

1 Overview

This document describes the:

- steps to set up a Wi-Fi solution using the NXP evaluation boards along with the SX-ULPAN-2401-SHIELD or WIFI-10-CLICK-SHIELD.
- hardware and software requirements.
- steps to connect the boards.
- steps to program and run the demo applications.

Two demo applications are available:

- one is for exercising the common Wi-Fi commands.
- one is a throughput demo.

This document is intended for software engineers, system engineers, and test engineers.

2 Hardware overview

This section describes:

- [Hardware configurations](#)
- [Assembly instructions](#)
- [Additional hardware](#)

2.1 Hardware configurations

The following hardware configurations are currently supported.

- FRDM-K22F plus SX-ULPAN-2401-SHIELD
- FRDM-KL46 plus SX-ULPAN-2401-SHIELD
- FRDM-K64F plus SX-ULPAN-2401-SHIELD
- FRDM-K82F plus SX-ULPAN-2401-SHIELD
- FRDM-KL28Z plus SX-ULPAN-2401-SHIELD
- FRDM-K32W042 plus SX-ULPAN-2401-SHIELD
- EVK-MIMXRT1050 plus SX-ULPAN-2401-SHIELD
- EVK-MIMXRT1060 plus SX-ULPAN-2401-SHIELD
- EVK-LPCXpresso55s69 plus WIFI-10-CLICK-SHIELD

Contents

1 Overview.....	1
2 Hardware overview.....	1
3 Software overview.....	4
4 Running the Wi-Fi shell demo.....	13
5 Running the Wi-Fi throughput demo....	16
6 References.....	19
7 Revision history.....	20



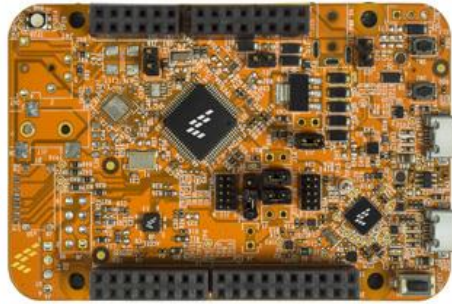


Figure 1. NXP FRDM-K22F board

The SX-ULPAN-2401-SHIELD and WIFI-10-CLICK-SHIELD kits contains the following parts.

- SX-ULPAN-2401 Wi-Fi Module (Soldered on Adapter Board)
- Adapter Board with Freedom compatible headers



Figure 2. Sillex SX-ULPAN-2401 Shield

- WIFI-10-CLICK Shield compatible with mikroBUS™ socket



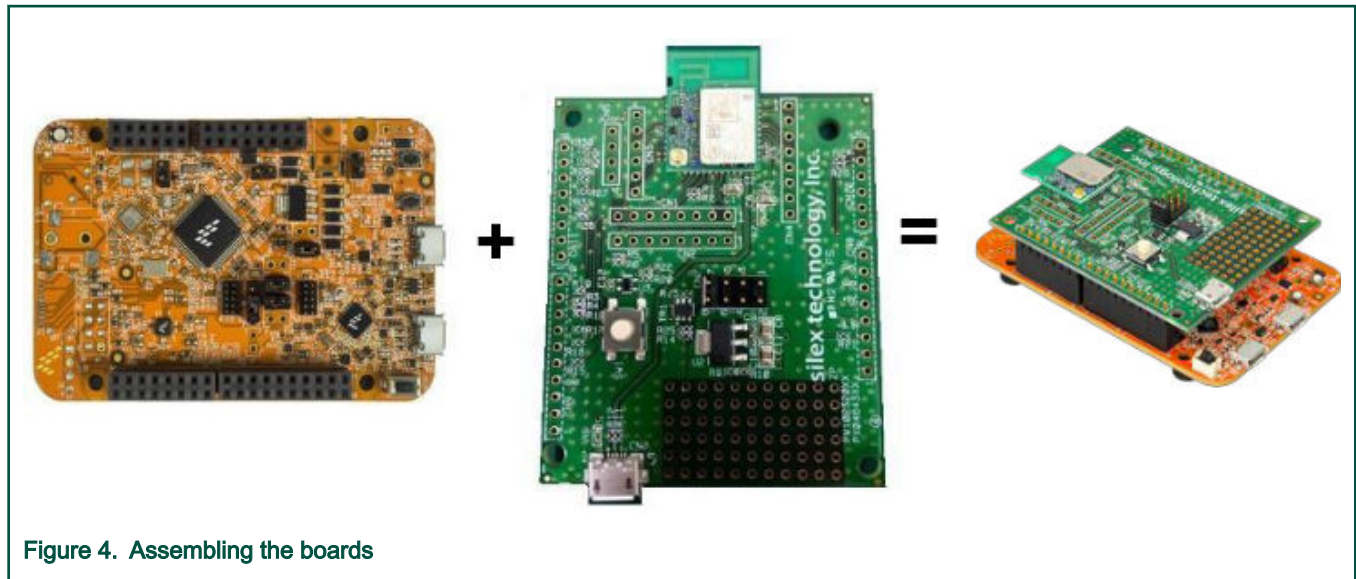
Figure 3. WIFI-10-CLICK Shield

2.2 Assembly instructions

To assemble the boards, plug the Silex/ULPAN shield into the Freedom board, taking the board orientation into consideration. In case of WIFI-10-CLICK just connect to the mikroBUS™ available on board.

NOTE

The USB connectors from the Freedom board must remain visible after the assembly is done.



2.3 Additional hardware

The additional hardware required includes:

- USB A to micro USB cable
- Personal computer

The QCA Wi-Fi application does not call for any special hardware configuration. Although not required, the recommendation is to leave the development board jumper settings and configurations in the default state when running this demo.

3 Software overview

The Freedomevaluation board in use should be programmed with the demo application binary before use. The demo projects are provided in the source code and must be built. The SX-ULPAN module is pre-programmed with the Wi-Fi firmware and no further actions are required. For custom Kinetis development, NXP offers the Kinetis SDK 2.x drivers as the current option.

3.1 Using Kinetis SDK 2.x

This section describes:

- [Software requirements](#)
- [Software installation](#)
- [Using demo application with MCUXpresso IDE](#)
- [Using demo application with IAR](#)
- [Migrating demo software to a custom design](#)

3.1.1 Software requirements

- QCA400x drivers, libraries, and demo applications available on www.nxp.com.
- Kinetis SDK v2.x available on <http://mcuxpresso.nxp.com/en/welcome>.

NOTE

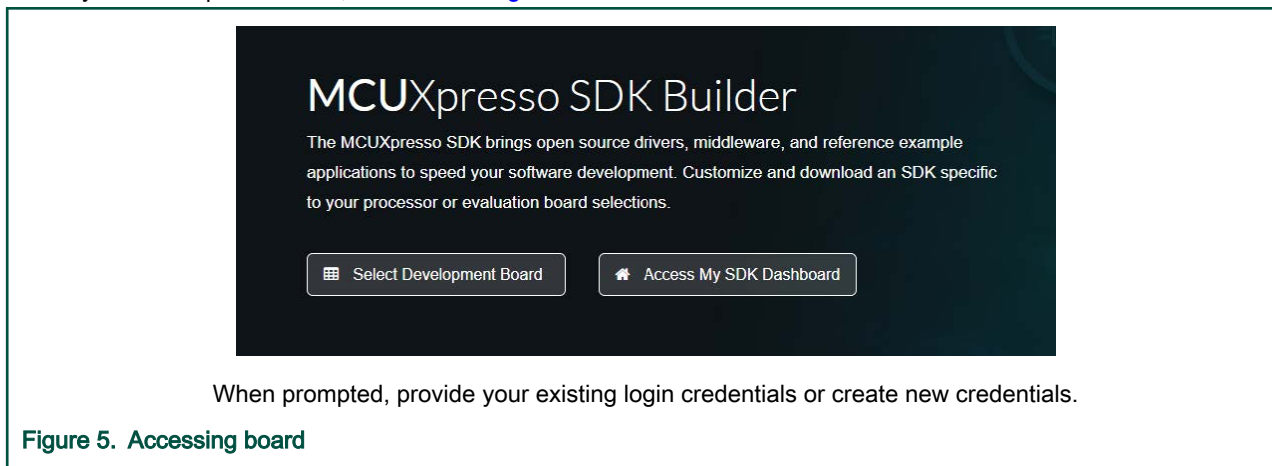
It is recommended to always use the latest version of SDKv2.

- MCUXpresso IDE available on <http://www.nxp.com/mcuxpresso/ide>.
- PC Virtual COM port software, such as TeraTerm, puTTY, and so on.

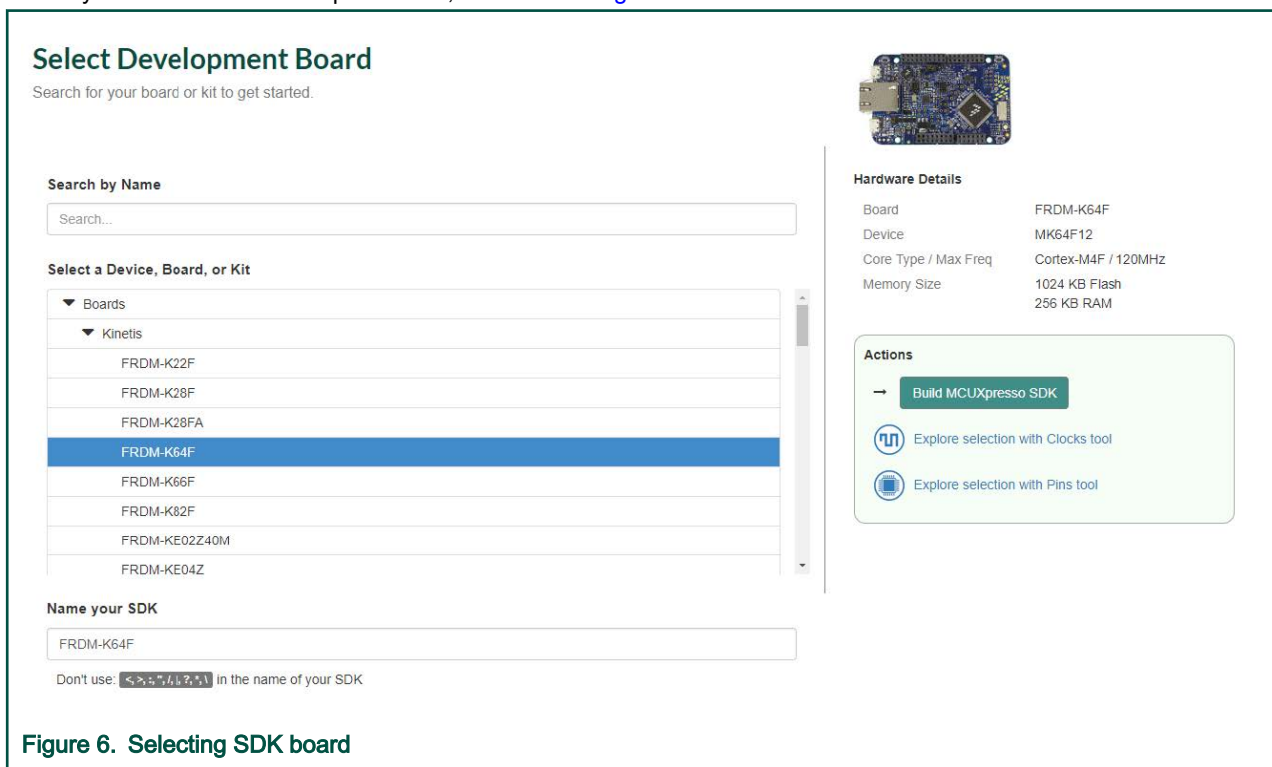
3.1.2 Software installation

To download the software, go to <https://mcuxpresso.nxp.com/en/welcome>.

1. Select your development board, as shown in Figure 5.



2. Select your board from the drop-down list, as shown in Figure 6.



3. Click **Build MCUXpresso SDK**, as shown in Figure 7.

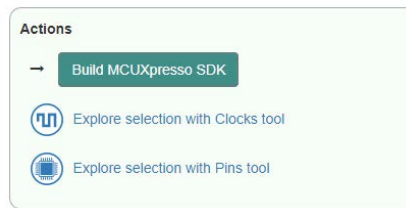


Figure 7. Building MCUXpresso SDK

4. Select your OS and then the preferable toolchain (or all of them), as shown in [Figure 8](#).

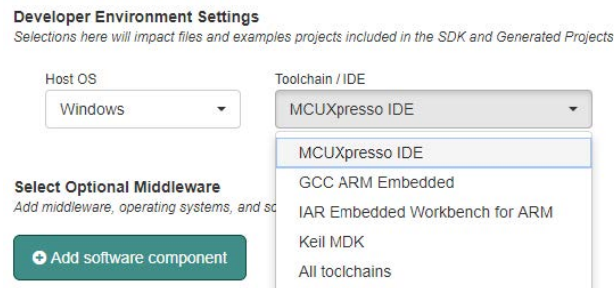


Figure 8. Selecting OS and toolchain

5. To select RTOS and Middleware, click **Add software component**.
6. Select all necessary software components for your project. For QCA400x, it is mandatory to select **Amazon- FreeRTOS Kernel** and **QCA400x WiFi**, as shown in [Figure 9](#).
7. Click **Save changes**.

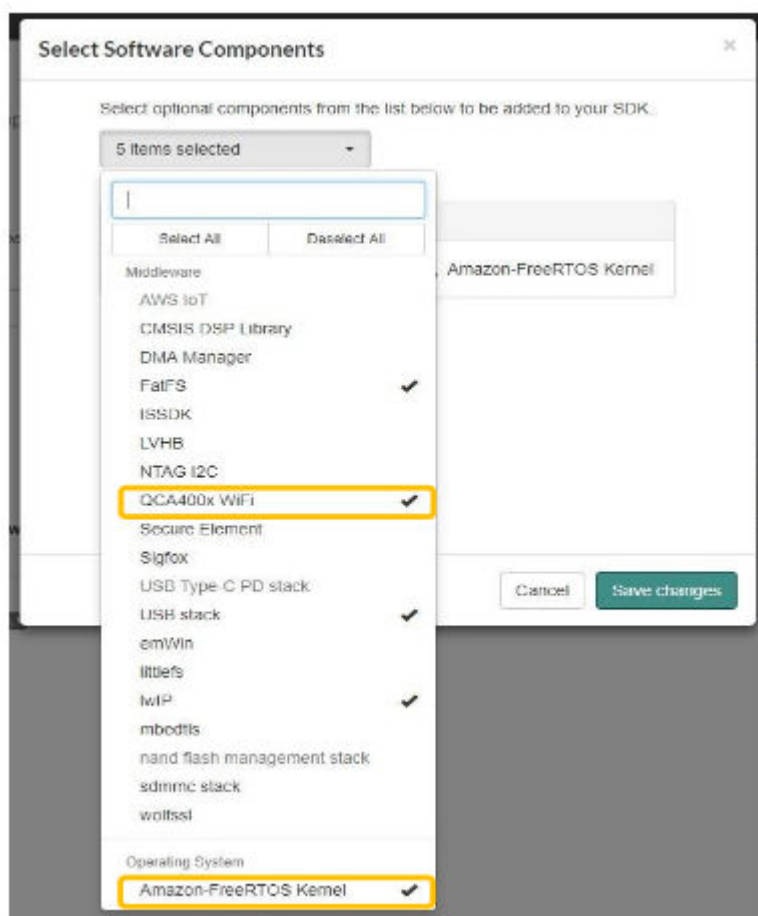


Figure 9. Additional components

8. Click **Request Build** to request a new Build, as shown in Figure 10.

Click the link below to request this specific MCUXpresso SDK Build
 In general, SDK builds should complete within a few minutes.
 You will be notified via email and notifications in the upper right corner of this webpage.

Request Build →

Archive Name

FRDM-K64F

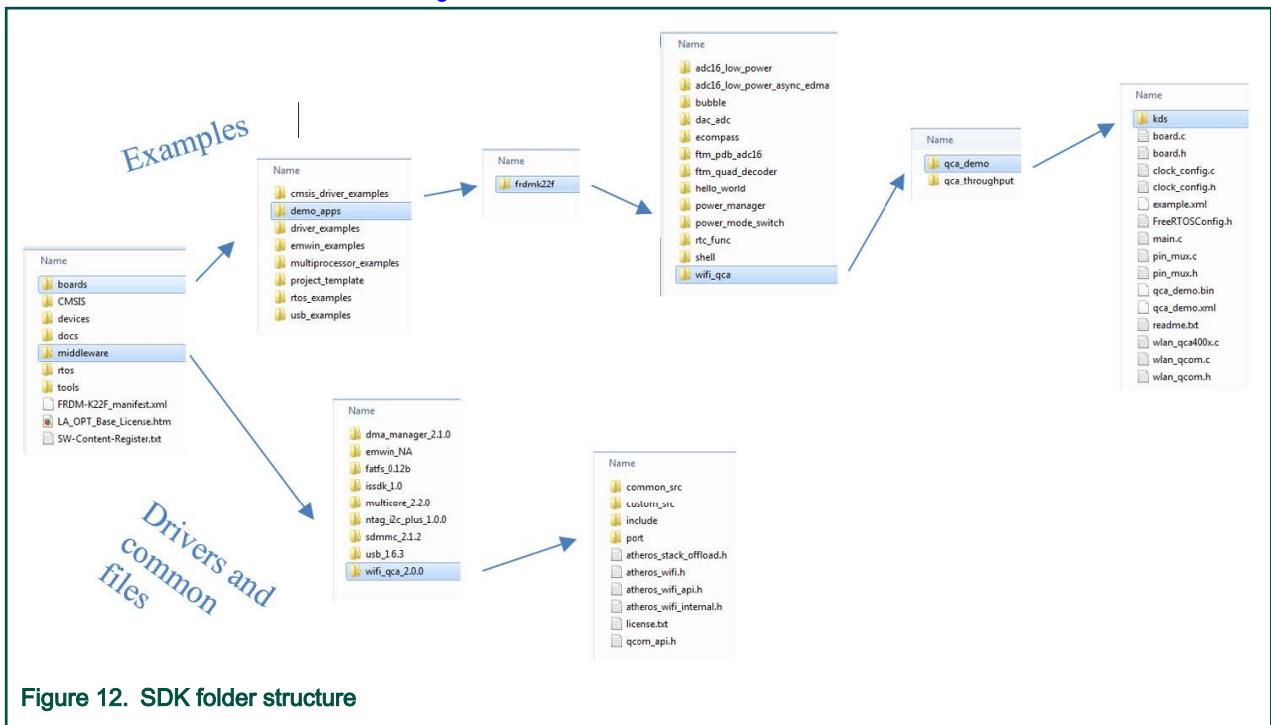
Don't use: <, >, ., -, /, \, ?, *, \ in the name of your SDK

Figure 10. Requesting build

9. You will receive an e-mail message when SDK starts to build and after it ends with a link to download. Another way to download the newly generated SDK is to enter the MCUXpresso SDK Dashboard and click the **Download** icon adjacent to the desired configuration.



10. Download and unzip the file to a local folder. For example, *C:\NXP\SDK_2.2_FRDM-K22F*.
11. Your local file structure is as shown in [Figure 12](#).



12. Downloaded zip file contains example applications in `/boards/demo_apps/<board_name>/wifi_qca` folder and wifi related middleware in `/middleware/wfi_qca_2.0.0` folder. Demo folder contains `board.c/board.h` and `mux.c/pin mux.h`.

As shown in [Figure 12](#), the examples can be accessed in the `wifi_qca` folder. In the `qca_demo` folder, you can access all board-specific files. In these files, you can change the pins, SPI, UART, and other items that are board-dependent. This is the first place you must change to migrate the demo project to your custom board.

In the `wifi_qca_2.0.0` folder, all the drivers and common files included in the demo projects are located. To understand how the demos work and/or add more features or functions to your custom project, navigate through these folders.

3.1.3 Using demo application with MCUXpresso IDE

1. Open MCUXpresso IDE. Drag and drop your FRDM-K64 SDK folder inside the installed SDK area, as shown in [Figure 13](#).

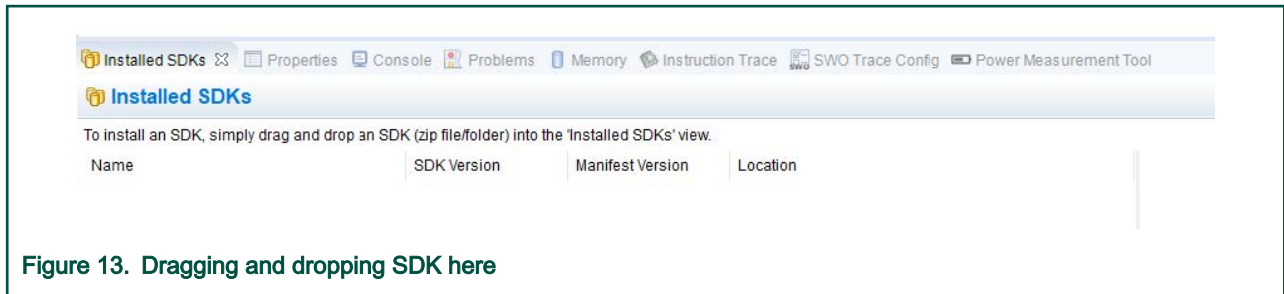
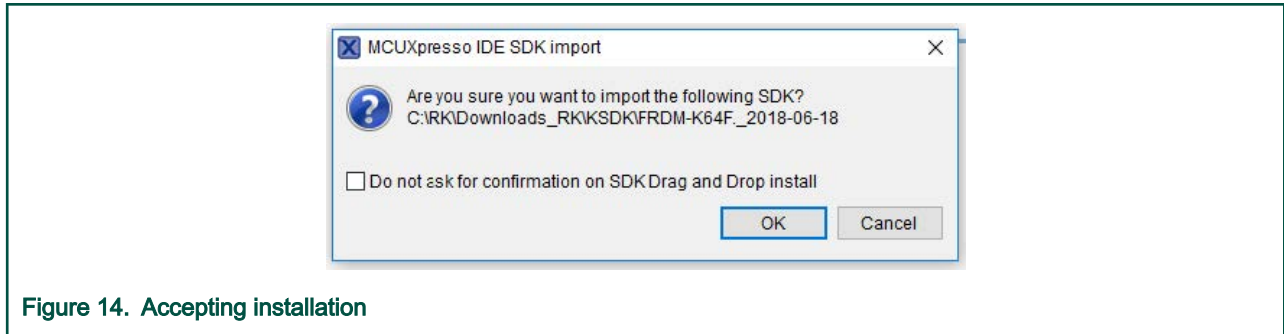


Figure 14 shows the project structure.



2. In the left bottom area of MCUXpresso, click **Import SDK Examples**, as shown in Figure 15.



3. A new window pops up, where you can select the desired board, as shown in Figure 16.

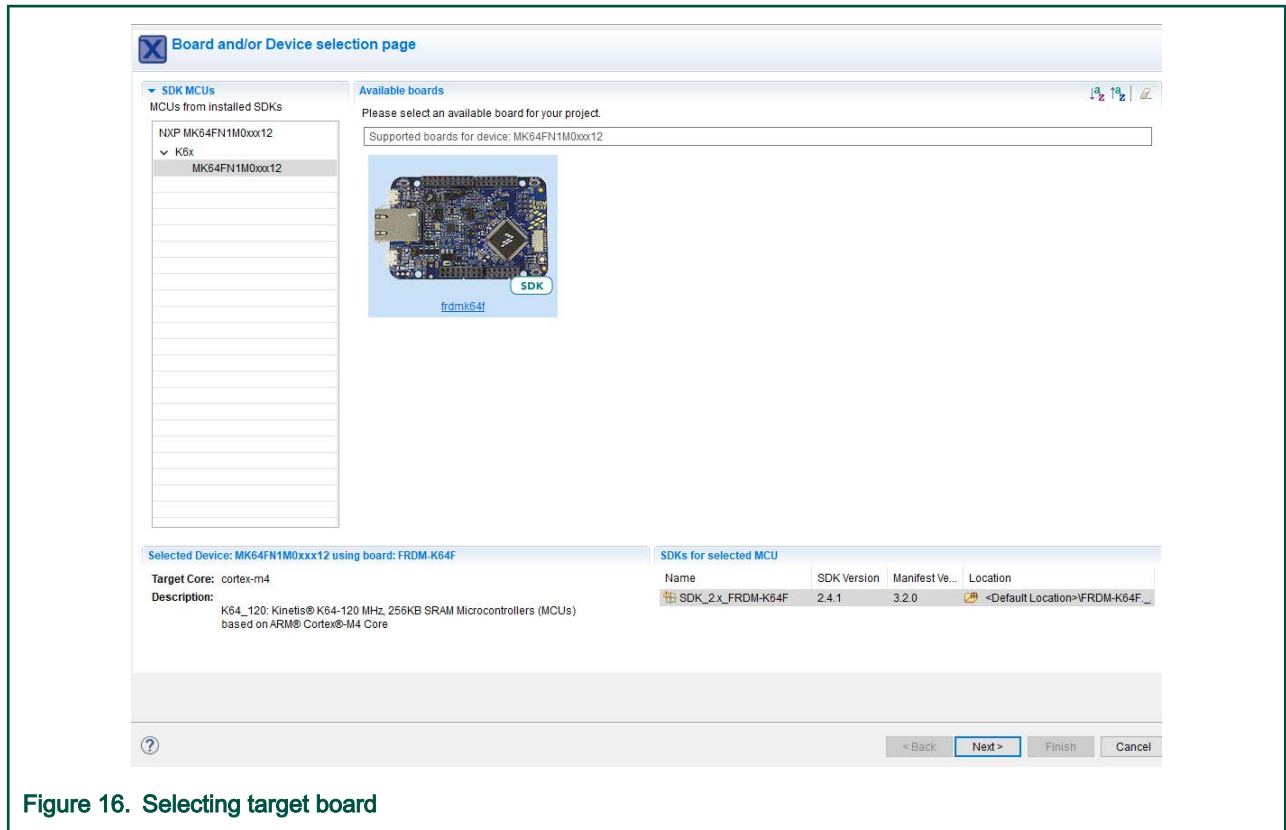


Figure 16. Selecting target board

- Click **Next** and select the examples you want to open in MCUXpresso IDE, as shown in Figure 17.

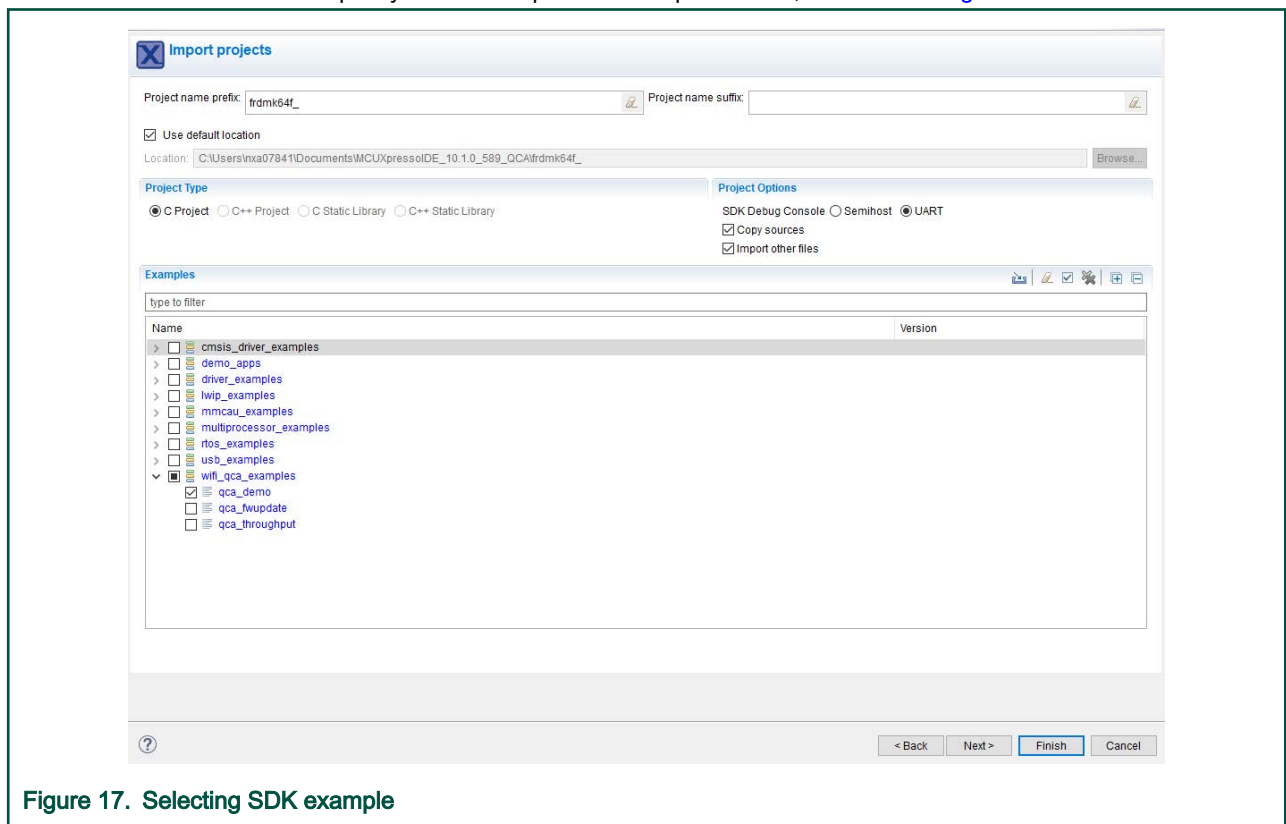


Figure 17. Selecting SDK example

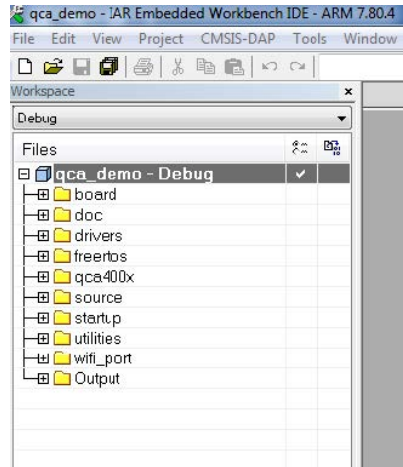


Figure 20. Workspace IAR IDE

In the **Menu**, select **Project - Rebuild All**. After the operation completes, enter **Project** again to select **Download and Debug**. The **debug** window opens in IAR and you can run/stop running the code.

3.1.5 Migrating demo software to a custom design

The demo software is supplied to test the FRDM-K22F and SX-ULPAN.

In *qca_demo*, the *middleware\wifi_qca\port\boards\frdmk64\freertos\silex2401* folder contains all the board-dependent files. The *wifi_shield_silex2401.h* file contains the main settings and it is the first to be reviewed to port and/or debug a custom board. It includes the definitions for the SPI module, correct pin selection for each SPI signal, Power-on Pin (GPIO used to turn the shield on or off), IRQ, and DMA channel. All of them must be assigned correctly.

Also, a session called *BOARD_InitSilex2401Shield* in *pin_mux.h* exists to set every GPIO necessary to use SX-ULPAN shield.

For a new or a custom board, all these files could be cloned and modified.

The GPIO settings are taken from the *pin_mux.h* file generated by the MCUXpresso PinmuxTool. Part of file for Silex2401 configurations.

```
/*FUNCTION*****
*
*   Function Name : BOARD_InitSilex2401Shield
*   Description   : Configures pin routing and optionally pin electrical features.
*
*END*****/ void
BOARD_InitSilex2401Shield(void) {
    CLOCK_EnableClock(kCLOCK_PortB);    /* Port B Clock Gate Control: Clock enabled */

    CLOCK_EnableClock(kCLOCK_PortD);    /* Port D Clock Gate Control: Clock enabled */

    PORT_SetPinMux(PORTB, PIN23_IDX, kPORT_MuxAsGpio);    /* PORTB23 (pin 69) is configured as PTB23 */
    PORTB->PCR[23] = ((PORTB->PCR[23] &
    (~ (PORT_PCR_PE_MASK | PORT_PCR_ISF_MASK)))    /* Mask bits to zero which are setting */
    | PORT_PCR_PE(PER_PCR_PE_ENABLED)    /* Pull Enable: Internal pullup or pulldown resistor is enabled on
    the corresponding pin, if the pin is configured as a digital input. */
    );
    PORT_SetPinMux(PORTB, PIN9_IDX, kPORT_MuxAsGpio);    /* PORTB9 (pin 57) is configured as PTB9 */
    PORTB->PCR[9] = ((PORTB->PCR[9] &
    (~ (PORT_PCR_PS_MASK | PORT_PCR_PE_MASK | PORT_PCR_ISF_MASK)))    /* Mask bits to zero which are setting
```

```

*/
| PORT_PCR_PS(PCR_PS_UP)      /* Pull Select: Internal pullup resistor is enabled on the corresponding
pin, if the corresponding PE field is set. */
| PORT_PCR_PE(PCR_PE_ENABLED)  /* Pull Enable: Internal pullup or pulldown resistor is enabled on
the corresponding pin, if the pin is configured as a digital input. */
);
PORT_SetPinMux(PORTD, PIN0_IDX, kPORT_MuxAlt2); /* PORTD0 (pin 93) is configured as SPI0_PCS0 */
PORT_SetPinMux(PORTD, PIN1_IDX, kPORT_MuxAlt2); /* PORTD1 (pin 94) is configured as SPI0_SCK */
PORT_SetPinMux(PORTD, PIN2_IDX, kPORT_MuxAlt2); /* PORTD2 (pin 95) is configured as SPI0_SOUT */
PORT_SetPinMux(PORTD, PIN3_IDX, kPORT_MuxAlt2); /* PORTD3 (pin 96) is configured as SPI0_SIN */
}

```

Other files are also used to configure the clock gating and functions for the pins. For example, *pin_mux.c*. However, these files are spread out in the KSDK examples. You can select and configure the pins using the MCUXpresso Pin Configuration tool (available at <https://mcuxpresso.nxp.com/>). The necessary signals are:

- SPI MOSI
- SPI MISO
- SPISCK
- SPICS
- WLAN IRQ(GPIO)
- WLAN PWRUP(GPIO)

The WLAN IRQ must be configured to enable the pull-up and it must support the GPIO interrupts. Some of the KL chips do not provide GPIO interrupts for all GPIO ports. The WLAN PWRUP must be set to pulldown.

If using another Freedom board (instead of FRDM-K22F), the information above is still valid. Some MCUs have a slightly different way to configure different peripherals, but the files you must change are the same as described.

For swap between shields or more supported boards (if the same structure is kept), a definition is set in order to select the correct board. In `middleware\wifi_qca\port\boards\frdmk64f\freertos\wifi_shield.h`, a selection between shields can be easily done:

```

/* Select specific shield support */
// #define WIFISHIELD_IS_GT202
#define WIFISHIELD_IS_SILEX2041

```

4 Running the Wi-Fi shell demo

This section describes:

- [Preparing the demo](#)
- [Exercising the console commands](#)

4.1 Preparing the demo

To prepare the demo, follow these steps:

- Connect the FRDM-K22F board to the PC using the USB cable. The USB connector used on the Freedom board is the OpenSDA USB.

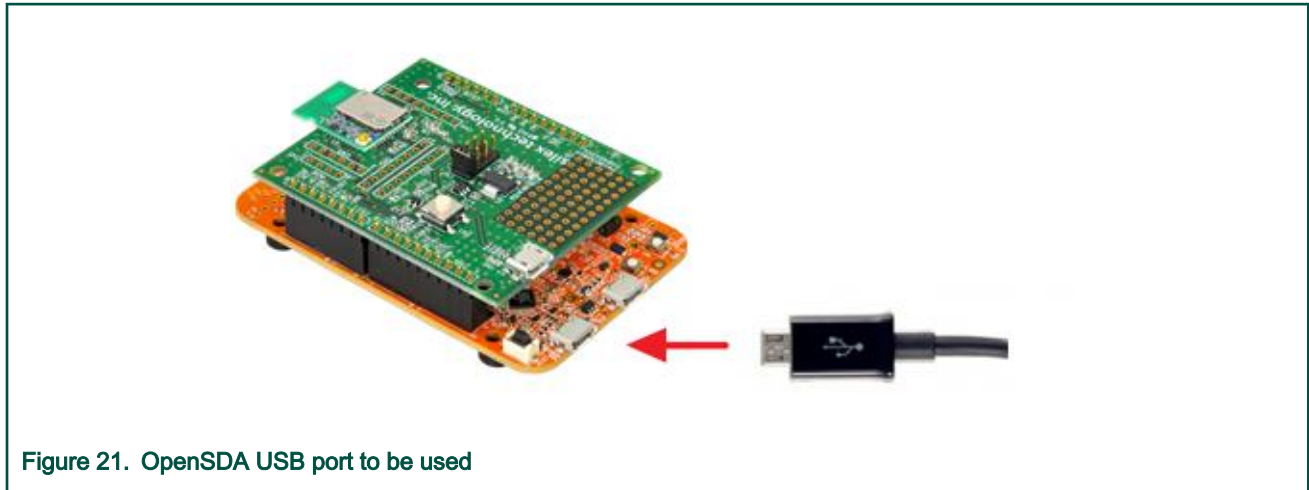


Figure 21. OpenSDA USB port to be used

- Wait for the debug and virtual COM port drivers to install on the PC.
- Within the IAR IDE, start a debug session to program the K22F chip. The debug configuration used depends on the debug interface used on the FRDM-K22F board (for example, Segger J-Link, PEmicro, OpenOCD, mbed CMSIS-DAP, and so on).
- After programming, terminate the debug session. The board is ready to be used.
- Open the serial COM port application on the PC (TeraTerm, puTTY) and configure the communication parameters for the port that corresponds to FRDM-K22F (available in the Device Manager): 115200 baud, 8N1, no flow control.
- Reset the FRDM-K22Fboard.
- The demo application starts and the terminal application shows the version information:

```
Host version: 3.3.0.0
Target version: 0x31c80997
Firmware version: 3.3.4.91
Interface version: 1
```

- The menu is displayed (it is displayed again each time the help is called by pressing the h key):

```
s AP Scan
c AP Connect (SSID='nxp', pass='NXP0123456789')
D AP Disconnect
d Get DHCP address
g HTTP GET nxp.com
w HTTP GET from gateway
p Ping gateway
P Ping nxp.com
i Print IP configuration
R Resolve some hosts
h Help (print this menu)
H Print extended help
```

4.2 Exercising the console commands

4.2.1 Connecting AP

Press **c** to connect to an AP with SSID=nxp and pass=NXP0123456789.

```
Key 'c': AP Connect (SSID='nxp', pass='NXP0123456789')
Reading connection params
opMode=0 (Station)
phyMode=mixed
ssid=nxp
EVENT: CLIENT connected
EVENT: 4 way handshake success for device=0
EVENT: CLIENT connected
EVENT: 4 way handshake success for device=0
```

4.2.2 Disconnecting AP

Press **D** to disconnect AP.

```
Key 'D': AP Disconnect
EVENT: CLIENT disconnect
```

4.2.3 Getting DHCP address

Press **d** after the connection to get and display the address assigned.

```
Getting DHCP address...
EVENT: CLIENT connected
EVENT: 4 way handshake success for device=0
DNS 0: 192.168.43.1
addr: 192.168.10.81 mask: 255.255.255.0 gw: 192.168.10.1
```

4.2.4 Getting HTTP address

Press **g** to get HTTP address from <http://www.nxp.com>.

```
Key 'g': HTTP GET www.nxp.com
*****
Looked up www.nxp.com as 104.80.15.112
HTTP GET from 104.80.15.112:80
GET / HTTP/1.0
User-Agent: Mozilla/5.0 (Windows NT 6.1; WOW64; rv:50.0) Gecko/20100101 Firefox/50.0
Accept-Language: en-us
Host: www.nxp.com
TCP sent 148 bytes
Waiting for response (with t_select)
qcom_rcv() receiving response
TCP received 461 bytes
(...)
```

4.2.5 Resolving hosts

Press **R** to get some host name resolved by dns.

```
Key 'R': Resolve some hosts
Looked up google.com as 172.217.29.110
```



```
Looked up cr.yp.to as 131.193.32.109
Looked up kernel.org as 198.145.29.83
Looked up www.nxp.com as 104.80.15.112
```

4.2.6 Pinging gateway

Press **p** to ping the AP.

```
Key 'p': Ping gateway
Pinging 192.168.10.1... OK (0 ms)
```

4.2.7 Pinging *nxp.com*

Press **P** to ping <http://www.nxp.com>.

```
Key 'P': Ping nxp.com
Looked up www.nxp.com as 23.44.183.148
Pinging 23.44.183.148... OK (20 ms)
```

4.2.8 Printing IP configurations

Press **i** to display the IP configurations.

```
Key 'i': Print IP configuration
addr: 192.168.10.81 mask: 255.255.255.0 gw: 192.168.10.1
```

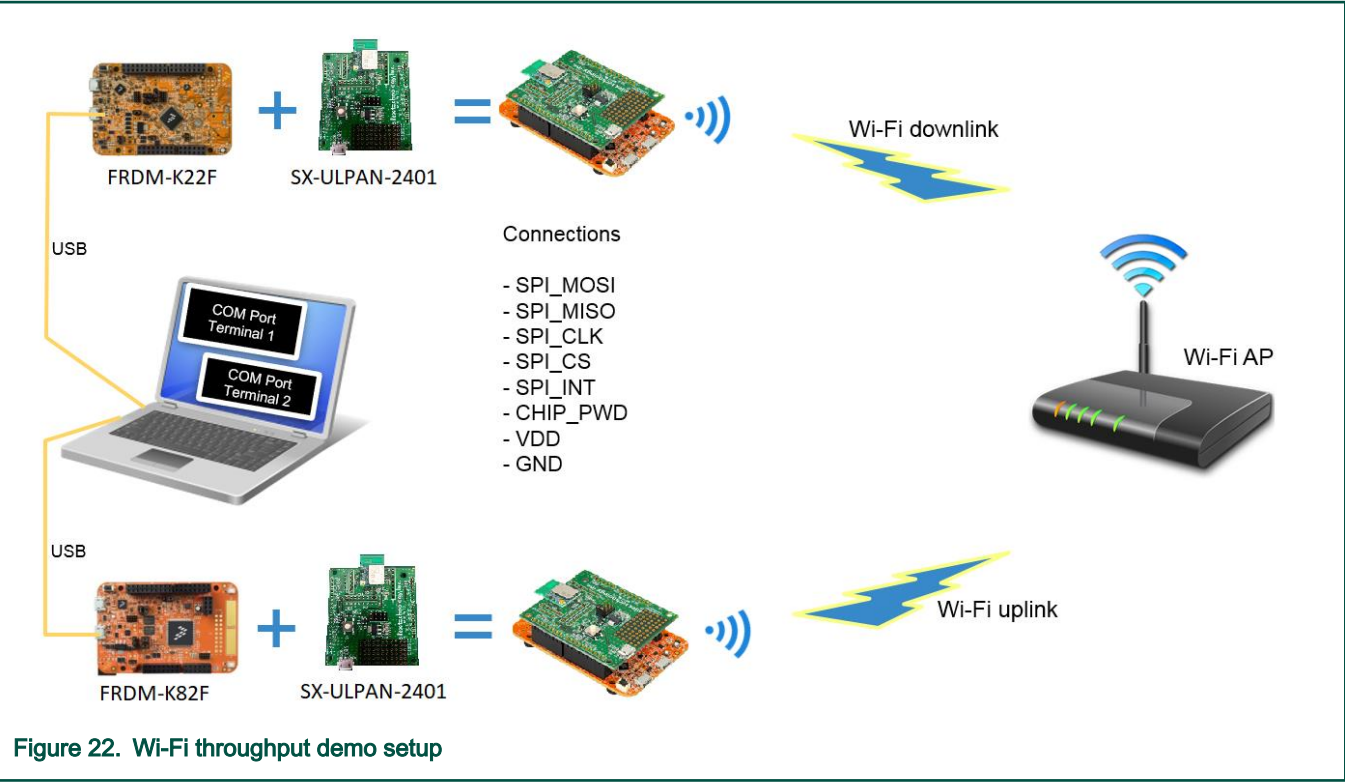
4.2.9 Resolving some hosts

Press **R** to resolve and display some (hardcoded) hosts.

```
Key 'R': Resolve some hosts
Looked up google.com as 216.58.209.110
Looked up cr.yp.to as 131.155.70.11
Looked up kernel.org as 199.204.44.194
Looked up nxp.com as 192.88.156.33
```

5 Running the Wi-Fi throughput demo

The throughput demo requires two FRDM+SX-ULPAN setups, not necessarily identical. Both setups connect to the same Access Point (AP) and the data are passed from one device to the other. The setup is as shown in [Figure 22](#).



The throughput example files are located at (for example, for FRDM-K22F).

```
<SDK_Install>\boards\frdmk22fdemo_apps\wifi_qca\qca_throughput\
```

If using similar setups, both Freedom boards must be programmed with the same firmware, otherwise a different example project (for example, for FRDM-K82F) must be added and built in the KDSK workspace.

When the two Freedom boards are programmed, the throughput test can be performed. To do this, the Freedom boards must be connected to the USB ports on a PC/laptop and two serial COM port terminal applications must be opened. Each terminal application connects the PC/laptop to one Freedom board.

Table 1 and Table 2 list the commands that must be executed on each device, always starting with Device 1 which is the TCP listener.

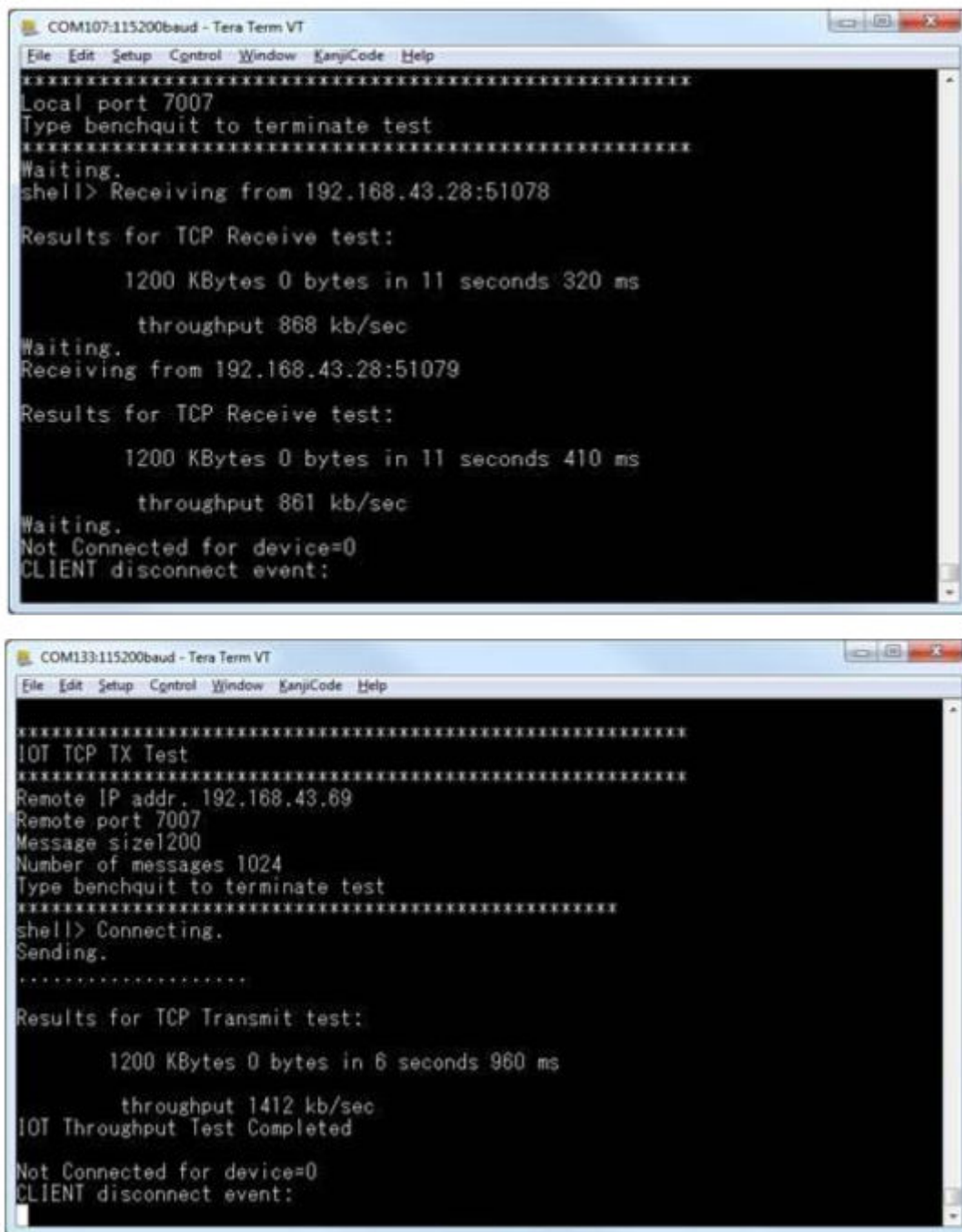
Table 1. Device 1

Command	Description
wmiconfig - p freescale	To provide the AP password.
wmiconfig - wpa 2 CCMP CCMP	To set up the security and encryption protocol.
wmiconfig - connect GL-iNet-d2d	To connect to AP (here called GL-iNet-d2d).
wmiconfig - ipdhcp	To ask the AP for IP address via DHCP.
ipconfig	To check the IP provided by the AP.
benchmode v4	To set up bench mode (Internet v4).
>benchr tcp 7007	To start listening on TCP port 7007.

Table 2. Device 2

Command	Description
<code>wmiconfig - p freescale</code>	To provide the AP password.
<code>wmiconfig - wpa 2 CCMP CCMP</code>	To set up the security and encryption protocol.
<code>wmiconfig - connect GL-iNet-d2d</code>	To connect to AP (here called GL-iNet-d2d).
<code>wmiconfig - ipdhcp</code>	To ask the AP for IP address via DHCP.
<code>ipconfig</code>	To check the IP provided by the AP.
<code>benchmode v4</code>	To set up bench mode (Internet v4).
<code>benchtx 192.168.8.145 7007 tcp 1024 1 2000 0</code>	To start the transmission speed test using Device 1 IP address and port number.

After the throughput test is completed, both shells display the connection speed, as shown in [Figure 23](#).



The figure consists of two screenshots of a Tera Term VT console window. The top window, titled 'COM107:115200baud - Tera Term VT', shows the results of a TCP Receive test. It displays the local port as 7007 and shows two successful receive operations from 192.168.43.28:51078 and 192.168.43.28:51079, both achieving a throughput of approximately 860-868 kb/sec. The bottom window, titled 'COM133:115200baud - Tera Term VT', shows the results of a TCP Transmit test. It displays the remote IP as 192.168.43.69 and remote port as 7007, with a message size of 1200 and 1024 messages. The transmit test achieved a throughput of 1412 kb/sec. Both windows end with a 'CLIENT disconnect event' message.

```
COM107:115200baud - Tera Term VT
File Edit Setup Control Window KanjiCode Help
*****
Local port 7007
Type benchquit to terminate test
*****
Waiting.
shell> Receiving from 192.168.43.28:51078

Results for TCP Receive test:

    1200 KBytes 0 bytes in 11 seconds 320 ms

    throughput 868 kb/sec
Waiting.
Receiving from 192.168.43.28:51079

Results for TCP Receive test:

    1200 KBytes 0 bytes in 11 seconds 410 ms

    throughput 861 kb/sec
Waiting.
Not Connected for device=0
CLIENT disconnect event:

COM133:115200baud - Tera Term VT
File Edit Setup Control Window KanjiCode Help
*****
IOT TCP TX Test
*****
Remote IP addr. 192.168.43.69
Remote port 7007
Message size 1200
Number of messages 1024
Type benchquit to terminate test
*****
shell> Connecting.
Sending.
.....

Results for TCP Transmit test:

    1200 KBytes 0 bytes in 6 seconds 960 ms

    throughput 1412 kb/sec
IOT Throughput Test Completed

Not Connected for device=0
CLIENT disconnect event:
```

Figure 23. Throughput result displayed on consoles output

6 References

- References to FRDM boards are available on NXP website:
<http://www.nxp.com/products/software-and-tools/hardware-development-tools/freedom-development-boards:FREDEVPLA?tid=vanFREEDOM>

- References to SX-ULPAN-2401-SHIELD and SX-ULPAN-2401-SHIELD(US) are in Silex Website:
<https://www.silextechnology.com/connectivity-solutions/embedded-wireless/sx-ulpan-shield>
- Full documentation can be downloaded after a quick registration:
<https://www.silextechnology.com/productspecs/sx-ulpan-2401-shield-product-specifications>
- References to GT-202 boards are available on Qualcomm and Arrow website:
 - <https://developer.qualcomm.com/hardware/qca4002-4?fsrch=1&sr=1&pageNum=1>
 - <https://www.arrow.com/en/products/search?q=GT202>

7 Revision history

Table 3. Revision history

Rev.	Date	Substantive change(s)
0	07/2018	Initial revision based on FRDMGT202QSG
1	09/2019	Added WIFI-10-CLICK-SHIELD

How To Reach Us

Home Page:

nxp.com

Web Support:

nxp.com/support

Information in this document is provided solely to enable system and software implementers to use NXP products. There are no express or implied copyright licenses granted hereunder to design or fabricate any integrated circuits based on the information in this document. NXP reserves the right to make changes without further notice to any products herein.

NXP makes no warranty, representation, or guarantee regarding the suitability of its products for any particular purpose, nor does NXP assume any liability arising out of the application or use of any product or circuit, and specifically disclaims any and all liability, including without limitation consequential or incidental damages. "Typical" parameters that may be provided in NXP data sheets and/or specifications can and do vary in different applications, and actual performance may vary over time. All operating parameters, including "typicals," must be validated for each customer application by customer's technical experts. NXP does not convey any license under its patent rights nor the rights of others. NXP sells products pursuant to standard terms and conditions of sale, which can be found at the following address: nxp.com/SalesTermsandConditions.

While NXP has implemented advanced security features, all products may be subject to unidentified vulnerabilities. Customers are responsible for the design and operation of their applications and products to reduce the effect of these vulnerabilities on customer's applications and products, and NXP accepts no liability for any vulnerability that is discovered. Customers should implement appropriate design and operating safeguards to minimize the risks associated with their applications and products.

NXP, the NXP logo, NXP SECURE CONNECTIONS FOR A SMARTER WORLD, COOLFLUX, EMBRACE, GREENCHIP, HITAG, I2C BUS, ICODE, JCOP, LIFE VIBES, MIFARE, MIFARE CLASSIC, MIFARE DESFire, MIFARE PLUS, MIFARE FLEX, MANTIS, MIFARE ULTRALIGHT, MIFARE4MOBILE, MIGLO, NTAG, ROADLINK, SMARTLX, SMARTMX, STARPLUG, TOPFET, TRENCHMOS, UCODE, Freescale, the Freescale logo, Altivec, C-5, CodeTEST, CodeWarrior, ColdFire, ColdFire+, C-Ware, the Energy Efficient Solutions logo, Kinetis, Layerscape, MagniV, mobileGT, PEG, PowerQUICC, Processor Expert, QorIQ, QorIQ Qonverge, Ready Play, SafeAssure, the SafeAssure logo, StarCore, Symphony, VortiQa, Vybrid, Airfast, BeeKit, BeeStack, CoreNet, Flexis, MXC, Platform in a Package, QUICC Engine, SMARTMOS, Tower, TurboLink, and UMEMS are trademarks of NXP B.V. All other product or service names are the property of their respective owners. AMBA, Arm, Arm7, Arm7TDMI, Arm9, Arm11, Artisan, big.LITTLE, Cordio, CoreLink, CoreSight, Cortex, DesignStart, DynamIQ, Jazelle, Keil, Mali, Mbed, Mbed Enabled, NEON, POP, RealView, SecurCore, Socrates, Thumb, TrustZone, ULINK, ULINK2, ULINK-ME, ULINK-PLUS, ULINKpro, μ Vision, Versatile are trademarks or registered trademarks of Arm Limited (or its subsidiaries) in the US and/or elsewhere. The related technology may be protected by any or all of patents, copyrights, designs and trade secrets. All rights reserved. Oracle and Java are registered trademarks of Oracle and/or its affiliates. The Power Architecture and Power.org word marks and the Power and Power.org logos and related marks are trademarks and service marks licensed by Power.org.

© NXP B.V. 2018-2019.

All rights reserved.

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

For sales office addresses, please send an email to: salesaddresses@nxp.com

Date of release: 12/2019

Document identifier: FRDMQCA400xQSG