator. This project is updated once the platform is built.

  The MicroEJ platform configuration file is automatically opened.
- From the MicroEJ platform configuration file, click on the link `Build Platform`

Figure 2.3. MicroEJ Platform Build



The build starts. This step may take several minutes. You can see the progress of the build steps in the MicroEJ console. Please wait for the final message:

```
BUILD SUCCESSFUL
```

At the end of the execution the MicroEJ platform is fully built for the STM32F412G-DISCO board and is ready to be linked into the Keil uVision project. Its name is `STM32F412GDISCO-My-Platform-CM4hardfp_ARMCC5`.

The MicroEJ platform is now ready for use and available in the MicroEJ platforms list of your MicroEJ repository (`Windows > Preferences > MicroEJ > Platforms in workspace`).

2.2. Run an Example on the MicroEJ Simulator

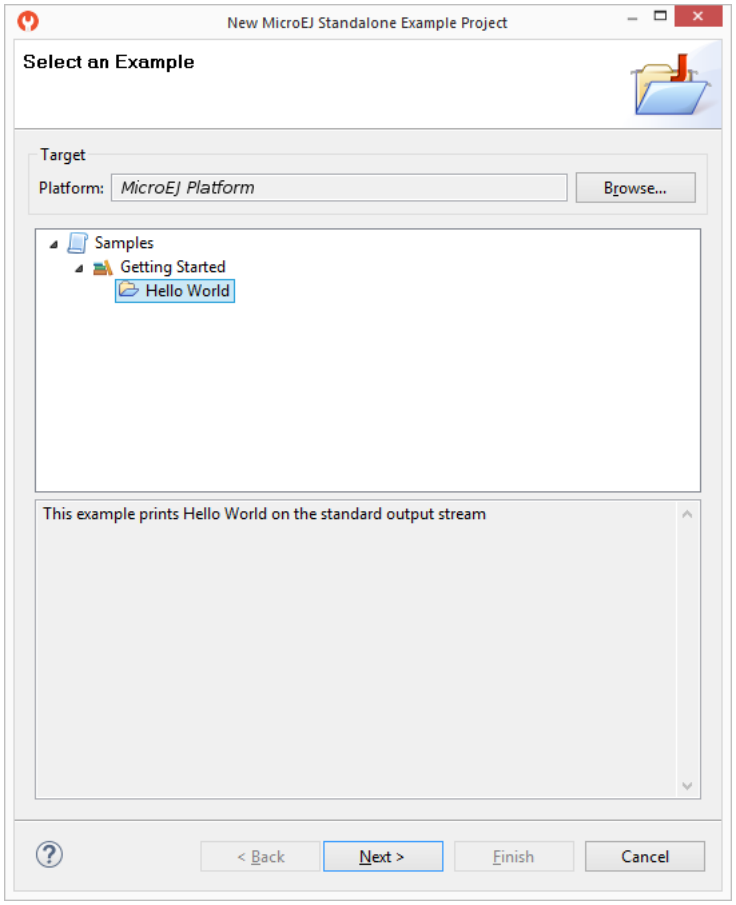
The aim of this chapter is to create a MicroEJ standalone application from a built-in example. Initially, this example will run on the MicroEJ simulator. Then, in the next section, this application will be compiled and deployed on the STM32F412G-DISCO board using the MicroEJ platform.

2.2.1. Create Example

- Open MicroEJ SDK.
- Open the `File > New > MicroEJ Standalone Example Project` menu.

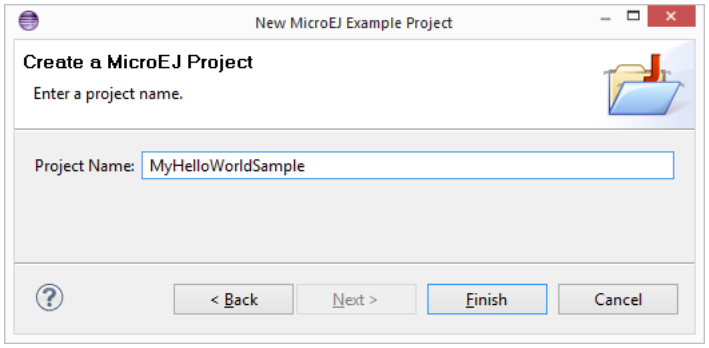
- Select the MicroEJ platform STM32F412GDISCO-MyPlatform-CM4hardfp_ARMCC5 from the combo box.
- Select the example `Samples > Getting Started > Hello World`.

Figure 2.4. MicroEJ Standalone Application Selection



- Click on Next. The next page suggests a name for the new project.

Figure 2.5. MicroEJ Standalone Application Naming



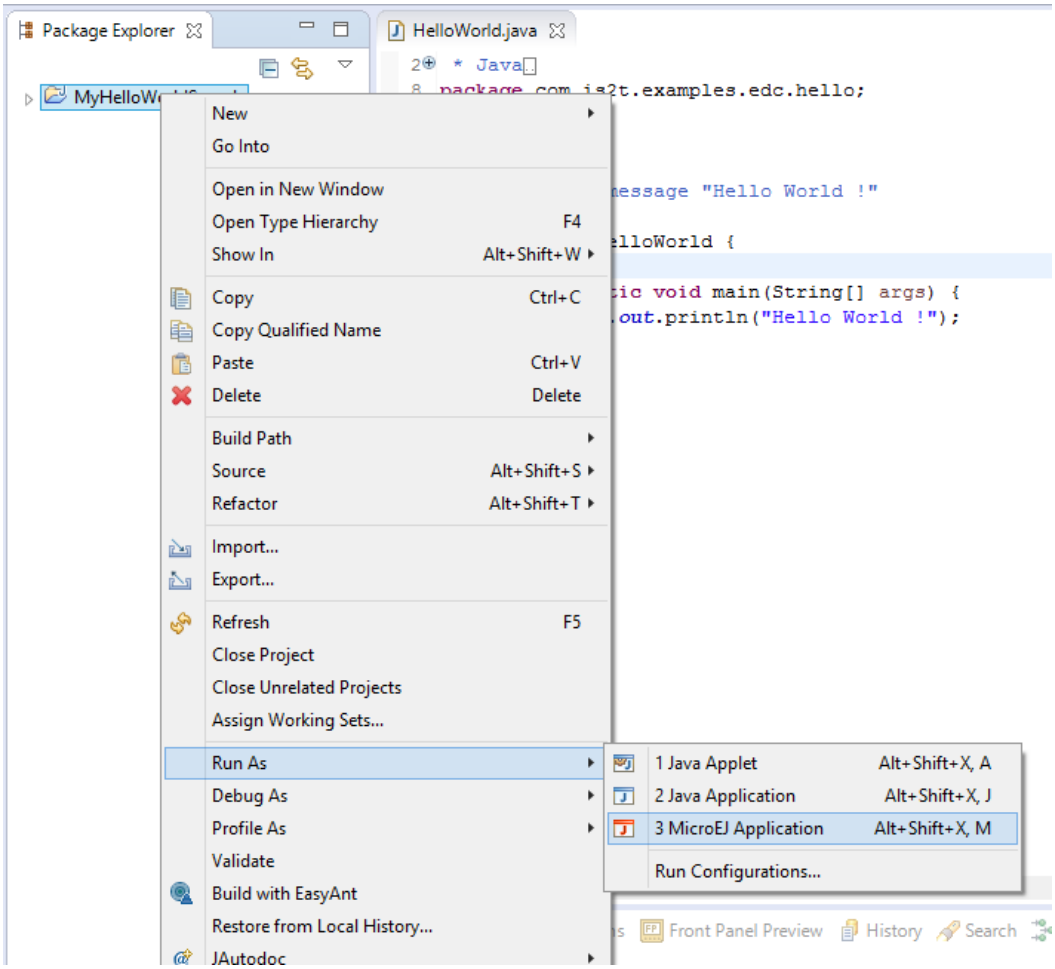
- Click on Finish. The selected example is imported into a project with the given name. The main class (the class which contains the `main()` method) is automatically opened.

2.2.2. Run Example

- Select the project in the Package Explorer tree

- Right-click on this project and select `Run As > MicroEJ Application`

Figure 2.6. MicroEJ Standalone Application Running



The application starts. It is executed on the MicroEJ simulator of the selected MicroEJ platform (STM32F412GDISCO-MyPlatform-CM4hardfp_ARMCC5). The result of the test is printed in the console:

```
Hello World !
```

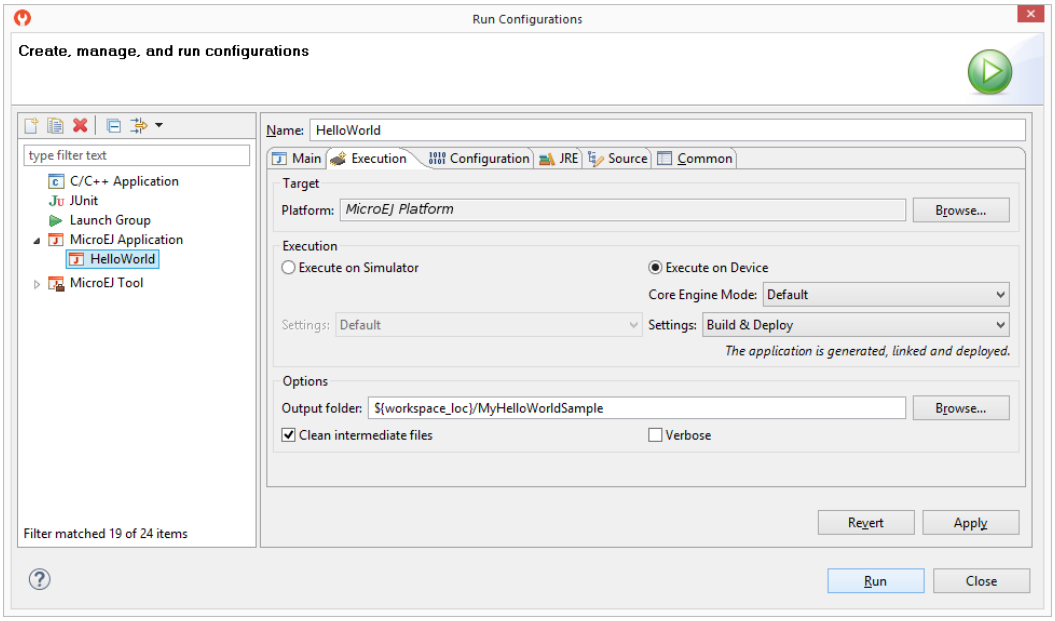
2.3. Run the Example on the STM32F412G-DISCO Board

2.3.1. Compile MicroEJ Standalone Application

- Open the run dialog (`Run > Run configurations...`).
- Select the MicroEJ Application launcher `HelloWorld`.

- Open `Execution` tab.
- Select `Execute on Device`.

Figure 2.7. Execution on Device



- Open `Configuration` tab and sub menu `Target > Deploy`. By default, an option is set to deploy the application library at a location known by the third-party IDE. If you want to deploy it elsewhere, unselect this option and enter your output path in the field below.
- Click Run: the application is compiled, and the compilation result (an ELF file) is copied into a well known location in the workspace. The Keil uVision BSP project will search for it there when it performs the final link.

2.3.2. Link and Deploy MicroEJ Standalone Application

The aim of the final step is to:

- Compile the BSP project (such as drivers).
- Link the BSP and the others libraries (MicroEJ Core Engine, C stacks, MicroEJ standalone application etc.).
- Deploy a MicroEJ standalone application on the STM32F412G-DISCO board.

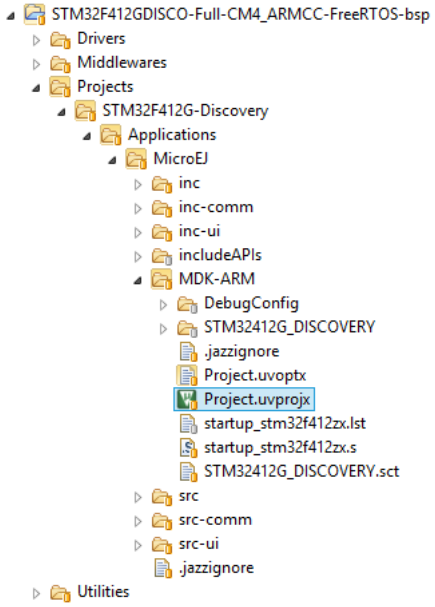
Note

This final step uses Keil uVision 5.18.0.0.

The following steps are performed within MicroEJ.

- In MicroEJ SDK, expand the project STM32F412GDISCO-MyPlatform-CM4hardfp_ARMCC5-bsp and the folder `Projects/STM32F412G-Discovery/Applications/MicroEJ/MDK-ARM`. A Keil uVision project file (`Project.uvprojx`) is available.

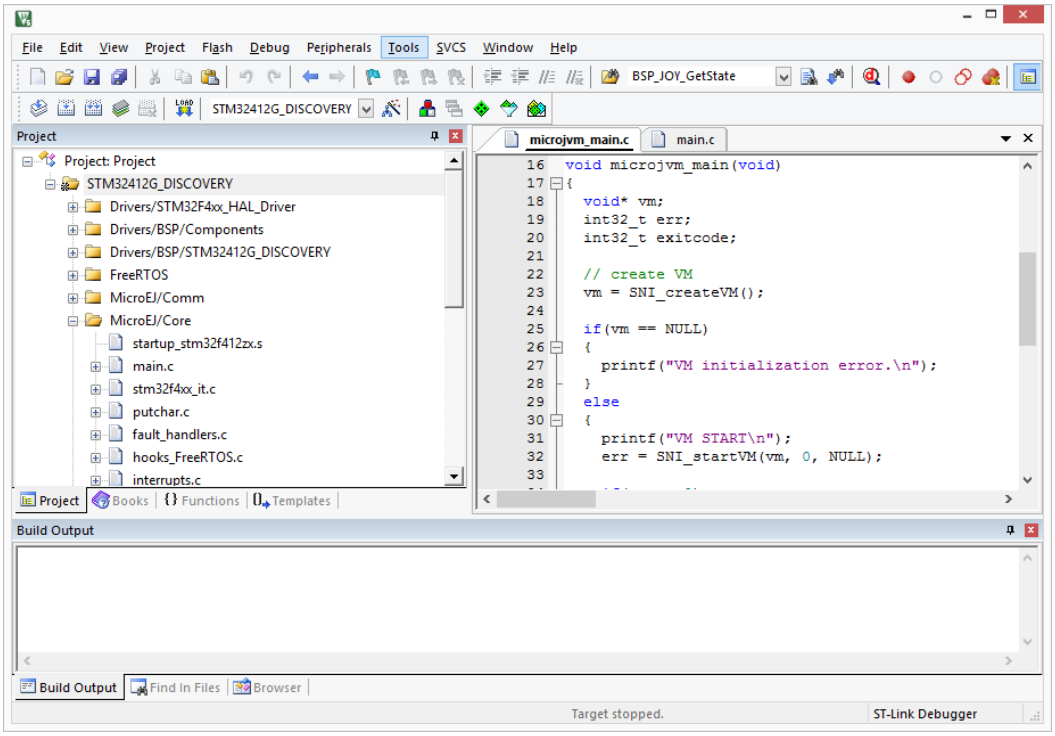
Figure 2.8. Keil uVision Project Selection



Double-click on this file to open Keil uVision.

The following steps are performed within Keil uVision.

- Figure 2.9. Keil uVision IDE



Build the Keil uVision project by clicking on the menu `Project > Build target`. The project is compiled and linked. See “Mandatory Connectors” to use the right connectors.

- Deploy the link result on the STM32F412G-DISCO board by clicking on the menu `Flash > Download`.

The application starts. The result of the execution is output on printf COM port. (See “Mandatory Connectors” to use the right connectors). Congratulations, you have deployed a MicroEJ standalone application on a MicroEJ platform.

Chapter 3. Specification

3.1. Overview

MicroEJ platform on STM32F412G-DISCO is based on board support package provided by STMicroelectronics: STM32CubeF4. It includes FreeRTOS, a graphical user interface and a serial connection. MicroEJ platform has been built against Keil µVision.

3.2. MicroEJ Platform Configuration

MicroEJ platform is based on MicroEJ architecture for ARM Cortex-M4.

Table 3.1. MCU Technical Specifications

MCU architecture	Cortex-M4 (STM32F412ZG)
MCU Clock speed	100 MHz
Internal Flash	1 MB
Internal RAM	256 KB
External Flash	16 MB (QSPI)

MicroEJ platform uses several architecture extensions. The following table illustrates the MicroEJ architecture and extensions versions.

Table 3.2. MicroEJ Configuration

Name	Version
MicroEJ architecture	6.9.0
UI	9.0.2

3.3. Platform Output stream

MicroEJ platform uses USB Virtual COM port as output print stream. The virtual COM port is available on USB ST-Link/V2 connector and is connected to the MCU USART 2.

Implementation Note

The COM port is also used as the output stream for the *printf* calls.

The COM port uses the following parameters:

- Baudrate: 115200
- Data bits bits: 8
- Parity bits: None
- Stop bits: 1
- Flow control: None

Implementation Note

On the STM32F412G-DISCO, the following parameters can be adjusted:

- Baudrate
- Parity bits
- Stop bits

3.4. RTOS Configuration

MicroEJ platform uses FreeRTOS 8.2.1. RTOS uses a heap to allocate all its objects: tasks stacks, task monitors, semaphores etc. The heap size is: 45 KB and is allocated in internal RAM. The following table illustrates the available tasks and their stack size.

Table 3.3. FreeRTOS Tasks

Task name	Size	Priority
Core Engine	12 KB	11
Touch	512 bytes	12
MCU Charge Calculation	512 bytes	15
Framerate Calculation	512 bytes	3

3.5. Memories

MicroEJ Plaform uses several internal and external memories. The following table illustrates the MCU and board memory layouts and sizes fixed by the MicroEJ platform.

Table 3.4. Internal RAM (256 KB)

Section Name	Size
Display buffers	14400 bytes
MicroUI working buffer	104384 bytes
MicroEJ standalone application stack blocks	512 * *n* bytes [a]
MicroEJ platform internal heap	*n* bytes [b]
Any RW	*n* bytes [c]
MicroEJ standalone application heaps	n bytes [d]

[a] *n* is the number of stack blocks defined in MicroEJ Application launcher options.

[b] *n* depends on memory configuration set in MicroEJ Application launcher options.

[c] *n* depends on MicroEJ application libraries used.

[d] Maximum size of the addition of MicroEJ heap size and MicroEJ immortal heap size. These sizes are defined in MicroEJ Application launcher options.

Table 3.5. Internal flash: AXIM interface (1 MB)

Section Name	Size
Any RO	*n* bytes [a]

[a] *n* depends on MicroEJ application, MicroEJ libraries, Board support package, RTOS, drivers, etc.

Table 3.6. External flash: QSPI (16 MB)

Section Name	Size
MicroEJ standalone application resources	*n* bytes [a]

[a] *n* is the size of all MicroEJ standalone application resources.

3.6. Graphical User Interface

MicroEJ plaform features a graphical user interface. It includes a display, a touch panel, a joystick and a runtime PNG decoder.

3.6.1. Display

The display module drives a 240 x 240 TFT display. The pixel format is 2 bits-per-pixel: black, white and two grays. The display device is clocked at 60Hz and the MicroEJ application drawings are synchronized on this display tick.

Implementation Note

The display stack implementation uses the double-buffering mode: the current MicroEJ application rendering is performed in a background buffer (called back buffer) and another buffer is used by the TFT display to refresh itself (called frame buffer). When the drawing is done, a copy of pixels data from the back buffer to the frame buffer is performed (stack `copy`). In order to avoid flickering, this copy is synchronized on display refresh tick.

Back buffer is allocated in internal RAM and frame buffer is located in LCD device. The size depends on the display size in pixels and on the number of bits-per-pixel (BPP):

`bufferSize = width * height * bpp / 8;`, where:

- `width` is the display width in pixels: 240
- `height` is the display width in pixels: 240
- `bpp` is the number of bits-per-pixel: 2

The buffers size is `14400 bytes`.

MicroUI requires a RAM buffer to store the dynamic images data. A dynamic is an image decoded at runtime (PNG image) or an image created by the MicroEJ application thanks the API `Image.create(width, height)`. This buffer is located in internal RAM and the reserved size is 104384 bytes.

Implementation Note

This buffer is called "working buffer". An image buffer size follows the same rule than the LCD buffer (see before).

3.6.2. Inputs

Touch panel: All touch panel events are sent to the MicroEJ application thanks a `Pointer` event generator.

Implementation Note

A touch *press* event is detected under interrupt. It wakes up a dedicated OS task. This task is used to communicate with the touch (I2C communication). For all next *drag* events, the touch task runs in polling mode. When a *release* is detected, the touch task goes to sleep and waiting a touch interrupt.

Joystick: All joystick events are sent to the MicroEJ application thanks a `Commands` event generator. The command are `UP, DOWN, RIGHT, LEFT`.

Implementation Note

The joystick events management are performed under interrupt.

3.7. Serial Communications

3.7.1. UART Connector

MicroEJ platform provides two serial communications (ECOM COMM) on USART1 and USART6. USART1 pins are (RTS/CTS mode is not used):

- TX: PA15; available on connector P2 40
- RX: PA10; available on connector P2 41

USART6 pins are (RTS/CTS mode is not used):

- TX: PG14; available on connector CN11 2
- RX: PG9; available on connector CN11 1

Implementation Note

This implementation uses interrupt and relies on the MicroEJ `LLCOMM_BUFFERED_CONNECTION` API. This API is FIFO oriented. It requires two distincts software buffer for reception and transmission: reception buffer uses 1024 bytes and transmission buffer uses 5 bytes. These buffers are statically allocated in internal RAM.

Chapter 4. Board Configuration

STM32F412G-DISCO provides several connectors, each connector is used by the MicroEJ Core Engine itself or by a foundation library.

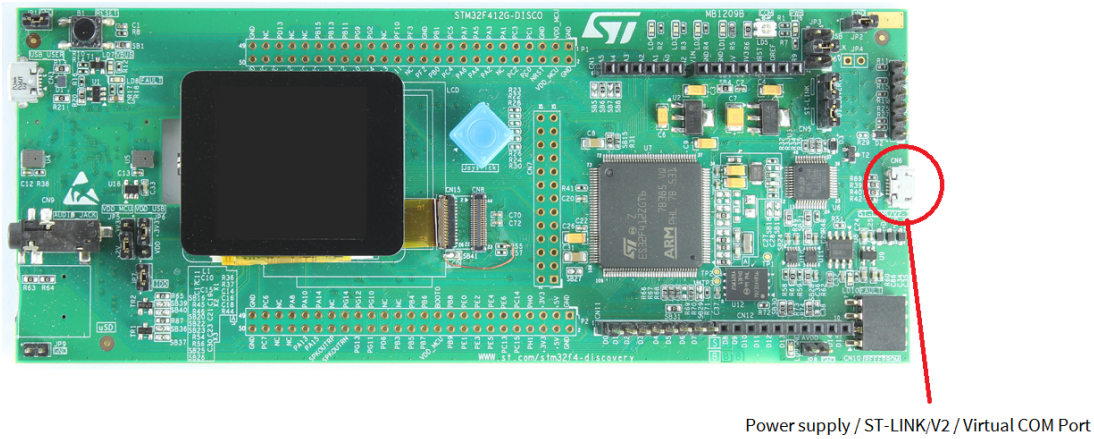
4.1. Mandatory Connectors

STM32F412G-DISCO provides a multi function USB port used as:

- Power supply connector
- Probe connector
- Virtual COM port

Plug a mini-USB cable to a computer to power on the board, be able to program an application on it and to see the MicroEJ standalone application `System.out.print` traces.

Figure 4.1. Mandatory Connectors

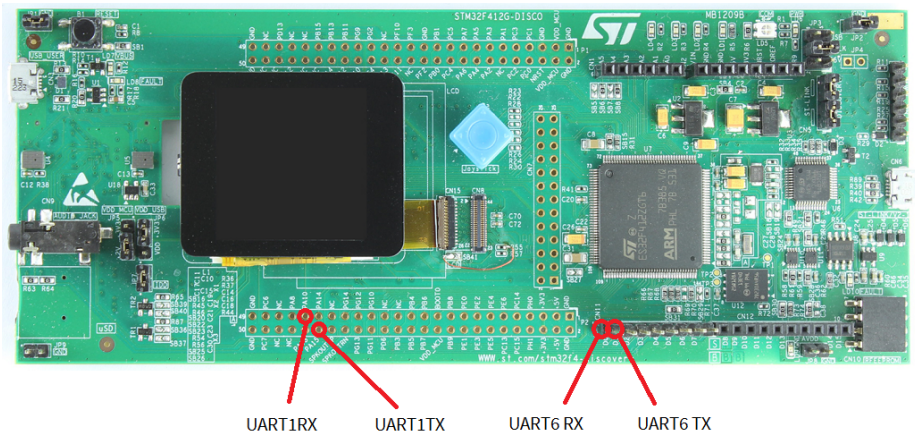


4.2. Communication Connectors

STM32F412G-DISCO provides several communication ports:

- Serial communication

Figure 4.2. Communication Connectors



Chapter 5. Keil MDK-ARM Configuration

5.1. Install Keil MDK-ARM

This section describes how to install a Keil MDK-ARM Development Kit, how to activate the MDK-ARM license required for evaluating MicroEJ SDK and how to download the Keil pack related to the target microcontroller.

A right to activate a MDK-ARM evaluation license is granted with the MicroEJ SDK evaluation version. This evaluation license is a one month time limited that allows to compile and link up to 256KB of code.

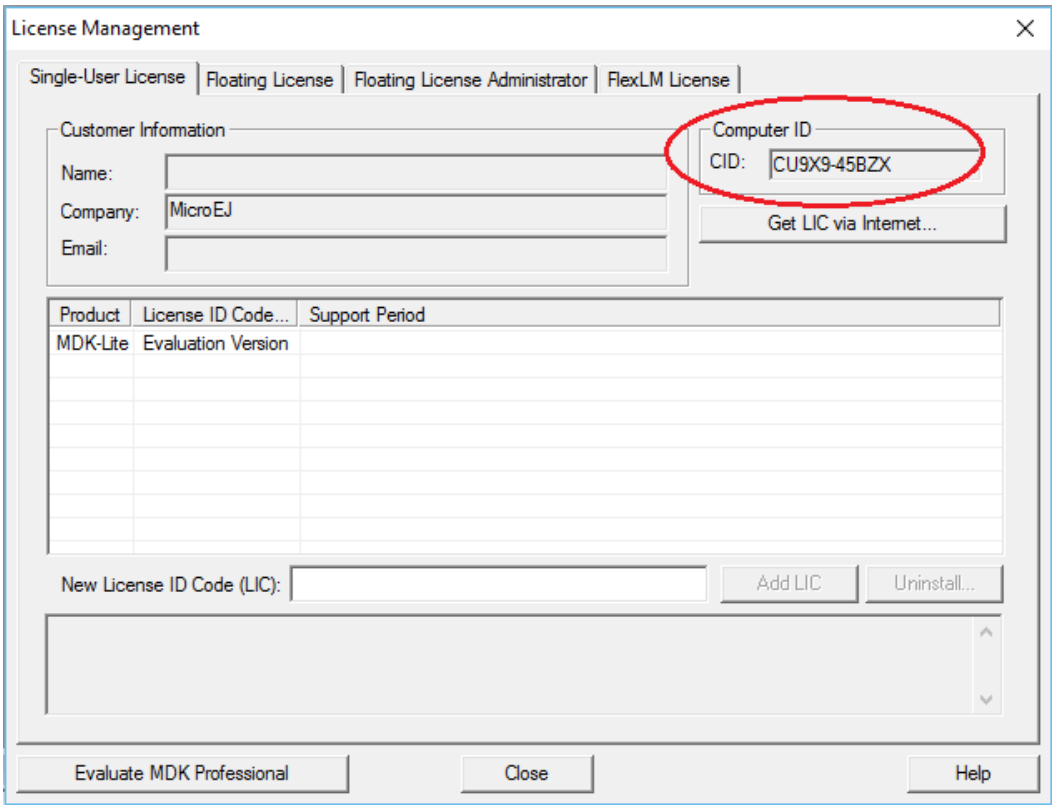
5.1.1. Download Keil MDK-ARM

- Go to `http://www.keil.com/demo/eval/arm.htm`.
- Fulfill the registration form and press `Submit` button.
- Download the executable file (e.g. `MDKxyz.exe`).
- Run executable file and follow installation steps. Install additional software and drivers if proposed. A new application named `Keil uVision5` shall have been created.

5.1.2. Activate the Evaluation License

- Start `Keil uVision5` application. If you are using Windows 7 or higher, you should run the application with administrator privileges (right-click on `Keil uVision5` shortcut `> Run as administrator`).
- Select `File > License Management...` and copy the computer ID (CID).

Figure 5.1. Keil MDK-ARM Computer ID



- Go to `https://www.keil.com/license/install.htm`.
- Fulfill the registration form with the computer ID and the following Product Serial Number (PSN):

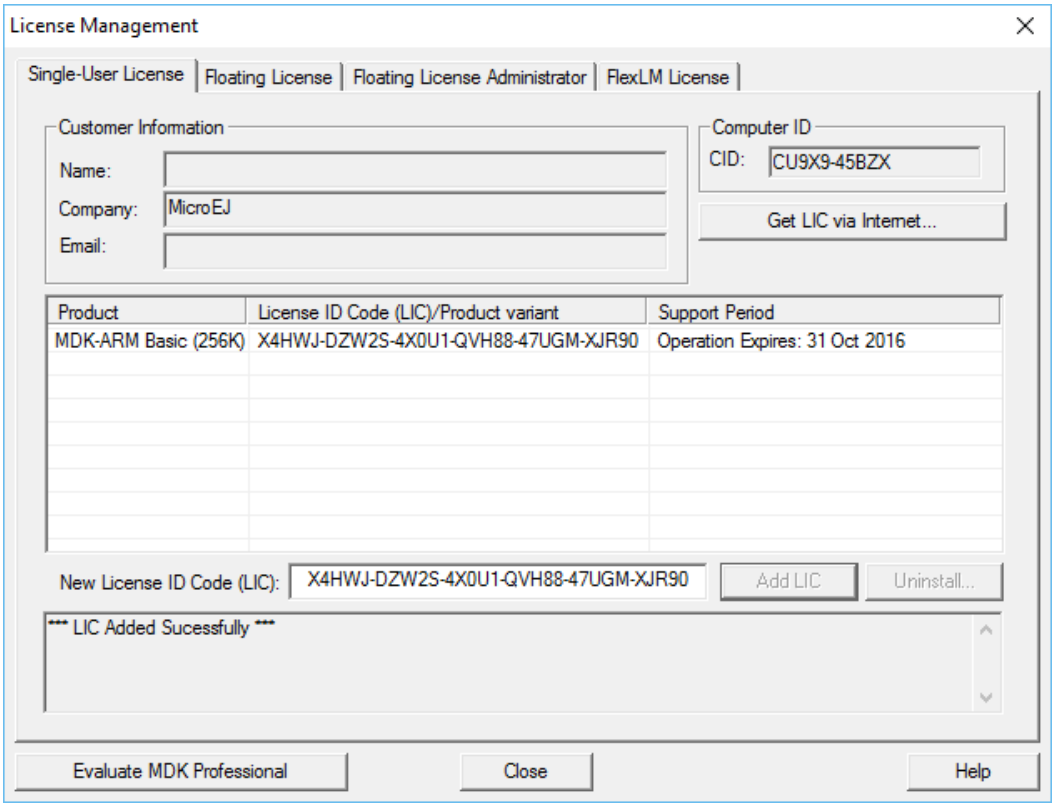
```
0YK7R-4V1PQ-R3KM9
```

Warning

This Product Serial Number is reserved to activate a Keil MDK-ARM product license solely for a usage through this MicroEJ SDK evaluation.

- Press `Submit` button.
- You should have received an email with an activation license.
- Copy/Paste the license ID code (LIC) included in the email and press on `Add LIC` button.
- License is now installed and should display the information as shown below.

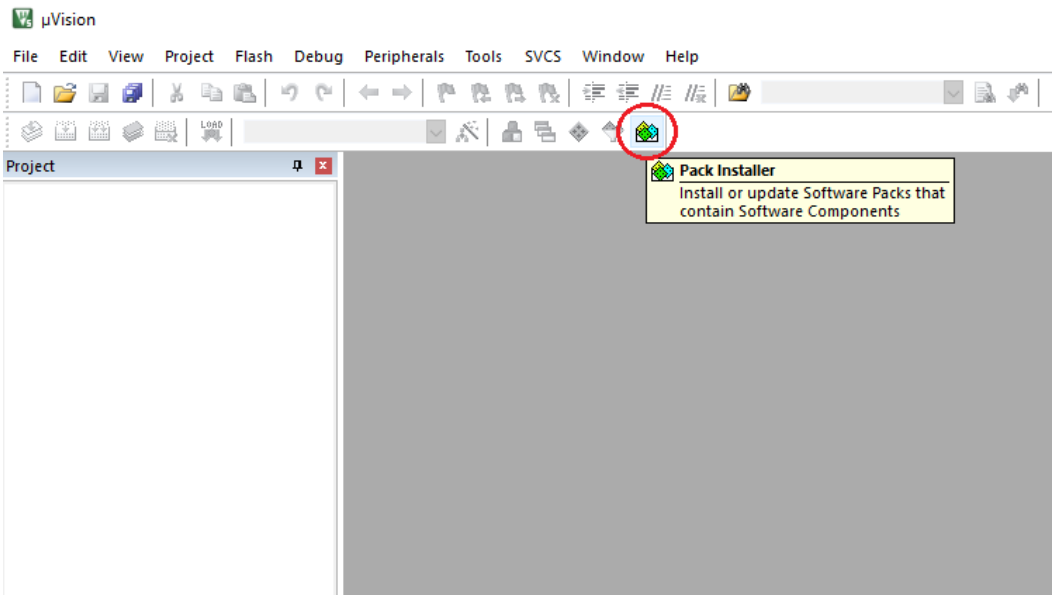
Figure 5.2. Keil MDK-ARM License Installation



5.1.3. Install Microcontroller Specific Pack

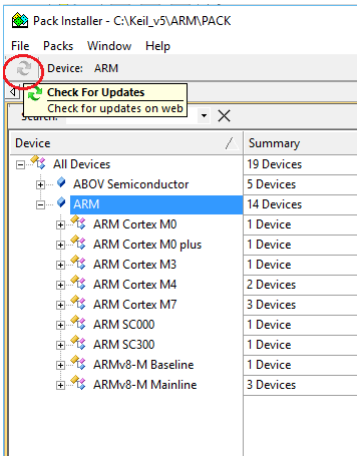
- In `Keil uVision5`, click on the pack installer icon.

Figure 5.3. Keil MDK-ARM Microcontroller Pack



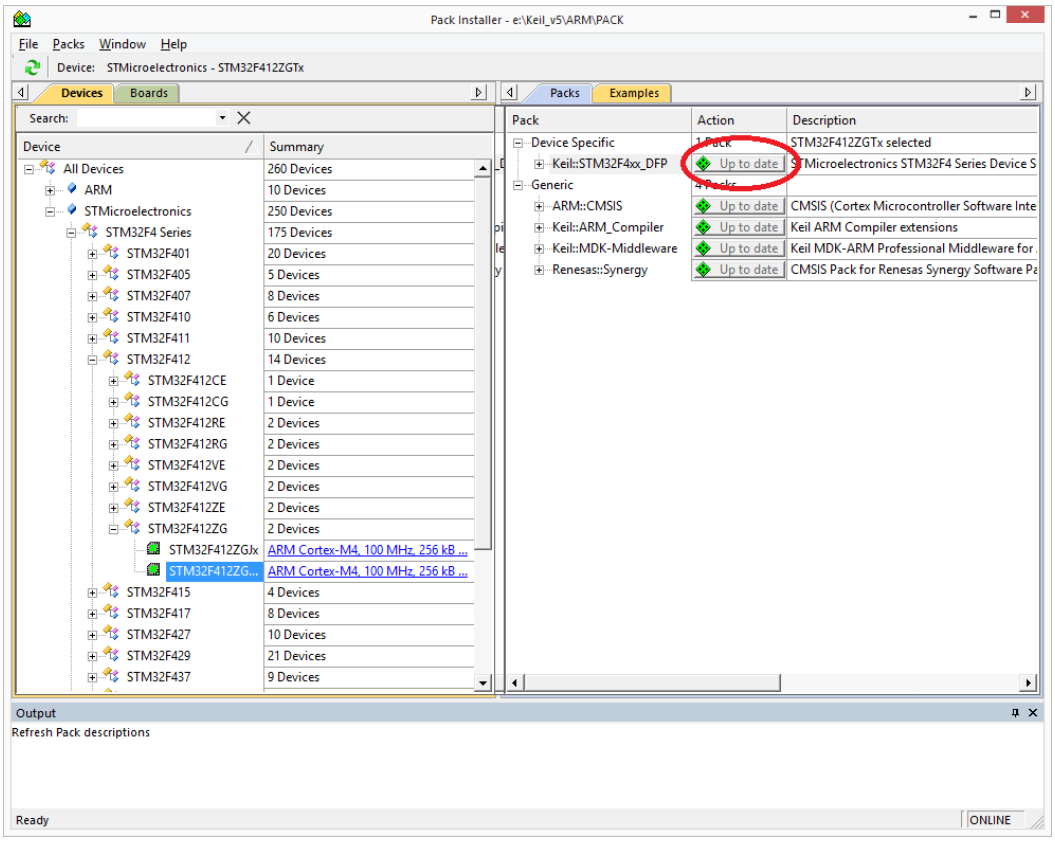
- Click on the refresh icon `Check for Updates` and wait until all available pack description are downloaded.

Figure 5.4. Microcontroller Pack Update



- In `Devices` tab, select the appropriate microcontrolleur part number `All Devices > STMicroElectronics > STM32F4 Series > STM32F412 > STM32F412ZG > STM32F412ZGTx`. In `Packs` Tab, select `Device Specific > Keil::STM32F4xxx_DFP` and click on `Install`.

Figure 5.5. Microcontroller Pack Selection



- The pack is downloaded and installed. Close the Pack Installer window.

5.2. BSP Project Structure

The MDK-ARM BSP project folder is included in a MicroEJ standard project. This project is visible from the MicroEJ workspace. This project uses the same tree than the computer file system:

- Drivers: all MCU drivers, board drivers and CMSIS drivers
- Middlewares: all 3rd-party files: FreeRTOS
- Projects: the MicroEJ platform project itself
- Utilities: miscellaneous C files required by drivers

The MDK-ARM BSP project file is `Projects/STM32F412G-Discovery/Applications/MicroEJ/MDK-ARM/Project.uvprojx`. This MDK-ARM BSP project has been written for µVision V5.18.0.0. The project follows the files structure of STMCubeF4 projects:

- Drivers/*: all MCU drivers, board drivers and CMSIS drivers
- FreeRTOS: FreeRTOS files
- MicroEJ/*: all MicroEJ platform implementation files

The MicroEJ platform implementation files are grouped by MicroEJ features:

- MicroEJ/Core: Core Engine implementation over STM32CubeF4 and FreeRTOS (always required)
- MicroEJ/Comm: ECOM COMM implementation over UART
- MicroEJ/Libs: MicroEJ platform C libraries
- MicroEJ/UI: UI implementation over STM32CubeF4

Chapter 6. Changelog

6.1. Version 1.1.0

- Fix packaging issue (documentation, architectures).
- MicroEJ 4.1 compliant

6.2. Version 1.0.3

Documentation updated about Front panel plugin.

6.3. Version 1.0.2

- LCD: Allow to choose the LCD orientation in "LLDISPLAY_configuration.h".
- Timer: Fix time returned by ""LLMJVM_getCurrentTime

6.4. Version 1.0.1

A wrong display initialization prevents renderings.

6.5. Version 1.0.0

Initial release of the platform.