NXP FRDM and GT202 Wi-Fi Solutions

1. Overview

This guide describes how to setup a Wi-Fi solution using the Freedom boards along with the GT202 Wi-Fi shield. The guide describes the required hardware and software, how to connect the boards, and how to program and run the demo applications.

There are two demo applications available; one for exercising the common Wi-Fi commands and the second one is a throughput demo.

This document is intended to be used by software engineers, system engineers, and test engineers.

Contents

2. Hardware overview

2.1. Hardware configurations

These hardware configurations are currently supported:

- FRDM-K22F plus GT202 shield
- FRDM-KL46 plus GT202 shield
- FRDM-K64F plus GT202 shield
- FRDM-K82F plus GT202 shield
- FRDM-KL28Z plus GT202 shield
- FRDM-K32W042 plus GT202 shield
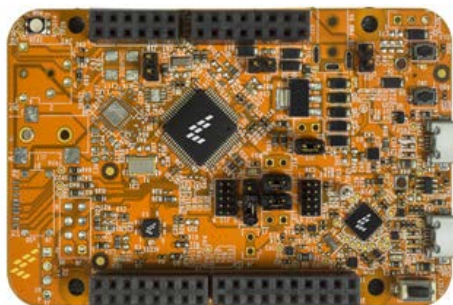- EVK-MIMXRT1020 plus GT202 shield

Figure 1. NXP FRDM-K22F board

The GT202 kit contains these parts:

- GT202 Wi-Fi module (soldered on the Freedom adapter).
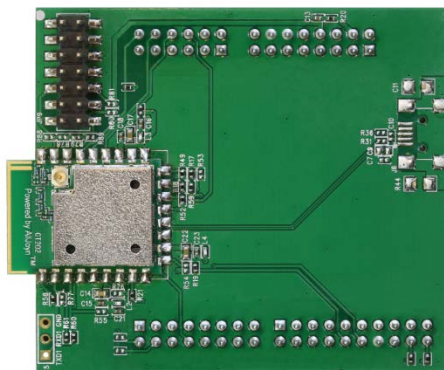- Freedom adapter.

Figure 2. GT202 shield

2.2. Assembly instructions

The assembly procedure is simple. Just plug the GT202 shield into the Freedom board, taking the board orientation into consideration. The USB connectors from the Freedom board must remain visible after the assembly is done.
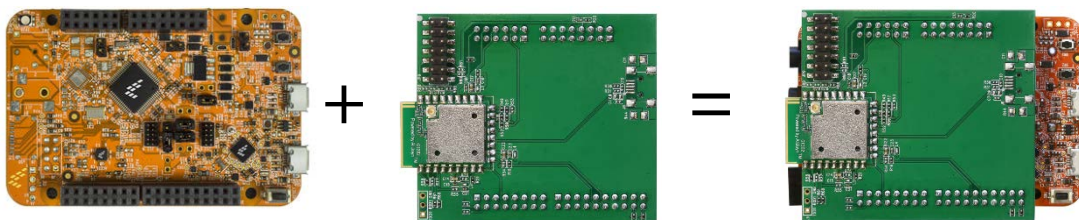

Figure 3. Assembling the boards

If the Freedom board to be used is the FRDM-K64F with a revision prior to E1, some workarounds are required to make the setup run. The next chapters describe these workarounds.

2.3. FRDM-K64F board errata

2.3.1. Early versions

As described in the *FRDM-K64F Board Errata* (document FRDMK64F_ERRATA) chapter 2, the PTA0 GPIO pin routed to Arduino J2.2 (pin 2 of J2) conflicts with the SWD_CLK input from the OpenSDA circuit. This makes it impossible to use J2.2 (PTA0) as WLAN_CHIP_POWER_PIN. For this reason, it is recommended to use PTC4 GPIO as WLAN_CHIP_POWER_PIN. This is the easiest solution because the PTC4 GPIO pin is routed to the Arduino connector J2.4, which is directly adjacent to J2.2.

Perform the following cut and soldering to make it work:

1. Cut off the J2.2 pin on the GT202.
2. On the backside of the GT202 (the side with the mini-USB connector and the QCA400x module), short the J2.2 and J2.4 pins with a short wire.

2.3.2. Later versions (Rev E and newer)

In the later versions of FRDM-K64F, the Arduino J2.2 conflict is fixed. Instead of PTA0, PTC12 is connected to J2.2 and this signal must be used as WLAN_CHIP_POWER_PIN. For this version, no

hardware modification is necessary. The only change is to use PTC12 GPIO as WLAN_CHIP_POWER_PIN.

2.4. Additional hardware

Besides the modules described above, the following hardware is also necessary:

- USB A to mini/micro USB cable.
- Personal computer.

The QCA WIFI 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 Freedom board to be used must be programmed with the demo application binary before use. The demo projects are provided in the source code and must be built. The GT202 module is pre-programed with the Wi-Fi firmware and no further actions are required. For custom Kinetis development, NXP offers the Kinetis SDK 2.0 drivers as the current option.

3.1. Using Kinetis SDK 2.x

3.1.1. Software requirements

- GT202 drivers and demo applications (available on www.nxp.com)
- Kinetis SDK v2.x (available on mcuxpresso.nxp.com)
- It is recommended to always use the latest version of SDK v2
- Kinetis Design Studio IDE v3.2.0 (available on www.nxp.com)
- PC Virtual COM port software (TeraTerm, puTTY, and other)

3.1.2. Software installation

To get the right software, visit mcuxpresso.nxp.com, use your login and password, and follow these steps:
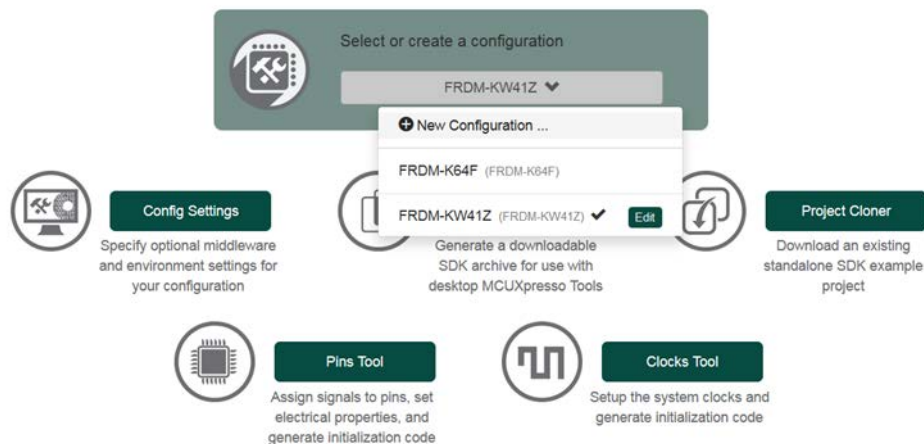
1. Select your board or MCU:

Figure 4. SDK Board selection

Click on "New Configuration" and select the board you want to use. If your board appears in the list, just select it (as shown in the following figures).
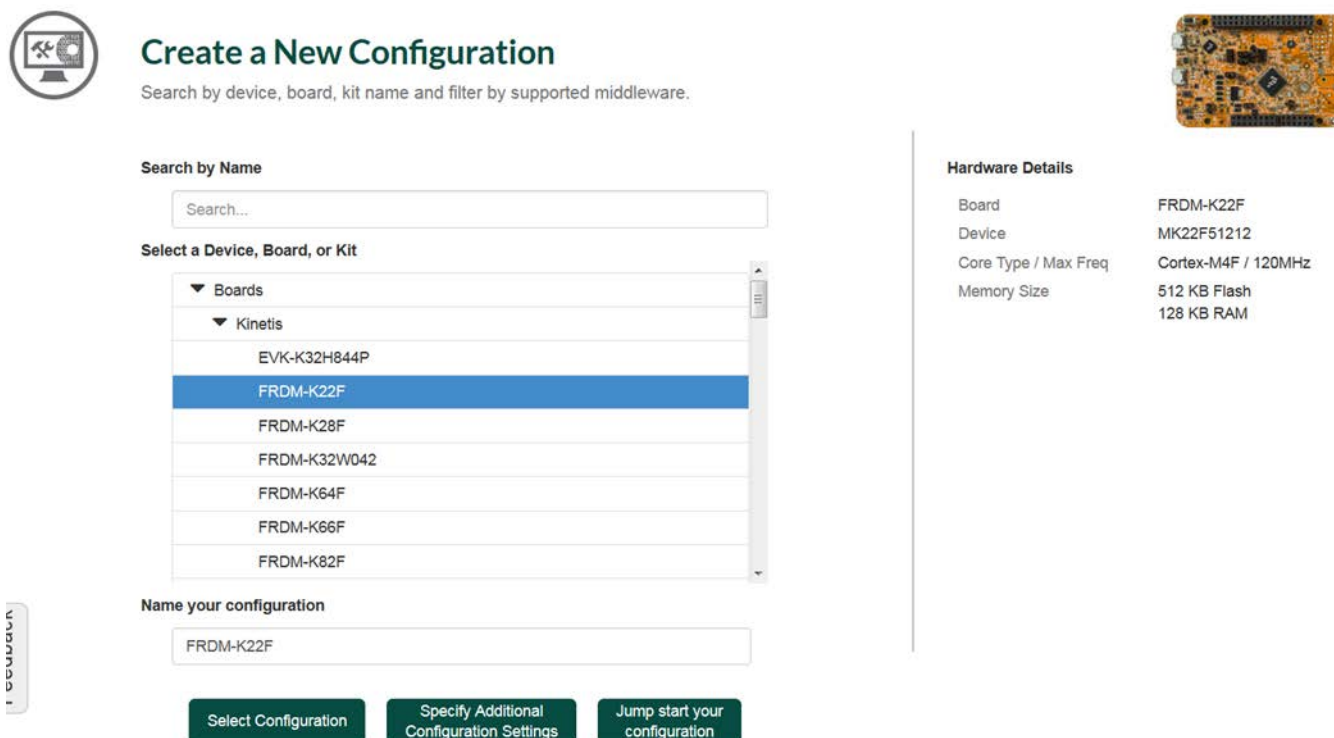

Figure 5. Create New Configuration

NXP FRDM and GT202 Wi-Fi Solutions, Quick Start Guide, Rev. 3.0, 06/2018

2. Click on **Specify Additional Configuration Settings**, select the "wifi_qca" drivers, and set the support to FreeRTOS. If you want to select all components at once, it is easier to include them in your project later.
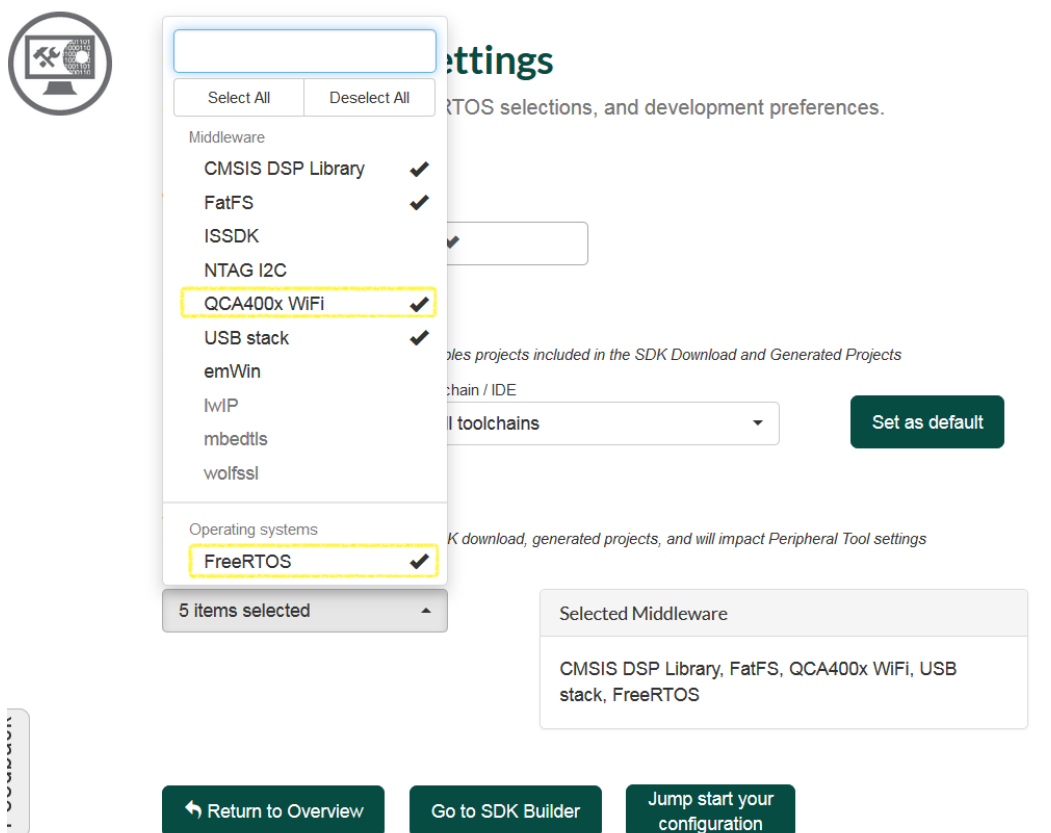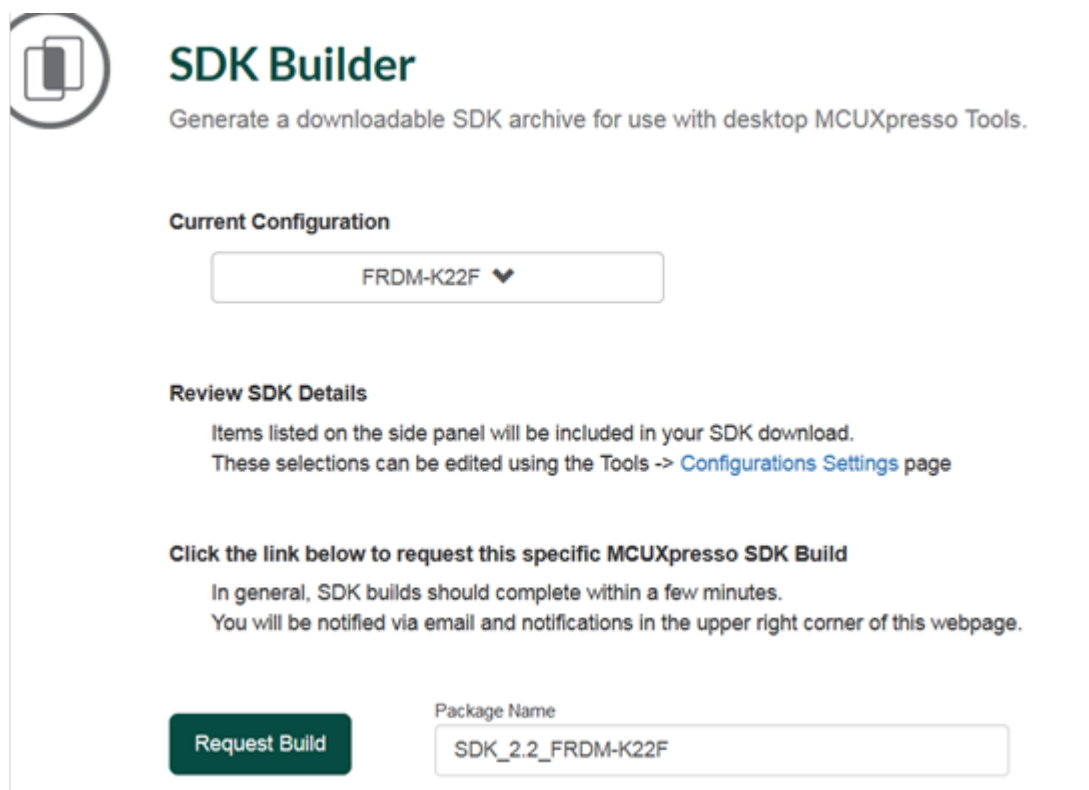

Figure 6. SDK additional items

3. Select `Go to SDK Builder` and then click on Request Build, as in image:

Figure 7. SDK Builder

4. A message will appear as soon as request process starts:

5. You should receive an e-mail when SDK starts to build and one after it ends with a link to Download. Another way to download the new generated SDK is enter in SDK Archive and Click on Download icon of desired configuration

Figure 8. SDK Archive

Figure 9. Downloading SDK

6. Download and unzip the file to a local folder (e.g. C:\NXP\SDK_2.2_FRDM-K22F)

7. Your local file structure will be like:
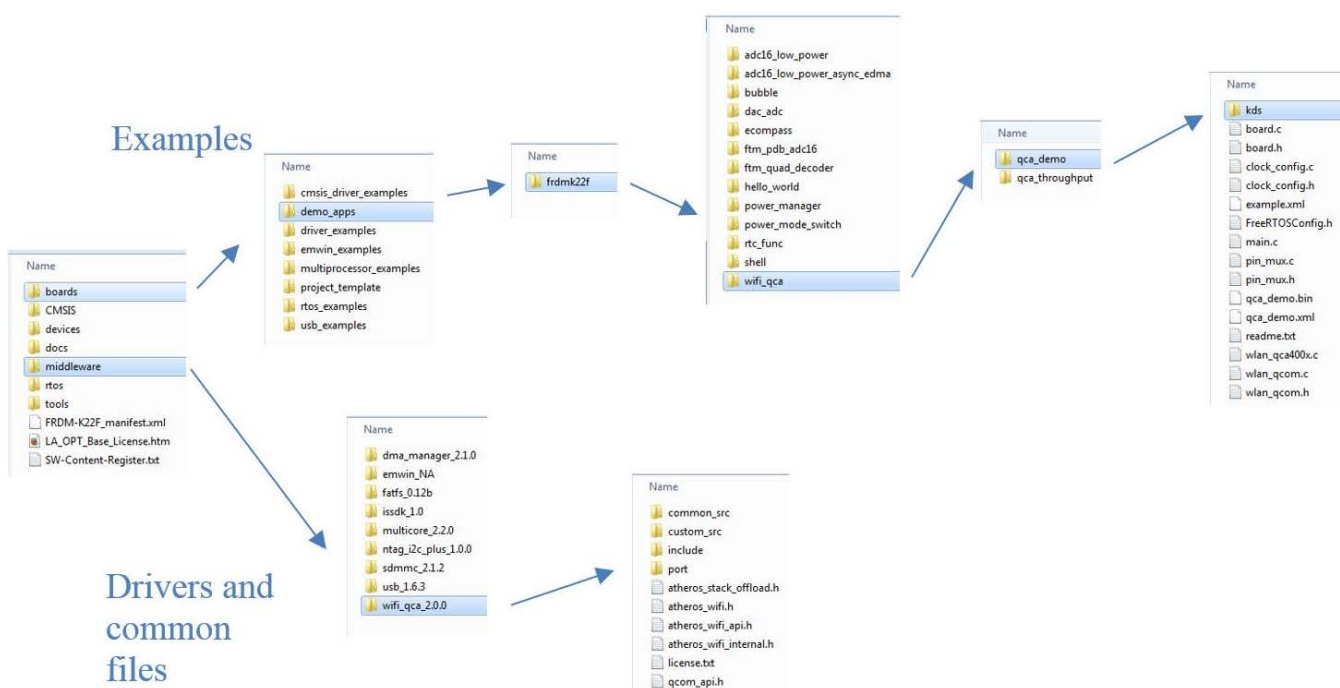
Figure 10.   SDK2 Folder Structure

As shown in the above figure, 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. Look at these folders to understand how the demos work and/or add more features or functions to your custom project.

3.1.3. Using demo application with IAR IDE

In the IAR IDE, open the *qca_demo.eww* file. It imports the *qca_demo* project files and configuration. The workspace looks like this:
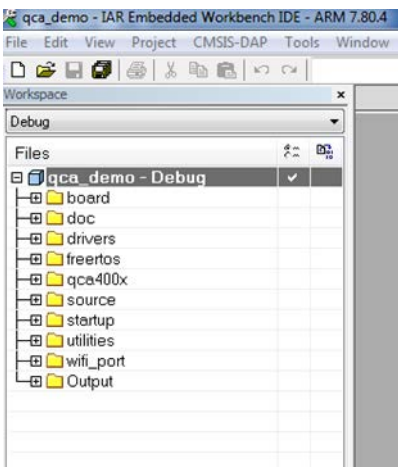

Figure 15. IAR IDE workspace

Select the "Project -> Rebuild All" menu. After the rebuild is complete, select "Project" and "Download and Debug". It opens the debug window in the IAR IDE and you can run/step run the code.

3.1.4. Migrating demo software to your custom design

The demo software is supplied to test the FRDM-K22F and GT202 boards and as a quick reference during the migration to the final product. There are many modifications that can be done, but the I/O changes are usually the first activities to be done.

In *qca_demo*, the *middleware\wifi_qca\port\boards\frdmk22f\freertos\gt202* folder contains all the board-dependent files. The *wifi_shield_gt202.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 GT202 board on or off), IRQ, and DMA channel. All of them must be assigned correctly.

The GPIO settings are taken from the *pin_mux.h* file generated by the MCUXpresso PinmuxTool.

```
/* Pinmux function, generated by pinmuxtool */
#define WIFISHIELD_PINMUX_INIT BOARD_InitGT202Shield

/* WLAN_IRQ signal */
#define WIFISHIELD_WLAN_IRQn (PORTC_IRQn)
#define WIFISHIELD_WLAN_ISR PORTC_IRQHandler
#define WIFISHIELD_WLAN_IRQ_DIRECTION (BOARD_INITGT202SHIELD_IRQ_DIRECTION)
#define WIFISHIELD_WLAN_IRQ_PORT (BOARD_INITGT202SHIELD_IRQ_PORT)
#define WIFISHIELD_WLAN_IRQ_GPIO (BOARD_INITGT202SHIELD_IRQ_GPIO)
#define WIFISHIELD_WLAN_IRQ_PIN (BOARD_INITGT202SHIELD_IRQ_GPIO_PIN)

/* WLAN_PWRON (PWRDWN) signal */
```

```
#define WIFISHIELD_WLAN_PWRON_DIRECTION (BOARD_INITGT202SHIELD_PWRON_DIRECTION)
#define WIFISHIELD_WLAN_PWRON_PORT (BOARD_INITGT202SHIELD_PWRON_PORT)
#define WIFISHIELD_WLAN_PWRON_GPIO (BOARD_INITGT202SHIELD_PWRON_GPIO)
#define WIFISHIELD_WLAN_PWRON_PIN (BOARD_INITGT202SHIELD_PWRON_GPIO_PIN)

/* SPI settings */
#define WIFISHIELD_SPI (SPI1)
#define WIFISHIELD_SPI_INIT_CS (kDSPI_Pcs0)
#define WIFISHIELD_SPI_XFER_CS (kDSPI_MasterPcs0)
#define WIFISHIELD_SPI_CLOCKSRC (kCLOCK_BusClk)
#define WIFISHIELD_SPI_BAUDRATE (20000000)
#define WIFISHIELD_SPI_THRESHOLD (8)

/* DMAMUX settings, interconnect SPI with DMA */
#define WIFISHIELD_DMAMUX (DMAMUX0)
#define WIFISHIELD_DMAMUX_RX_REQ (kDmaRequestMux0SPI1)

/* EDMA settings */
#define WIFISHIELD_DMA (DMA0)
#define WIFISHIELD_DMA_RX_CHNL (0)
#define WIFISHIELD_DMA_IM_CHNL (1)
#define WIFISHIELD_DMA_TX_CHNL (2)
```

There are other files that are used to configure the clock gating and functions for the pins (such as *pin_mux.c*), but these files are usually spread out in the KSDK examples. You can select and configure the pins using the MCUXpresso Pin Configuration tool (available as a web service at https://mcuxpresso.nxp.com/). The necessary signals are:

- SPI MOSI
- SPI MISO
- SPI SCK
- SPI CS
- 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 pull down.

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.

4. Running the Wi-Fi shell demo

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.
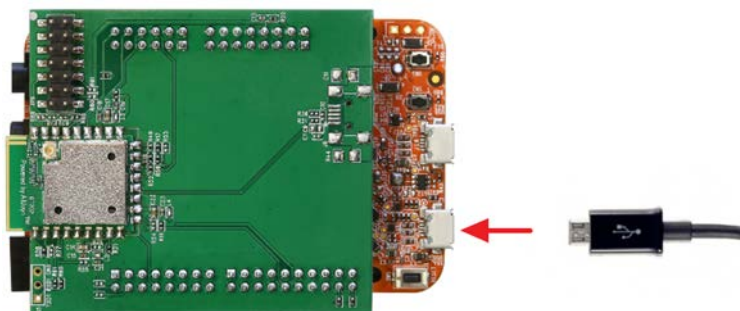
Figure 16. OpenSDA USB port

- 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 other).
- 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-K22F board.
- 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):

```
  a  Connect, DHCP, ping gateway,
start UDP echo server
  s  AP Scan
  c  AP Connect (SSID='nxp',
pass='NXP0123456789')
  C  AP Disconnect
  d  Get DHCP address
  g  HTTP GET cr.yp.to
  w  HTTP GET from gateway
  p  Ping gateway
  P  Ping dr.dk
  u  Start UDP echo server
  i  Print IP configuration
  R  Resolve some hosts
  v  Toggle tracing of QCOM API
  h  Help (print this menu)
  H  Print extended help
```

4.2. Exercising console commands

4.2.1. AP Connect

Press "c" to connect to an AP with the SSID='nxp', 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. AP Disconnect

Press "C" to disconnect:

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

4.2.3. Get DHCP address

Press "d" to get the address assigned:

```
Key 'd': Get DHCP address
Getting DHCP address...
DNS 0: 192.168.137.1
addr: 192.168.137.183 mask: 255.255.255.0 gw: 192.168.137.1
```

4.2.4. HTTP GET cr.yp.to

Press "g" to get the cr.yp.to HTTP page:

```
Key 'g': HTTP GET cr.yp.to
****************************************
Looked up cr.yp.to as 131.155.70.11
HTTP GET from 131.155.70.11:80
GET / HTTP/1.0
User-Agent: frdm+gt202
Accept-Language: en-us

TCP sent 70 bytes
Waiting for response (with t_select)
qcom_recv() receiving response
TCP received 185 bytes
HTTP/1.0 404 file does not exist
Server: publicfile
Date: Wed, 13 Apr 2016 12:23:12 GMT
Content-Length: 47
Content-Type: text/html

<html><body><h1>Beta Site</h1></body></html>
```

4.2.5. HTTP GET from gateway

Press "w" to get the start page of the web server on the gateway (usually your PC):

```
Key 'w': HTTP GET from gateway
****************************************
HTTP GET from 192.168.12.1:80
GET / HTTP/1.0
User-Agent: frdm22k+gt202
Accept-Language: en-us

TCP sent 70 bytes
Waiting for response (with t_select)
qcom_recv() receiving response
TCP received 620 bytes
HTTP/1.1 200 OK
Server: nginx/1.9.3 (Ubuntu)
Date: Fri, 15 Apr 2016 06:47:23 GMT
Content-Type: text/html
Connection: close

<html>
<head><title>Test Page</title></head>
<body>
<h1>Test Page</h1>
</body>
</html>
```

4.2.6. Ping gateway

Press "p" to ping the AP:

```
Key 'p': Ping gateway
Pinging 192.168.137.1... OK (15 ms)
```

4.2.7. Ping *www.nxp.com*

Press "P" to ping *www.nxp.com*:

```
Key 'P': Ping www.nxp.com
Looked up nxp.com as 23.202.218.160
Pinging 23.202.218.160... OK (165 ms)
```

4.2.8. Start UDP echo server

Press "u" to start the UDP echo server:

```
Key 'u': Start UDP echo server
UDP echo server listening on port 9000

Test UDP echo server from your Unix workstation with the netcat command:
  echo 'hello' | netcat -u -q1 192.168.12.1 9000
```

When executing the above, the following is displayed on the console:

```
UDP received 6 bytes from 192.168.12.1:37100
UDP sent/echoed 6 bytes
```

4.2.9. Print IP configuration

Press "i" to display the IP configuration:

```
Key 'i': Print IP configuration
addr: 192.168.137.183 mask: 255.255.255.0 gw: 192.168.137.1
```

4.2.10. Resolve hosts

Press "R" to resolve and display the (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-GT202 setups, not necessarily identical. Both setups connect to the same Access Point (AP) and the data are passed from one device to the other. This figure describes the setup:
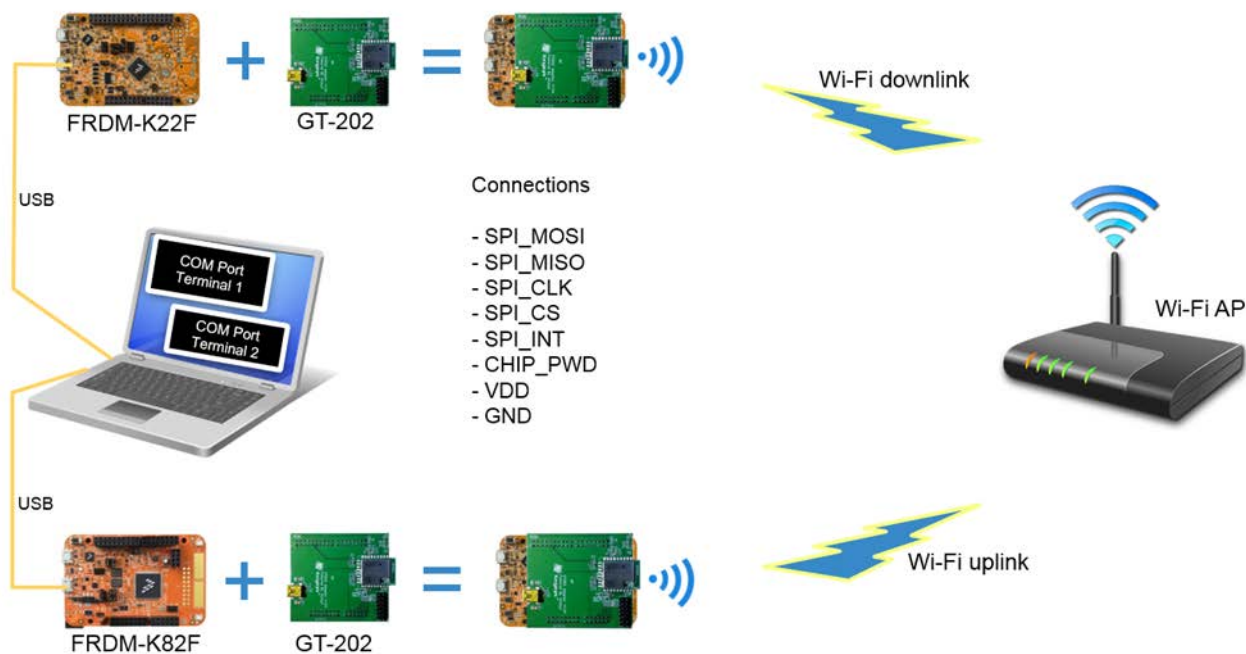
Figure 17. Wi-Fi throughput demo setup

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

<SDK_Install>\boards\frdmk22f\demo_apps\wifi_qca\qca_throughput

If using similar setups, both Freedom boards must be programed 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.

The following tables 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	Provide the AP password
wmiconfig --wpa 2 CCMP CCMP	Set up the security and encryption protocol
wmiconfig --connect GL-iNet-d2d	Connect to the AP (here called GL-iNet-d2d)
wmiconfig --ipdhcp	Ask the AP for the IP address via DHCP
ipconfig	Check the IP provided by the AP
benchmode v4	Set up the bench mode (Internet v4)
benchrx tcp 7007	Start listening on TCP port 7007

Table 2. Device 2

Command	Description
wmiconfig --p freescale	Provide the AP password
wmiconfig --wpa 2 CCMP CCMP	Set up the security and encryption protocol
wmiconfig --connect GL-iNet-d2d	Connect to the AP (here called GL-iNet-d2d)
wmiconfig --ipdhcp	Ask the AP for the IP address via DHCP
ipconfig	Check the IP provided by the AP
benchmode v4	Set up the bench mode (Internet v4)
benchtx 192.168.8.145 7007 tcp 1024 1 2000 0	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 this figure:

Figure 19. Throughput result displayed on consoles' outputs

6. References

A detailed list of all throughput demo commands is in this document:

https://developer.qualcomm.com/download/qca4002-4/sp140-141-demo-application-user-guide.pdf

The references to the FRDM boards are available on the NXP website:

http://www.nxp.com/products/software-and-tools/hardware-development-tools/freedom-development-boards:FREDEVPLA?tid=vanFREEDOM

The references to the GT202 boards are available on the Qualcomm and Arrow websites:
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

The following table summarizes the changes done to this document since the initial release:

Table 3.   Revision history

Revision number	Date	Substantive changes
0	08/2016	Initial revision.
0.1	08/2016	Update after review. Added throughput demo example.
1.0	04/2017	Updated Chapter 3 with new images and descriptive procedure
2.0	08/2017	Update from FRDM-K32W042 release Drop KDS support
3.0	06/2018	Updated Section 2.1, "Hardware configurations" with EVK-MIMXRT1020 plus GT202 shield

arm

NXP