

Freescal Bare Metal USB Stack v5.0 Beta User's Guide

1 Read Me First

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

Contents

1	Read Me First	1
2	Requirements for Building USB Examples	2
3	USB Code Structure	8
4	Compiling or Running the USB Stack and Examples	10
5	USB Stack Configuration	23



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 board
- (Optional) TWR-SER and Elevator
- J-Link debugger
- USB cables

2.2 Software

- Freescale BM USB Stack Beta release package v 5.0
- IAR Embedded Workbench for ARM Version 6.70.3, available for Kinetis devices
 - Patch for K64F120M
- Keil uVision4 Integrated Development Environment Version 5.0.5.15, available for Kinetis ARM® CortexM4 devices
 - Patch for K64F120M
- GNU Tools for ARM Embedded Processors 4.7 2013Q3
- MinGW v3.82.90 with the **mingw32-base** and **msys-base** packages installed

2.3 Board jumper settings

This document focuses on the USB related jumper settings on the CPU board. For the other jumper settings, refer to the board related user guide.

2.3.1 TWR-K22F120M

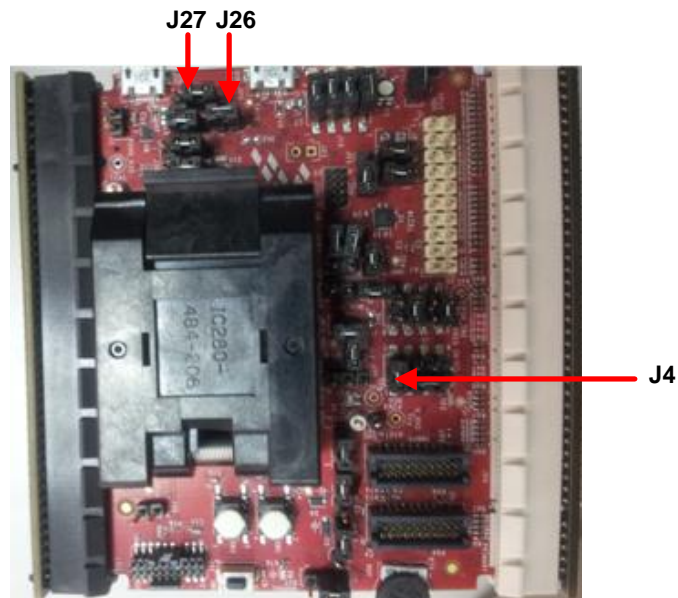


Figure 1 Board jumper settings

- J4 1-2: enables the P3V3_VOUT
- J26 1-2: enables 5V VBUS for the Host mode

For **TWR-K22F120M board rev A**:

- J27 1-2: uses the mini USB receptacle on the TWR-SER board
- J27 2-3: uses the micro USB receptacle on the TWR-K22F120M board

For **TWR-K22F120M board rev B or later**:

- J27 1-2: uses the micro USB receptacle on the TWR-K22F120M board
- J27 2-3: uses the mini USB receptacle on the TWR-SER board

2.3.2 FRDM-K64F120M REV C

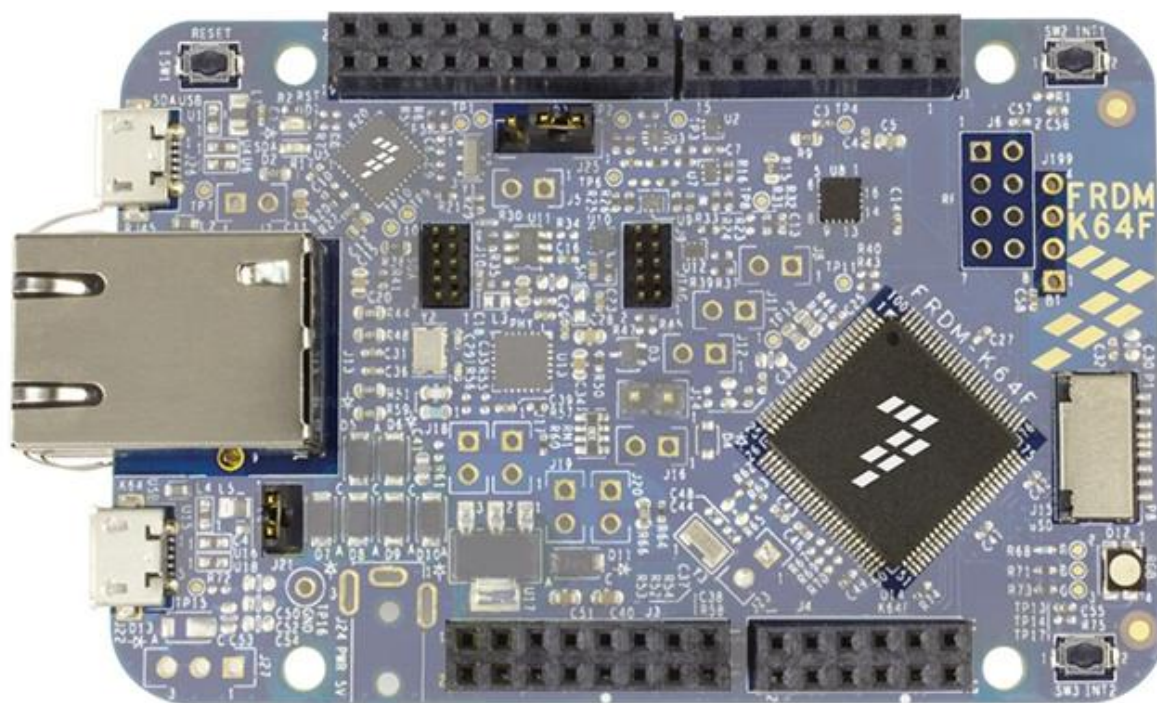


Figure 2 FRDM-K64F120M

- J21 1-2: enables the USB Host VBUS

2.3.3 TWR-K64F120M

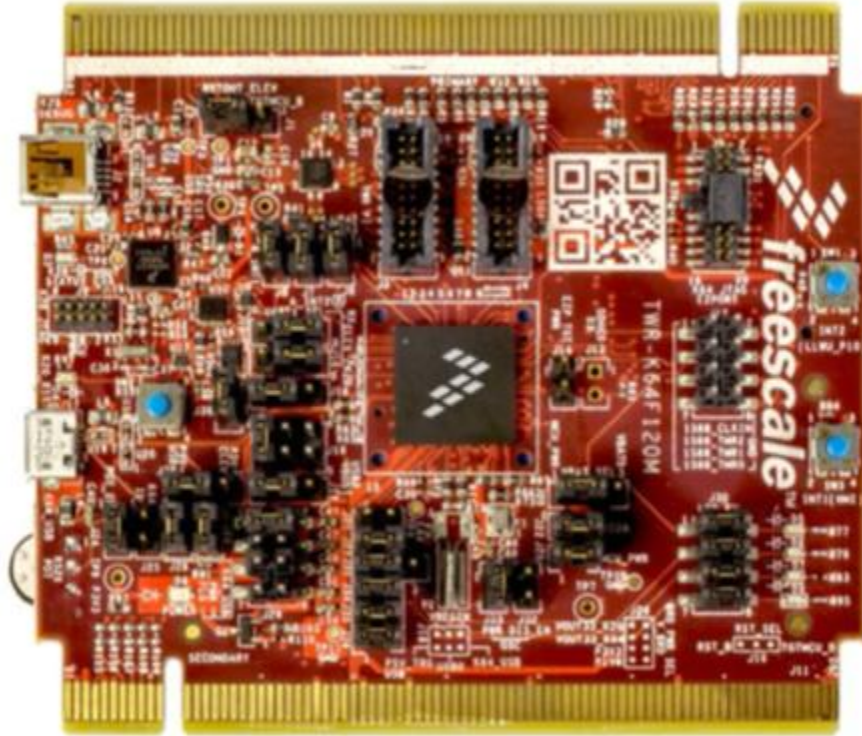


Figure 3 TWR-K64F120M

- J19 1-2: enables VREGIN from Micro USB on TWR-K64F120M
- J19 2-3: enables VERGIN from ELEVATOR
- J26 1-2: enables 5V VBUS for the Host mode

Notes: If you need to test OTG for TWR-K64F120M or want to use the USB port on TWR-SER, you need to insert R522 and R523, and remove R63 and R65 from TWR-K64F120M.

2.3.4 TWR-K70F120M

- J1 1-2: enables the VREGIN

2.3.5 FRDM-KL25

No USB related jumper setting needed

2.3.6 FRDM-KL26

No USB related jumper setting needed

2.3.7 TWR-SER



Figure 4 TWR-SER

- J10 1-2: enables the VBUS for Host
- J10 2-3: source 5V from USB
- J16 1-2: Host mode
- J16 3-4: Device mode
- J16 5-6 and J11 5-6: OTG mode



Figure 5 TWR-SER2

- J21 1-2: Connects USB VBUS EN to Elevator
- J24 1-2: Device Mode

3 USB Code Structure

The USB stack is located in the **Src** folder. There are seven subfolders in it as follows:

 adapter	File Folder	3/21/2014 1:44 PM
 build	File Folder	3/24/2014 11:54 AM
 example	File Folder	3/21/2014 1:44 PM
 os	File Folder	3/21/2014 1:44 PM
 output	File Folder	3/24/2014 11:54 AM
 platform	File Folder	3/21/2014 1:44 PM
 usb_core	File Folder	3/24/2014 11:54 AM

Figure 6 Src folder structure

- adapter

Includes the adapter files which allow the USB stack to run on different RTOS with the same USB core code.

- build

Includes the GCC make files.

- example

Includes all the source code and project files of the USB examples. The examples are organized as following structure:

ROLE/CLASS/APP/bm/COMPILER/BOARD_APP_NAME

- The ROLE could be “device”, “host”, or “otg”.
- The CLASS could be “audio”, “cdc”, “hid”, “msd”, or “phdc”.
- The APP is the application example name like “hid_mouse”.
- The COMPILER could be “iar”, “make”, or “uv4”, which represent IAR, ARM_GCC, and KEIL in turn.
- BOARD_APP_NAME could be “dev_hid_keyboard_frdmkl26z” or “dev_hid_keyboard_frdmkl25z”.

- os

Includes the source code of bare metal system services, such as msgq and semaphore.

- output

The USB library binary file is generated into this folder and all the USB related public header files are copied to this folder. The examples need to include one folder as the including path in the example project settings.

- platform

Includes the source files and header files related to SOC, BSP and ARCH.

-
- usb_core

Include the USB source files, such as HAL, controller driver, and class drivers. It also includes the USB library projects.

4 Compiling or Running the USB Stack and Examples

4.1 Step by step guide for IAR

This section takes IAR as an example to show how to build examples. The other tool chains have similar steps. Take TWR-K22F120M board as an example in this section.

1. Open IAR as follows.

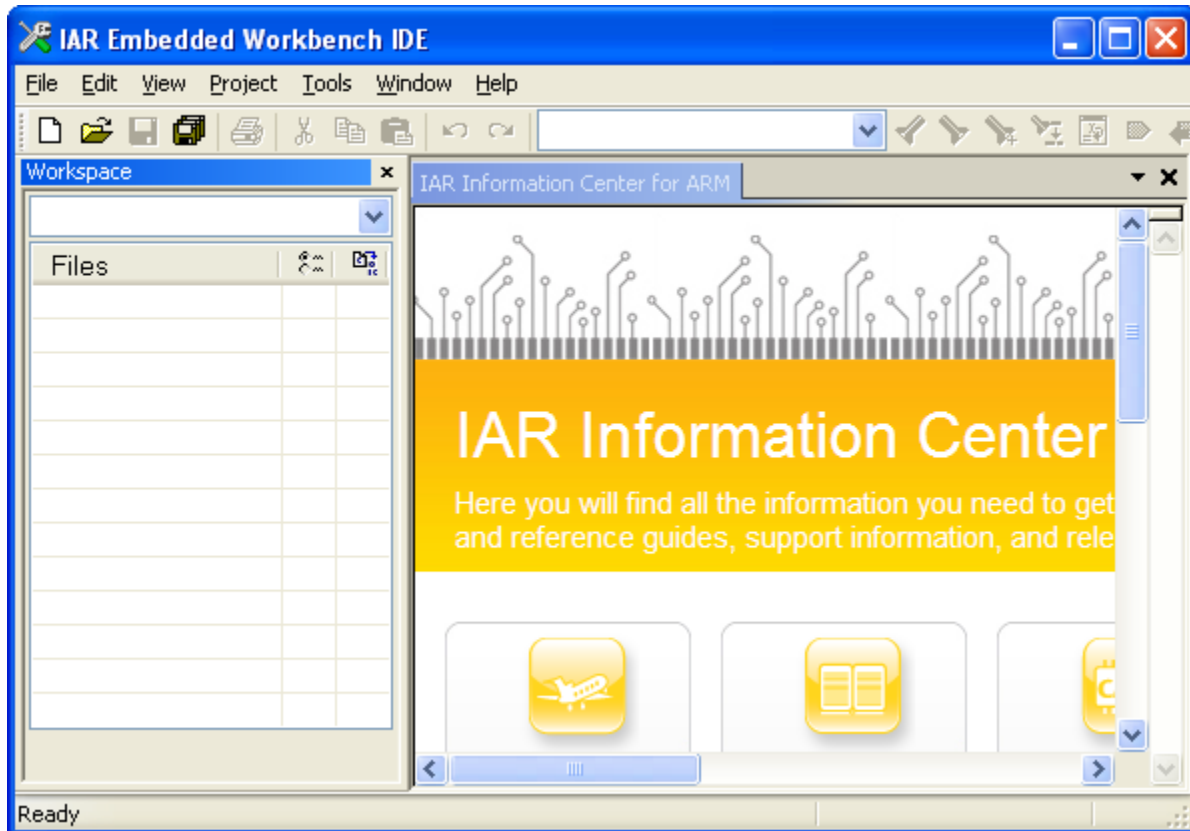


Figure 7 IAR

2. Add an USB stack library project by clicking **Project → Add Existing Project...**

You can find the corresponding IAR project files in the following paths:

- USB Device Stack
usb_core/device/build/iar/usbd_bm_twrk22f120m
- USB Host Stack
usb_core/host/build/iar/usbh_bm_twrk22f120m
- USB OTG Stack
usb_core/otg/build/iar/usbotg_bm_twrk22f120m

Notes: The OTG Stack is available for Full Speed Controller only. The OTG functionality doesn't work on the High Speed Controller.

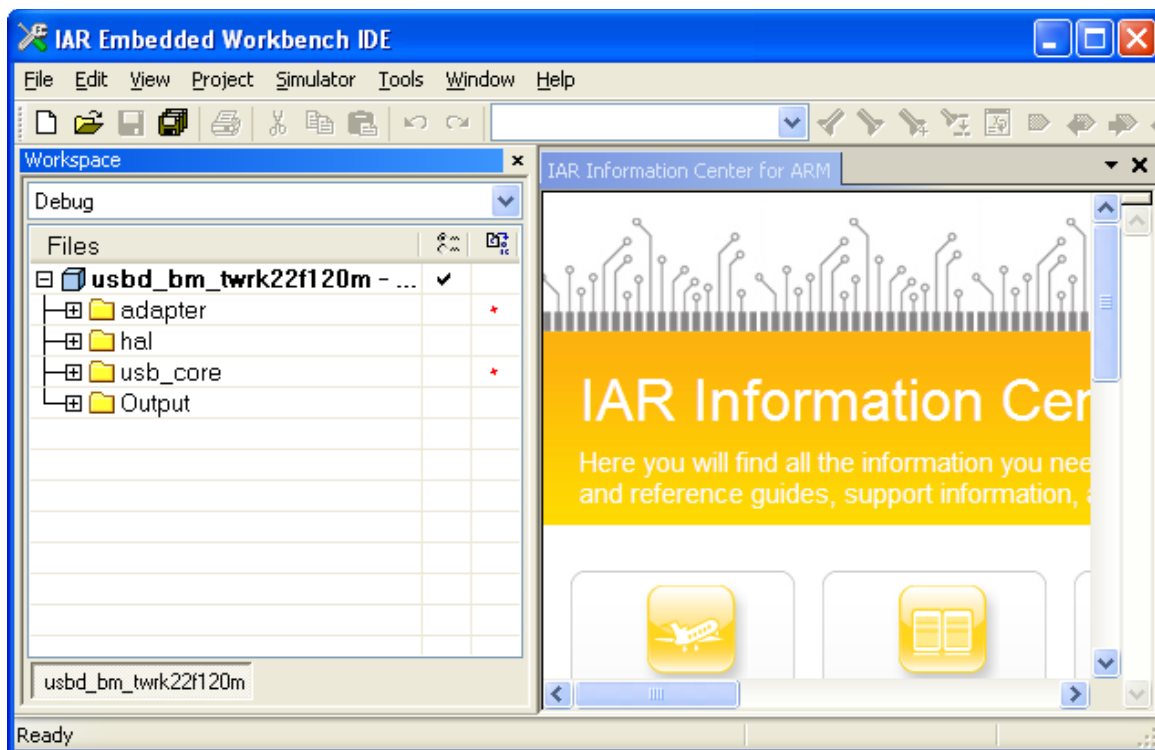


Figure 8 Folder name of USB device stack

3. Add a USB example project.

All the USB examples are located in the **example** folder. The folder structure is as follows.

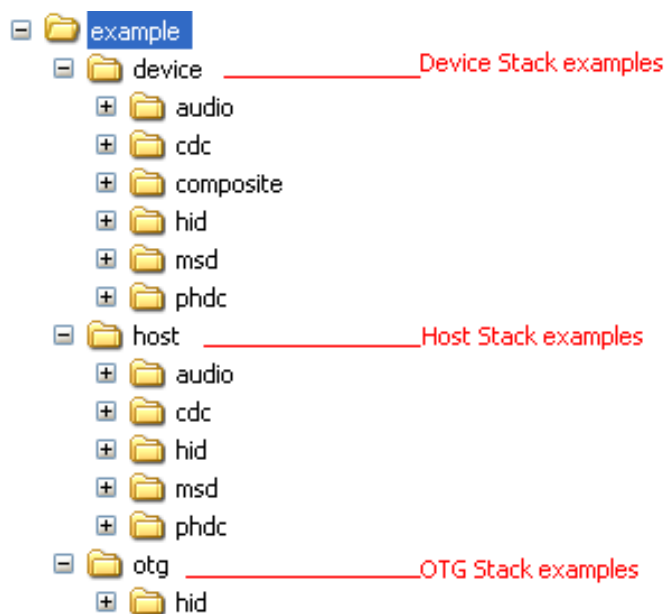


Figure 9 Example Folder Structure

This guide adds the USB device HID mouse example.

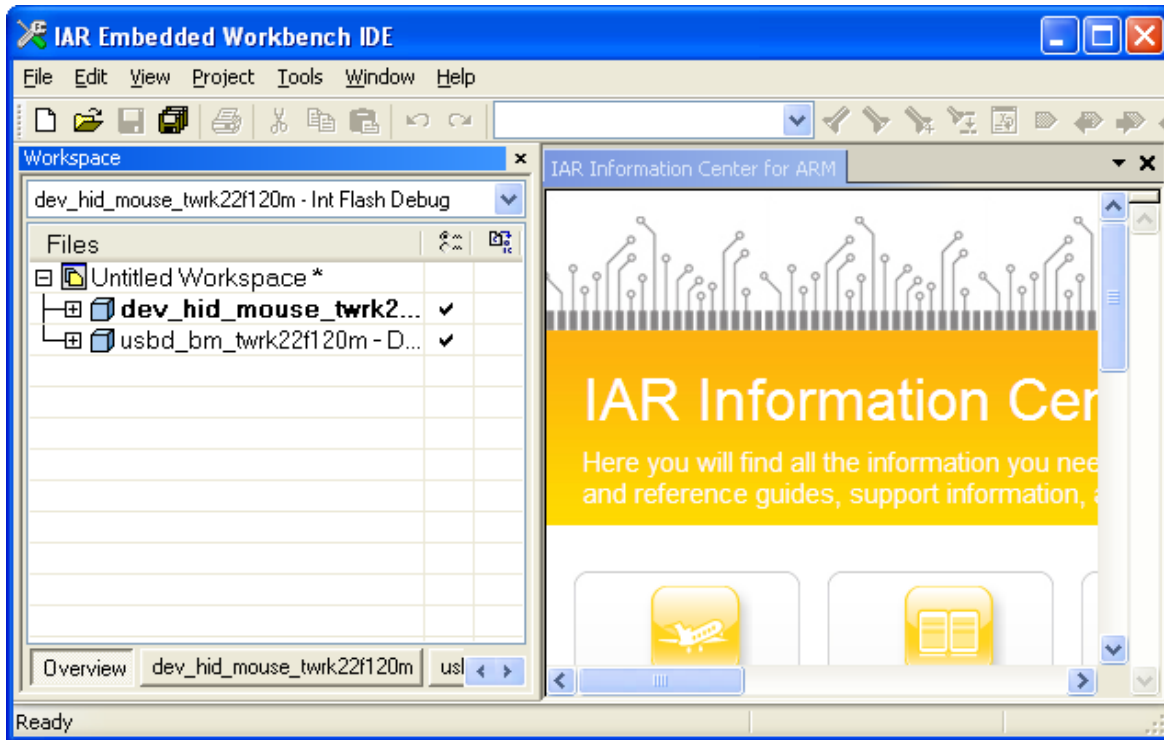


Figure 10 Adding a USB example project

4. Build the USB stack library.

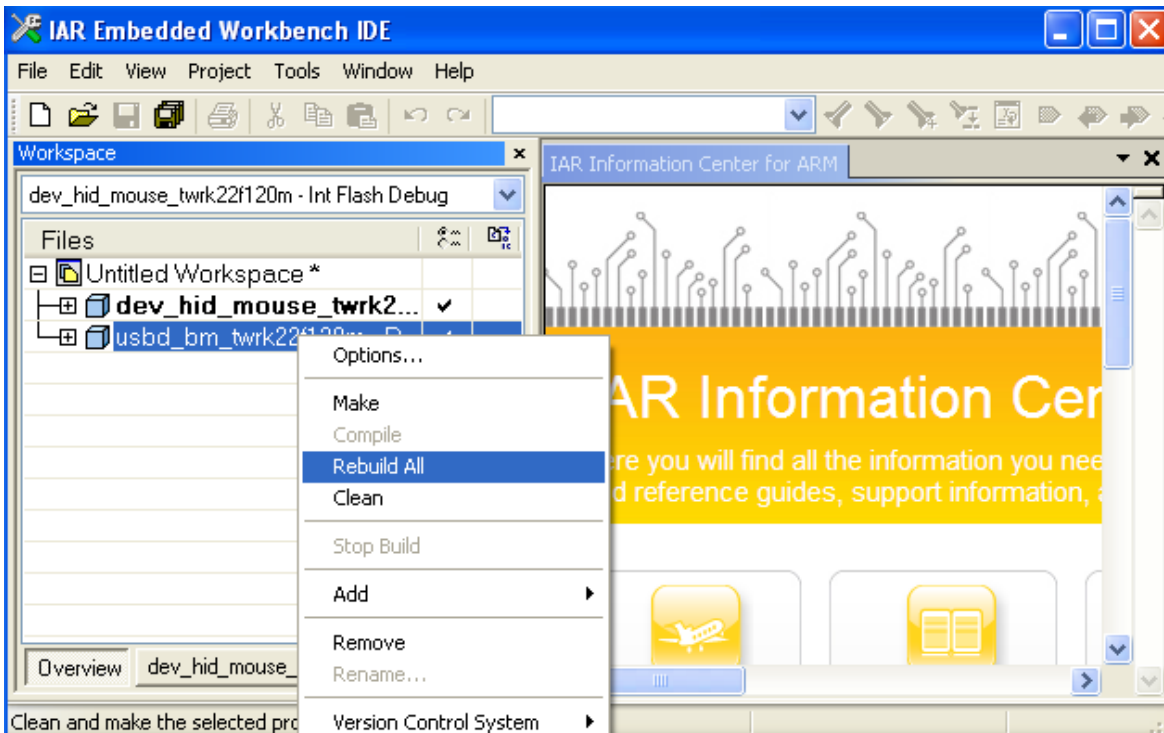


Figure 11 Building a USB stack library

A dialog box appears to ask you to save a workspace.

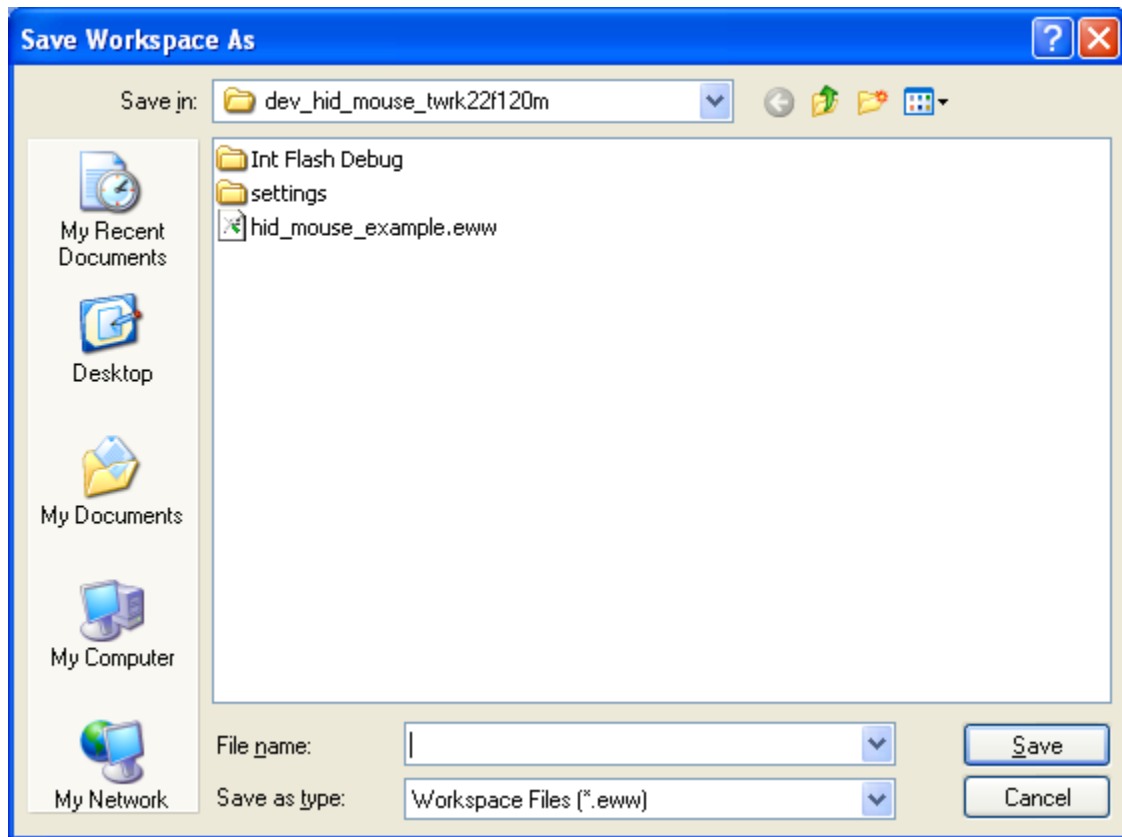


Figure 12 Save Workspace As dialog box

5. Check the USB library build result.
 - After the USB library is built, you can find the generated library binary file (usbd.a) under `output/twrk22f120m.iar/debug/usbd/bm/`.
 - In addition, all the USB related public header files are copied to this folder.
6. Build the USB device HID mouse example. The USB library must compile firstly. Otherwise, the build for the example project may fail.

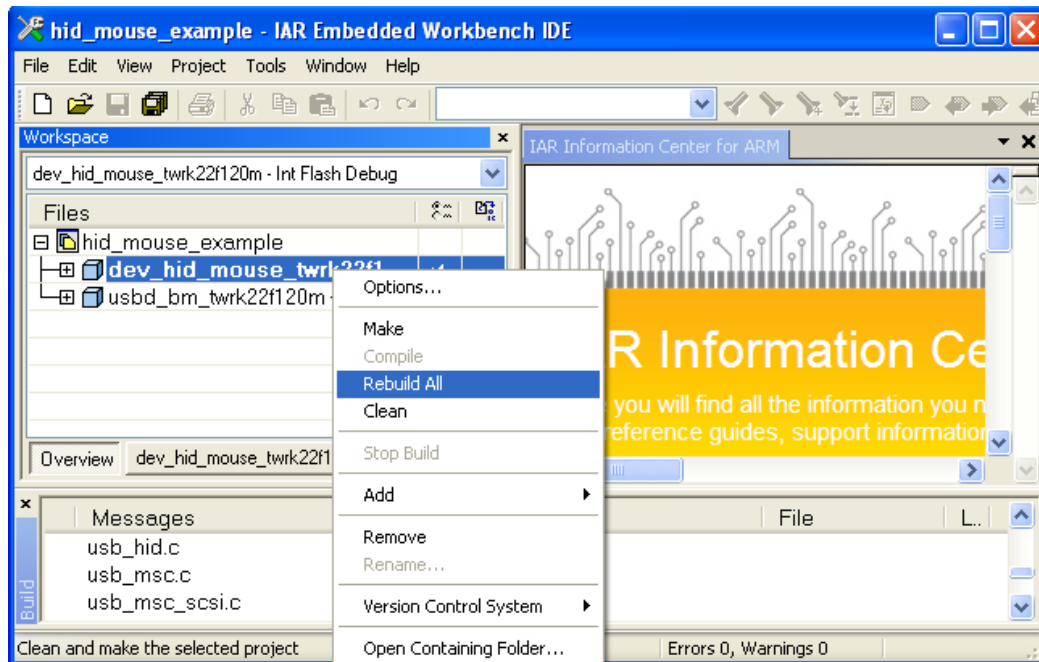


Figure 13 Building the USB device HID mouse example

7. Connect the J-Link to the JTAG port (J31) on TWR-K22F120M.

Notes: The KL26 and KL25 default debugger is PEmicro OpenSDA. You can refer to the *FRDM-KL26Z-QSG.pdf* file (in www.freescale.com) to load the PEmicro OpenSDA application.

8. Connect the micro USB cable from a PC to J25 of TWR-K22F120M to power on the board.
9. Click **Download and Debug**. Wait for the downloading to be finished.

For FRDM-K64, make sure that the CMSIS-DAP firmware on the board is updated and CMSIS-DAP is set as the debugger. Go to the website: <http://mbed.org/handbook/CMSIS-DAP-MDK> and download firmware here: <http://mbed.org/handbook/Firmware-FRDM-K64F>.

10. Click **Go** to run the example.
11. Connect the micro USB cable from a PC to the J32 port of TRW-K22F120M to enable the USB mouse device to work on the PC.

After the mouse device is enumerated by the PC, the mouse is active, and the mouse pointer draws a rectangle on the PC.

4.2 Additional actions for Keil

The compilation process for Keil is similar to that for IAR. This section focuses on the parts of the Keil downloading process that are different from IAR.

Before we can download the binary to the target board with Keil, we need to set a programming algorithm as follows.

1. Access the options for the target project by right-clicking the target project.

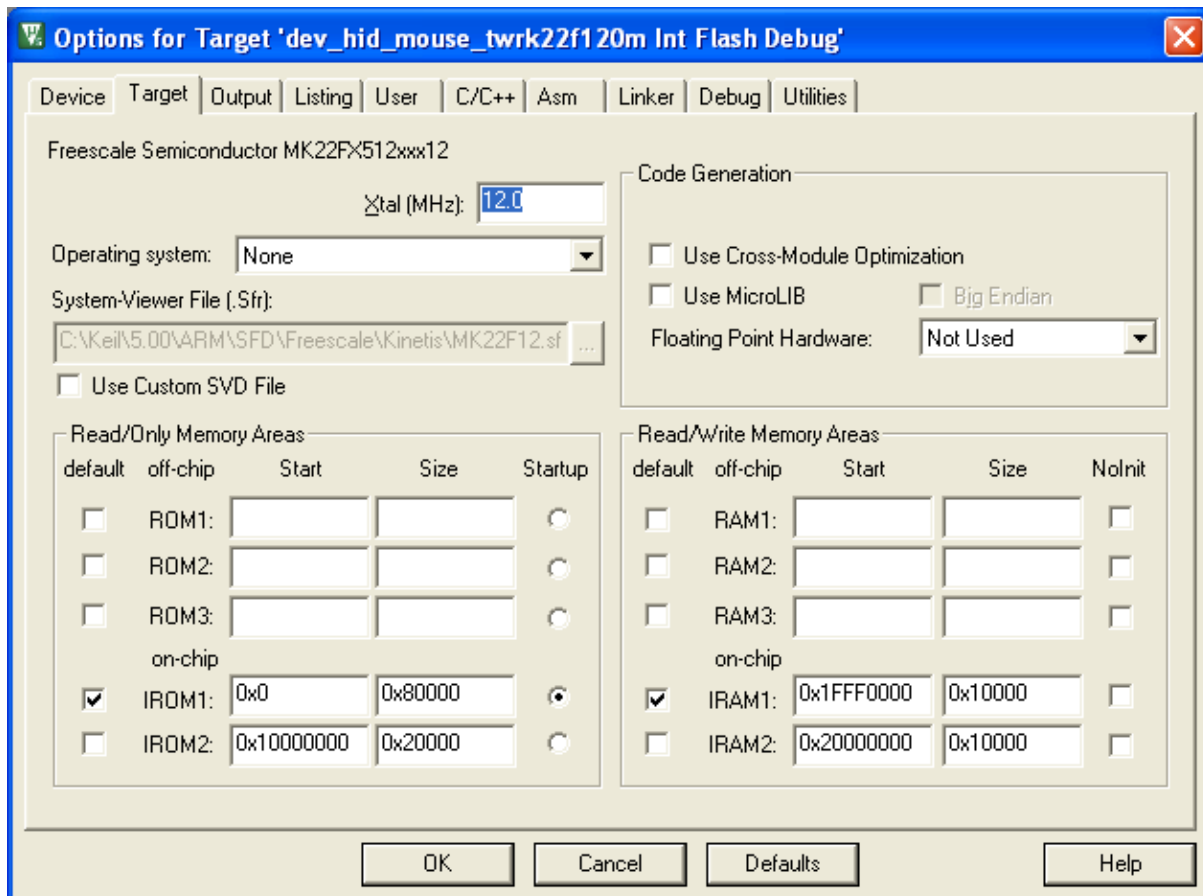


Figure 14 Options for the target project

2. Click the **Debug** tab.

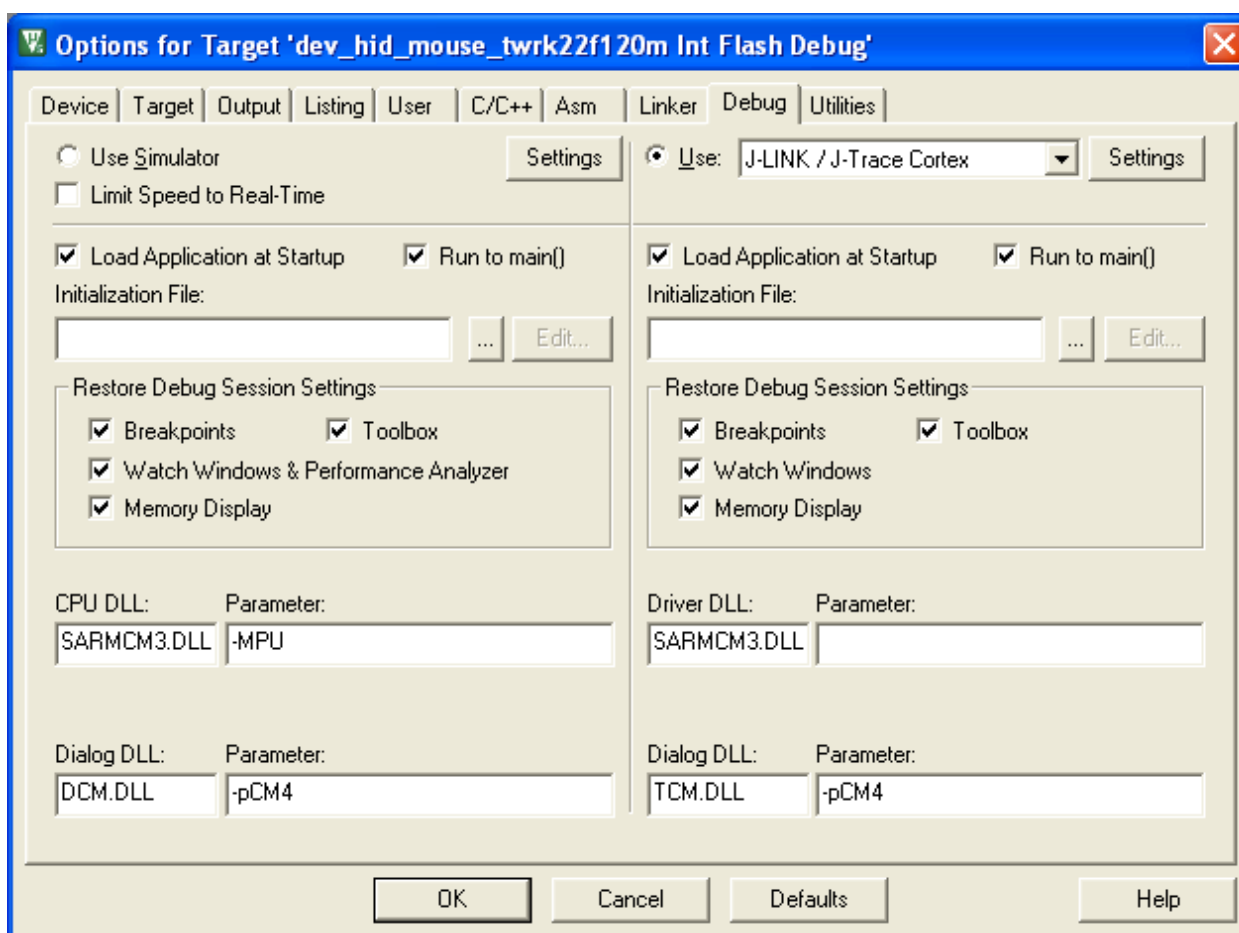


Figure 15 Debug tab

- Click **Setting** next to the **Use: J-Link/J-Trace Cortex** option. The **Cortex JLink/JTrace Target Driver Setup** dialog box appears.

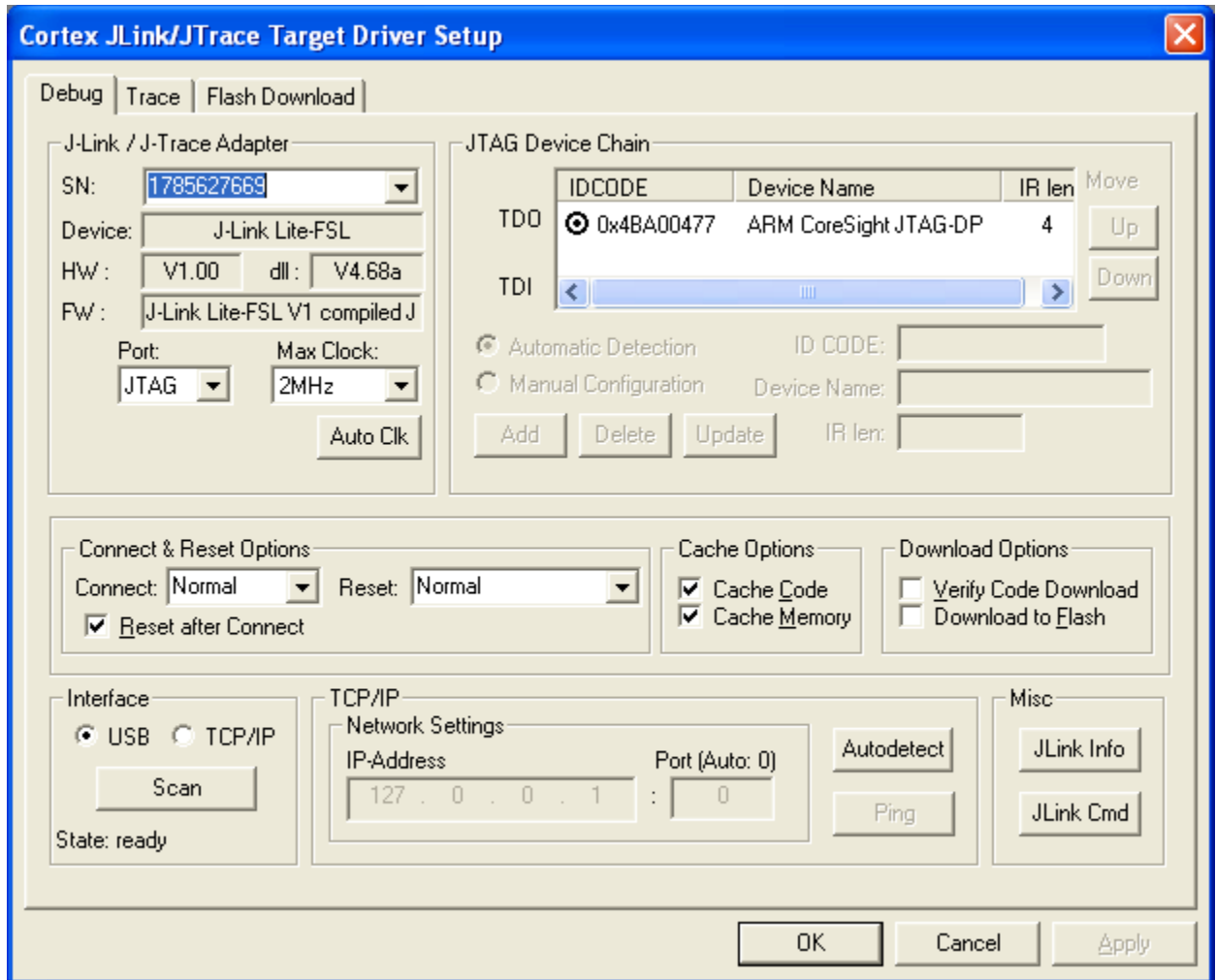


Figure 16 Cortex JLink/JTrace Target Driver Setup dialog box

For FRDM-K64, make sure that the CMSIS-DAP firmware on the board is updated and the CMSIS-DAP is set as the debugger. Go to the website: <http://mbed.org/handbook/CMSIS-DAP-MDK> and download firmware here: <http://mbed.org/handbook/Firmware-FRDM-K64F>.

- Click the **Flash Download** tab.



Figure 17 Cortex JLink/JTrace Target Driver Setup dialog box - Flash Download tab

5. Click **Add** and select **MKXX50Mhz 512kB Prog Flash**.

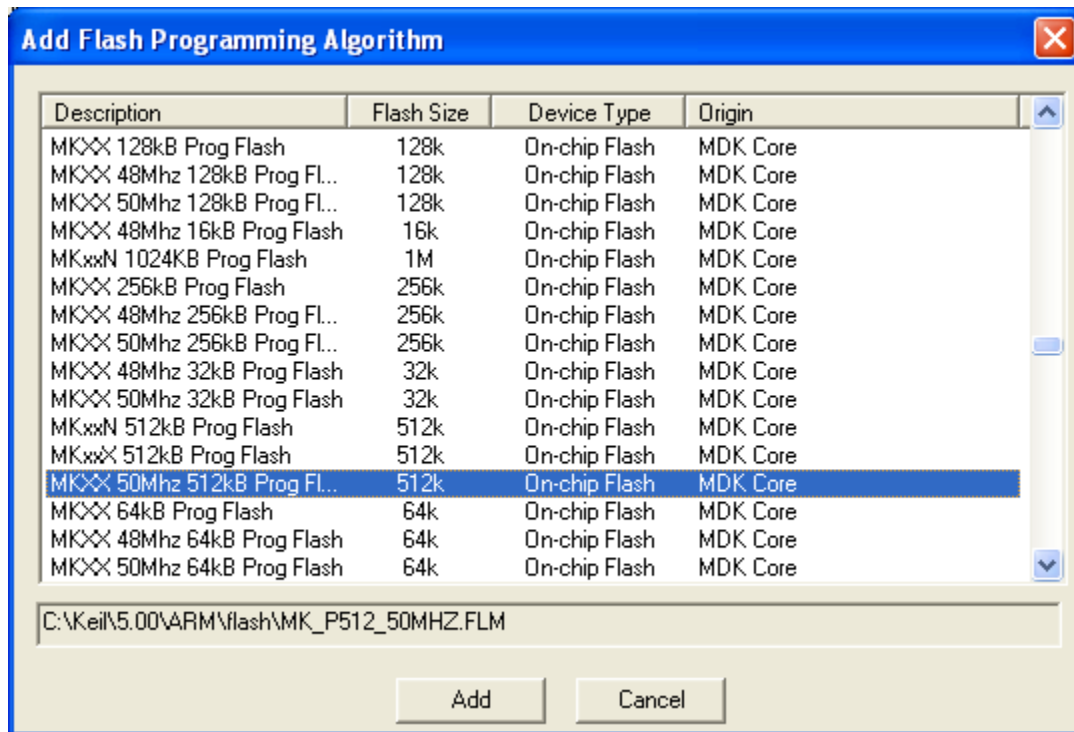


Figure 18 Selecting MKXX 50Mhz 512kB programming flash

4.3 Downloading GNU tools ARM embedded 4.7

The compilation process of GNU Tools ARM Embedded is similar to that of IAR and Keil. You need to access the corresponding folder and run “mingw32-make” to compile the project from the command line or just run the corresponding batch file “build_gcc_arm.bat” for each example.

Note: Before the GCC ARM compiler can work, you need to make sure that the TOOLCHAIN_ROOTDIR in src/build/common/make/global.mak is valid (By default, TOOLCHAIN_ROOTDIR is not configured). Otherwise, the compilation process will fail. In addition, the path should be in the short file name format. You can get the short file name by using the following command:

```
for %A in ("C:/Program Files/GNU Tools ARM Embedded/4.7 2013q3") do @echo %~sA
```

The string C:/Program Files/GNU Tools ARM Embedded/4.7 2013q3 in the above command should be replaced by the correct target long file name.

Some strange issues may occur when the default installation path of the GCC Tool Chain is changed, hence it is recommended not to change it. In addition, make sure that the **mingw32-base** and **msys-base** packages are installed in your system and the corresponding path (MINGW/bin) is added into the system path, and MINGW/msys/1.0/bin is not added into the system path.

The downloading steps are as follows:

1. Run **J-Link GDB Server**.

This application is installed along with the J-Link. Select MK22FN512xxx12 as the target device and click **OK**.

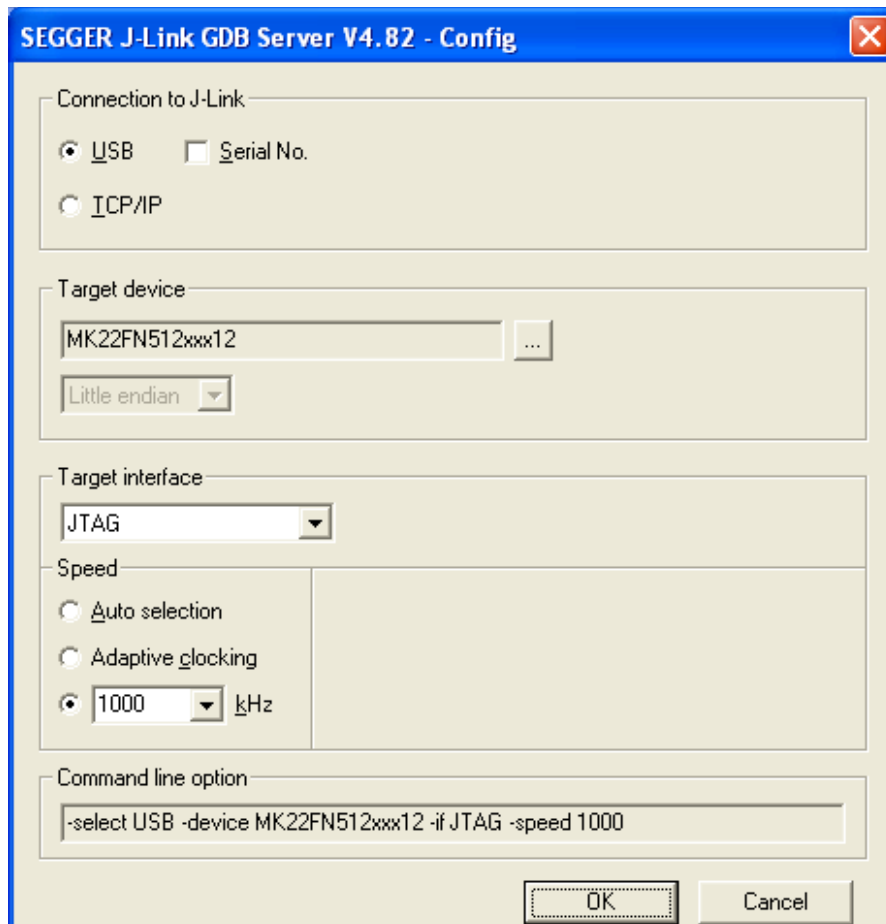


Figure 19 J-Link GDB Server Configuration dialog box

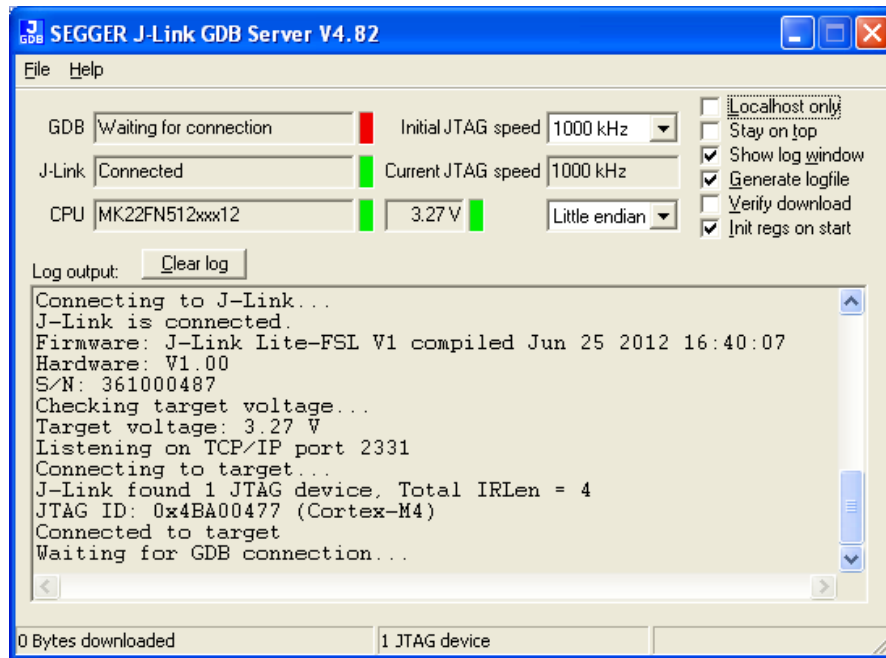


Figure 20 J-Link GDB server configuration result

2. Run **arm-none-eabi-gdb** under the folder where the target binary is located.

In the example, this folder is under the following path:

example/device/hid/hid_mouse/bm/make/dev_hid_mouse_twrk22f120m/gcc_arm/512KB_Pflash_DEV_release

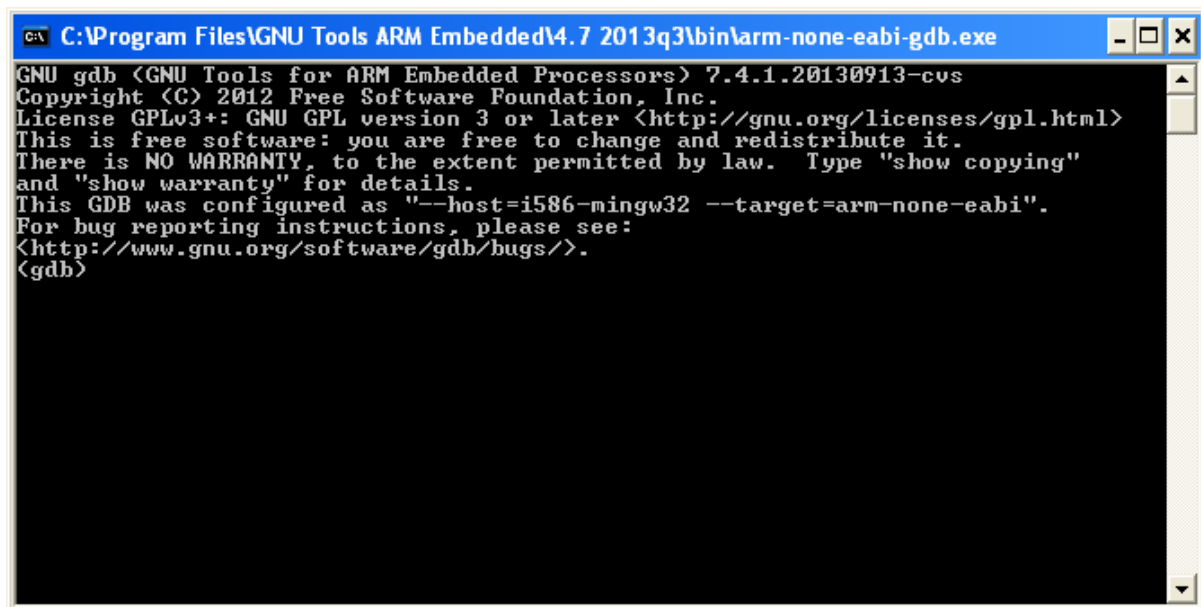
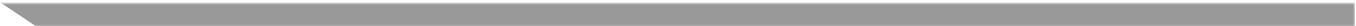


Figure 21 arm-none-eabi-gdb folder

3. On the gdb client, run the following commands:

```
target remote localhost:2331
```



```
monitor reset
monitor flash device = MK22FN512xxx12
load dev_hid_mouse_twrk22f120m.elf
monitor reg pc = (0x00000004)
monitor reg sp = (0x00000000)
monitor go
```

The mouse becomes active on the PC.

5 USB Stack Configuration

5.1 Device configuration

All the device configurations are listed in the following file:

```
usb_core/device/include/BOARD_NAME/usb_device_config.h
```

BOARD_NAME could be “twrk22f120m”, “frdmkl26z”, or other board names.

We can enable or disable the USB class driver through this file, and we can configure the object number to decrease the memory usage or increase the object number to meet some specific requirements.

If you change the configuration of the device stack, both the USB library project and the example project need to be re-built.

Notes

The composite device examples work only with:

```
USBCFG_DEV_COMPOSITE    1
```

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

```
USBCFG_DEV_COMPOSITE    0
```

If incorrect settings are configured, a build error will occur and will need to be modified.

5.2 Host configuration

All the host configurations are listed in the following file:

```
usb_core/host/include/BOARD_NAME/usb_host_config.h
```

BOARD_NAME could be “twrk22f120m”, “frdmkl26z”, or other board names.

We can enable or disable the USB class driver through this file, and we can configure the object number to decrease the memory usage or increase the object number to meet some specific requirements.

If you change the configuration of the host stack, both the USB library project and the example project need to be re-built.

Notes

There are two USB receptacles available for TWR-K22F120M if the TWR-SER and elevator are used. We need to configure both SW and HW to switch between the two USB receptacles.

- To use the micro receptacle on the TWR-K22F120M, the jumper settings should be (for both device and host):
 - J4 1-2
 - J27 2-3 (for rev. A)
 - J27 1-2 (for rev. B)

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 board, the jumper settings should be (for both device and host):
 - J4 1-2
 - J27 1-2 (for rev. A)
 - J27 2-3 (for rev. B)
 - Please refer to the TWR-SER user guide to get the jumper setting on TWR-SER board

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

```
USBCFG_HOST_PORT_NATIVE    0
```

There is no such configuration for the device, because switching between the two USB receptacles doesn't require changing any code in the device mode.

5.3 OTG configuration

All the OTG configurations are listed in the following files:

```
usb_core/device/include/twrk22f120m/usb_device_config.h
usb_core/host/include/twrk22f120m/usb_host_config.h
```

We can enable or disable the USB class driver through these files, and we can configure the object number to decrease the memory usage or increase the object number to meet some specific requirements.

If you change the configuration of the OTG stack, both the USB library project and the example project need to be re-built.

Notes

The OTG example for TWR-K22F120M requests to use the mini receptacle on the TWR-SER board, the jumper settings should be:

- J4 1-2
- J27 1-2 (for rev. A)
- J27 2-3 (for rev. B)
- Please refer to the TWR-SER user guide to get the jumper setting on TWR-SER board

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
```

5.4 Example specific configuration

Some configurations items can be specific for each single example,

For USB Device HID mouse example, you can find them in

```
example/device/hid/hid_mouse/bm/COMPILE_TOOL/dev_hid_mouse_twrk22f120m/user_config.h
```

COMPILE_TOOL could be iar, uv4, or make.

You can enable or disable the printf functionality or time delay functionality or other configurations through this file.

In addition, you can change the HIGH_SPEED macro (you can find it in corresponding example head file) manually to switch between High Speed controller and Full Speed controller. By default, this HIGH_SPEED macro is set to 0 in all the platforms, and it can be set to 1 only on the platforms which have high speed controller like TWR-K70F120M.

If you change the configuration in the user_config.h, the USB library project does not need to be re-built, only the example project needs to be re-built.

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, Kinetis, and the Freescale logo are trademarks of Freescale Semiconductor, Inc., Reg. U.S. Pat. & Tm. Off. All other product or service names are the property of their respective owners. The ARM Powered Logo is a trademark of ARM Limited.

©2014 Freescale Semiconductor, Inc.