User's Guide

# MQX<sup>™</sup> RTOS USB Stack User's Guide

### 1 Overview

This document describes how to compile the USB stack and examples, download a binary image, and run the examples. It takes TWRK22F120M Tower System module as an example, and provides the board specific information.

#### Contents

1	Overview	1
2	Requirements for Building USB Examples	2
3	USB Code Structure	7
4	Compiling or Running the USB Stack and Exan	nples 9
5	USB Stack Configuration	26
Notic	e:	28

## 2 Requirements for Building USB Examples

TWR-K22F120M is taken as an example in the rest of the document. The operations of compiling, downloading, or running the examples on all the other boards are similar, except for the explicit description for specific boards.

### 2.1 Hardware

- TWR-K22F120M Tower System module
- (Optional) TWR-SER Tower System module and Elevator
- J-Link debugger(optional)
- USB cables

### 2.2 Software

- Freescale MQX<sup>™</sup> RTOS 4.2 release package
- IAR Embedded Workbench for ARM<sup>®</sup> Version 7.30.4
- Keil μVision5 Integrated Development Environment Version 5.13, available for Kinetis ARM Cortex<sup>®</sup>-M4 devices
  - Keil.Kinetis\_K20\_DFP.1.2.0 pack
  - Keil.Kinetis\_K60\_DFP.1.3.0 pack
- Kinetis Design Studio IDE v2.5
- CodeWarrior 10.6 with all updates
- DS5 20.1 32bit
- GCC ARM 4.8-2014-q3

### 2.3 Board Jumper Settings

This document focuses on the USB-related jumper settings on the board. For other jumper settings, see the board-related user's guide. The board jumper settings are provided for all supported boards.



J2 OpenSDA, Power source and UART console output (if J10 & J15 are set to 2-3)

Re-direct UART1 output from K64 to OpenSDA. Select 2-3 for both J10 & J15

J29 Board power selection:1-2 to power from OpenSDA3-4 to power from K64 MCU5-6 to power from Elevator

Figure-1 TWR-K64 Tower System module

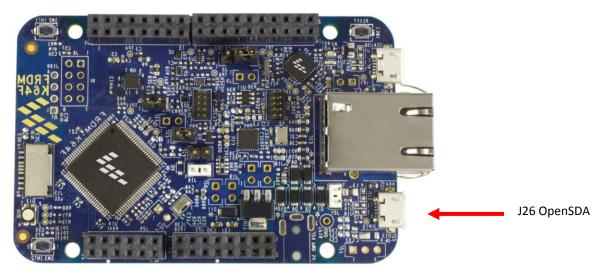
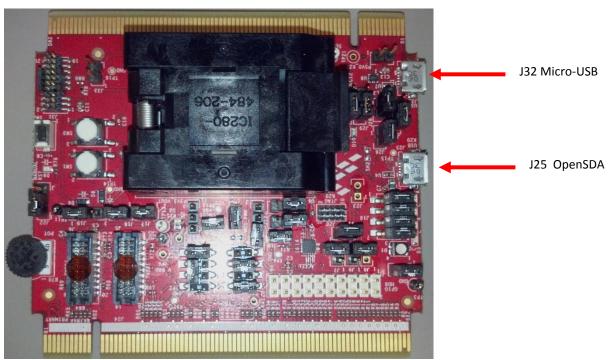


Figure-2 Freescale Freedom FRDM-K64F development platform

Freescale MQX USB Stack User's Guide, Rev. 1, 04/2015



Connect J21 Install a 2.2uF capacitor in the position C33 as shown.

Figure-3 TWR-K22 Tower System module

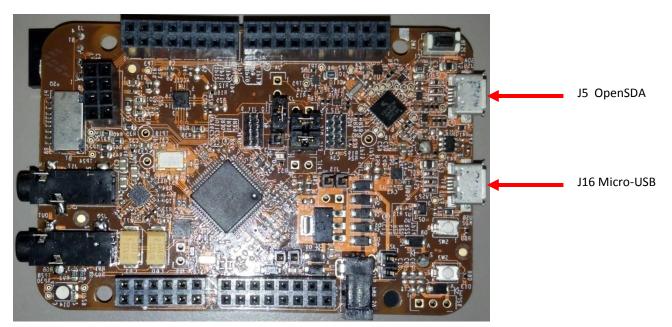


Figure-4 Freescale Freedom FRDM-K22 development platform

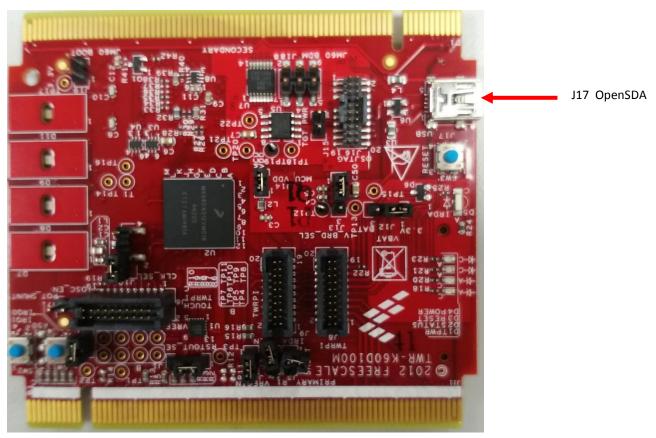


Figure-5 TWR-K60 Tower System module

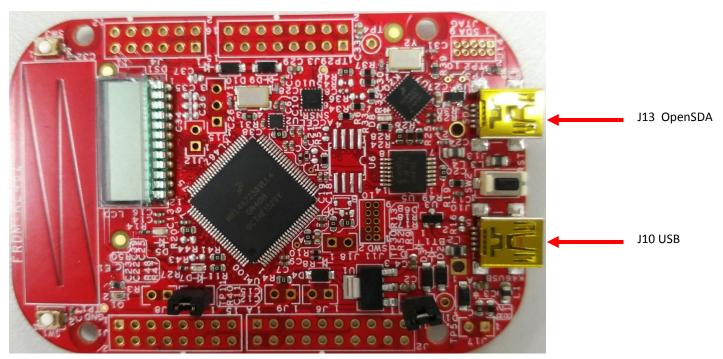


Figure-6 Freescale Freedom FRDM-KL46 development platform

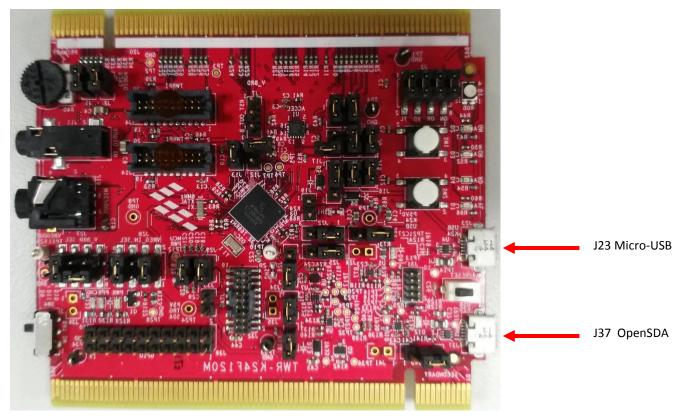


Figure-7 TWR-K24 Tower System module

Freescale MQX USB Stack User's Guide, Rev 1, 04/2015

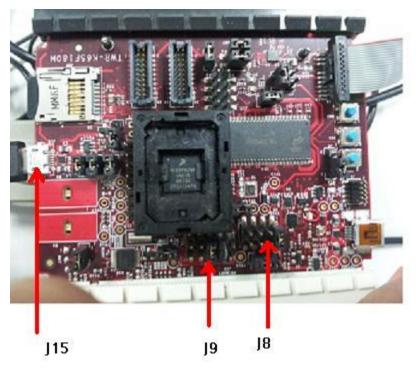


Figure-8 TWR-K65 Tower System Board jumper settings

- J8 1-2: ON
- J9 3-4: ON
- J9 5-6: ON

Note:

Only J15 on K65F180M can be used as the High Speed USB port, the USB port on TWR-SER2 board is not supported.

### 3 USB Code Structure

New USB stack is located in the "**usb\_v2**" folder, in the root level of the MQX RTOS4.2 folder. There are some subfolders in it as shown in figure below:

4 퉬 Freescale_MQX_4_	2		
🖻 퉲 build			
🛛 📗 config			
🖻 퉲 demo			
⊳ 퉲 doc			
Þ 鷆 ffs			
🖻 퉲 mcc			
🖻 퉲 mfs			
Þ 퉲 mqx			
Þ 퉲 mqx1588			
🛛 📗 rtcs			
🖻 퉲 shell			
🛛 퉲 tools			
🖻 퉲 usb			
4 🍌 usb_v2	]		
🖻 🍌 adapter	1		
Þ 퉲 build			
🛛 📗 example			
鷆 output			
b lb usb_core			

#### Figure-8 New USB stack folder structure

• adapter

This subfolder includes the adapter files to support the USB stack running on a different RTOS with the same USB core code.

• build

This subfolder includes arm gcc tool chains supported in USB stack.

• example

This subfolder includes all source code and project files for the USB examples.

• output

The USB library binary file will be generated into this folder and all the USB related public header files which may be used by the user will be copied to this folder so that the examples need to include one folder as the including path in the example project settings.

• usb\_core

This subfolder includes the USB source files, such as HAL, controller driver, class drivers, and the USB library projects.

### 4 Compiling or Running the USB Stack and Examples

### 4.1 Step-by-step guide for IAR

This section describes how to build and run USB example on IAR. The "**host\_hid\_mouse**" example will be used as an example.

- 1. Open IAR.
- 2. Add projects to IAR by clicking on **Project** tag and the select Add Existing Project. You can find corresponding IAR project files as below links.
  - a. bsp library project <install\_dir>/mqx/build/iar/bsp\_twrk22f120m

  - c. USB host library project <install\_dir>/usb\_v2/usb\_core/host/build/iar/usbh\_mqx\_twrk22f120m
  - d. USB host hid mouse example project
     <install\_dir>/usb\_v2/example/host/hid/mouse/mqx/iar/host\_hid\_mouse\_twrk22f
     120m

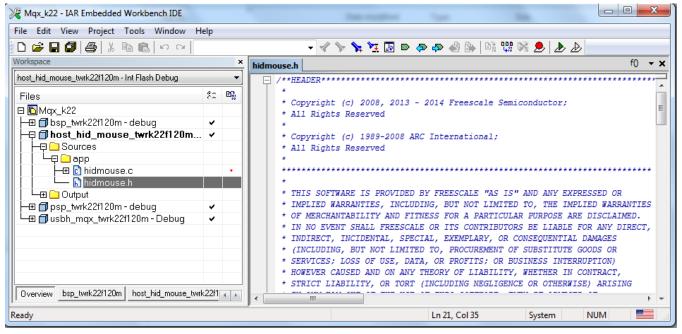


Figure-9 IAR workspace

- 3. Build the MQX RTOS bsp and psp library.
- 4. Build the usbh\_mqx\_twrk22f120m library.
- 5. Check the USB library build result.
- 6. After the USB library is built, the generated library binary file (libusbh\_mqx.a) is located in <install\_dir>/usb\_v2/output/twrk22f120m.iar/debug/usbh/mqx.
- 7. All USB-related public header files are copied to this folder.
- 8. Build the host\_hid\_mouse\_twrk22f120m\_mqx example.

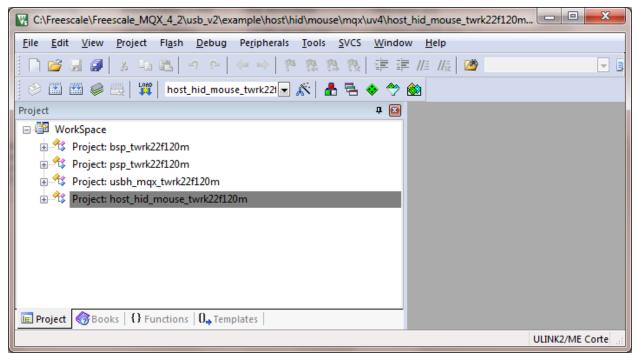
- 9. Connect the micro USB cable from a PC to the J25 of the TWR-K22F120M Tower System module to power on the board.
- 10. Click the "Download and Debug" button. Wait for the download to complete.
- 11. Click the "Go" button to run the example.

### 4.2 Step-by-step guide for KEIL

This section describes how to build USB stack and USB example on KEIL.

- 1. Open KEIL
- 2. Add project to KEIL by clicking on **Project** tag then select "**New Multi-Project Workspace...**" You can find corresponding KEIL project files as below links.
  - a. bsp library project
     <install dir>/mqx/build/uv4/bsp twrk22f120m

  - c. USB host library project <install\_dir>/usb\_v2/usb\_core/host/build/uv4/usbh\_mqx\_twrk22f120m
  - d. USB host hid mouse example project
     <install\_dir>/usb\_v2/example/host/hid/mouse/mqx/uv4/host\_hid\_mouse\_twrk22f
     120m



#### Figure-10 KEIL Workspace

- 3. Build the MQX RTOS bsp and psp library.
- 4. Build the usbh\_mqx\_twrk22f120m library.
- 5. Check the USB library build result.

- 6. After the USB library is built, the generated library binary file (libusbh\_mqx.a) is located in <install\_dir>/usb\_v2/output/twrk22f120m.uv4/debug/usbh/mqx.
- 7. All USB-related public header files are copied to this folder.
- 8. Build the host\_hid\_mouse\_twrk22f120m\_mqx example.
- 9. Connect the micro USB cable from a PC to the J25 of the TWR-K22F120M Tower System module to power on the board
- 10. Click the "Start/Stop" debug session button. Wait for the download to complete.
- 11. Click the "Go" button to run the example.

### 4.3 Step-by-step guide for the Kinetis Design Studio IDE

- 1. Unlike IAR or KEIL, the Kinetis Design Studio IDE doesn't have a workspace. As a result, create a workspace and import Kinetis Design Studio IDE USB examples, platform libraries, and the USB stack library.
- 2. Select "File" then "Import" from the KDS IDE Eclipse menu.
- 3. Expand the General folder and select "Existing Projects into Workspace". Then, click the "Next" button.

🥦 Import	- • ×
Select Create new projects from an archive file or directory.	Ľ
Select an import source:	
type filter text	
<ul> <li>✓ General</li> <li>Marchive File</li> <li>✓ Existing Projects into Workspace</li> <li>✓ File System</li> <li>✓ Preferences</li> <li>✓ C/C++</li> <li>✓ Component Development Environment</li> <li>✓ CVS</li> <li>✓ Git</li> <li>✓ Install</li> <li>✓ Processor Expert</li> <li>✓ Remote Systems</li> </ul>	
(?) < <u>Back</u> <u>Next</u> > <u>Finish</u>	Cancel

Figure-11 Selection of the correct import type in KDS IDE

4. Point the KDS IDE to the **bsp\_twrk22f120m** in **<install\_dir>/mqx/build/kds**. The import projects directory selection window should resemble this figure

🨹 Import		_ <b>D X</b>
Import Projects Select a directory to sear		
<ul> <li>Select root directory:</li> <li>Select <u>a</u>rchive file:</li> </ul>	C:\Freescale\Freescale_MQX_4_2\mqx\build	Browse Browse
Projects:	(C:\Freescale\Freescale_MQX_4_2\mqx\build\kc	<u>S</u> elect All <u>D</u> eselect All R <u>e</u> fresh
Options     Search for nested pro     Copy projects into w	-	
Working sets Add projec <u>t</u> to work Working sets:	ing sets	S <u>e</u> lect
?	< <u>Back</u> <u>N</u> ext > <u>Finish</u>	Cancel

Figure-12 Selection of the K22 ksdk\_platform\_lib project

5. Following the same step to import the USB host library and the USB example, after importing them, the window should like this

🛞 C/C++ - usbh_mqx_twrk22f120m/usb_core/host/in	clude/usb_host_config.h - Kinetis Design Studio - C:\Users\hain 🗔 💷 💻 🌉
	Project       Processor Expert       Run       Window       Help <sup>™</sup>
Project Explorer S       Project Explorer S       S	<pre>     usb_host_config.h ☆</pre>
	0 errors, 6 warnings, 0 others
0 items selected	1

Figure-13 USB projects workspace

6. Choose the appropriate build target: "**Debug**" or "**Release**" by left-clicking the arrow next to the hammer icon as shown here.

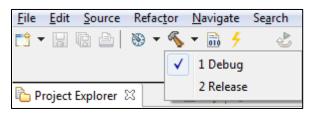


Figure-14 The hammer button

- 7. If the library build does not begin after selecting the desired target, left-click the hammer icon to start the build.
- 8. Following the same step to build the bsp\_twrk22f120m library, psp\_twrk22f120m library the usbh\_mqx\_twrk22f120m library and the host\_hid\_mouse\_twrk22f120m example.
- 9. To check the debugger configurations, click the down arrow next to the green debug button and select "**Debug Configurations**".

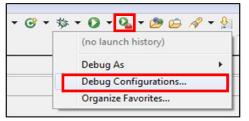


Figure-15 Debug configurations

10. After verifying that the debugger configurations are correct, click on the "Debug" button.

🎘 Debug Configurations	 
Create, manage, and run configurations	Ť.
Image: Second	Variables       Search Project       Browse         f120m       Browse         e launching         Use Active       ▼         ♥ Select configuration using 'C/C++ Application'         © Disable auto build
?	Debug Close

#### Figure-16 Kinetis Design Studio Debug configurations

- 11. The application is downloaded to the target and automatically run to main().
- 12. Run the code by clicking the "Resume" button to start the application

or Expe	ert R	un	Wi	ndov
3		88		8-3
	Res	sum	e (F8	)

Figure-17 Resume button

### 4.4 Step-by-step guide for the DS5

- 1. Same with the Kinetis Design Studio, the DS5 also doesn't have a workspace. As a result, create a workspace and import Kinetis Design Studio USB examples, platform libraries, and the USB stack library. We use twrvf65gs10\_a5 as example for DS5 build and download.
- 2. Select "File" then "Import" from the DS5 menu.
- 3. Expand the General folder and select "Existing Projects into Workspace". Then, click the "Next" button

🖨 Import	
Select Create new projects from an archive file or directory.	Ľ
Select an import source:	
type filter text	
<ul> <li>▲ General</li> <li>Archive File</li> <li>☆ Existing Projects into Workspace</li> <li>→ File System</li> <li>→ Preferences</li> <li>&gt; ↔ C/C++</li> <li>&gt; ↔ C/S</li> <li>&gt; ↔ Install</li> <li>&gt; ↔ Remote Systems</li> <li>&gt; ☆ Run/Debug</li> <li>&gt; ☆ Scatter File Editor</li> <li>&gt; ☆ Target Configuration Editor</li> <li>&gt; &lt; Team</li> </ul>	
(?) < <u>Back</u> Next > <u>Finish</u>	Cancel

#### Figure 18 Import projects

4. Point the KDS IDE to the **bsp\_twrvf65gs10\_a5** in **<install\_dir>/mqx/build/ds5**. The import projects directory selection window should resemble this figure

🖨 Import		
Import Projects Select a directory to sear	ch for existing Eclipse projects.	
<ul> <li>Select roo<u>t</u> directory:</li> <li>Select <u>a</u>rchive file:</li> <li><u>P</u>rojects:</li> </ul>	IQX_4_2\mqx\build\ds5\bsp_twrvf65gs10_a5 →	Browse Browse
▼ bsp_twrvf65gs10	_a5 (C:\Freescale\Freescale_MQX_4_2\mqx\build\	<u>S</u> elect All <u>D</u> eselect All R <u>e</u> fresh
Options     Searc <u>h</u> for nested pro <u>C</u> opy projects into w     Working sets     Add project to work     Working sets:	orkspace	S <u>e</u> lect
?	< <u>Back</u> <u>N</u> ext > <u>Finish</u>	Cancel

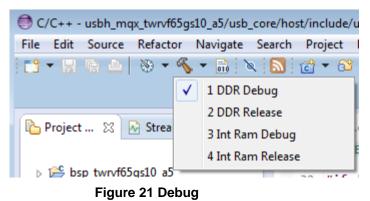
Figure 19 Import Eclipse projects

5. Following the same step to import the USB host library and the USB example, after importing them, the window should like this

	<pre></pre>	nost_config.h ⊠		
Image: bsp_twn/f65gs10_a5       Image: bsp_twn/f65gs10_a5       Image: bsp_twn/f65gs10_a5       Image: bsp_twn/f65gs10_a5	<pre>1@ /**HEADER***********************************</pre>	******	***************************************	□□ ↓ <sup>a</sup> <sub>Z</sub> ≷ ≷ ♥ ₩ # _usbhost_cnfg_h. # USBCFG_HOST_KF
c	<pre>stdefine USBCFG_HOST_HKL1 stdefine USBCFG_HOST_EHCI 40 41@ /* 42 ** Maximum number of USB instat 3 ** MGGT: <option #define="" *="" **="" *<="" ,="" 44="" 45="" 46="" 470="" 48="" 49="" 50="" 51="" 52="" 53@="" 54="" 55="" 56="" <option="" maximum="" mgct:="" number="" of="" pipes.="" th="" type="number" usbcfg_host_default_nax="" usbcfg_host_max_pipes="" usbcfg_host_num=""><th>nce. /&gt; /&gt; ( /&gt; x_NAK_COUNT (</th><th>0) 1) 2) 16) 3000) +</th><th><pre># USBCFG_HOST_N # USBCFG_HOST_M # USBCFG_HOST_C # USBCFG_HOST_C # USBCFG_HOST_C # USBCFG_HOST_M # USBCFG_HOST_M # USBCFG_HOST_M # USBCFG_HOST_M # USBCFG_HOST_M # USBCFG_HOST_M # USBCFG_HOST_M # USBCFG_HOST_M # USBCFG_HOST_K # USBCFG_HOST_K # USBCFG_HOST_K # USBCFG_HOST_K # USBCFG_HOST_K # USBCFG_HOST_K</pre></th></option></pre>	nce. /> /> ( /> x_NAK_COUNT (	0) 1) 2) 16) 3000) +	<pre># USBCFG_HOST_N # USBCFG_HOST_M # USBCFG_HOST_C # USBCFG_HOST_C # USBCFG_HOST_C # USBCFG_HOST_M # USBCFG_HOST_M # USBCFG_HOST_M # USBCFG_HOST_M # USBCFG_HOST_M # USBCFG_HOST_M # USBCFG_HOST_M # USBCFG_HOST_M # USBCFG_HOST_K # USBCFG_HOST_K # USBCFG_HOST_K # USBCFG_HOST_K # USBCFG_HOST_K # USBCFG_HOST_K</pre>

Figure 20 Importing projects

6. Choose the appropriate build target: "**Debug**" or "**Release**" by left-clicking the arrow next to the hammer icon as shown here.



- 7. If the library build does not begin after selecting the desired target, left-click the hammer icon to start the build.
- 8. Following the same step to build the bsp\_twrvf65gs10\_a5 library, psp\_twrvf65gs10\_a5 library the usbh\_mqx\_twrvf65gs10\_a5 library and the host\_hid\_mouse\_twrvf65gs10\_a5 example.

Freescale MQX USB Stack User's Guide, Rev. 1, 04/2015

9. To check the debugger configurations, click the down arrow next to the green debug button and select "**Debug Configurations**".

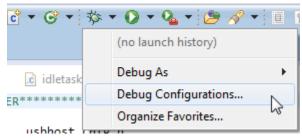


Figure 22 Debug configurations

10. After verifying that the debugger configurations are correct, click on the "Debug" button.

🖨 Debug Configurations				
Create, manage, and run configurations				
Create, edit or choose a configuration to launch a DS-5 debuggi	ng session.			
Image: Second Secon	Name: host_hid_mouse_twrvf65gs10_a5_DDR_Debug			
🤗 Python unittest ⊒ Remote Java Application	K			
Filter matched 22 of 22 items	Apply Revert			
?	Debug			

Figure 23 Debug button

- 11. The application is downloaded to the target and automatically run to main().
- 12. Run the code by clicking the "Resume" button to start the application

🔿 🕶 🖂 🕶		00	₽	P	.P	⇔ si	
	-						

Freescale MQX USB Stack User's Guide, Rev 1, 04/2015

### 4.5 Step-by-step guide for the ARM GCC and KDS IDE GCC

### 4.5.1 Setup tool chains

#### 4.5.1.1 Install GCC ARM Embedded tool chain

Download and install the installer from www.launchpad.net/gcc-arm-embedded.

#### 4.5.1.2 Install MinGW

- 1. Download the latest mingw-get-setup.exe.
- 2. Install the GCC ARM Embedded toolchain. The recommended path is C:/MINGW, however, you may install to any location. Note that the installation path may not contain a space.
- 3. Ensure that the mingw32-base and msys-base are selected under Basic Setup.
- 4. Finally, click "Installation" and "Apply changes".

Installation Package Settings								
Basic Setup		Package	Class	Installed	Repository Version	Description		
All Packages		mingw-developer-toolkit	bin		2013072300	An MSYS Installation for MinGW Developers (meta		
	5	mingw32-base	bin		2013072200	A Basic MinGW Installation		
		mingw32-gcc-ada	bin		4.8.1-4	The GNU Ada Compiler		
		mingw32-gcc-fortran	bin		4.8.1-4	The GNU FORTRAN Compiler		
		mingw32-gcc-g++	bin		4.8.1-4	The GNU C++ Compiler		
		mingw32-gcc-objc	bin		4.8.1-4	The GNU Objective-C Compiler		
	9	msys-base	bin		2013072300	A Basic MSYS Installation (meta)		

Figure 24 Setup MinGW and MSYS

5. Add paths C:/MINGW/msys/1.0/bin;C:/MINGW/bin to the system environment. Note that if the GCC aRM Embedded tool chain was installed somewhere other than the recommended location, the system paths added should reflect this change. An example using the recommended installation locations are shown below.

#### NOTE

There is a high chance that, if the paths are not set correctly, the tool chain will not work properly.

	;
Edit System Variat	ble
Variable name: Variable value:	Path c:\MinGW\msys\1.0\bin;c:\MinGW\bin;b:\Pr OK Cancel
System variables	
Variable	Value
Variable Path PATHEXT PROCESSOR_A PROCESSOR_ID	c:\/MinGW\/msys\1.0\/bin;C:\/Program File .COM;.EXE;.BAT;.CMD;.VBS;.VBE;.JS;

Figure 25 Add Path to systems environment

### 4.5.1.3 Add new system environment variable ARMGCC\_DIR

Create a new system environment variable ARMGCC\_DIR. The value of this variable should be the short name of the ARM GCC Embedded tool chain installation path.

dit System Varia	ible 📃 💌
Variable name:	ARMGCC_DIR
Variable value:	c:\GNU_Tools_ARM_Embedded\4.8_2014q
vstem variables	OK Cancel
ystem variables Variable	OK Cancel Value
Variable ARMGCC_DIR	Value c:\GNU_Tools_ARM_Embedded\4.8_20
Variable ARMGCC_DIR ComSpec	Value c:\GNU_Tools_ARM_Embedded\4.8_20 C:\Windows\system32\cmd.exe
Variable ARMGCC_DIR ComSpec DEFLOGDIR	Value c:\GNU_Tools_ARM_Embedded\4.8_20 C:\Windows\system32\cmd.exe C:\ProgramData\McAfee\DesktopProtec
Variable ARMGCC_DIR ComSpec	Value c:\GNU_Tools_ARM_Embedded\4.8_20 C:\Windows\system32\cmd.exe C:\ProgramData\McAfee\DesktopProtec

Figure 26 Add ARMGCC\_DIR system variable

#### 4.5.1.4 Add new system environment variable KDSGCC\_DIR

Create a new system environment variable KDSGCC\_DIR. The value for the variable is the name of the Kinetis Design Studio (KDS) IDE ARM GCC installation path. By default, KDS IDE is installed to the C:/Freescale/KDS\_1.1.0/toolchain location.

dit System Varia	ble 📃 🚬
Variable name:	KDSGCC_DIR
Variable value:	c:\Freescale\KDS_1.1.0\toolchain
ystem variables	OK Cancel
Variable	Value 4
Variable KDSGCC_DIR	Value c:\Freescale\KDS_1.1.0\toolchain
Variable	Value 1
Variable KDSGCC_DIR MQX_PATH	Value  c: \Freescale \KDS_1.1.0 \toolchain  C: \Freescale \Freescale MQX_4_1  C: \Freescale \Freescale \Freescale MQX_4_1

Figure 27 Add KDSGCC\_DIR in system variable

### 4.5.2 Set the ARM Gcc Tool Chain Environment

The TOOLCHAIN\_ROOTDIR and the GCC\_VERSION should be unmasked and set to the correct value. These variables are in <install\_dir>/usb\_v2/build/common/make/global.mak. For example:

```
ifeq ($(TOOL),gcc_arm)
GCC_VERSION ?=4.8.4
TOOLCHAIN_ROOTDIR = C:/PROGRA~1/GNUTOO~1/43F2B~1.720
endif
```

### 4.5.3 Build the MQX RTOS library

To build the platform library, follow these instructions:

- Open a GCC ARM Embedded tool chain command window.
- Change the directory of the command window to the MQX RTOS **bsp** and **psp** lib directory in the MQX RTOS4.2 (one of these):

```
<install_dir>/mqx/build/make/bsp_twrk22f120m/
<install dir>/mqx/build/make/psp_twrk22f120m/
```

- Run "build gcc arm.bat"
- The mqx library is generated in these directories according to the build target.

### 4.5.4 Build the USB host/device library

• Change the directory to the project directory

<install\_dir>/usb\_v2/usb\_core/host/build/make/usbh\_mqx\_twrk22f120m

• Run "build\_gcc\_arm.bat"

#### 4.5.5 Build the USB demo

• Change the directory to the project directory:

<install\_dir>/usb\_v2/example/host/hid/mouse/mqx/make/host\_hid\_mouse\_twrk22f120m

• Run "build\_gcc\_arm.bat"

#### 4.5.6 Run a demo application

This section describes steps to run a demo application using J-Link GDB Server application.

- 1. Connect the J-Link debug pod to the SWD/JTAG connector of the board.
- 2. Open the J-Link GDB Server application and modify your connection settings as shown in this figure.

SEGGER J-Link GDB Server V4.90e - Config	×
Connection to J-Link	
O TCh/Ib	
Target device	
MK22DN512xxx5	
Little endian 💌	
Target interface	
JTAG	
Speed	
C Adaptive <u>c</u> locking	
C 1000 <u>k</u> Hz	
Command line option	
-select USB -device MK22DN512xxx5 -if JTAG -speed auto	
OK Cancel	]

Figure 28 SEGGER J-Link GDB Server configuration

3. Once connected, the screen should resemble this figure:

SEGGER J-Link GDB Server V4.90e		
<u>F</u> ile <u>H</u> elp		
GDB Waiting for connection J-Link Connected CPU MK22DN512xxx5	Initial JTAG speed Auto	✓       Localhost onlyi         Stay on top         ✓       Show log window         ✓       Generate logfile         ✓       Verify download         ✓       Init regs on start
Log output: <u>Clear log</u> Connecting to J-Link J-Link is connected. Firmware: J-Link Lite-FSL V Hardware: V1.00 S/N: 361000583 Checking target voltage Target voltage: 3.30 V Listening on TCP/IP port 23 Connecting to target J-Link found 1 JTAG device,	31 Total IRLen = 4	.6:40:07
JTAG ID: 0x4BA00477 (Cortex Connected to target Waiting for GDB connection.		= ~
0 Bytes downloaded 1 J	JTAG device	//

Figure 29 SEGGER J-Link GDB Server screen after successful connection

4. Open the ARM GCC command prompt and change the directory to the output directory of the desired demo. For this example, the directory is:

<install\_dir>/usb\_v2/example/host/hid/mouse/mqx/make/host\_hid\_mouse\_twrk22f120m

5. Run the command "arm-none-eabi-gdb.exe <DEMO\_NAME>.elf". For this example, it is "arm-none-eabi-gdb.exe host\_hid\_mouse\_twrk22f120m.elf".

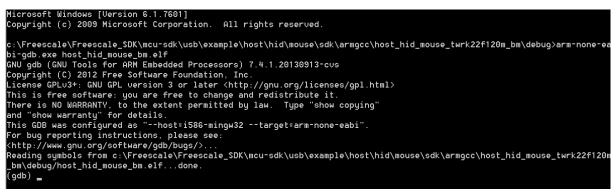


Figure 30 Run arm-none-eabi-gdb

- 6. Run these commands:
  - a. "target remote localhost: 2331"
  - b. "monitor reset"
  - c. "monitor halt"
  - d. "load"
  - e. "monitor reset"
- 7. The application is downloaded and connected. Execute the "monitor go" command to start the demo application.

### 5 USB Stack Configuration

### 5.1 Device configuration

All device configurations are listed in this file:

<install\_dir>/usb\_core/device/include/BOARD\_NAME/usb\_device\_config.h

Replace BOARD\_NAME with the name of the board.

This file is used to either enable or disable the USB class driver. The object number is configurable either to decrease the memory usage or to meet specific requirements.

If the device stack configuration is changed, rebuild both the USB library and the example projects.

#### NOTE

The composite device examples works only with this setting:

```
USBCFG_DEV_COMPOSITE
```

All the other non-composite device examples work only with this setting:

USBCFG\_DEV\_COMPOSITE 0

If incorrect settings are configured, a build error occurs.

### 5.2 Host configuration

All the host configurations are listed in this file:

<install\_dir>/usb\_core/host/include/BOARD\_NAME/usb\_host\_config.h

Replace BOARD\_NAME with the name of the board.

This file is used to either enable or disable the USB class driver. The object number is configurable either to decrease the memory usage or to meet specific requirements.

If the device stack configuration is changed, rebuild both the USB library and the example projects.

#### NOTE

Micro and mini receptacles are available for the TWR-K22F120M Tower System module if the TWR-SER and elevator are used. Configure the software and hardware to switch between the two USB receptacles.

• To use the micro receptacle on the TWR-K22F120M Tower System module, the jumper settings should be (for both device and host):

o J4 1-2

- J27 2-3 (for rev. A)
- o J27 1-2 (for rev. B)

Freescale MQX USB Stack User's Guide, Rev 1, 04/2015

If the host stack is used, the additional configuration is needed:

```
USBCFG_HOST_PORT_NATIVE 1
```

- To use the mini receptacle on the TWR-SER Tower System module, the jumper settings should be (for both device and host):
  - o J4 1-2
  - J27 1-2 (for rev. A)
  - J27 2-3 (for rev. B)
  - See the appropriate TWR-SER user's guide for the jumper settings on TWR-SER Tower System module.

If the host stack is used, the additional configuration is needed:

USBCFG\_HOST\_PORT\_NATIVE 0

Additional configurations are not needed for the device because switching between the two USB receptacles doesn't require changing code in the device mode.

### 5.3 OTG configuration

All OTG configurations are listed in these files:

```
<install_dir>/usb_core/device/include/twrk22f120m/usb_device_config.h
<install_dir>/usb_core/host/include/twrk22f120m/usb_host_config.h
```

These files either enable or disable the USB class driver. The object number is configurable either to decrease the memory usage or to meet specific requirements.

If the OTG stack configuration is changed, rebuild both USB library and example projects.

#### NOTE

The OTG example for the TWR-K22F120M Tower System module requires the mini receptacle on the TWR-SER Tower System module. The jumper settings should be:

o J4 1-2

- J27 1-2 (for rev. A)
- J27 2-3 (for rev. B)
- See the appropriate TWR-SER user's guide for the jumper settings on the TWR-SER Tower System module.

The additional configuration is needed for the host mode:

USBCFG\_HOST\_PORT\_NATIVE 0

The additional configuration is needed for the device mode:

```
USBCFG_DEV_COMPOSITE
```

0

#### NOTE

1. If the USB mini port (J14) on the serial board needs to be used as a device connector on the K64 Tower System module, J19 must be set to 2-3.

2. Because the K64\_USB\_DP and K64\_USB\_DN are not connected to the elevator micro USB port on the TWR-K64F120M Tower System module, the R522 and the R523 are not placed and only the micro USB port can be used.

3. If the TWRK-K64 Tower System module is a USB device when no OpenSDA power is supplied, the jumpers need to be set up like this:

J29, 5-6 J19, 2-3 J18, 2-3.

4. If the Freescale Freedom FRDM-K64F is a USB device when no OpenSDA power is supplied, add a 0-ohm resistor on R61 to power on the P3V3\_SDA.

#### Notice:

• Khei host can't work on HSRUN mode on TWRK65 platform. Because if K65 SYSCLOCK run in 180M, USB module can't get the 48M clock.

- Please change the MQX\_ENABLE\_HSRUN MACRO from 1 to 0 for KHCI host test .

• If you run the USB example on High speed mode, please make sure the HIGH\_SPEED MACRO is set to 1, and example running in FULL speed mode, the MACRO should be set to 0.

Freescale MQX USB Stack User's Guide, Rev 1, 04/2015

#### How to Reach Us:

Home Page: www.freescale.com

Web Support: www.freescale.com/support Information in this document is provided solely to enable system and software implementers to use Freescale 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.

Freescale reserves the right to make changes without further notice to any products herein. Freescale makes no warranty, representation, or guarantee regarding the suitability of its products for any particular purpose, nor does Freescale 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 Freescale 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. Freescale does not convey any license under its patent rights nor the rights of others. Freescale sells products pursuant to standard terms and conditions of sale, which can be found at the following address: freescale.com/SalesTermsandConditions.

Freescale, the Freescale logo, and Kinetis are trademarks of Freescale Semiconductor, Inc., Reg. U.S. Pat. & Tm. Off. Tower is a trademark of Freescale Semiconductor, Inc. All other product or service names are the property of their respective owners. ARM is a registered trademark of ARM Limited (or its subsidiaries) in the EU and/or elsewhere. All rights reserved.

©2015 Freescale Semiconductor, Inc.



