

AN12134

A71CH Quick start guide for Windows

Rev. 1.0 — 09 July 2018
492510

Application note
COMPANY PUBLIC

Document information

Info	Content
Keywords	Security IC, A71CH, Quick-start guide, Windows
Abstract	This document gives information on how to get started with the A71CH Mini PCB board (OM3710A71CHPCB) for evaluation of A71CH features in a Windows computer.



Revision history

Rev	Date	Description
1.0	20180709	First release

Contact information

For more information, please visit: <http://www.nxp.com>

1. Introduction

This document gives information on how to get started with the A71CH Mini PCB board (OM3710A71CHPCB) for evaluation of A71CH features in a Windows computer. It gives an overview of the required hardware and software options. It also gives step-by-step instructions for setting up the development environment as well as full directions for running the example application in a Windows-based environment.

2. A71CH Overview

The A71CH is a ready-to-use solution, enabling ease-of-use security for IoT device makers. It is a secure element capable of securely storing and provisioning credentials, securely connecting IoT devices to public or private clouds and performing cryptographic device authentication.

The A71CH solution provides an outstanding level of security measures to protect the IC against many physical and logical attacks. It can be integrated in various host platforms and host operating systems to secure a broad range of applications. In addition, it is complemented by a comprehensive product support package, offering easy design-in with plug & play host application code, easy-to-use development kits, reference designs, documentation and IC samples for product evaluation.

3. System description

The A71CH evaluation setup presented in this document consists of an A71CH security IC connected to a Windows-based platform through a FRDM-K64F or FRDM-K82F board programmed to act as a virtual COM port and the A71CH Arduino compatible kit (OM3710A71CHARD). Fig 1 shows a basic diagram of the system architecture.

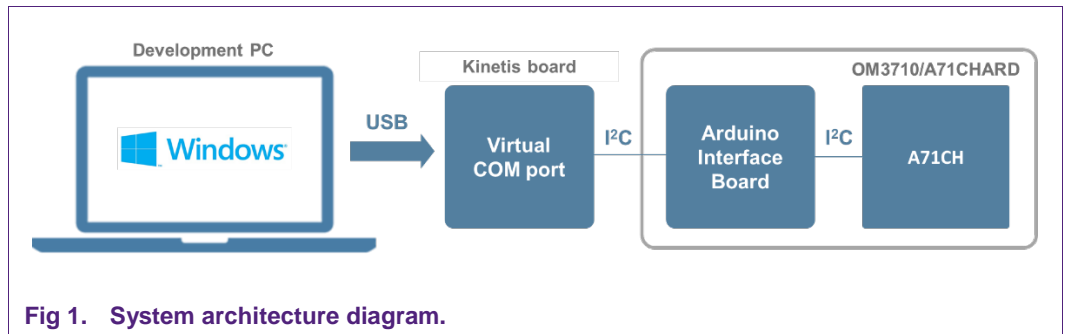


Fig 1. System architecture diagram.

The document explains the following parts:

- **Hardware overview:** It describes the FRDM-K64F and the FRDM-K82F development platforms as well as the A71CH Arduino compatible kit (OM3710/A71CHARD).
- **Hardware setup:** It describes how to connect the A71CH Arduino compatible kit with the FRDM-K64F or the FRDM-K82F boards.
- **Software setup:** It describes how to configure the development environment and how to import the required software packages

- **A71CH application examples execution:** It describes how to run the A71CH Windows-based application examples contained in the A71CH Host software package for
- **RJCT server:** It describes how to use the RJCT server to communicate with A71CH from a remote or local client process.

In addition, the document includes the following appendix:

- **USB to I²C bird:** It describes how to use the USB to I²C bird – Ascot adaptor (OM3710/B001) to interface a Windows platform as an alternative to a Kinetis board configured as a virtual COM port.
- **VCOM driver installation:** It describes how to troubleshoot VCOM driver installation issues.

4. Hardware overview

This setup uses a Windows laptop as a host MCU while the A71CH security IC acts as the protected storage module. The following material is needed:

1. A computer running Microsoft Windows 7 (32 or 64 bit) or later.
2. The A71CH Arduino compatible development kit (OM3710/A71CAHRD).
3. The FRDM-K64F or FRDM-K82F evaluation board.

4.1 A71CH Arduino compatible development kit (OM3710/A71CHARD)

The OM3710/A71CHARD is an Arduino development kit containing two items as well:

1. An A71CH Mini PCB board (OM3710/A71CHPCB)
2. An Arduino interface board, allowing the user to connect the A71CH to any host featuring an Arduino compatible header (e.g., many LPC, Kinetis and i.MX boards in the industry).

4.1.1 A71CH Mini PCB board (OM3710/A71CHPCB)

The OM3710/A71CHPCB board is a small PCB containing the A71CH solution and a set of jumpers for the I²C or SPI host interface selection (Note that only the I²C driver is available; SPI support might be added in future revisions).

Fig 2 shows an image of the MiniPCB. It features two connectors that can be used depending on which communication interface is employed. The figure shows the jumpers configuration that enables the use of the A71CH I²C interface.

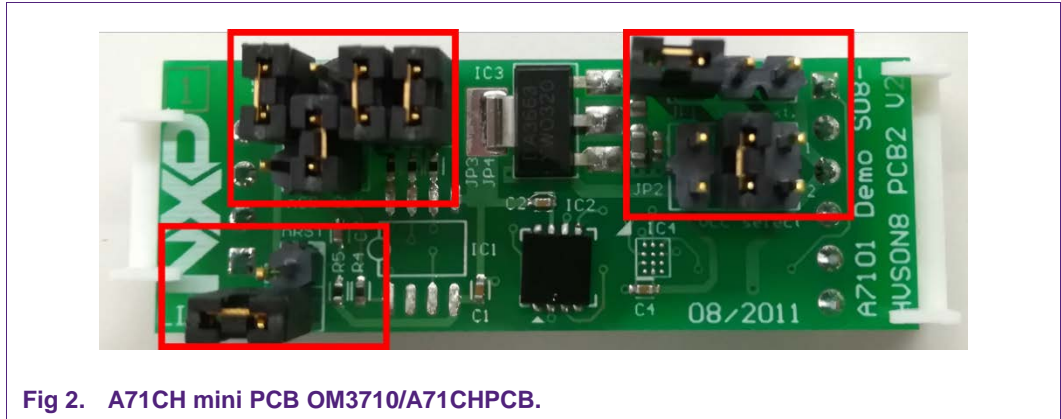


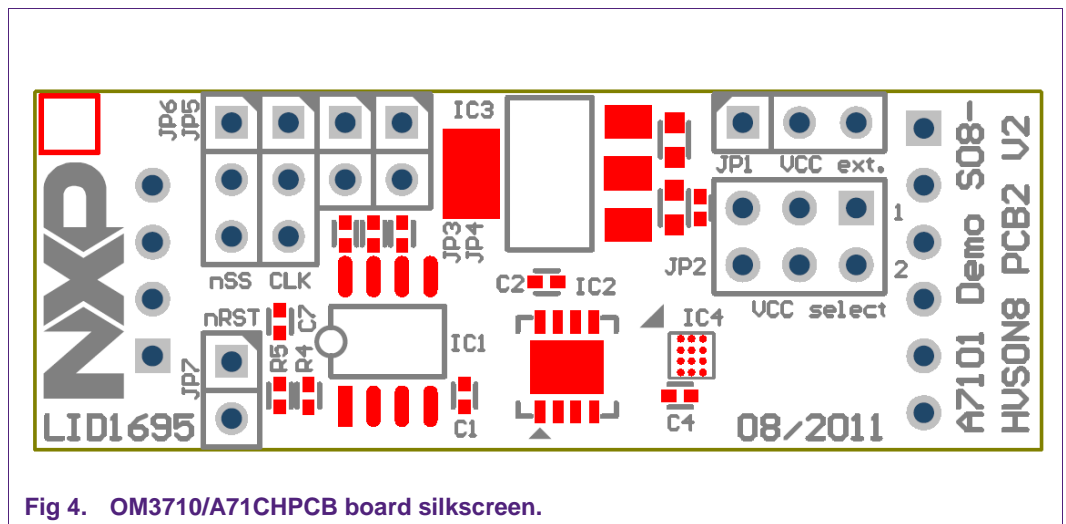
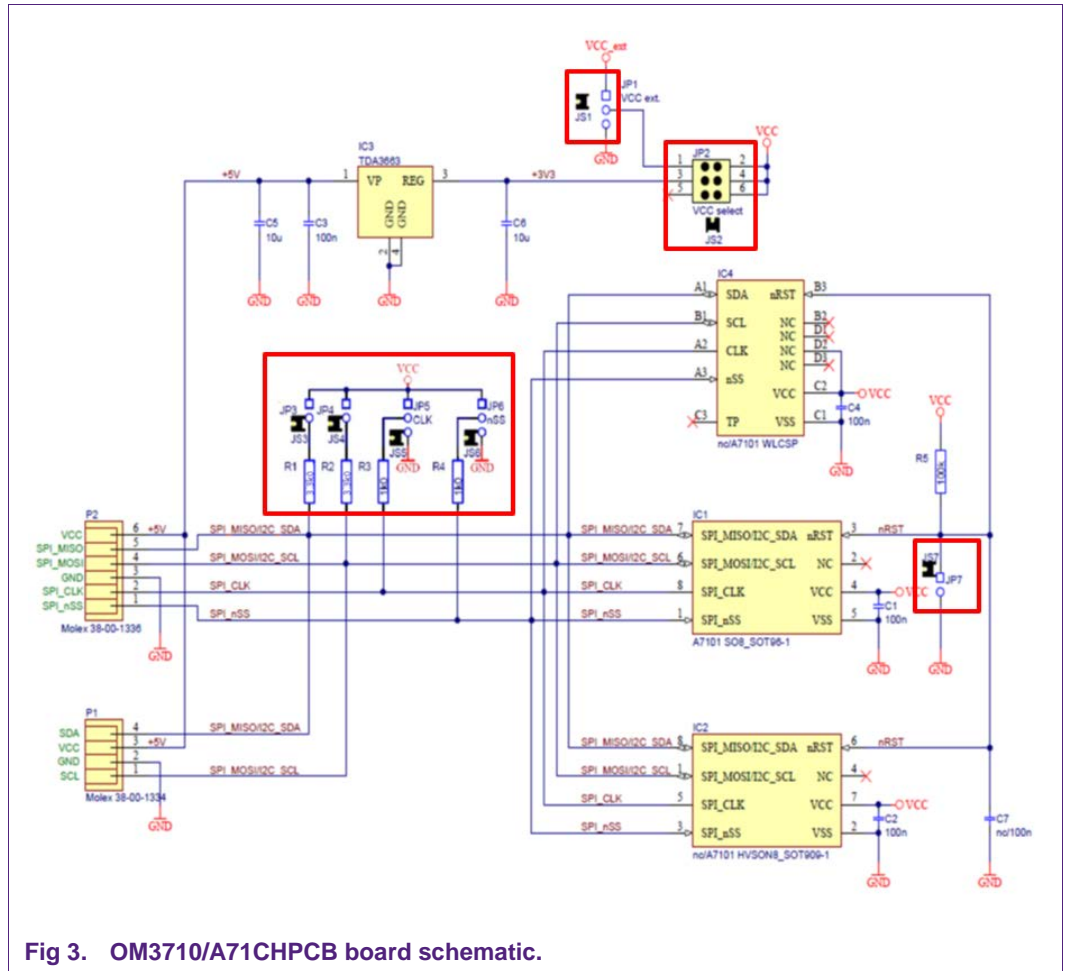
Fig 2. A71CH mini PCB OM3710/A71CHPCB.

To enable the I²C communication protocol, it is necessary to configure JP5/6 according to Table 1. JP2 connects the A71CH to the on-board 3.3V voltage regulator on the MiniPCB board. The jumpers JP3 and JP4 enable the I²C SDA/SCL pull-up resistors. JP7 can be used to connect the A71CH reset signal.

Table 1. Default OM3710/A71CHPCB Jumper settings

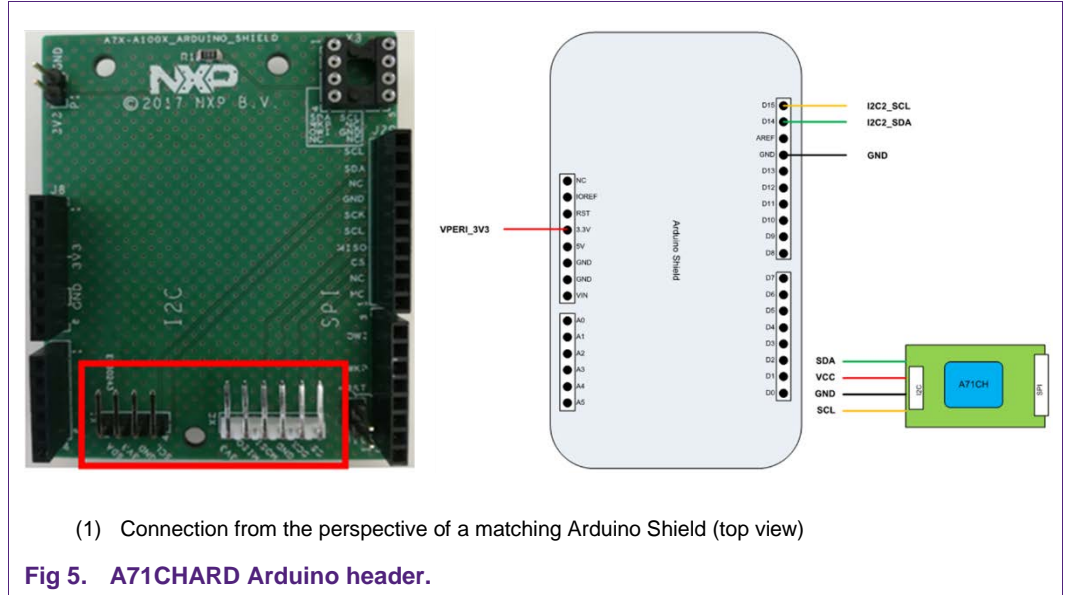
Jumper	Setting	Usage
JP1	Not set	External VCC connection
JP2	3-4	Connect A71CH to 3.3V regulator on MiniPCB
JP3	Set	Connect I ² C SDA pull-up resistor
JP4	Set	Connect I ² C SCL pull-up resistor
JP5	1-2	Use I ² C address 0x92/0x93
	2-3 (Default)	Use I ² C address 0x90/0x91
JP6	1-2	Activate I ² C interface
JP7	Not set (Default)	A71CH operates
	Set	A71CH IC reset

The board schematic and layout are shown in Fig 3 and Fig 4.



4.1.2 Arduino interface board

The Arduino header board permits the user to interface the A71CH OM3710/A71CHPCB with other platforms with dedicated Arduino headers (e.g., FRDM-K64F and FRDM-K82F Kinetis boards). Fig 5 shows the board pinout.



4.2 Freedom development platforms for Kinetis

The section details the Freedom development platforms for Kinetis supported by the A71CH support package for Windows machines.

4.2.1 FRDM-K64F

The Kinetis FRDM-K64F [FRDM_K64F] development platform is a simple, yet sophisticated design, featuring a Kinetis K64 series microcontroller, built on the ARM® Cortex®-M4 core. The FRDM-K64F can be used to evaluate the K64, K63, and K24 Kinetis K series devices. It features the MK64FN1M0VLL12 MCU, which boasts the maximum operation frequency of 120 MHz, 1 MB of flash, 256 KB RAM, a full-speed USB controller, Ethernet controller, secure digital host controller, and analog and digital peripherals.

The FRDM-K64F hardware is form-factor compatible with the Arduino R3 pin layout, providing a broad range of expansion board options. The onboard interface includes a six-axis digital accelerometer & magnetometer, RGB LED, SDHC, add-on Bluetooth module, add-on RF module, Ethernet and OpenSDAv2, the NXP open-source hardware embedded serial and debug adapter running an open-source bootloader.

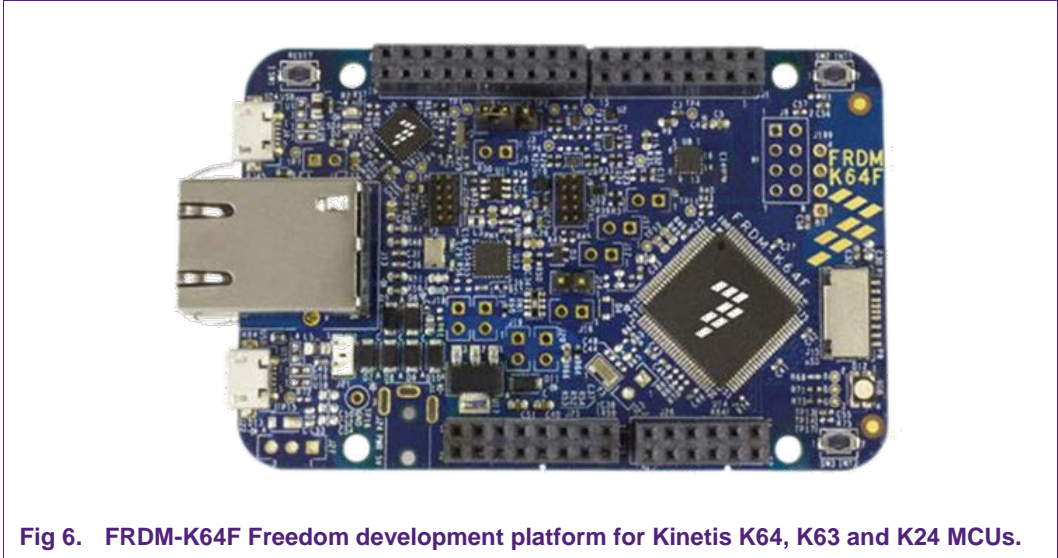


Fig 6. FRDM-K64F Freedom development platform for Kinetis K64, K63 and K24 MCUs.

4.2.2 FRDM-K82F

The Freescale Freedom K82 hardware [FRDM-K82F] is a simple yet sophisticated design featuring a Kinetis K series microcontroller built on the ARM® Cortex®-M4 core which features a floating-point unit (FPU).

The FRDM-K82F can be used to evaluate the K80, K81, and K82 Kinetis K series devices. The FRDMK82F board features the K82FN256VLL15 MCU, which boasts a maximum operation frequency of 150 MHz, 256 KB of flash, a 256 KB RAM, a full-speed USB controller with available crystal-less operation, and analog and digital peripherals.

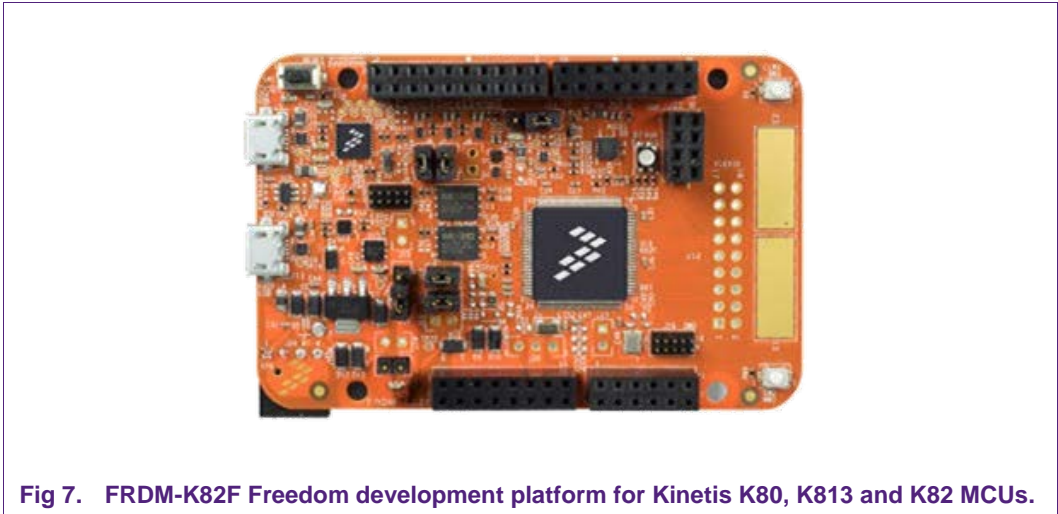
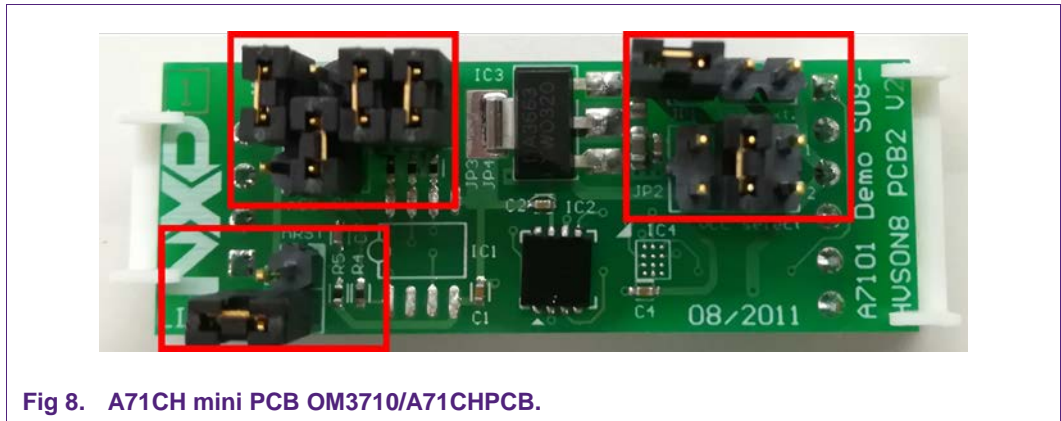


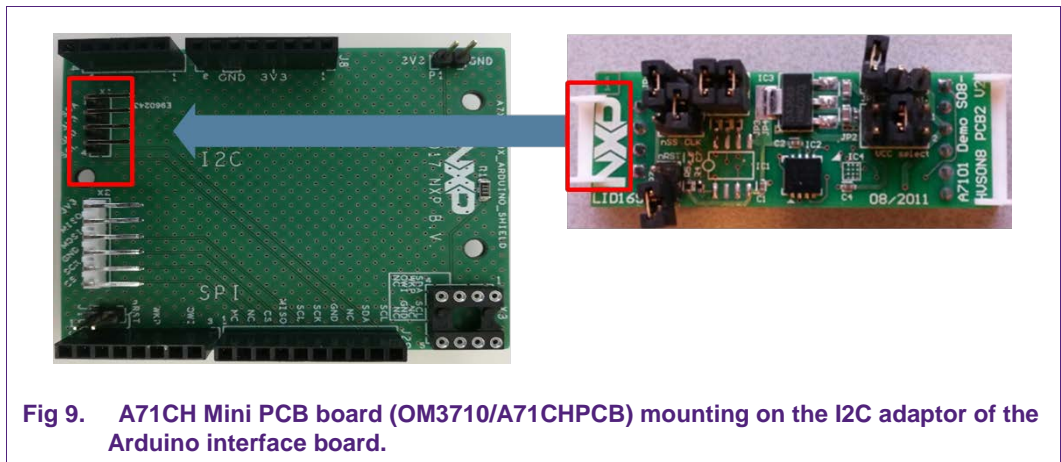
Fig 7. FRDM-K82F Freedom development platform for Kinetis K80, K813 and K82 MCUs.

5. Hardware setup

The hardware setup consists of mounting the different system parts together. Three simple steps are required. First, make sure the jumpers of the OM3710/A71CHPCB are properly configured.



Then, plug the A71CH Mini PCB board (OM3710/A71CHPCB) to the I²C adaptor of the Arduino interface board (Fig 9).



Finally, plug the A71CH into the Kinetis board using the Arduino adaptors (Fig 10). Please note the Arduino shield board comes with male connectors below.

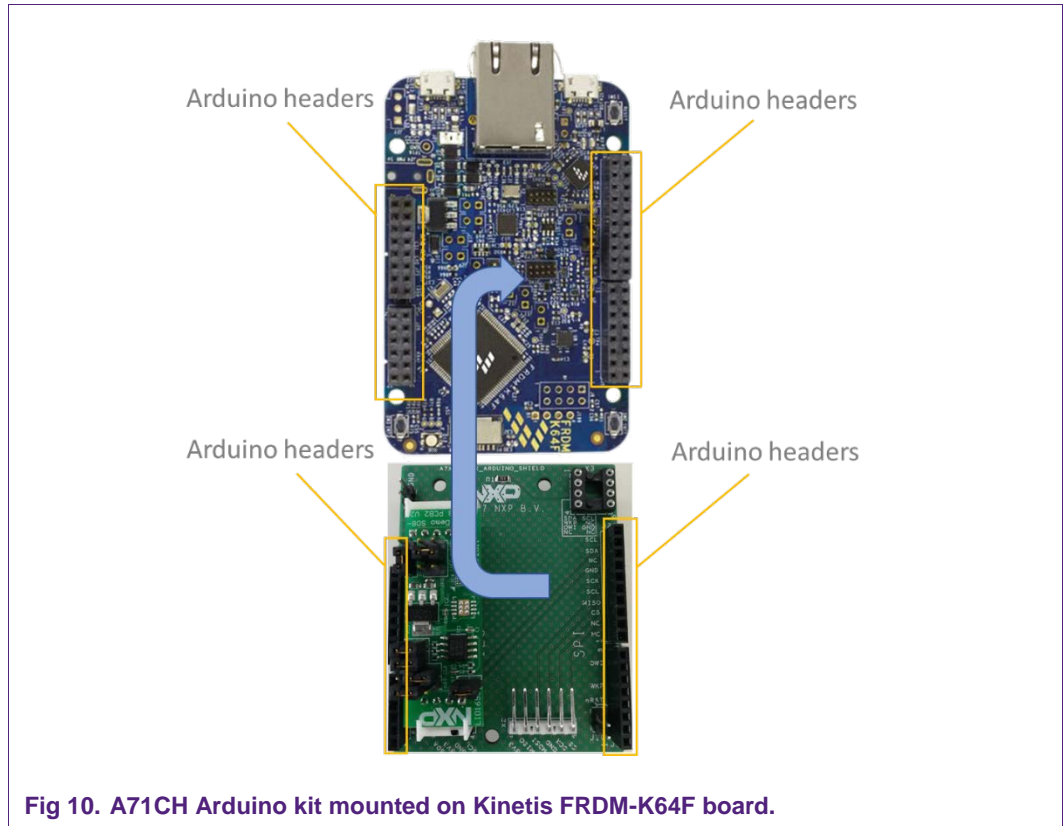


Fig 10. A71CH Arduino kit mounted on Kinetis FRDM-K64F board.

Then, A71CH security IC is connected to the Kinetis board through the Arduino interface board (Fig 11).

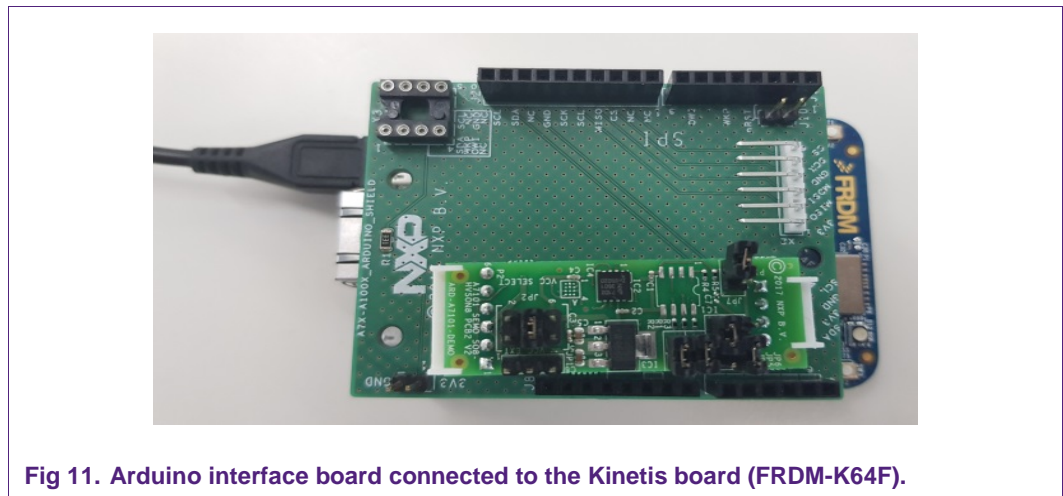


Fig 11. Arduino interface board connected to the Kinetis board (FRDM-K64F).

As can be observed, there are two USB connectors in the FRDM-K64F and FRDM-K82F boards (Fig 12). The USB connector highlighted in red corresponds to OpenSDA serial port. This port is used by the development PC to configure the Kinetis board as a virtual COM port. The USB connector highlighted in yellow corresponds to the virtual COM connector port.

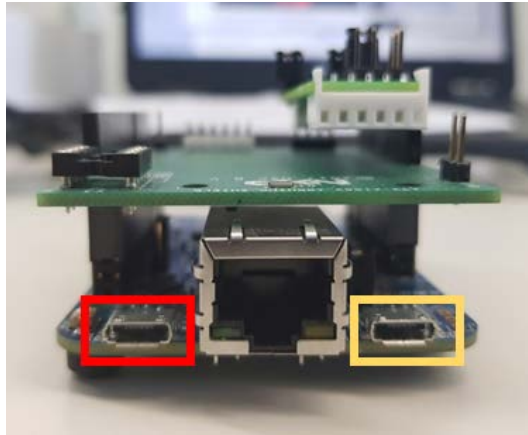


Fig 12. Red USB indicates OpenSDA serial port. Yellow USB indicates virtual COM connector port (FRDM-K64F).

6. Software setup

This section details the required steps to complete the software setup for running the A71CH application examples in a Windows-based platform.

Note: This section details the software setup for running the A71CH application examples in Windows-based platforms based on A71CH HostLib v1.4.0. If you are using a different A71CH HostLib version, the screenshots or project names indicated in this section may differ.

6.1 MCUXpresso IDE installation

MCUXpresso IDE is a fully featured software development environment for NXP's ARM-based MCUs, and includes all the tools necessary to develop high-quality embedded software applications in a timely and cost-effective fashion.

MCUXpresso IDE is based on the Eclipse IDE and includes the industry standard ARM GNU toolchain. It brings developers an easy-to-use and unlimited code size development environment for NXP MCUs based on Cortex-M cores (LPC, Kinetis and i.MX RT). The IDE combines the best of the widely popular LPCXpresso and Kinetis Design Studio IDEs, providing a common platform for all NXP Cortex-M microcontrollers.

MCUXpresso IDE is a free toolchain providing developers with no restrictions on code or debug sizes. It provides an intuitive and powerful interface with profiling, power measurement on supported boards, GNU tool integration and library, multicore capable debugger, trace functionality and more. MCUXpresso IDE debug connections support Freedom, Tower, EVK, LPCXpresso and custom development boards with industry leading open-source and commercial debug probes including LPC-Link2, P&E and SEGGER.

The fully featured debugger supports both SWD and JTAG debugging, and features direct download to on-chip and external flash memory

The installation file of MCUXpresso can be found in [MCUXPRESSO_IDE]. The setup wizard will guide the user through the process of installing MCUXpresso correctly. Since MCUXpresso requires extra drivers during the installation, check all the items on the list to allow the installation of the drivers. Make sure the checkbox for installing the NXP debug drivers is activated (Fig 13).

Note: Please, install MCUXpresso IDE version 10.2.0 or higher

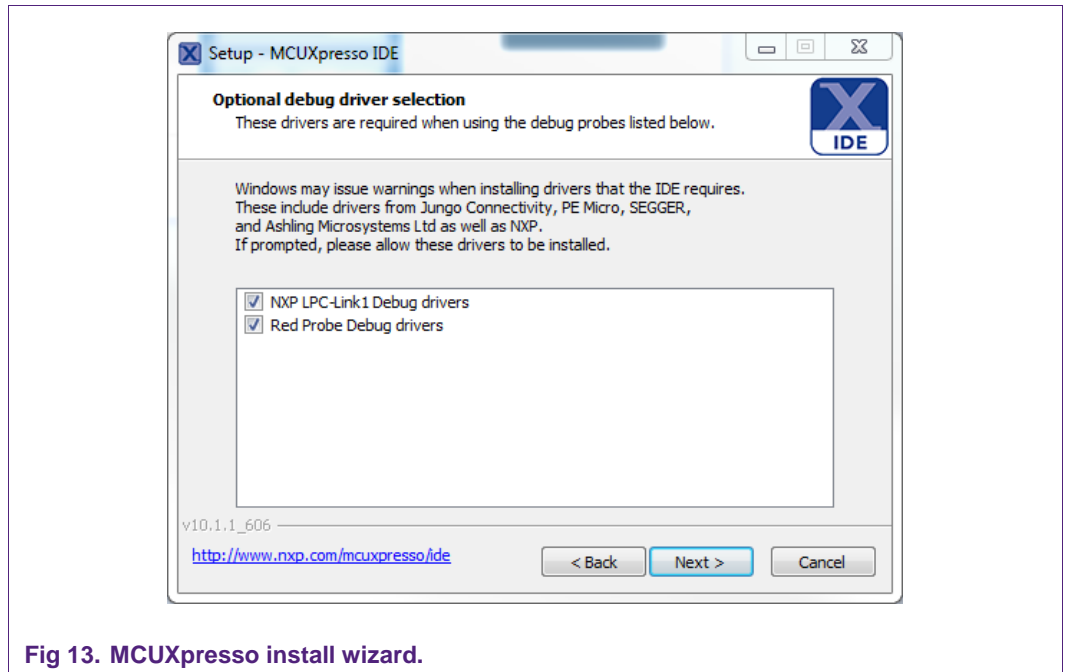


Fig 13. MCUXpresso install wizard.

6.2 OpenSDA configuration

OpenSDA is a serial and debug adapter built into the Kinetis board. It provides a bridge between the development PC and the Kinetis MCU, which can be used for debugging, flash programming and serial communication all over USB.

Note: This section explains how to install the correct OpenSDA bootloader firmware version to the Kinetis FRDM board. This needs to be done for debugging, flash programming, and serial communication over a single USB connection between a host and an embedded target processor. If this section is not followed carefully, it is possible that the examples will not be executed.

To configure OpenSDA into the Kinetis board, an OpenSDA bootloader (.bin file) should be downloaded from OpenSDA website [OPENSADA_FIRMWARE]. Scroll down the page to section 'Compatible Evaluation Boards' and search for the target Kinetis board. For

instance, Fig 14 depicts the OpenSDA bootloader version defined for the Kinetis FRDM K64: version 2.0.

Compatible Evaluation Boards

The following eval boards have been tested with the J-Link OB firmware and are known to be working. Other eval board may work as well but are not guaranteed to do so. In case of doubt, please consult SEGGER.

Evaluation Board	OpenSDA bootloader version
FRDM-K22F	2.1
FRDM-K28F	2.1
FRDM-K64F	2.0

Fig 14. OpenSDA bootloader version for the Kinetis FRDM-K64F.

Once the OpenSDA bootloader version is identified, click on the 'Downloads' section, and scroll down until 'J-Link OpenSDA – Generic Firmwares' appears and download the desired version. Fig 15 illustrates the process; in this case OpenSDA V2 Bootloader has been selected according to the compatible evaluation boards table previously mentioned.

Resources

- User Manual
- Downloads
- Release Notes
- Update Notification
- Pricing
- Support

J-Link OpenSDA - Generic Firmwares

- Supports all ARM based NXP boards which comes with on-board OpenSDA (e.g. Freedom board, Tower System, etc.)
- Implements SWD debug protocol and virtual COM port functionality
- More information

Click for downloads

	Version	Date	File size	
OpenSDA V1 Bootloader		[2017-11-16]	56 KB	DOWNLOAD
OpenSDA V2 Bootloader		[2017-11-16]	68 KB	DOWNLOAD

Fig 15. Desired firmware for the Kinetis FRDM-K64F.

To write the downloaded firmware into the Kinetis board, the bootloader mode must be enabled. For this, press the Kinetis board 'Reset' button and, while holding it down, connect a USB cable to the Kinetis board (e.g., FRDM K64, Fig 16).

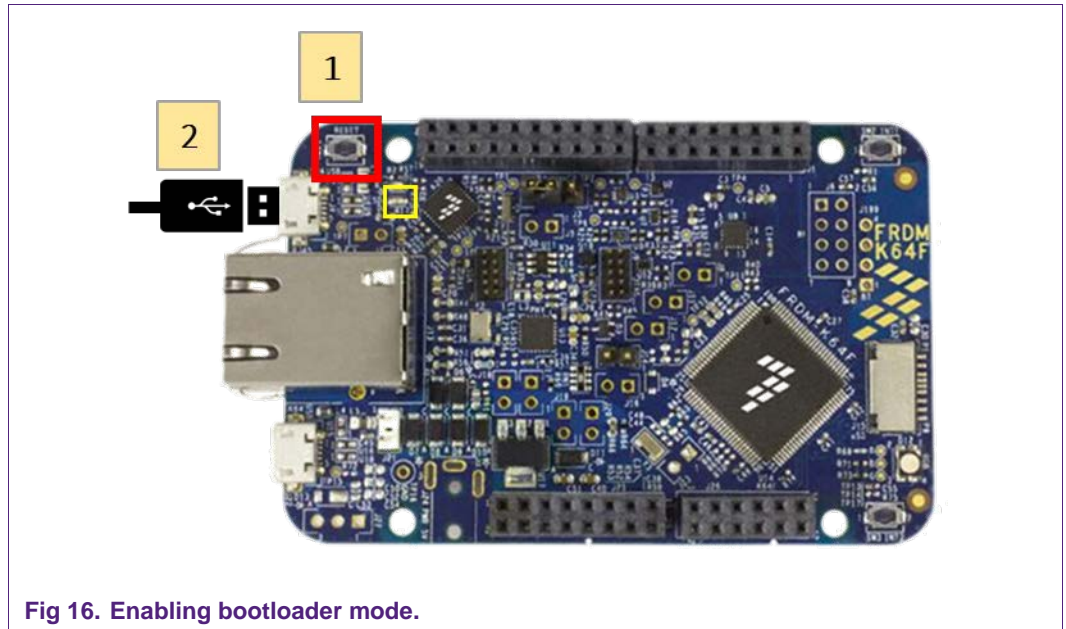


Fig 16. Enabling bootloader mode.

After connecting the USB cable to the Kinetis board, the green led located inside the yellow square will start blinking and the development PC will show a new drive called 'BOOTLOADER'.

Drag the downloaded firmware directly into the drive (Fig 17). Once the file is copied inside the 'BOOTLOADER' drive, unplug the Kinetis board and plug it again. The green led remains still, thus indicating that the OpenSDA bootloader firmware has been configured correctly.

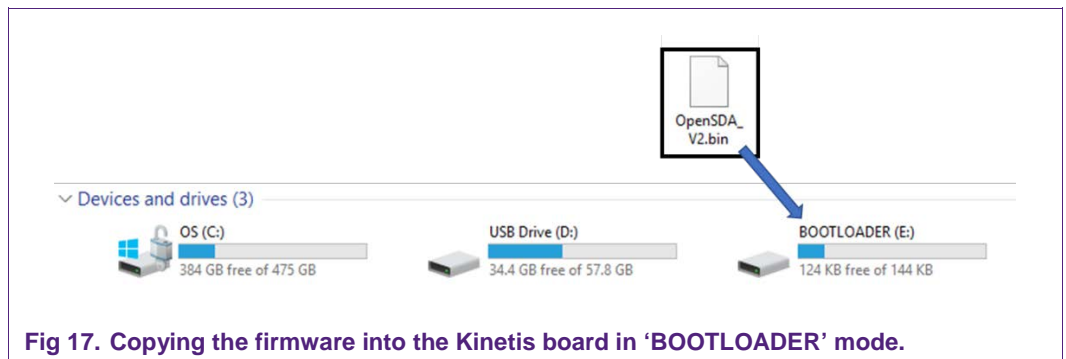


Fig 17. Copying the firmware into the Kinetis board in 'BOOTLOADER' mode.

6.3 Kinetis SDK package for A71CH

To generate and download your customized SDK for your Kinetis FRDM board, you can enter the MCUXpresso SDKBuilder website [SDKBuilder] and follow these steps

1. Select your Kinetis FRDM board and click on '*Build MCUXpresso SDK*'; in this case the selected board is the *FRDM-K64F* (Fig 18).

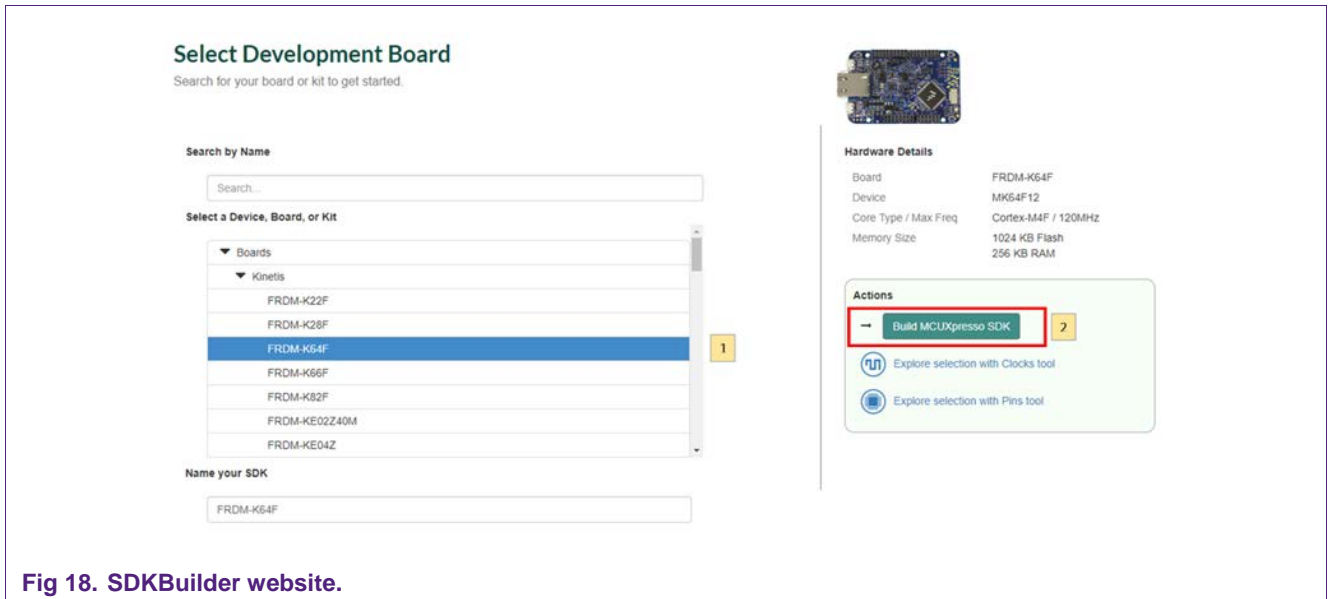


Fig 18. SDKBuilder website.

In the next screen, select the software components (Fig 19):

1. Select '*Add software component*'.
2. Select the middleware; choose options '*FatFS*', '*USB stack*', '*lwIP*', '*mbedtls*', '*Secure Element*', '*Amazon-Freertos Kernel*' and '*AWS IoT*'.
3. Click on '*Save changes*'.
4. Finally, click on '*Download SDK*'.

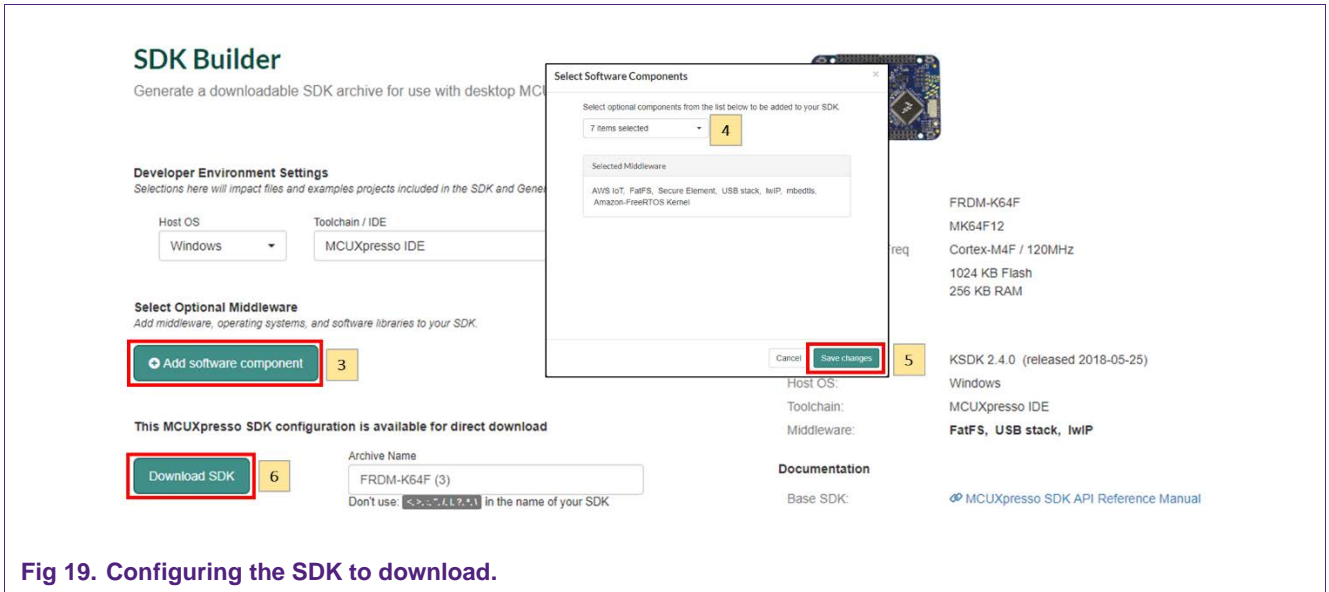


Fig 19. Configuring the SDK to download.

The downloaded SDK should be imported in MCUXpresso IDE. To import the SDK into MCUXpresso IDE, drag and drop the SDK file inside the red square ('Installed SDKs') and then click 'OK' to confirm the operation (Fig 20).

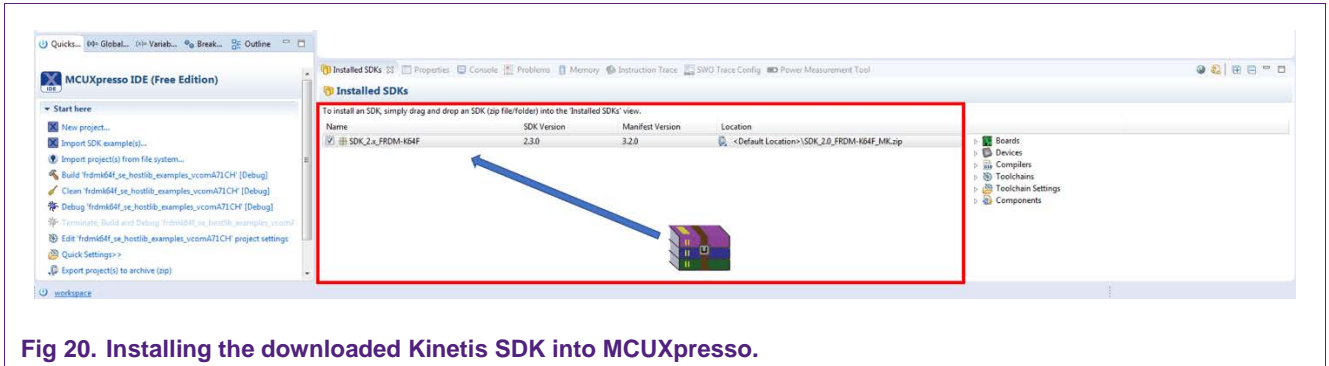


Fig 20. Installing the downloaded Kinetis SDK into MCUXpresso.

6.4 Importing a project in MCUXpresso IDE

There are two possible ways to import A71CH project examples in MCUXpresso IDE, depending on if we are using the MCUXpresso project files bundled with the A71CH Host Software package installer or if the installed SDK package already contains the A71CH middleware:

- Importing the A71CH example projects from local drive (bundled with installer)
- Importing the A71CH example projects from the installed SDK.

6.4.1 Importing the A71CH example projects from the installed SDK

The first option is to import the A71CH example projects from the installed SDK:

1. Select 'Import SDK example(s)...' to import available example projects to the workspace.

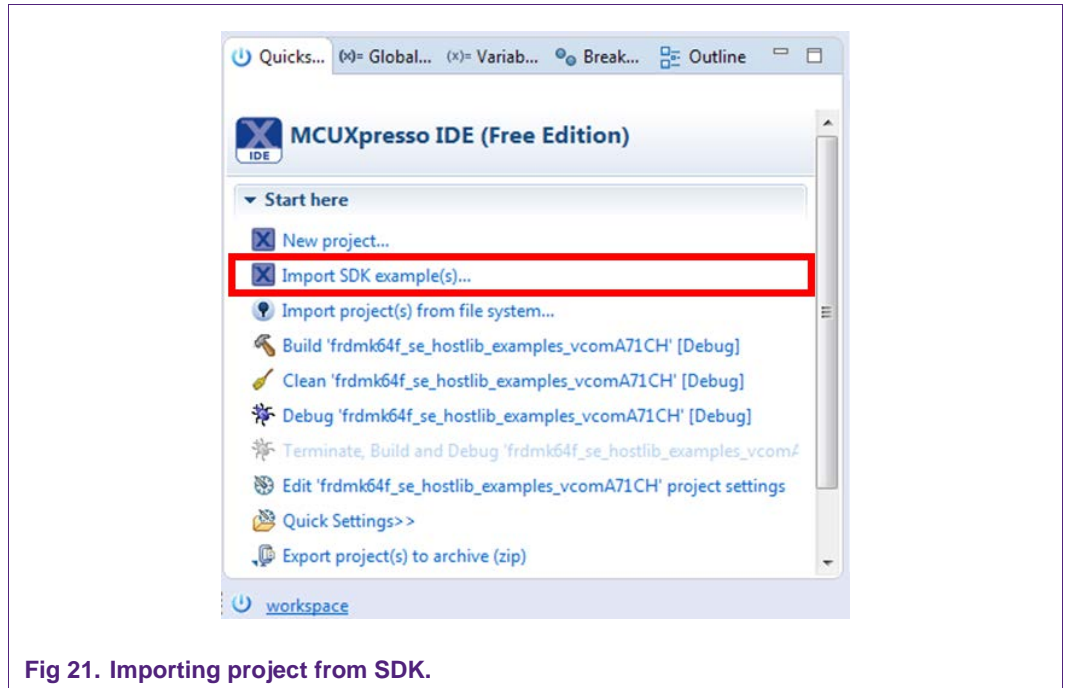


Fig 21. Importing project from SDK.

An SDK Wizard window will pop-up:

2. Select 'frdmk64f' from Available boards and then click the next button (Fig 22).

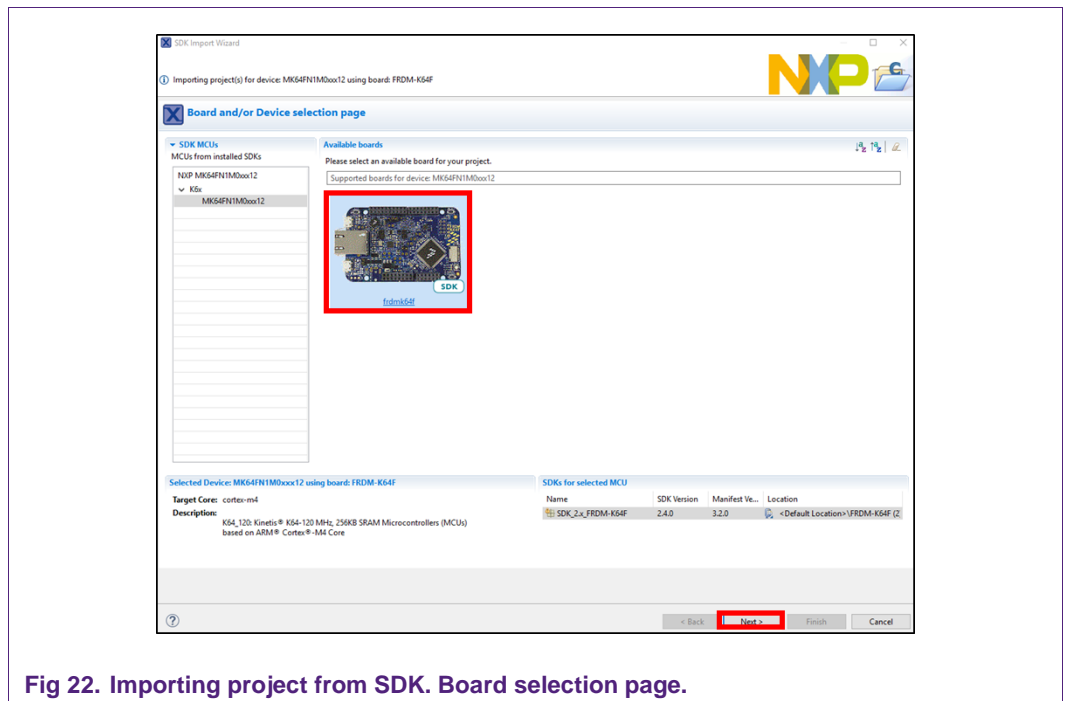


Fig 22. Importing project from SDK. Board selection page.

3. A list with different elements included in the SDK will appear; click on 'se-hostlib-examples' and 'Finish' (Fig 23).

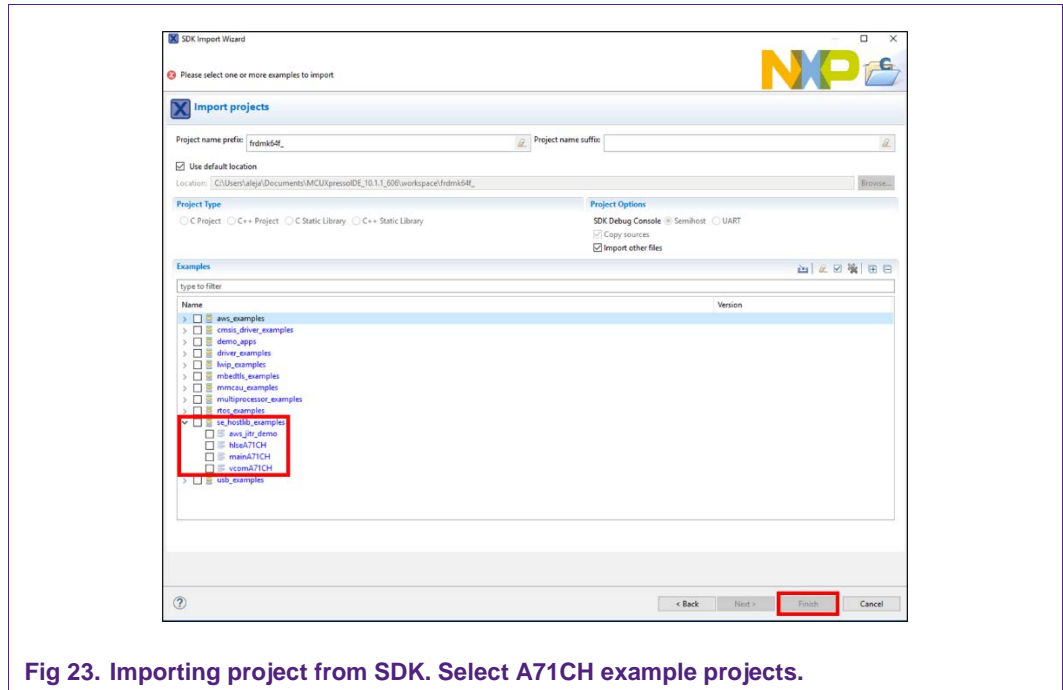


Fig 23. Importing project from SDK. Select A71CH example projects.

The imported examples will appear in the workspace window

6.4.2 Importing A71CH example projects from local drive (bundled with installer)

The second option is to use the project files bundled with the A71CH Host Software package installer. The A71CH Host Software Package can be downloaded from [A71CH_HOST_SW]. For instance, these are in 'A71CH_v<libversion>/frdmk64f_projects' in the case of the FRDM K64F board. The content of this folder is illustrated in Fig 24. As can be seen, there are three example projects:

- **A71CH Host API usage project:** demonstrates the usage of various functionalities of the A71CH in combination with mbedTLS cryptographic library.
- **VCOM project:** allows the Kinetis board to be used as a bridge between the PC and the A71CH and enables the execution of the A71CH Configure tool and other utilities from the development PC. It will be used in this document.
- **AWS JITR demo project.**

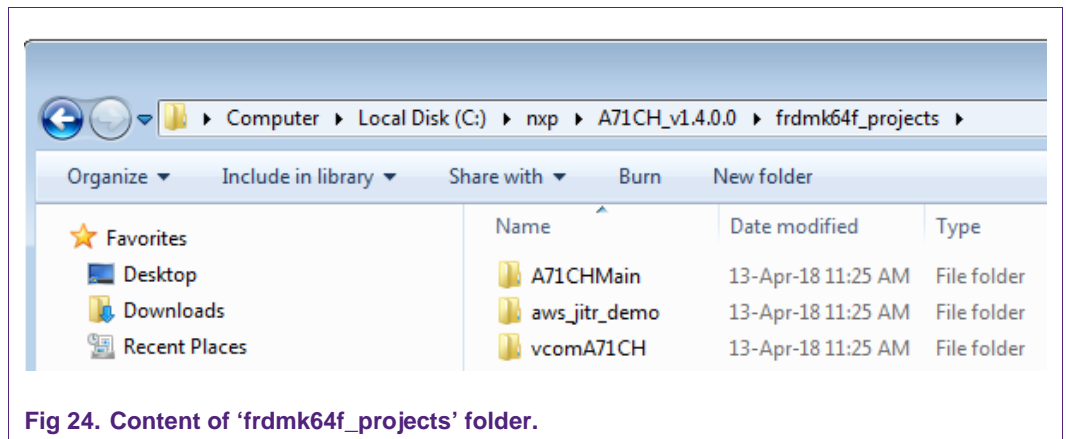


Fig 24. Content of 'frdmk64f_projects' folder.

To import a project from file system, click on 'Import project(s) from file system...' in the 'Quick start Panel' located in the bottom left (Fig 25).



Fig 25. Import project from file system.

After clicking the import option, a new pop-up will open. In the 'Project directory (unpacked)' field, browse and point to the correct project directory (Fig 26). Then, click on 'Next'.

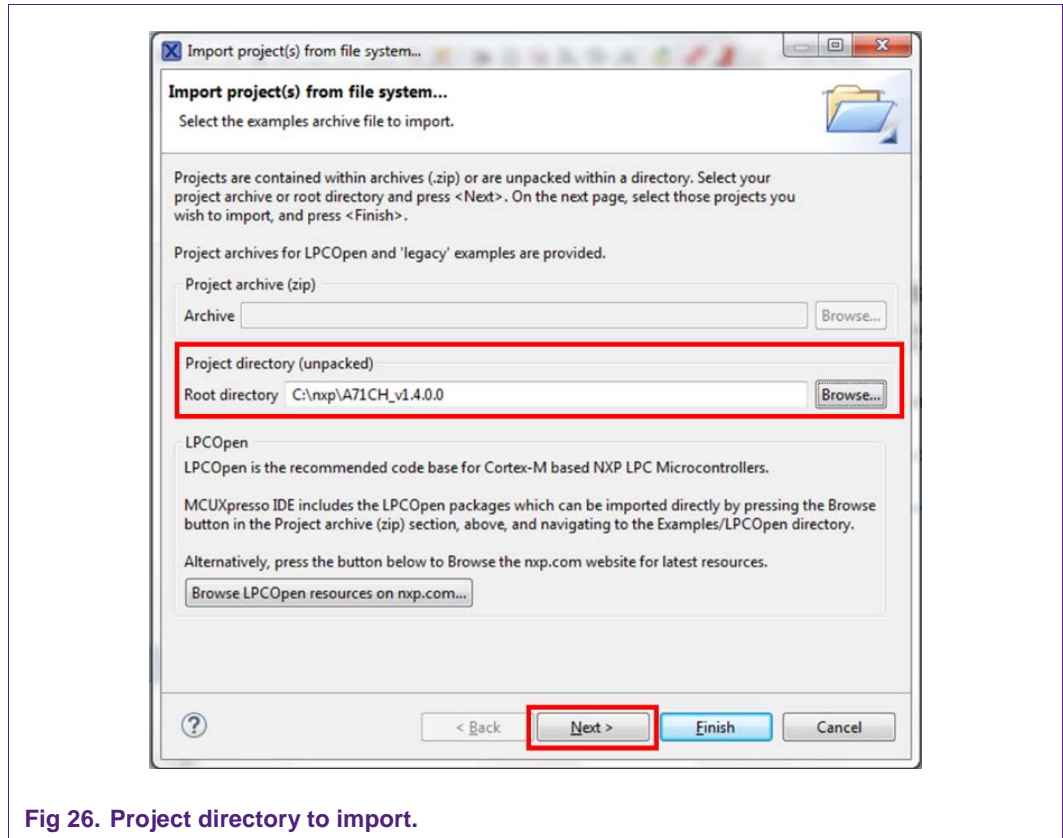


Fig 26. Project directory to import.

Finally, select all the available A71CH example project and then click on 'Finish'.

Note: It is of most importance to uncheck the "Copy projects into workspace", as shown in Fig 27.

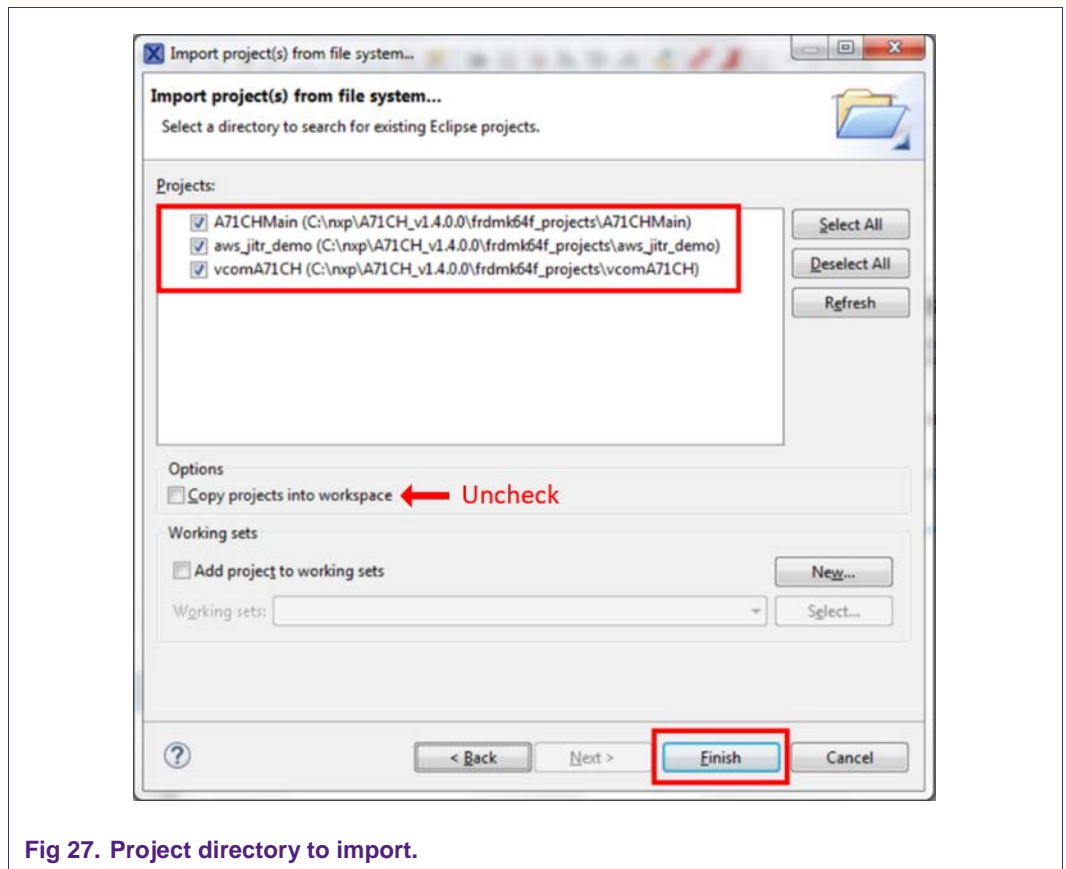


Fig 27. Project directory to import.

6.5 Microsoft Visual Studio IDE installation

Microsoft Visual Studio is an integrated development environment (IDE) from Microsoft. It is used to develop computer programs, as well as web sites, web apps, web services and mobile apps. It is only needed to build and run the Host API examples. The A71CH configuration tool is delivered as well as precompiled binary in the host library package.

- Start the built application from a Windows terminal, passing the virtual COM port address as input argument.

7.1 Set the Kinetis board as virtual COM port

Before running the A71CH example applications from a Windows platform, the Kinetis must be programmed as a virtual COM port. If the SDK has already been installed, and the project examples have been imported as explained in section 6.4, open MCUXpresso IDE and take the following steps (Fig 29):

1. Select 'frdmk64f_se_hostlib_examples_vcomA71CH' project.
2. Click on Debug 'frdmk64f_se_hostlib_examples_vcomA71CH [Debug]' (Note that the name can be slightly different in future versions of the Host software package).
3. Select J-Link OpenSDA probe and click on 'OK'. Make sure the OpenSDA serial port is connected to the Windows platform (Fig 12, highlighted in red)
4. If a 'Terms of use' pop-up appears, check 'Do not show this message again for today' box and click 'Accept'.

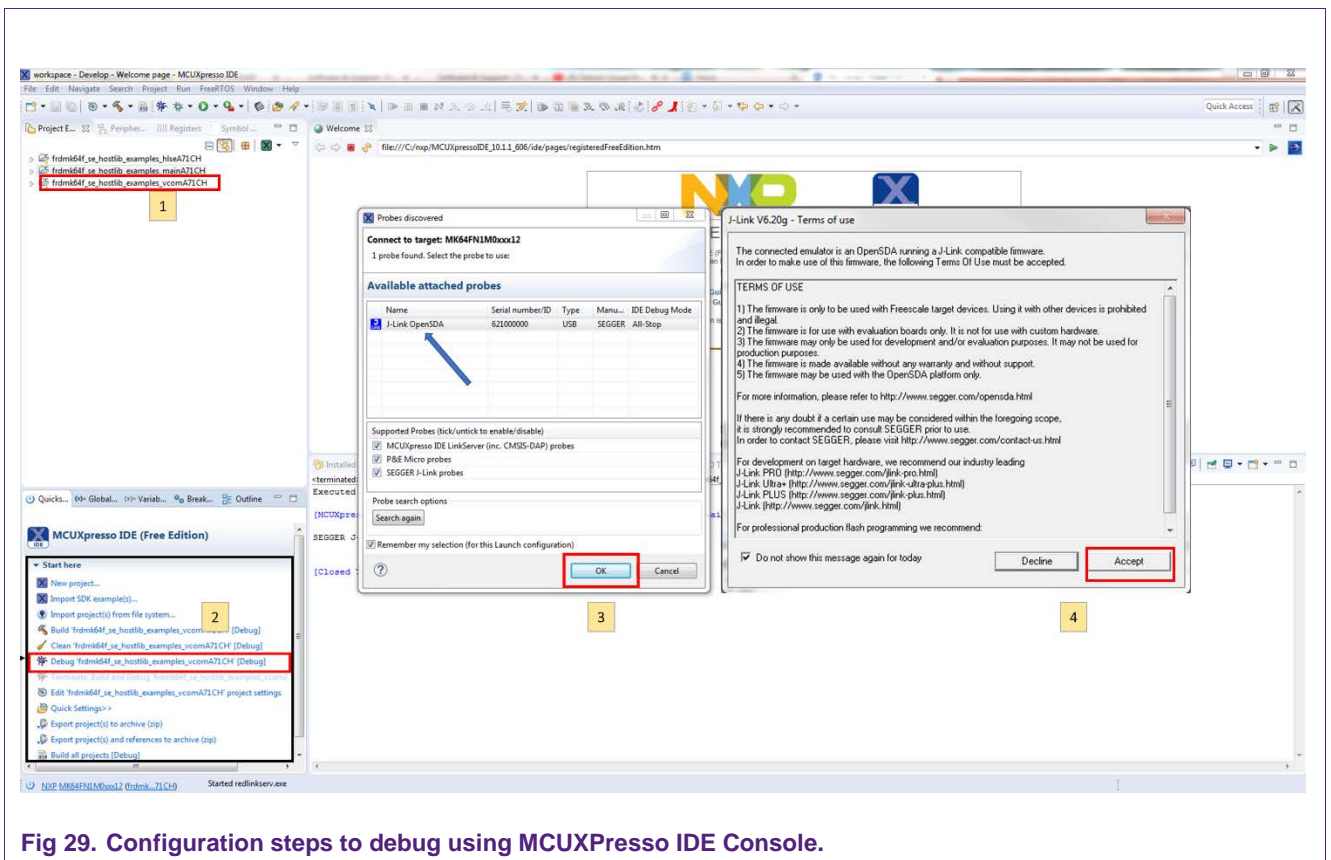


Fig 29. Configuration steps to debug using MCUXpresso IDE Console.

After that, the project will start to compile and execute automatically.

7.1.1 Connect the Kinetis board to the Windows platform over USB

Once the Kinetis board has been configured as a virtual COM port, connect it to the Windows-based platform through the USB port highlighted in yellow in Fig 12. If the Kinetis board is not recognized, check Appendix B for drivers troubleshooting.

7.2 Running A71CH Configure tool

In order to run the A71CH Configure tool, the Kinetis board is programmed to behave as a USB to I2C adapter and the A71CH Configure tool application is executed from a development PC running a Windows platform. Fig 30 illustrates the system architecture of this scenario.

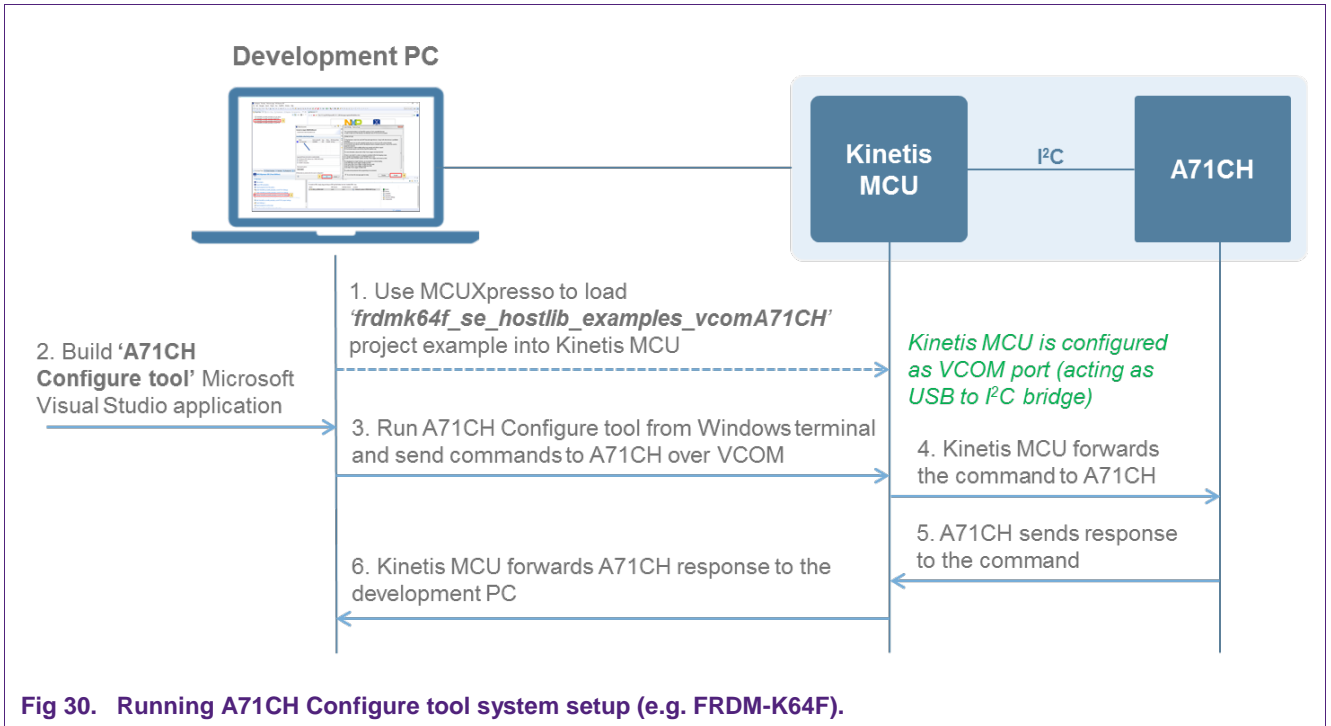


Fig 31 indicates the path to the A71CH Configure tool files (depending on the Microsoft Visual Studio installed version). As can be observed, there are '_vcom_' projects and '_socket_' projects. The first ones are meant to output the APDU commands through a virtual COM port, while the '_socket_' projects are meant to output the APDU commands through a TCP/IP socket (these will be further presented).

Open the '_vcom_' project by double-clicking '_openssl_a71ch_vcom_x86.sln'. Microsoft Visual Studio will be automatically opened.

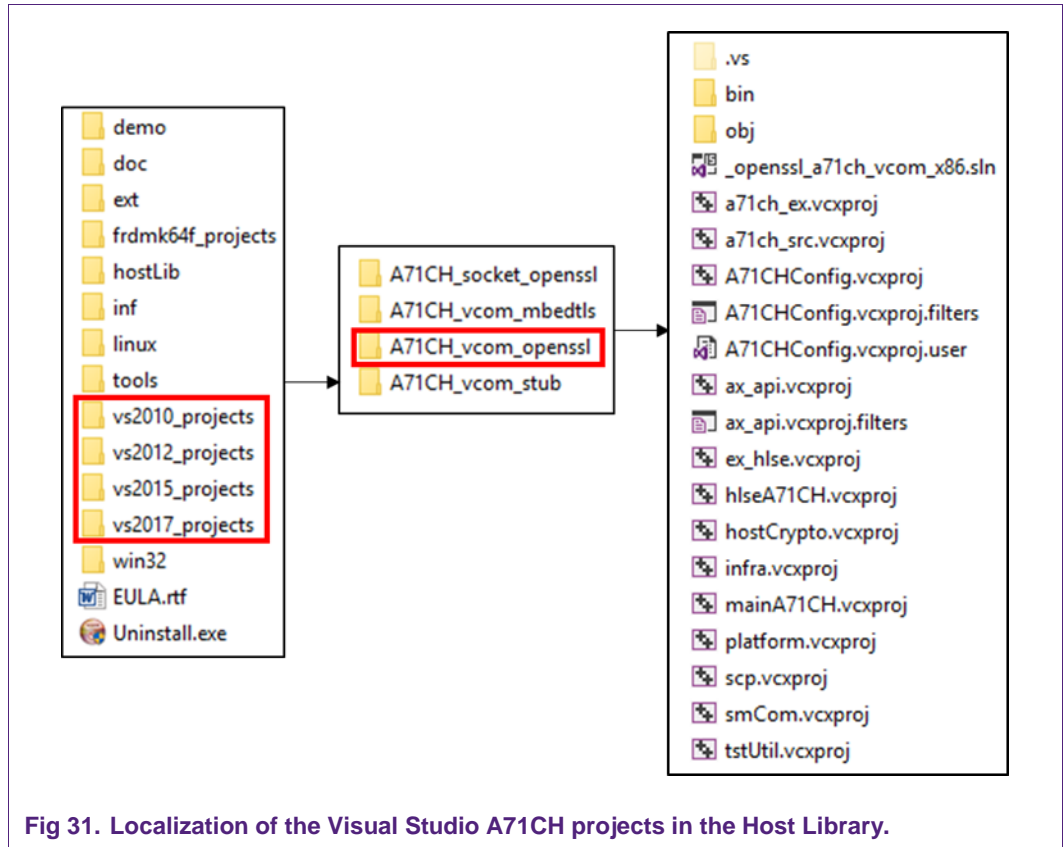


Fig 31. Localization of the Visual Studio A71CH projects in the Host Library.

To configure the virtual COM address into the A71CHConfig project, do the following:

1. Right-click 'A71CHConfig' in the 'Solution Explorer' tab and open the 'Properties' window.
2. Then, in 'Configuration Properties – Debugging' tab make sure the virtual COM address is set in the 'Command Arguments' field.
3. Once the VCOM port has been defined, invoke menu *Build - Build Solution* to create the executable.

Fig 32 illustrates the above-mentioned steps to build the project in Microsoft Visual Studio. The 'Build Solution' option has been highlighted in red, while the 'Properties' and 'Command Arguments' field have been highlighted in green. In addition, each step has been numbered in the figure.

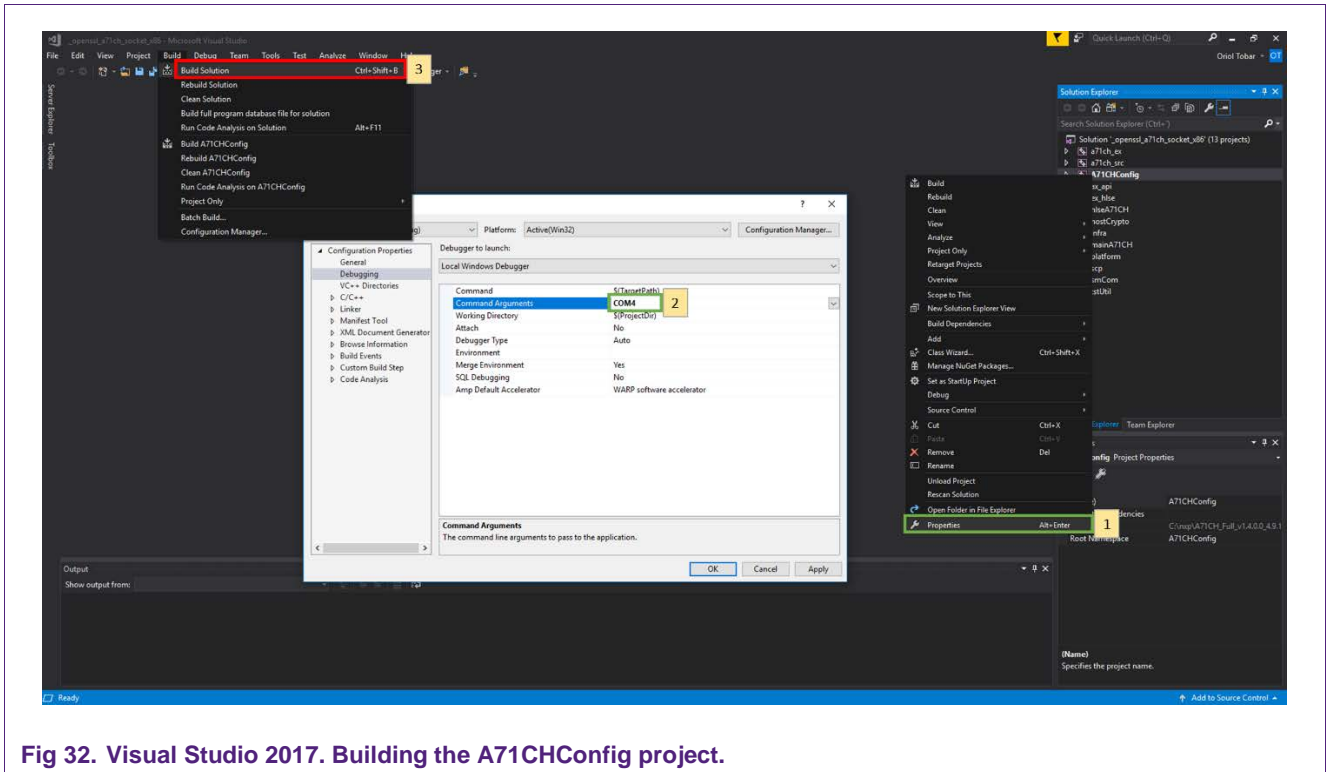


Fig 32. Visual Studio 2017. Building the A71CHConfig project.

The resulting executable will be named 'A71CHConfig_vcom.exe' and will be in directory 'tools'. To run the obtained executable, open a PowerShell window in 'tools' folder. This can be easily done by right clicking the folder while pressing the shift key. Once in the PowerShell window, it is possible to search and list all the existing .exe files (executables) with the following command:

```
dir *.exe
```

As shown in Fig 33, the previously built 'A71CHConfig' project generated an executable file inside the 'tools' folder.

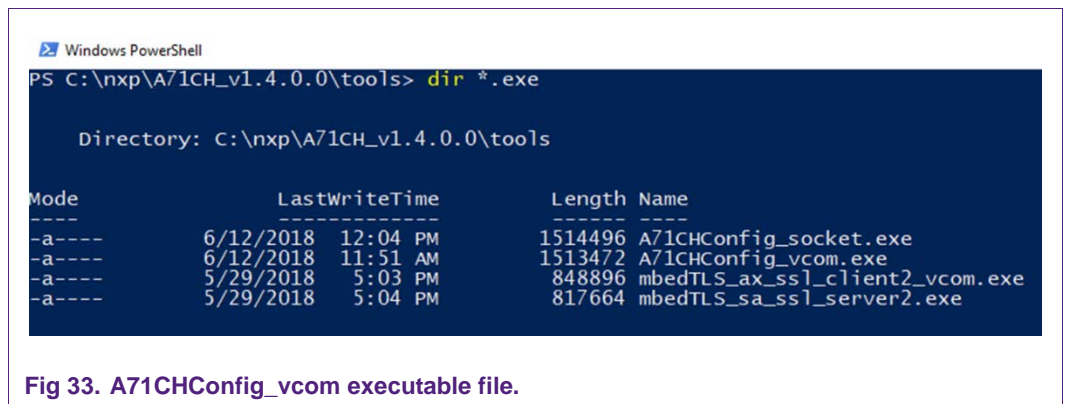


Fig 33. A71CHConfig_vcom executable file.

Finally, the A71CH Configure tool can be run with the following command:

```
.\A71CHConfig_vcom.exe COM4 <input_arguments>
```

The A71CH Configure tool will be launched and the connection between the Windows platform and the A71CH over the Kinetis board will be established.

7.3 Running A71CH Host API usage example

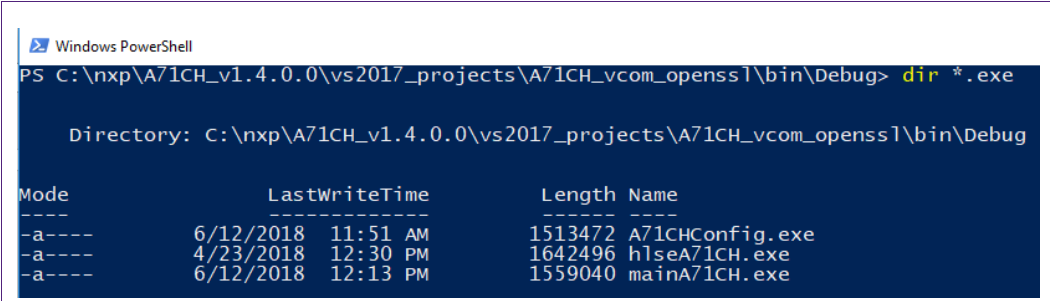
The A71CH Host API usage examples are built with Microsoft Visual Studio:

1. In this case, right-click '*mainA71CH*' in the '*Solution Explorer*' tab and open the '*Properties*' window.
2. Then, in '*Configuration Properties – Debugging*' tab make sure the virtual COM address is set in the '*Command Arguments*' field.
3. Once the VCOM port has been defined, invoke menu *Build - Build Solution* to create the executable.

Finally, the A71CH Host API usage example can be executed in two ways.

7.3.1 Run the A71CH Host API usage executable from a terminal

An executable named '*mainA71CH.exe*' will be generated in '*/bin/Debug*' folder



```

Windows PowerShell
PS C:\npx\A71CH_v1.4.0.0\vs2017_projects\A71CH_vcom_openss\bin\Debug> dir *.exe

Directory: C:\npx\A71CH_v1.4.0.0\vs2017_projects\A71CH_vcom_openss\bin\Debug

Mode                LastWriteTime         Length Name
----                -
-a----             6/12/2018  11:51 AM      1513472 A71CHConfig.exe
-a----             4/23/2018  12:30 PM      1642496 h1seA71CH.exe
-a----             6/12/2018  12:13 PM      1559040 mainA71CH.exe

```

Fig 34. mainA71CH executable file.

To launch it, open a PowerShell by right clicking the '*Debug*' folder while pressing the shift key. Then, start the executable with the following command:

```
.\mainA71CH.exe COM6
```

Where '*COM6*' is the number assigned to the Kinetis configured as a virtual COM port. The A71CH Host API usage example will be launched and the connection between the Windows platform and the A71CH over the Kinetis board will be established (Fig 35).

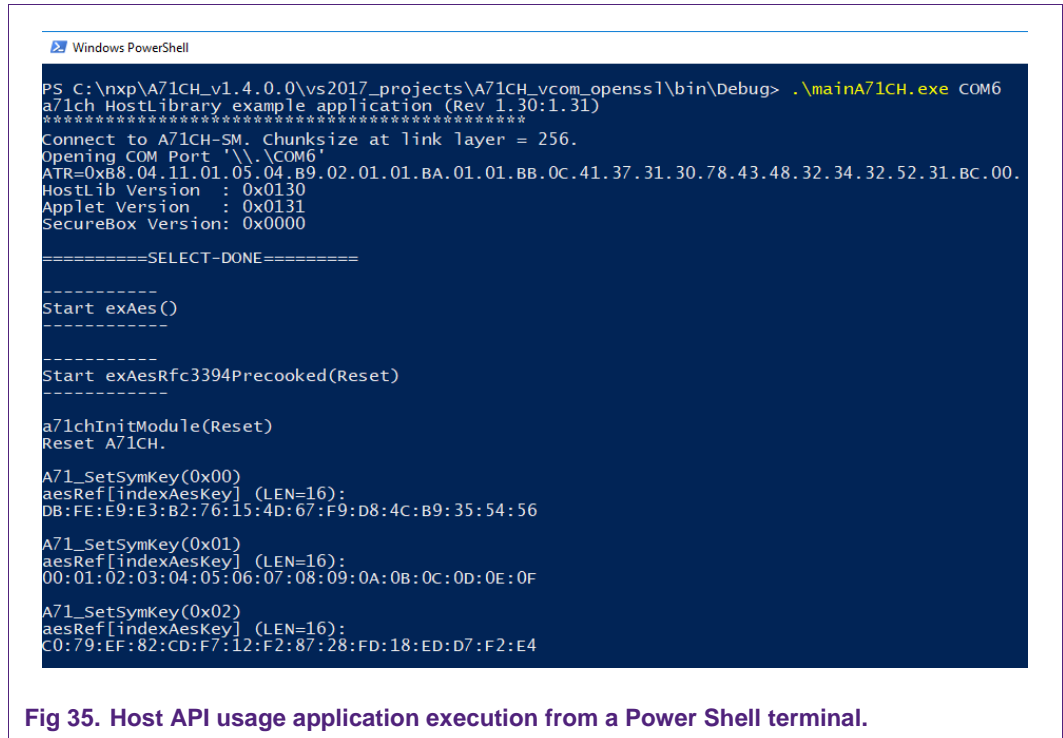


Fig 35. Host API usage application execution from a Power Shell terminal.

7.3.2 Run the A71CH Host API usage from Visual Studio

Alternatively, the A71CH Host API usage application can be directly executed from the Microsoft Visual Studio. For this, just click on the ‘*Local Windows Debugger*’ as highlighted in Fig 36. A new window will pop-up showing the execution. Also, the user can add or remove test points to debug the code.

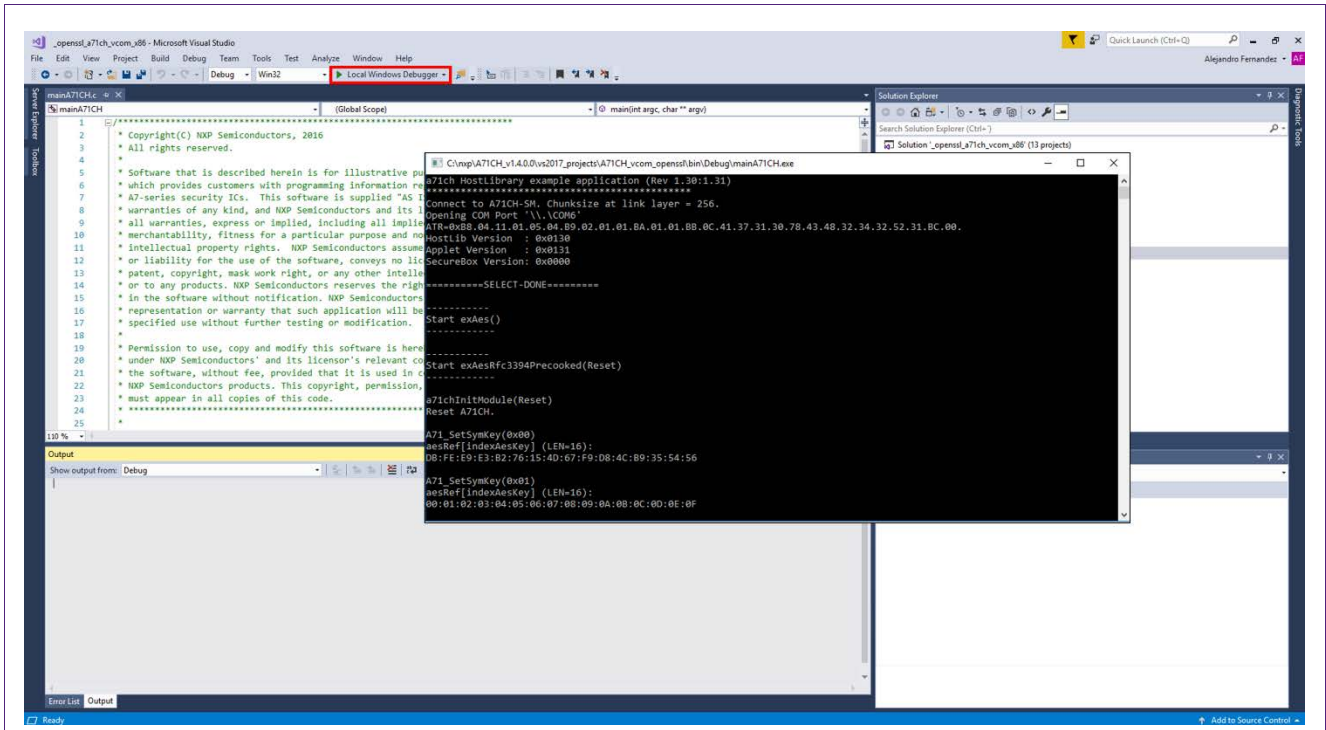


Fig 36. Host API usage application execution from Visual Studio.

8. RJCT server

The A71CH Host software package includes a Remote Java Card Terminal server (RJCT). The RJCT server is a standalone process that can establish a communication session with a Secure Element on behalf of a client process. Therefore, the RJCT server can be used to communicate with A71CH from a remote or local client process.

Fig 37 illustrates the communication between client process running on a remote development PC and the A71CH connected to an embedded target. As an example, the i.MX6UltraLite will be used as the embedded platform. A quick start guide explaining how to setup the i.MX6UltraLite to start developing with the A71CH can be found in [QUICK_START_IMX6].

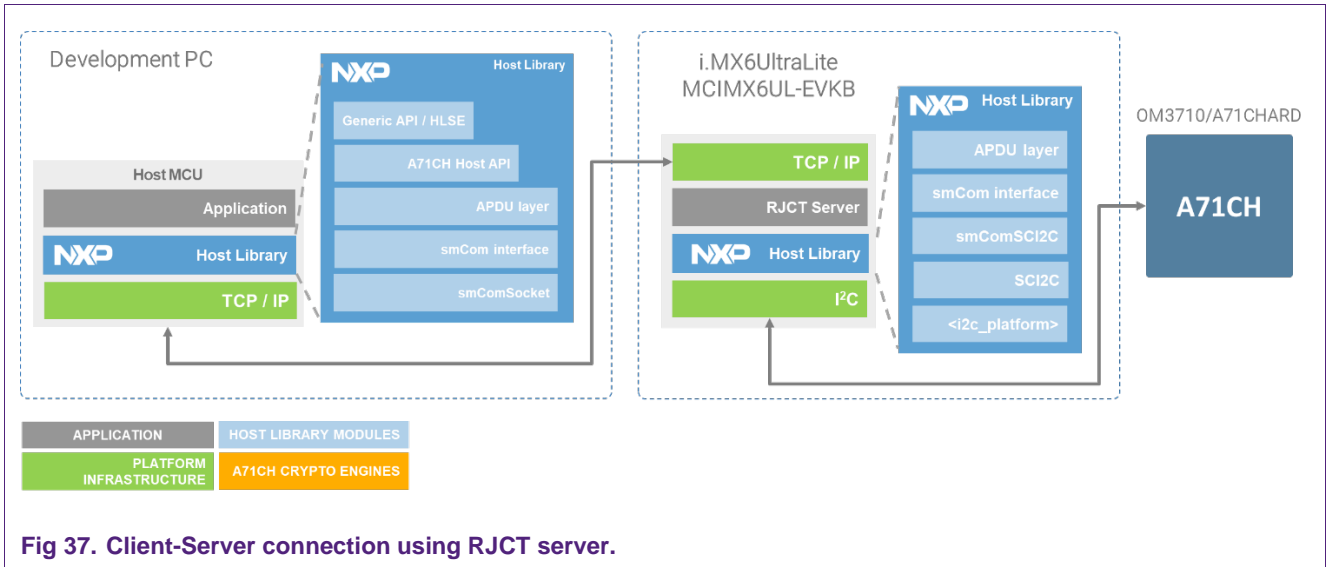


Fig 37. Client-Server connection using RJCT server.

The following steps must be followed to establish a communication between a Windows platform (remote development PC) and the i.MX6UltraLite (embedded platform) over the RJCT server. In this example, the A71CH Configure tool will be executed.

1. Open a Tera Term terminal to interact with the i.MX6UltraLite platform (Fig 38, 6.3 [QUICK_START_IMX6]). Make sure the i.MX6UltraLite is connected to the ethernet.

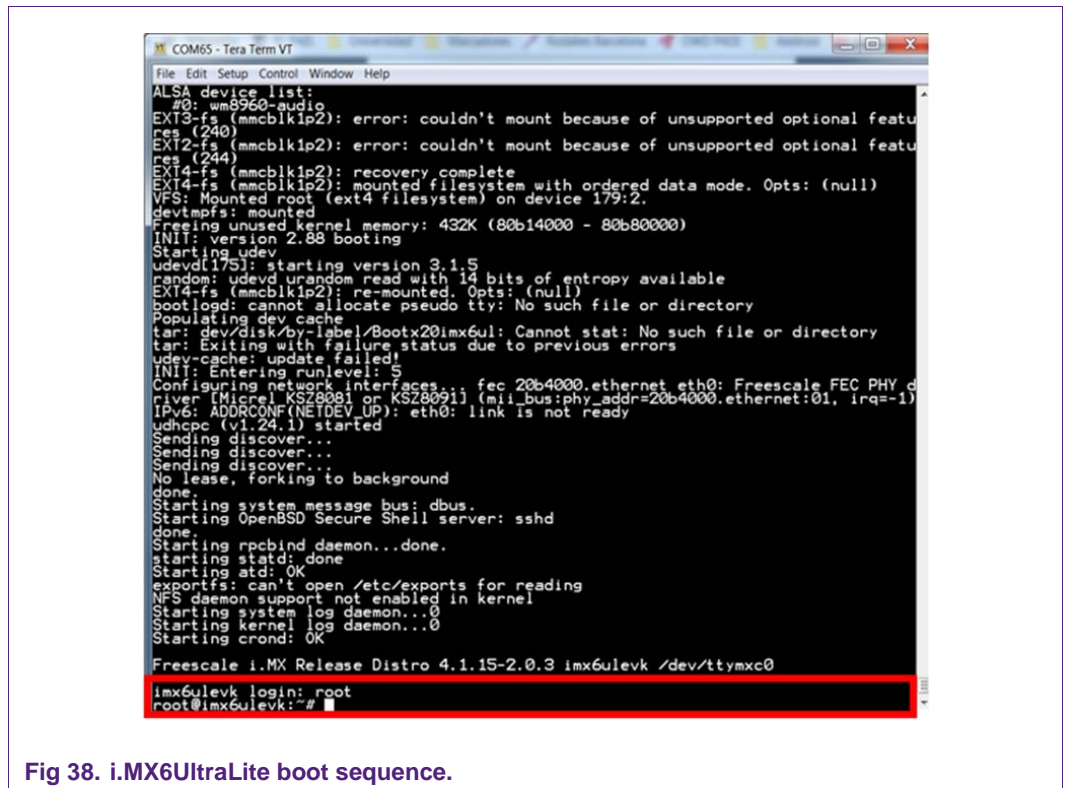


Fig 38. i.MX6UltraLite boot sequence.

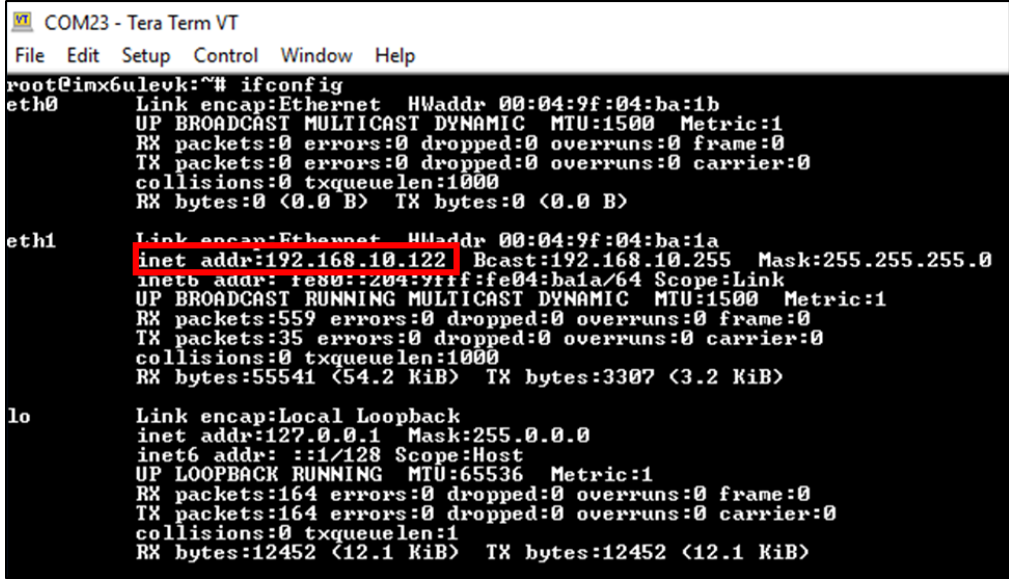
2. Navigate to the folder where the RJCT executable is located:

```
root@imx6ulevk:~# cd axHostSw/linux
```

3. Check the IP address with the following command:

```
root@imx6ulevk:~# ifconfig
```

In this case, the i.MX6UltraLite address is 192.168.10.122



```

COM23 - Tera Term VT
File Edit Setup Control Window Help
root@imx6ulevk:~# ifconfig
eth0      Link encap:Ethernet  HWaddr 00:04:9f:04:ba:1b
          UP BROADCAST MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)

eth1      Link encap:Ethernet  HWaddr 00:04:9f:04:ba:1a
          inet addr:192.168.10.122  Bcast:192.168.10.255  Mask:255.255.255.0
          inet6 addr: fe80::204:9f:fe04:baba/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:559 errors:0 dropped:0 overruns:0 frame:0
          TX packets:35 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:55541 (54.2 KiB)  TX bytes:3307 (3.2 KiB)

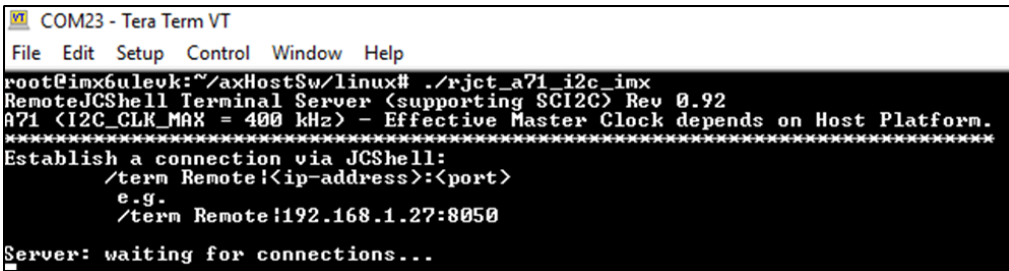
lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:65536  Metric:1
          RX packets:164 errors:0 dropped:0 overruns:0 frame:0
          TX packets:164 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1
          RX bytes:12452 (12.1 KiB)  TX bytes:12452 (12.1 KiB)

```

Fig 39. Check i.MX6UltraLite IP address.

4. Launch the RJCT server:

```
root@imx6ulevk:~# ./rjct_a71_i2c_imx
```



```

COM23 - Tera Term VT
File Edit Setup Control Window Help
root@imx6ulevk:~/axHostSw/linux# ./rjct_a71_i2c_imx
RemoteJCSHELL Terminal Server (supporting SCI2C) Rev 0.92
A71 (I2C_CLK_MAX = 400 kHz) - Effective Master Clock depends on Host Platform.
*****
Establish a connection via JCSHELL:
  /term Remote<ip-address>:<port>
  e.g.
  /term Remote:192.168.1.27:8050
Server: waiting for connections...

```

Fig 40. Launch RJCT server in the i.MX6UltraLite.

The RJCT starts listening to incoming APDUs from the remote development PC. Once in the Windows platform:

1. Open the Visual Studio solution ‘_openssl_a71ch_socket_x86.sln’. Note that in this case we will be using the ‘_socket_’ solution since the APDUs will be sent over TCP/IP to the i.MX6UltraLite.
2. Select the ‘A71CHConfig’ project and set ‘localhost:8050’ in its properties, as shown is shown in Fig 32. Then, build the selected project.
3. An executable file called ‘A71CHConfig_socket’ will be generated in the ‘tools’ folder (Fig 33).
4. Open a Power Shell by right clicking the ‘tools’ folder while pressing the shift key. Then, start the executable with the following command:

```
.\A71CHConfig_socket.exe 192.168.10.122:8050 interactive
```

It will launch the A71CH Configure tool in interactive mode (Fig 41).

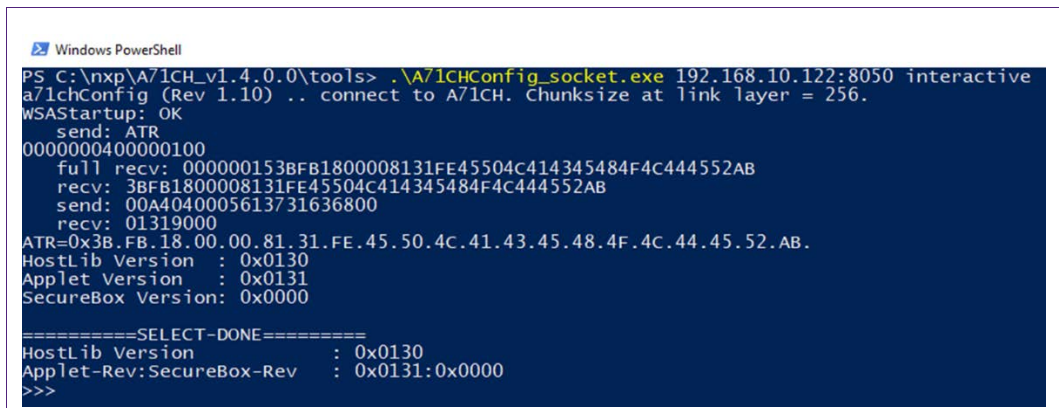


Fig 41. A71CH Configure tool executed in interactive mode.

As can be observed in Fig 42, the RJCT server (in the i.MX6UltraLite) establishes a connection with the development PC, receives the packed APDUs from the development PC and will re-direct these APDUs to the A71CH.

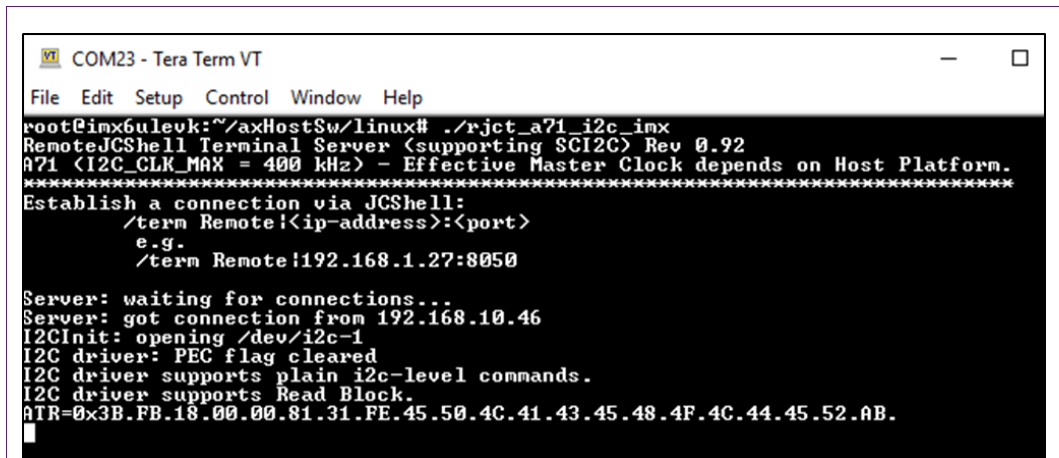


Fig 42. Connection established with the RJCT server running in the i.MX6UltraLite.

Then, the response from the A71CH will be sent to the development PC and printed in the PowerShell terminal. For instance, Fig 43 shows the command *'info all'* executed from the development PC. The response to this command is prompted in the terminal after being sent to the i.MX6UltraLite and consequently to the A71CH.

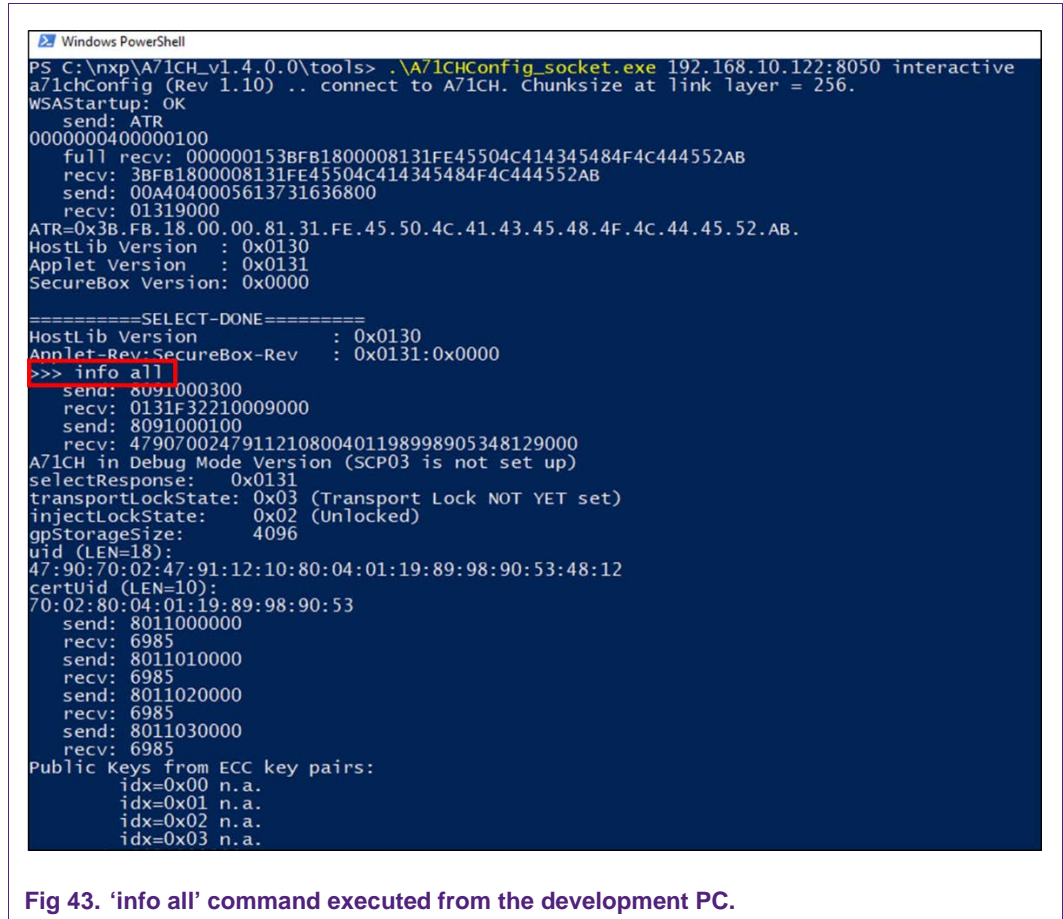


Fig 43. 'info all' command executed from the development PC.

9. Appendix A: Running A71CH application examples with USB to I²C bird

The USB to I²C bird – Ascot adaptor (OM3710/B001) can be used to interface a Windows platform as an alternative to a Kinetis board configured as a virtual COM port. In this case, the development PC will be connected to the A71CH Mini PCB board (OM3710/A71CHPCB) through the USB to I²C bird (OM3710/B001). This appendix presents how to setup the involved hardware and software to execute A71CH applications using the USB to I²C bird (OM3710/B001).

9.1 USB/I²C bird (OM3710/B001)

The OM3710/B001 board is an I²C to USB converter enabling the connection of an A71CH via USB to a PC. The OM3710/B001 comes with a USB I²C Bird / Ascot adaptor and I²C bus cable.

Note: For ordering information: please contact your local sales representative.

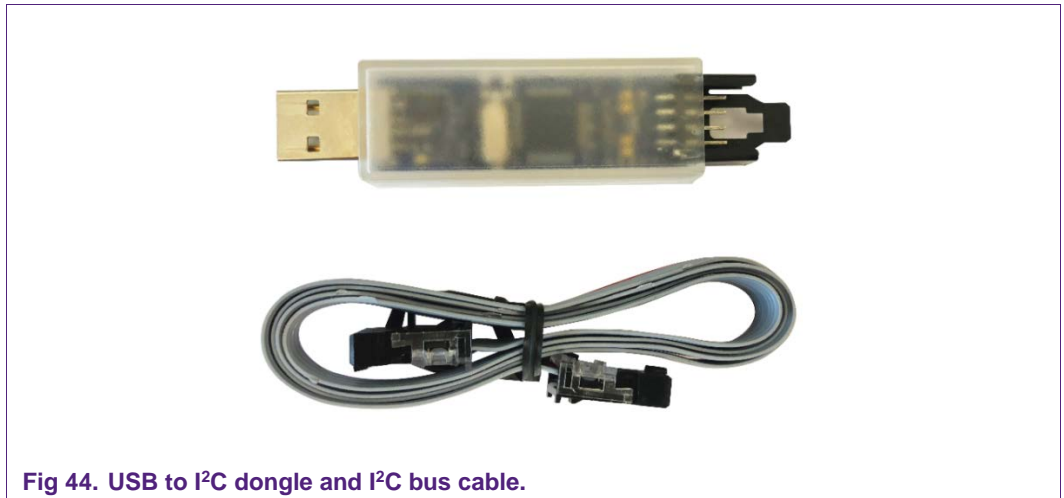


Fig 44. USB to I²C dongle and I²C bus cable.

9.2 Hardware setup

Connect the A71CH mini PCB to the USB to I²C Bird as shown in the following Fig 45. Please note the connector polarity.

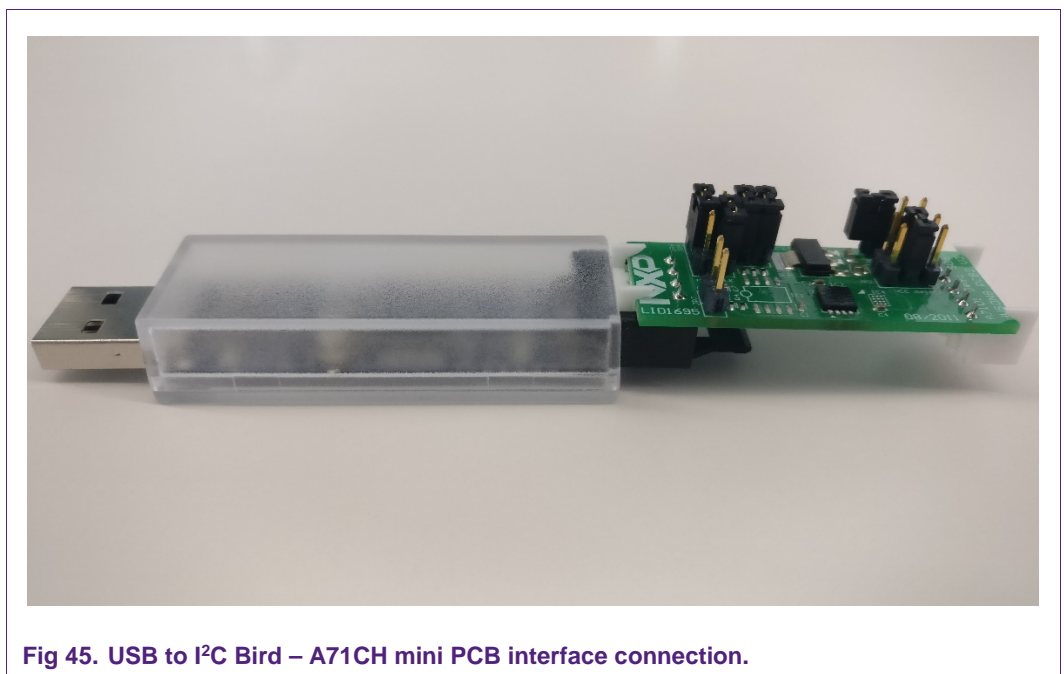


Fig 45. USB to I²C Bird – A71CH mini PCB interface connection.

Finally, connect the USB to I²C Bird to the Windows platform USB port.

9.3 Software setup

Fig 46 illustrates the system architecture, including a detailed view of the windows platform and illustrating the software layers involved when using the USB to I²C Bird solution.

A local server will be created to listen to the IP address where the A71CH application example APDUs are sent. This local server will be created using the *NXPCardServer* java application contained in the A71CH Host software package and will be in charge of waiting for requests from the IP address, capturing and prompting the APDU commands in a window, and re-directing these APDUs to the USB to I²C adaptor I²C address. As such, a TCP/IP address needs to be passed as an argument when running an A71CH example application.

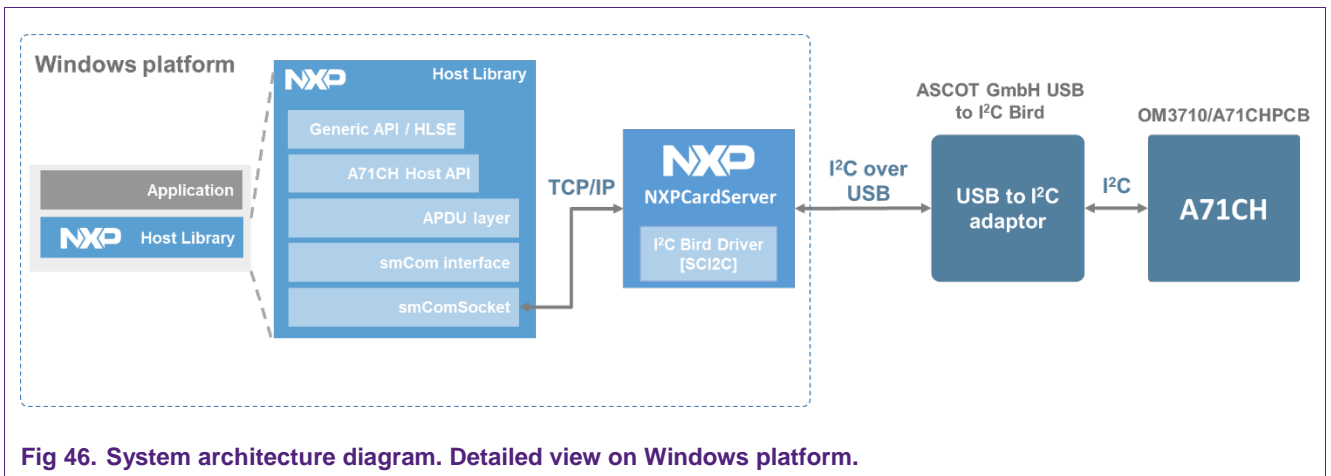


Fig 46. System architecture diagram. Detailed view on Windows platform.

The following steps are required to run the A71CH Windows-based applications using the USB to I²C bird:

- Install USB to I²C adaptor drivers.
- Open a local server with *NXPCardServer* java application. Local server will listen to *localhost:8050* port and will redirect data to I²C address *90h*. This is a transparent operation for the user.
- Build the target A71CH application with Microsoft Visual Studio.
- Start the built application from a Windows terminal passing the *localhost:8050* address as input argument.

9.3.1 USB to I²C ASCOT adaptor – Drivers

To use the USB to I²C Ascot adaptor, it is necessary to install the required drivers. These can be found in [I2C_BIRD].

Install the executable for your processor (either 32 or 64 bits) and follow the setup wizard until it is finished. Once the driver has been correctly installed, the USB to I²C adaptor can be used to establish a communication between the Windows platform and the A71CH contained in the OM3710/A71CHPCB. Fig 47 shows a capture of the setup wizard.

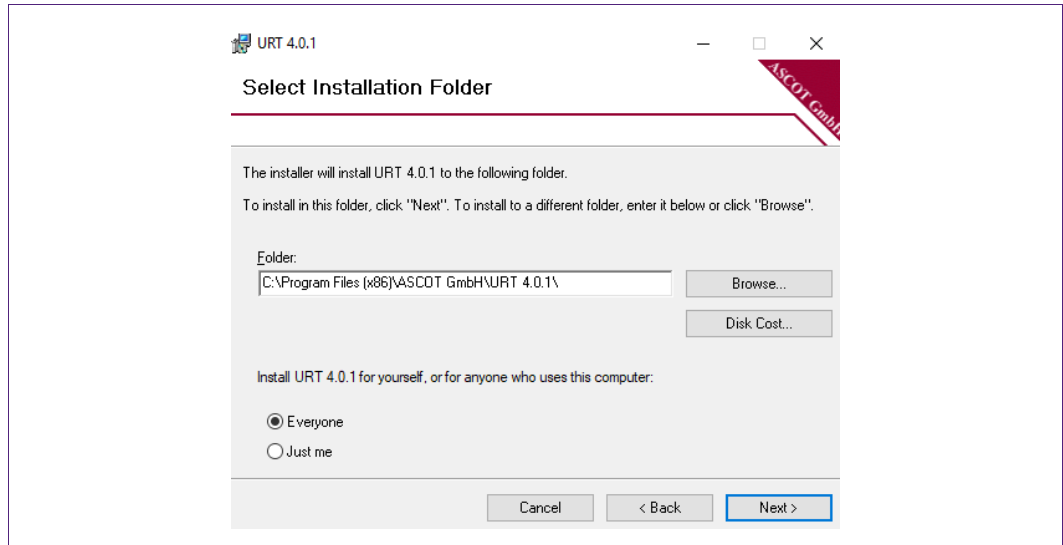


Fig 47. USB to I²C Ascot adaptor. Drivers installation wizard.

9.3.2 NXPCardServer

NXPCardServer is a Java application that is called by using 'javaws' command in the Command Prompt. This command launches Java Web Start, which executes Java applications or applets hosted on a network and creates a local server instance. This local server will be constantly listening to the *localhost:8050* IP address, and will re-direct the input commands to the I²C address of the USB to I²C Ascot adaptor.

The NXPCardServer Java application requires a 32-bit Java Runtime Environment (JRE). It is known to be compatible with Java6, Java7 and Java8.

All the required files for launching the NXPCardServer application can be found in 'ext/NXPCardServer' folder of the A71CH Host SW package. This folder contains several windows batch files (.bat) and application extensions (.dll), as shown in Fig 50.

When working with the USB to I²C adaptor connected to a Windows platform, the 'NXPCardServer_I2C_Bird.bat' file should be run to execute the 'javaws' command and start the server.

This .bat file will call the 'NXPCardServer.bat' executable with the I²C Bird configuration. 'NXPCardServer.bat' will first check if Java Runtime environment is installed in the Windows device. Then, the following environment variable will be automatically defined:

- **JRE_HOME=%PROGRAMFILES(X86)%\Java\jre_version**

Where **jre_version** belongs to the installed JRE version. This variable defines where the JRE is installed and where the 'javaws' is. Therefore, this must be correctly defined to call the 'javaws' function and open the local server.

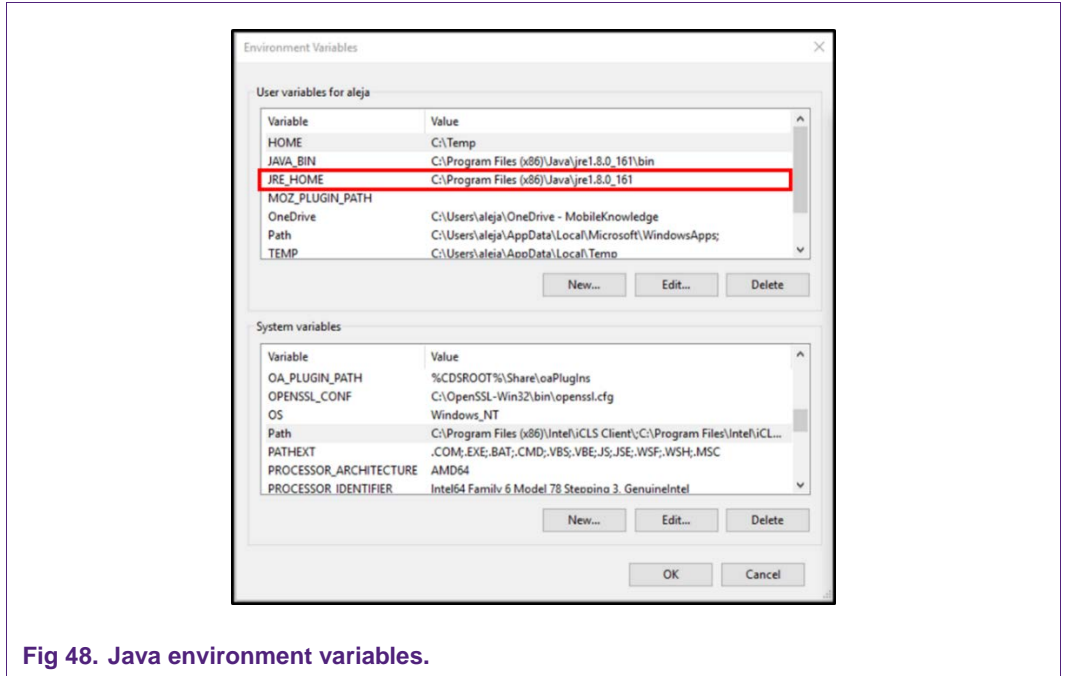


Fig 48. Java environment variables.

The user should make sure that JRE_HOME variable has been correctly created or edited by opening the system environment variables editor of Windows. Fig 48 shows the system environment variables editor on Windows. As can be observed, JAVA_BIN and JRE_HOME variables are defined in the user variables window, while PATH variable includes the contents of JAVA_BIN (right figure).

Finally, the 'javaws' command will be called and the NXPCardServer Java applet will be started. If NXPCardServer is successfully launched, the window presented in Fig 49 will be opened and the server will start to listen to the localhost:8050 IP address.

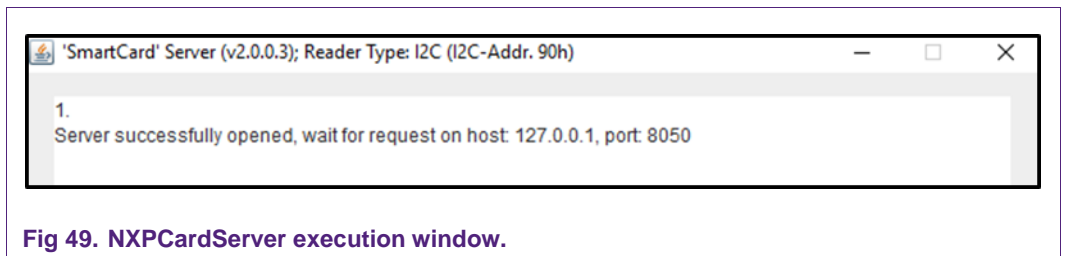


Fig 49. NXPCardServer execution window.

In case these environment variables are not properly defined, the 'javaws' command will not be executed; thus, the server will not be started.

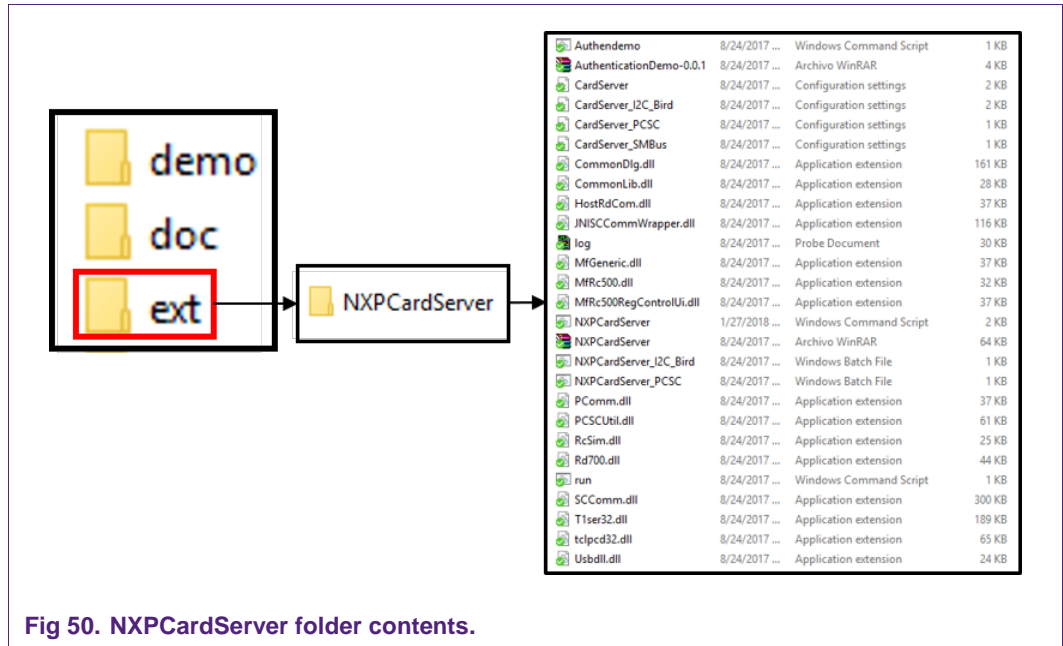


Fig 50. NXPCardServer folder contents.

9.4 Running A71CH Configure tool

To run the A71CH Configure tool:

1. Open the Visual Studio solution ‘_openssl_a71ch_socket_x86.sln’. Note that in this case we will be using the ‘_socket_’ solution since the APDUs will be sent over TCP/IP to the NXPCardServer application.
2. Right-click ‘mainA71CH’ in the ‘Solution Explorer’ tab and open the ‘Properties’ window.
3. Then, in ‘Configuration Properties – Debugging’ tab make sure the IP address localhost:8050 is set in the ‘Command Arguments’ field.
4. Once the IP address has been defined, invoke menu *Build - Build Solution* to create the executable.
5. An executable file called ‘A71CHConfig_socket’ will be generated in the ‘tools’ folder.

Considering the NXPCardServer is running, open a PowerShell by right clicking the ‘tools’ folder while pressing the shift key. Then, start the A71CH Configure tool with the following command:

```
.\A71CHConfig_socket.exe localhost:8050 debug reset
```

And prompt the A71CH status information with:

```
.\A71CHConfig_socket.exe localhost:8050 info status
```

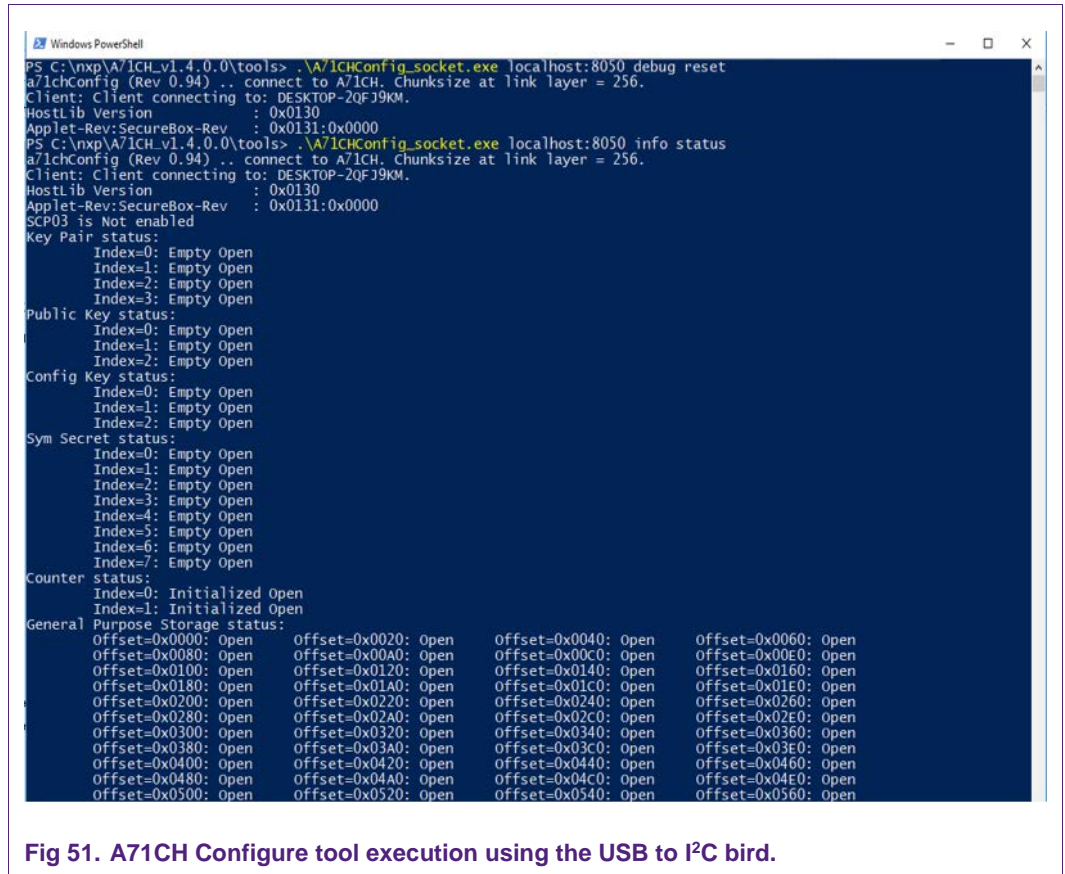


Fig 51. A71CH Configure tool execution using the USB to I²C bird.

9.5 Running A71CH Host API usage example

Similarly, the A71CH Host API usage example can be built with Microsoft Visual Studio and launched from a PowerShell terminal. The following command can be used to start it:

```
.\mainA71CH localhost:8050
```

10. Appendix B: VCOM driver installation troubleshooting

If the Kinetis board is connected (Fig 52) and has been configured to perform as an USB to I²C adaptor, the Windows platform should detect and assign it a virtual COM port number.

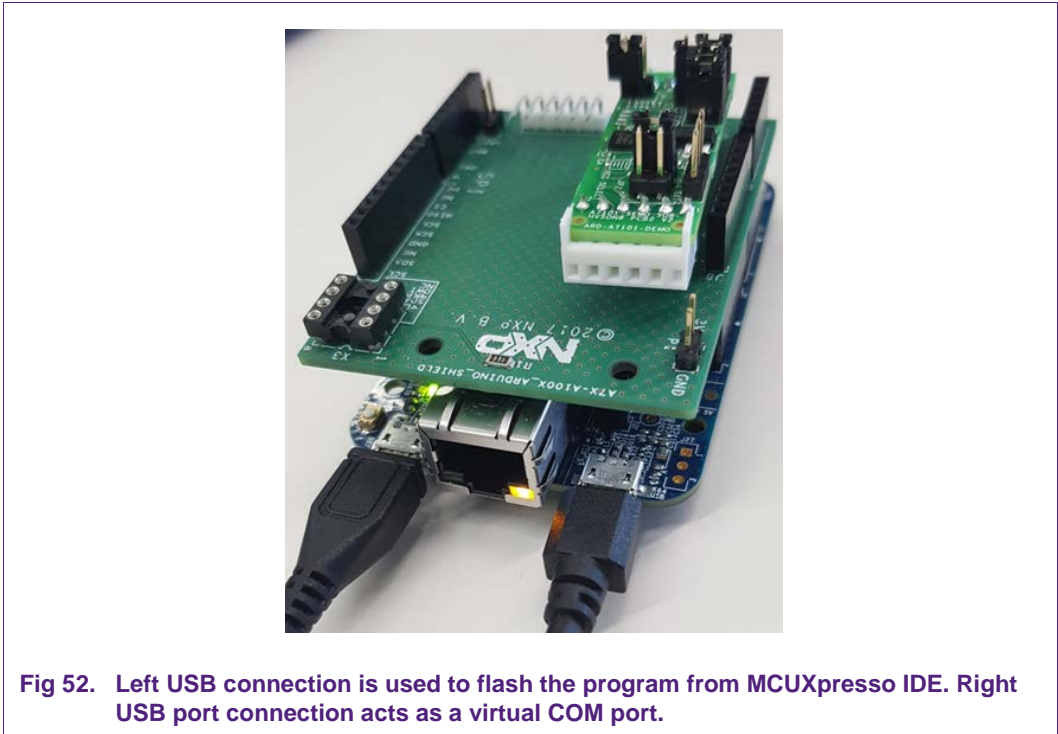


Fig 52. Left USB connection is used to flash the program from MCUXpresso IDE. Right USB port connection acts as a virtual COM port.

To ensure that the Kinetis is correctly recognized, open the ‘Device Manager’ control panel. The Kinetis board should be detected and labeled as ‘...VCOM Port (COMX)’ within the ‘Ports (COM & LPT)’ drop-down (Fig 53).



Fig 53. Drivers correctly installed.

If the Windows OS does not recognize the Kinetis board, it would appear labeled as ‘MCU VIRTUAL COM DEMO’ within ‘Other devices’ as shown in Fig 54. In this case, it means that the drivers need to be updated.

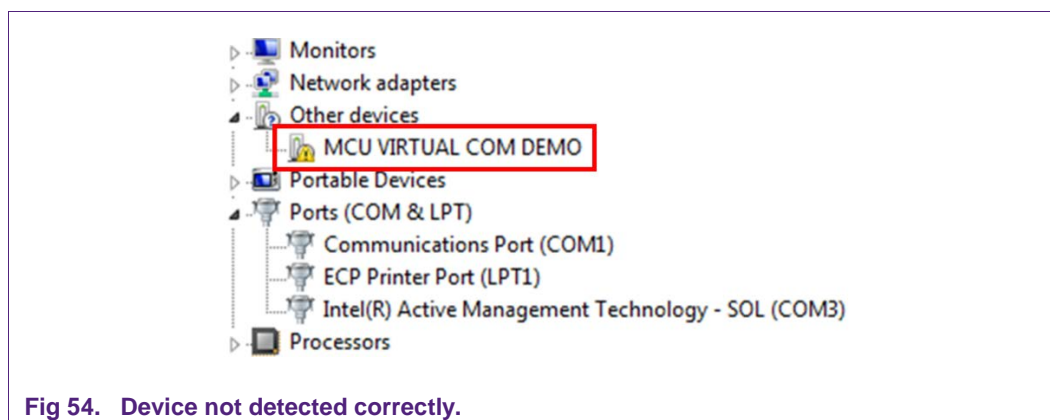


Fig 54. Device not detected correctly.

In order to update the drivers, follow the next steps:

6. Right-click on 'MCU VIRTUAL COM DEMO'.
7. Click on Update Driver.
8. 'Browse my computer for driver software'.
9. 'Let me pick from a list of device drivers on my computer'.
10. Select Ports (COM & LPT).
11. Un-check 'Show compatible hardware'.
12. Select 'NXP' and 'LPC USB VCOM Port'.
13. Ignore the warning message and click on 'Yes'.

11. Appendix C: Running examples using Cygwin

This appendix specifies how to install Cygwin and use it to run the examples and shell scripts available in the A71CH support package from a Windows machine without the need of using Visual Studio IDE.

Cygwin is a large collection of GNU and Open Source tools which provide functionality like a Linux Distribution on Windows. It consists of two parts; a Dynamic Link Library (DLL) and an extensive collection of software tools and applications that provide a Unix-like environment.

11.1 Cygwin Installation

Cygwin programs are installed by running Cygwin's setup program. This program downloads the necessary packages from different repositories in the Internet. To download this setup program, go to [CYGWIN]. Choose the adequate option for your system (either 32-bit or 64-bit) and click on it so the download can start. Save it in your machine and run it at any time to update or install Cygwin packages.

Once the setup program is executed, it will guide the user through the installation process. The first step is to choose how to install the packages, which can be done downloading it from the Internet, installing it from a local directory, or even downloading the packets without installing them. Choose 'Install from the internet' and click 'Next'. Once an installation directory is chosen, it is necessary to choose the Internet

Connection type. As can be observed in Fig 55, different options can be chosen depending on the what the user considers appropriate.

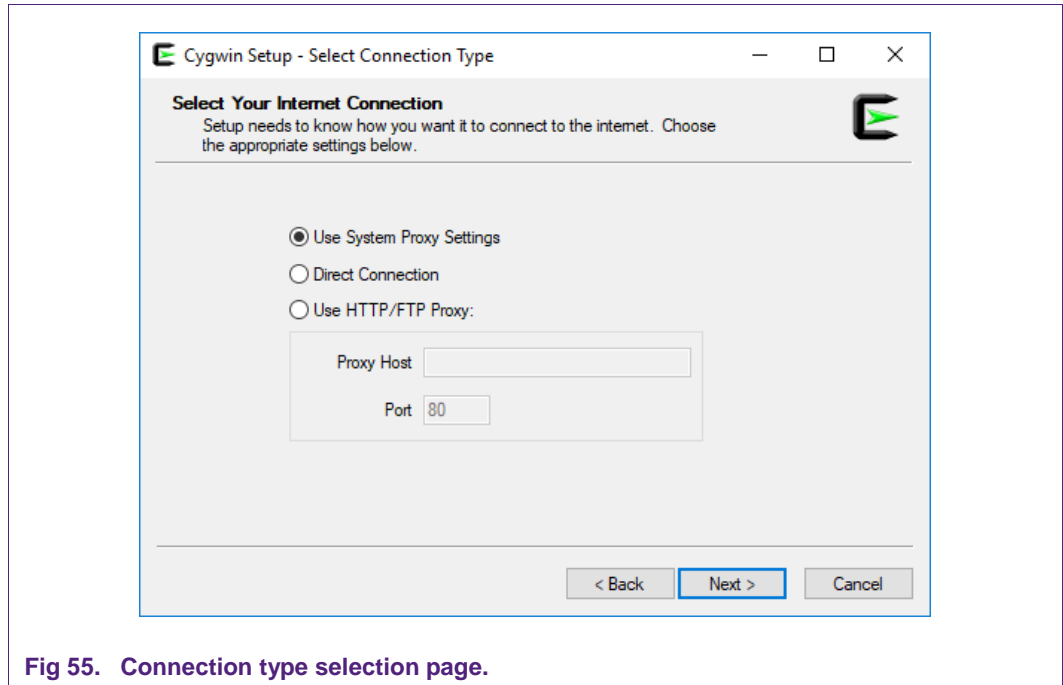


Fig 55. Connection type selection page.

Then, a list of sites from where to download the packages is provided. Either select one of them or define a known site in the corresponding box. After a site is chosen, or defined, the list of packages will appear, as in Fig 56.

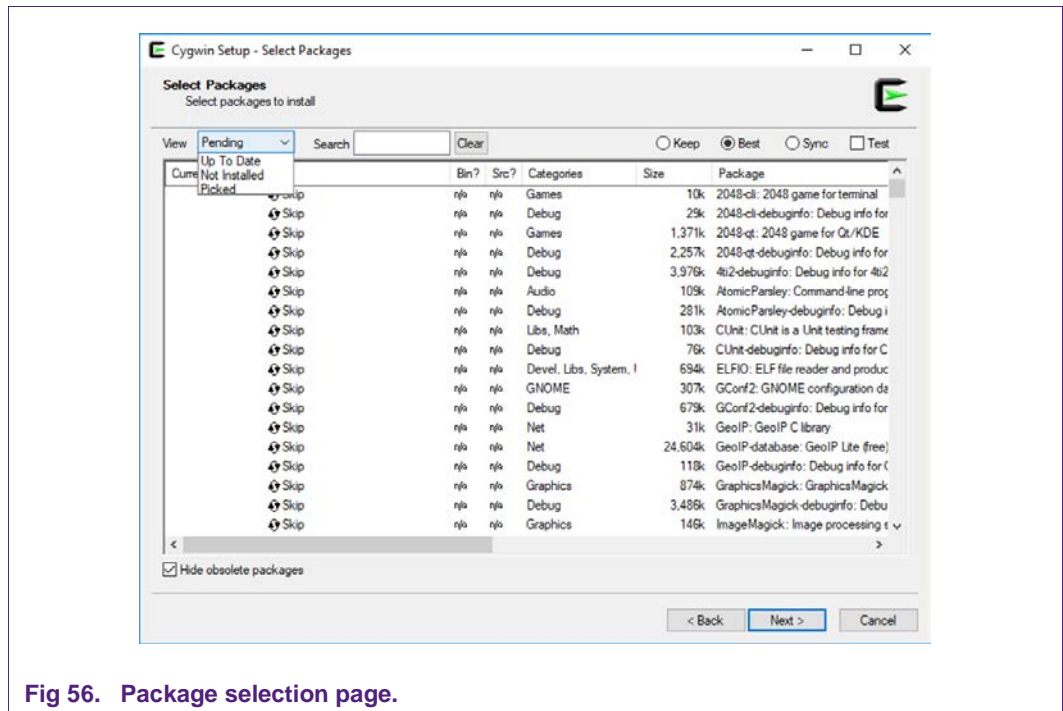


Fig 56. Package selection page.

In the View tab, the user can select which packages are shown (the full set of packages, those with updates available, the ones that have not been installed, etc.). To download a certain package, click on it and it will be selected for downloading.

To make use of Cygwin with A71CH, it is necessary to download the following packages: *gcc*, *make*, *Openssl* and *Openssl-devel*. Fig 57 shows the packages that must be downloaded.

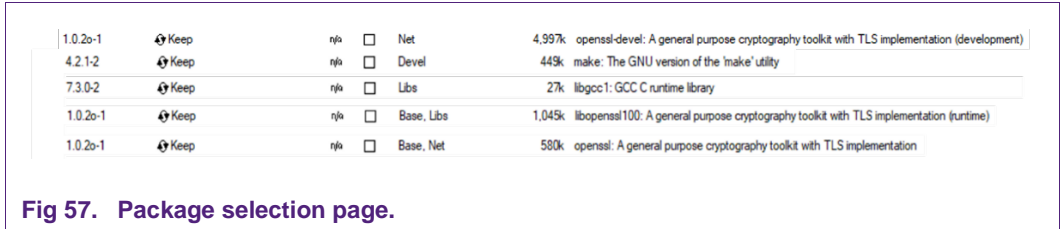


Fig 57. Package selection page.

Once all the packages the user wants to download have been selected, click 'Next'. Confirm the packages to be downloaded and click 'Next' again. The setup program will start downloading the selected packages.

11.2 A71CH application examples execution

It has already been said that Cygwin allows working in a Unix-like environment. By using it, the examples available in the Host Library can be executed from a Windows machine without the need of running Visual Studio.

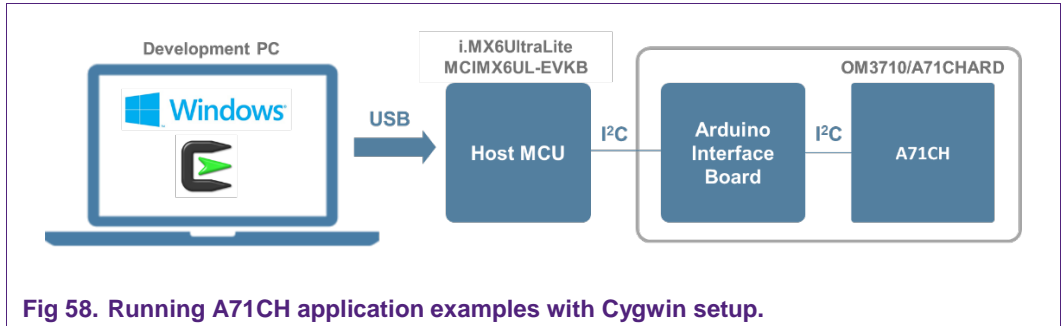


Fig 58. Running A71CH application examples with Cygwin setup.

First, it is necessary to download the Bash Installer of the A71CH Host Software package [A71CH_HOST_SW]. An SH file named 'A71CH-HOST-LINUX' will be downloaded. Run the Cygwin terminal and go to the folder where the Bash Installer has been saved to run it (Fig 59).

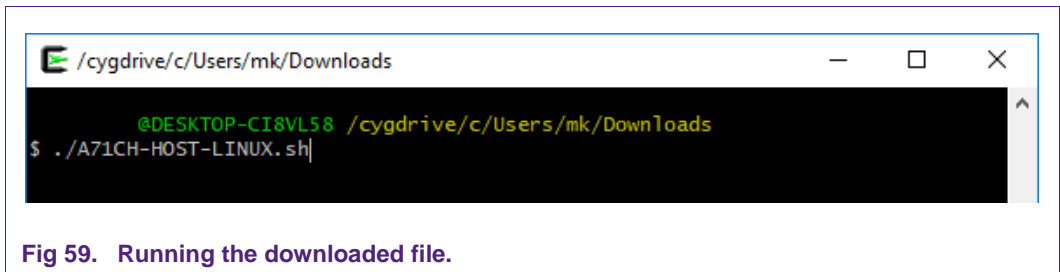


Fig 59. Running the downloaded file.

This will create a folder named 'axHostSw' where the Host Library will be located. Using Cygwin, move to the 'linux' folder and build the A71CH binaries, as in Fig 60.

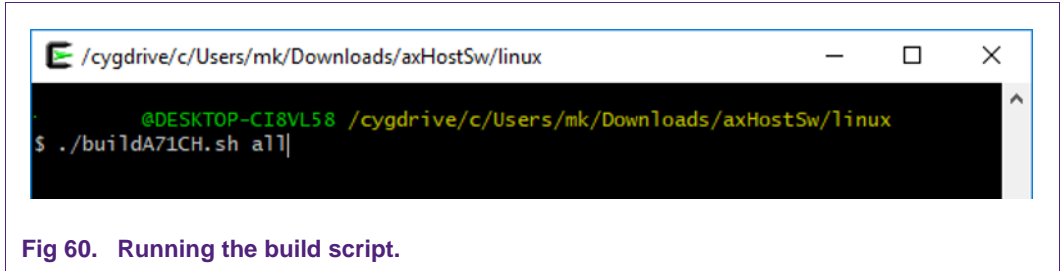


Fig 60. Running the build script.

Once this script is executed, the following files will be installed in the 'linux' folder.

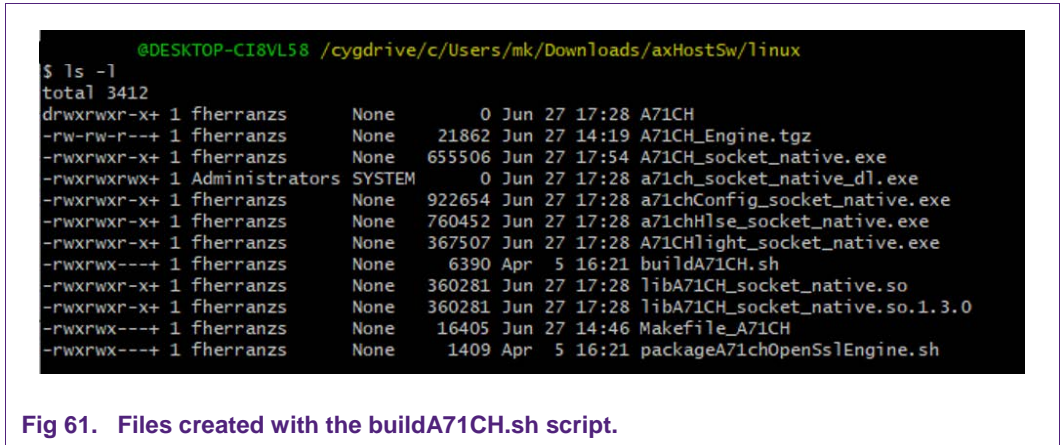


Fig 61. Files created with the buildA71CH.sh script.

This way, the user can use the A71CH Configure Tool, the Host API usage examples and the OpenSSL Engine examples from the Cygwin terminal. To do that, connect the i.MX6UltraLite board as explained in [QUICK_START_IMX6], check its IP address, as shown in Section 8, and run the Remote Java Card Terminal, also as explained in Section 8.

11.2.1 Running A71CH OpenSSL Engine examples

The OpenSSL Engine examples are available in the */hostLib/embSeEngine/a71chDemo/scripts*. For instance, the user can run the *tlsPrepareClient.sh* script to provision an A71CH security IC for TLS connection. In the Cygwin terminal, run the example specifying the IP address of the i.MX6UL evaluation board, as shown in Fig 62.

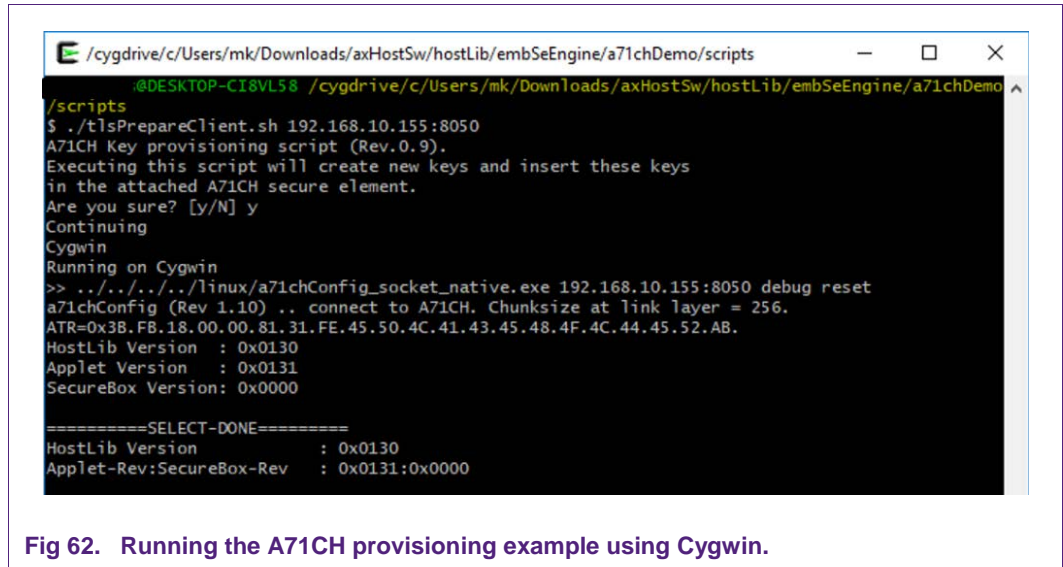


Fig 62. Running the A71CH provisioning example using Cygwin.

Once the script is run successfully, the A71CH will be provisioned and ready to establish a TLS connection to a server.

11.2.2 Running the A71CH Configure Tool

Using Cygwin, it is possible to use the A71CH Configure Tool without using Visual Studio. As an example, to run the 'info all' command, just run the command shown in Fig 63 (with the appropriate IP address and port).

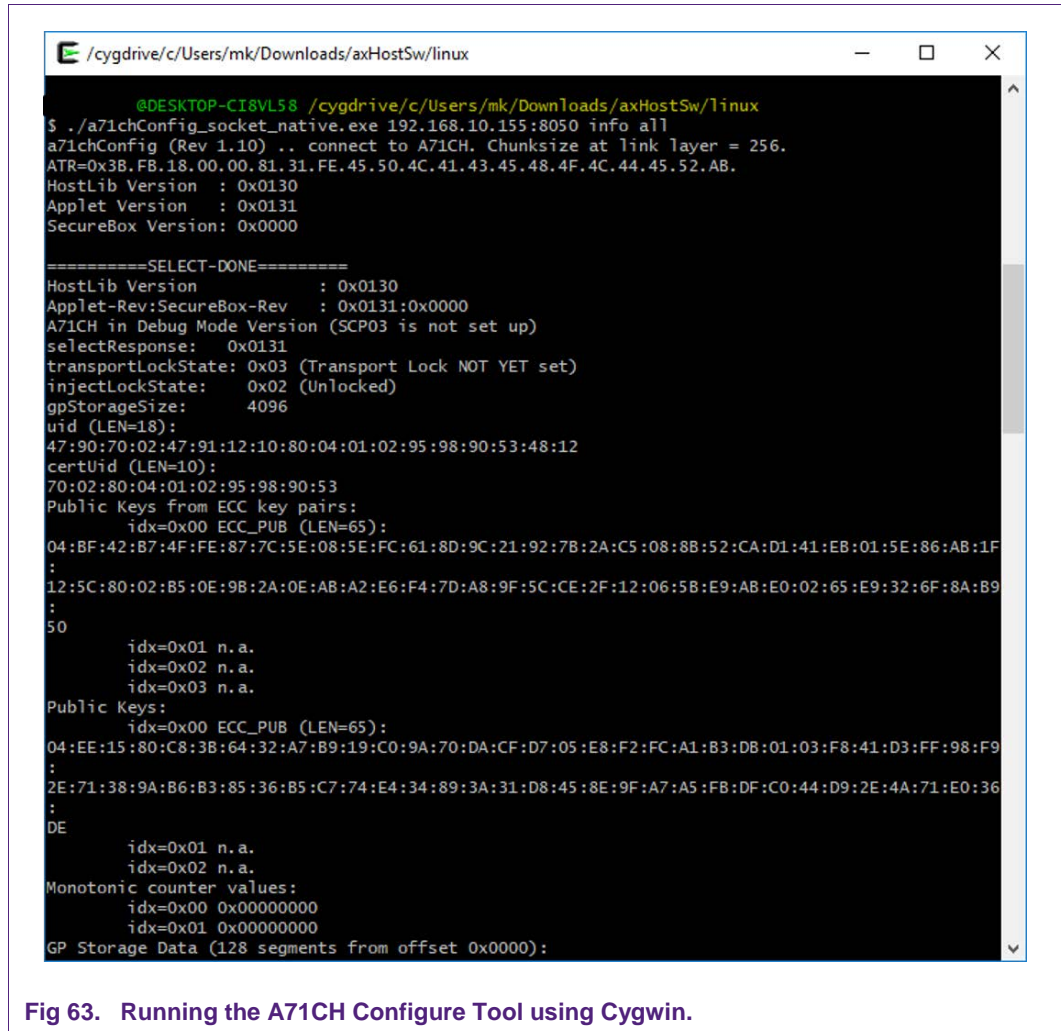


Fig 63. Running the A71CH Configure Tool using Cygwin.

11.2.3 Running A71CH Host API usage examples

Similarly, it is possible to run the Host API usage example by running the `A71CH_socket_native.exe` application. The information about the A71CH Host API usage examples can be found in [AN_A71CH_HOST_SW] and in the Doxygen documentation [A71CH_HOST_SW]. Fig 64 shows the result of this command in the Cygwin terminal.

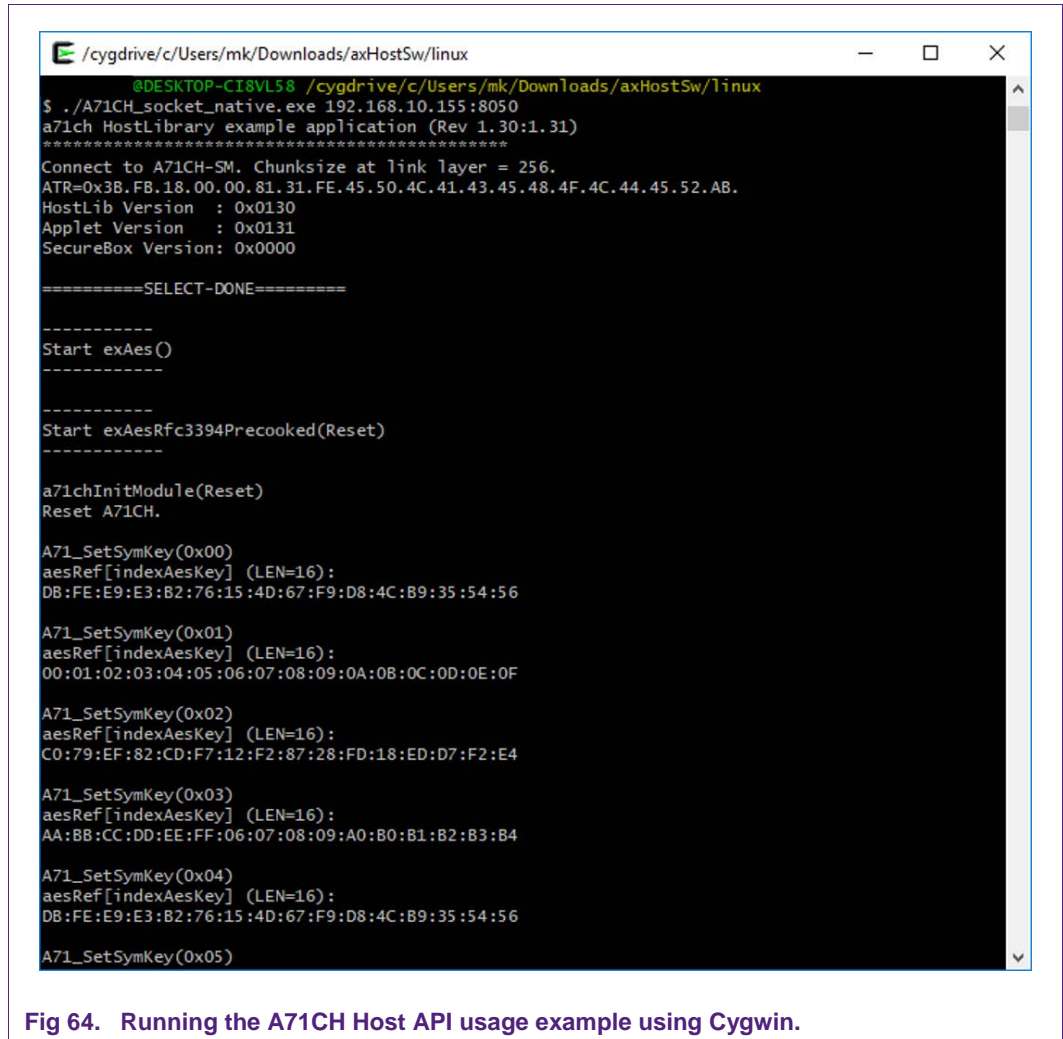


Fig 64. Running the A71CH Host API usage example using Cygwin.

12. References

All the references contained in this document are listed in the following table.

Table 2. References

[SCI2C]	SCI2C Protocol Specification Rev 1.5 or later (restricted to 1.x Revision, NOT compatible with Rev. 2.x)
[OPEN_SSL]	OpenSSL Cryptography and SSL/TLS Toolkit information - www.openssl.org
[A71CH_APDU]	APDU Specification of A71CH Security Module - ds409420
[A71CH_HOST_SW]	A71CH Host Software Package (Windows Installer) – DocStore, document number sw4673xx ¹ , Version 01.03.00 (or later), available on www.nxp.com/A71CH A71CH Host Software Package (Bash installer) – DocStore, document number sw4672xx ¹ , Version 01.03.00 (or later), available on www.nxp.com/A71CH
[FRDM_K64F]	Kinetis FRDM-K64F - https://www.nxp.com/products/processors-and-microcontrollers/arm-based-processors-and-mcus/kinetis-cortex-m-mcus/k-seriesperformancem4/k2x-usb/freedom-development-platform-for-kinetis-k64-k63-and-k24-mcus:FRDM-K64F
[FRDM_K82F]	Kinetis FRDM-K82F - https://www.nxp.com/products/processors-and-microcontrollers/arm-based-processors-and-mcus/kinetis-cortex-m-mcus/k-seriesperformancem4/k8x-secure/freedom-development-platform-for-kinetis-k82-k81-and-k80-mcus:FRDM-K82F
[MCUXPRESSO_IDE]	MCUXpresso IDE - https://www.nxp.com/support/developer-resources/software-development-tools/mcuxpresso-software-and-tools/mcuxpresso-integrated-development-environment-ide:MCUXpresso-IDE
[OPENSDA_FIRMWARE]	OpenSDA / OpenSDA V2 website - https://www.segger.com/products/debug-probes/j-link/models/other-j-links/opensda-sda-v2/
[MBED_TLS]	mbedTLS website - https://tls.mbed.org/
[SDKBUILDER]	MCUXpresso SBKBuilder website - https://mcuxpresso.nxp.com/en/select
[QUICK_START_IMX6]	AN12119 Quick start guide for OM3710A71CHARD i.MX6 – Application note, document number 4582 ^{**1}
[I2C_BIRD]	http://www.ascot-gmbh.de/download/download.de.html
[AN_A71CH_HOST_SW]	AN12133 A71CH Host software package documentation – Application note, document number 4643 ^{**1}
[CYGWIN]	Cygwin Setup download - https://cygwin.com/install.html

13. Legal information

13.1 Definitions

Draft — The document is a draft version only. The content is still under internal review and subject to formal approval, which may result in modifications or additions. NXP Semiconductors does not give any representations or warranties as to the accuracy or completeness of information included herein and shall have no liability for the consequences of use of such information.

13.2 Disclaimers

Limited warranty and liability — Information in this document is believed to be accurate and reliable. However, NXP Semiconductors does not give any representations or warranties, expressed or implied, as to the accuracy or completeness of such information and shall have no liability for the consequences of use of such information. NXP Semiconductors takes no responsibility for the content in this document if provided by an information source outside of NXP Semiconductors.

In no event shall NXP Semiconductors be liable for any indirect, incidental, punitive, special or consequential damages (including - without limitation - lost profits, lost savings, business interruption, costs related to the removal or replacement of any products or rework charges) whether or not such damages are based on tort (including negligence), warranty, breach of contract or any other legal theory.

Notwithstanding any damages that customer might incur for any reason whatsoever, NXP Semiconductors' aggregate and cumulative liability towards customer for the products described herein shall be limited in accordance with the *Terms and conditions of commercial sale* of NXP Semiconductors.

Right to make changes — NXP Semiconductors reserves the right to make changes to information published in this document, including without limitation specifications and product descriptions, at any time and without notice. This document supersedes and replaces all information supplied prior to the publication hereof.

Suitability for use — NXP Semiconductors products are not designed, authorized or warranted to be suitable for use in life support, life-critical or safety-critical systems or equipment, nor in applications where failure or malfunction of an NXP Semiconductors product can reasonably be expected to result in personal injury, death or severe property or environmental damage. NXP Semiconductors and its suppliers accept no liability for inclusion and/or use of NXP Semiconductors products in such equipment or applications and therefore such inclusion and/or use is at the customer's own risk.

Applications — Applications that are described herein for any of these products are for illustrative purposes only. NXP Semiconductors makes no representation or warranty that such applications will be suitable for the specified use without further testing or modification.

Customers are responsible for the design and operation of their applications and products using NXP Semiconductors products, and NXP Semiconductors accepts no liability for any assistance with applications or customer product design. It is customer's sole responsibility to determine whether the NXP Semiconductors product is suitable and fit for the customer's applications and products planned, as well as for the planned application and use of customer's third party customer(s). Customers should provide appropriate design and operating safeguards to minimize the risks associated with their applications and products.

NXP Semiconductors does not accept any liability related to any default, damage, costs or problem which is based on any weakness or default in the

customer's applications or products, or the application or use by customer's third party customer(s). Customer is responsible for doing all necessary testing for the customer's applications and products using NXP Semiconductors products in order to avoid a default of the applications and the products or of the application or use by customer's third party customer(s). NXP does not accept any liability in this respect.

Export control — This document as well as the item(s) described herein may be subject to export control regulations. Export might require a prior authorization from competent authorities.

Translations — A non-English (translated) version of a document is for reference only. The English version shall prevail in case of any discrepancy between the translated and English versions.

Evaluation products — This product is provided on an "as is" and "with all faults" basis for evaluation purposes only. NXP Semiconductors, its affiliates and their suppliers expressly disclaim all warranties, whether express, implied or statutory, including but not limited to the implied warranties of non-infringement, merchantability and fitness for a particular purpose. The entire risk as to the quality, or arising out of the use or performance, of this product remains with customer.

In no event shall NXP Semiconductors, its affiliates or their suppliers be liable to customer for any special, indirect, consequential, punitive or incidental damages (including without limitation damages for loss of business, business interruption, loss of use, loss of data or information, and the like) arising out of the use of or inability to use the product, whether or not based on tort (including negligence), strict liability, breach of contract, breach of warranty or any other theory, even if advised of the possibility of such damages.

Notwithstanding any damages that customer might incur for any reason whatsoever (including without limitation, all damages referenced above and all direct or general damages), the entire liability of NXP Semiconductors, its affiliates and their suppliers and customer's exclusive remedy for all of the foregoing shall be limited to actual damages incurred by customer based on reasonable reliance up to the greater of the amount actually paid by customer for the product or five dollars (US\$5.00). The foregoing limitations, exclusions and disclaimers shall apply to the maximum extent permitted by applicable law, even if any remedy fails of its essential purpose.

13.1 Licenses

ICs with DPA Countermeasures functionality



NXP ICs containing functionality implementing countermeasures to Differential Power Analysis and Simple Power Analysis are produced and sold under applicable license from Cryptography Research, Inc.

13.2 Trademarks

Notice: All referenced brands, product names, service names and trademarks are property of their respective owners.

FabKey — is a trademark of NXP B.V.

PC-bus — logo is a trademark of NXP B.V.

14. List of figures

Fig 1.	System architecture diagram.	3	Fig 31.	Localization of the Visual Studio A71CH projects in the Host Library.	25
Fig 2.	A71CH mini PCB OM3710/A71CHPCB.	5	Fig 32.	Visual Studio 2017. Building the A71CHConfig project.	26
Fig 3.	OM3710/A71CHPCB board schematic.	6	Fig 33.	A71CHConfig_vcom executable file.	26
Fig 4.	OM3710/A71CHPCB board silkscreen.	6	Fig 34.	mainA71CH executable file.	27
Fig 5.	A71CHARD Arduino header.	7	Fig 35.	Host API usage application execution from a Power Shell terminal.	28
Fig 6.	FRDM-K64F Freedom development platform for Kinetis K64, K63 and K24 MCUs.	8	Fig 36.	Host API usage application execution from Visual Studio.	29
Fig 7.	FRDM-K82F Freedom development platform for Kinetis K80, K813 and K82 MCUs.	8	Fig 37.	Client-Server connection using RJCT server.	30
Fig 8.	A71CH mini PCB OM3710/A71CHPCB.	9	Fig 38.	i.MX6UltraLite boot sequence.	30
Fig 9.	A71CH Mini PCB board (OM3710/A71CHPCB) mounting on the I2C adaptor of the Arduino interface board.	9	Fig 39.	Check i.MX6UltraLite IP address.	31
Fig 10.	A71CH Arduino kit mounted on Kinetis FRDM-K64F board.	10	Fig 40.	Launch RJCT server in the i.MX6UltraLite.	31
Fig 11.	Arduino interface board connected to the Kinetis board (FRDM-K64F).	10	Fig 41.	A71CH Configure tool executed in interactive mode.	32
Fig 12.	Red USB indicates OpenSDA serial port. Yellow USB indicates virtual COM connector port (FRDM-K64F).	11	Fig 42.	Connection established with the RJCT server running in the i.MX6UltraLite.	32
Fig 13.	MCUXpresso install wizard.	12	Fig 43.	'info all' command executed from the development PC.	33
Fig 14.	OpenSDA bootloader version for the Kinetis FRDM-K64F.	13	Fig 44.	USB to I ² C dongle and I ² C bus cable.	34
Fig 15.	Desired firmware for the Kinetis FRDM-K64F.	13	Fig 45.	USB to I ² C Bird – A71CH mini PCB interface connection.	34
Fig 16.	Enabling bootloader mode.	14	Fig 46.	System architecture diagram. Detailed view on Windows platform.	35
Fig 17.	Copying the firmware into the Kinetis board in 'BOOTLOADER' mode.	14	Fig 47.	USB to I ² C Ascot adaptor. Drivers installation wizard.	36
Fig 18.	SDKBuilder website.	15	Fig 48.	Java environment variables.	37
Fig 19.	Configuring the SDK to download.	16	Fig 49.	NXPCardServer execution window.	37
Fig 20.	Installing the downloaded Kinetis SDK into MCUXpresso.	16	Fig 50.	NXPCardServer folder contents.	38
Fig 21.	Importing project from SDK.	17	Fig 51.	A71CH Configure tool execution using the USB to I ² C bird.	39
Fig 22.	Importing project from SDK. Board selection page.	17	Fig 52.	Left USB connection is used to flash the program from MCUXpresso IDE. Right USB port connection acts as a virtual COM port.	40
Fig 23.	Importing project from SDK. Select A71CH example projects.	18	Fig 53.	Drivers correctly installed.	40
Fig 24.	Content of 'frdmk64f_projects' folder.	19	Fig 54.	Device not detected correctly.	41
Fig 25.	Import project from file system.	19	Fig 55.	Connection type selection page.	42
Fig 26.	Project directory to import.	20	Fig 56.	Package selection page.	42
Fig 27.	Project directory to import.	21	Fig 57.	Package selection page.	43
Fig 28.	Microsoft Visual Studio 2017.	22	Fig 58.	Running A71CH application examples with Cygwin setup.	43
Fig 29.	Configuration steps to debug using MCUXpresso IDE Console.	23	Fig 59.	Running the downloaded file.	43
Fig 30.	Running A71CH Configure tool system setup (e.g. FRDM-K64F).	24	Fig 60.	Running the build script.	44
			Fig 61.	Files created with the buildA71CH.sh script.	44

Fig 62. Running the A71CH provisioning example using
Cygwin.45

Fig 63. Running the A71CH Configure Tool using
Cygwin.46

Fig 64. Running the A71CH Host API usage example
using Cygwin.....47

15. List of tables

Table 1. Default OM3710/A71CHPCB Jumper settings..5
Table 2. References.....48

16. Contents

1.	Introduction	3	9.	Appendix A: Running A71CH application examples with USB to I²C bird	33
2.	A71CH Overview	3	9.1	USB/I ² C bird (OM3710/B001)	33
3.	System description	3	9.2	Hardware setup	34
4.	Hardware overview	4	9.3	Software setup	35
4.1	A71CH Arduino compatible development kit (OM3710/A71CHARD)	4	9.3.1	USB to I ² C ASCOT adaptor – Drivers	35
4.1.1	A71CH Mini PCB board (OM3710/A71CHPCB)	4	9.3.2	NXPCardServer	36
4.1.2	Arduino interface board	7	9.4	Running A71CH Configure tool	38
4.2	Freedom development platforms for Kinetis	7	9.5	Running A71CH Host API usage example	39
4.2.1	FRDM-K64F	7	10.	Appendix B: VCOM driver installation troubleshooting	39
4.2.2	FRDM-K82F	8	11.	Appendix C: Running examples using Cygwin	41
5.	Hardware setup	8	11.1	Cygwin Installation	41
6.	Software setup	11	11.2	A71CH application examples execution	43
6.1	MCUXpresso IDE installation	11	11.2.1	Running A71CH OpenSSL Engine examples	44
6.2	OpenSDA configuration	12	11.2.2	Running the A71CH Configure Tool	45
6.3	Kinetis SDK package for A71CH	15	11.2.3	Running A71CH Host API usage examples	46
6.4	Importing a project in MCUXpresso IDE	16	12.	References	48
6.4.1	Importing the A71CH example projects from the installed SDK	16	13.	Legal information	49
6.4.2	Importing A71CH example projects from local drive (bundled with installer)	18	13.1	Definitions	49
6.5	Microsoft Visual Studio IDE installation	21	13.2	Disclaimers	49
7.	A71CH application examples execution	22	13.1	Licenses	49
7.1	Set the Kinetis board as virtual COM port	23	13.2	Trademarks	49
7.1.1	Connect the Kinetis board to the Windows platform over USB	24	14.	List of figures	50
7.2	Running A71CH Configure tool	24	15.	List of tables	52
7.3	Running A71CH Host API usage example	27	16.	Contents	53
7.3.1	Run the A71CH Host API usage executable from a terminal	27			
7.3.2	Run the A71CH Host API usage from Visual Studio	28			
8.	RJCT server	29			

Please be aware that important notices concerning this document and the product(s) described herein, have been included in the section 'Legal information'.