Freescale Semiconductor
User's Guide

Document Number: MQXTWRK22512RUG
Rev. 1.0, 04/2014

Freescale MQX USB Stack for TWR-K22F120M512R GA 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. This document also provides the board-specific information related to TWR-K22F120M512R.

Contents

2 Requirements for Building USB Examples

2.1 Hardware

- TWR-K22F120M System module Rev. B with a PK22FN512VDC processor
- (Optional) TWR-SER and Elevator
- J-Link debugger
- USB cables

2.2 Software

- Freescale MQX RTOS For TWR-K22F120M512R release package
- IAR Embedded Workbench for ARM Version 6.70.3, available for Kinetis devices
- Keil uVision4 Integrated Development Environment Version 5.0.5.15, available for Kinetis ARM® CortexM4 devices
- GNU Tools for ARM Embedded Processors 4.7 2013Q3
- MinGW v3.82.90 with the **mingw32-base** and **msys-base** packages installed
- J-Link ARM dll with the latest version

2.3 Board jumper settings

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

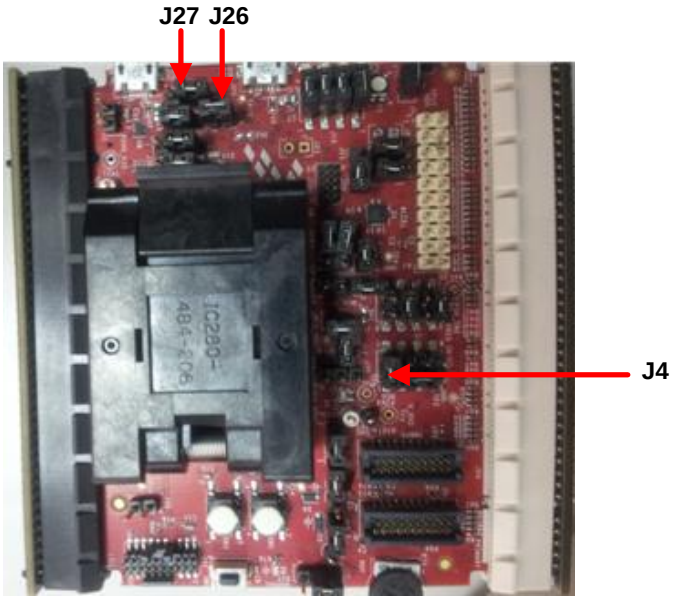


Figure-1 Board jumper settings

- 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

3 USB Code Structure

The USB stack is located in the usb_v2 subfolder of the MQX root folder. There are five subfolders in it:

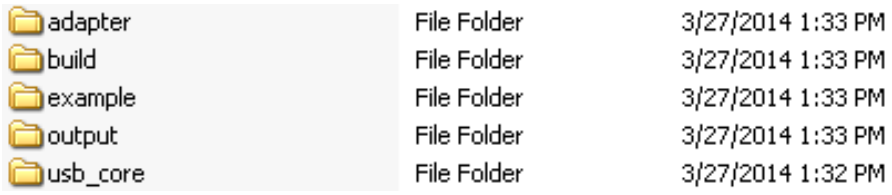


Figure-2 usb_v2 folder structure

- 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.

- output

  Contains the generated USB library binary file and all USB-related public header files. The examples need to include this folder to the `include` path in the project settings.

- usb_core

  Includes 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.

1. Open IAR as follows.

Figure-3 IAR

2. Add MQX bsp and psp projects by clicking **Project → Add Existing Project**.
   You can find the corresponding IAR project files in the following paths:

   - bsp
     `<install_dir>/mqx/build/iar/bsp_twrk22f120m`
   - psp
     `<install_dir>/mqx/build/iar/psp_twrk22f120m`

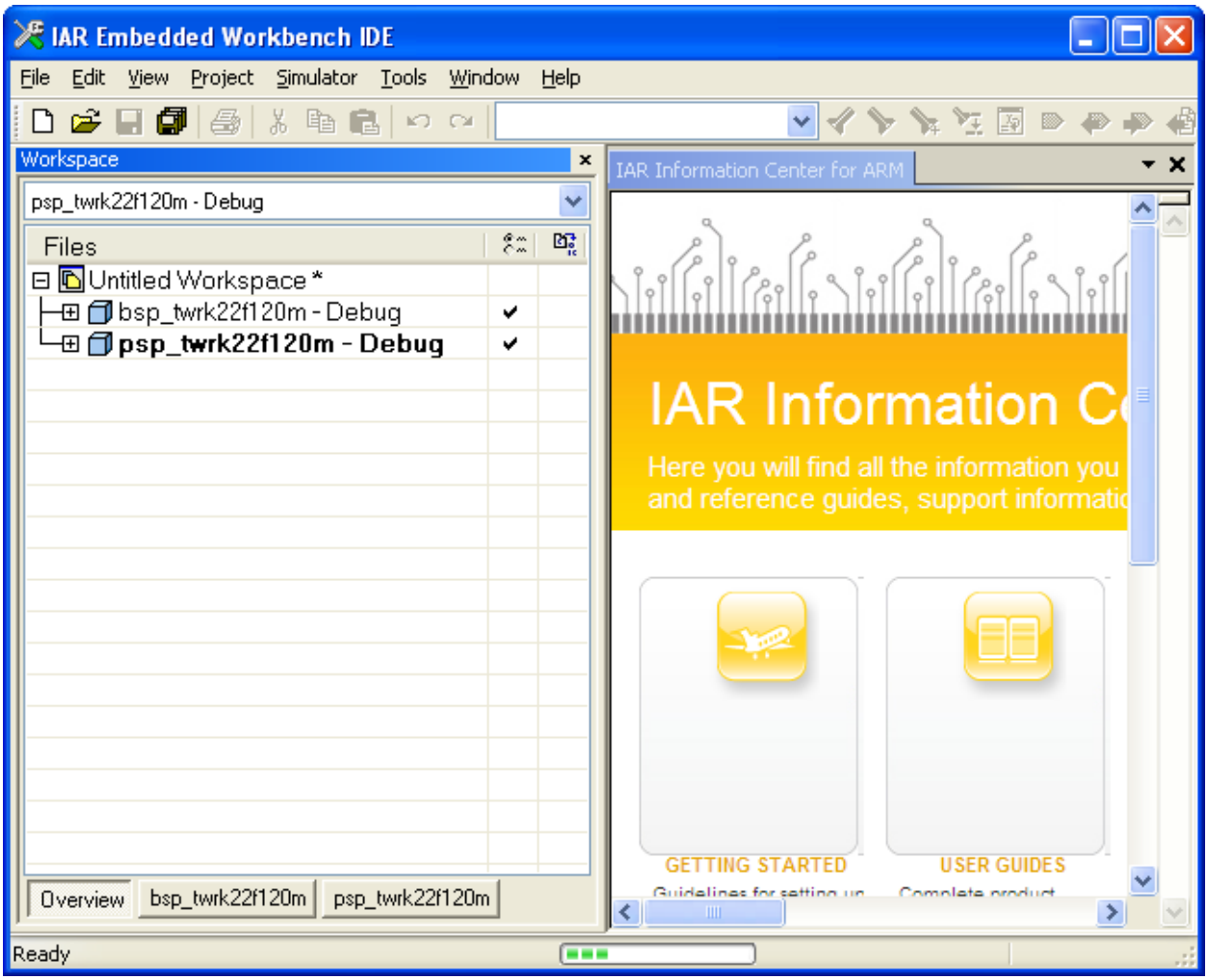


Figure-4 bsp and psp projects

3. Add a USB stack library project in the following paths:

   - USB Device Stack
     `<install_dir>/usb_v2/usb_core/device/build/iar/usbd_mqx_twrk22f120m`
   - USB Host Stack
     `<install_dir>/usb_v2/usb_core/host/build/iar/usbh_mqx_twrk22f120m`
   - USB OTG Stack
     `<install_dir>/usb_v2/usb_core/otg/build/iar/usbotg_mqx_twrk22f120m`

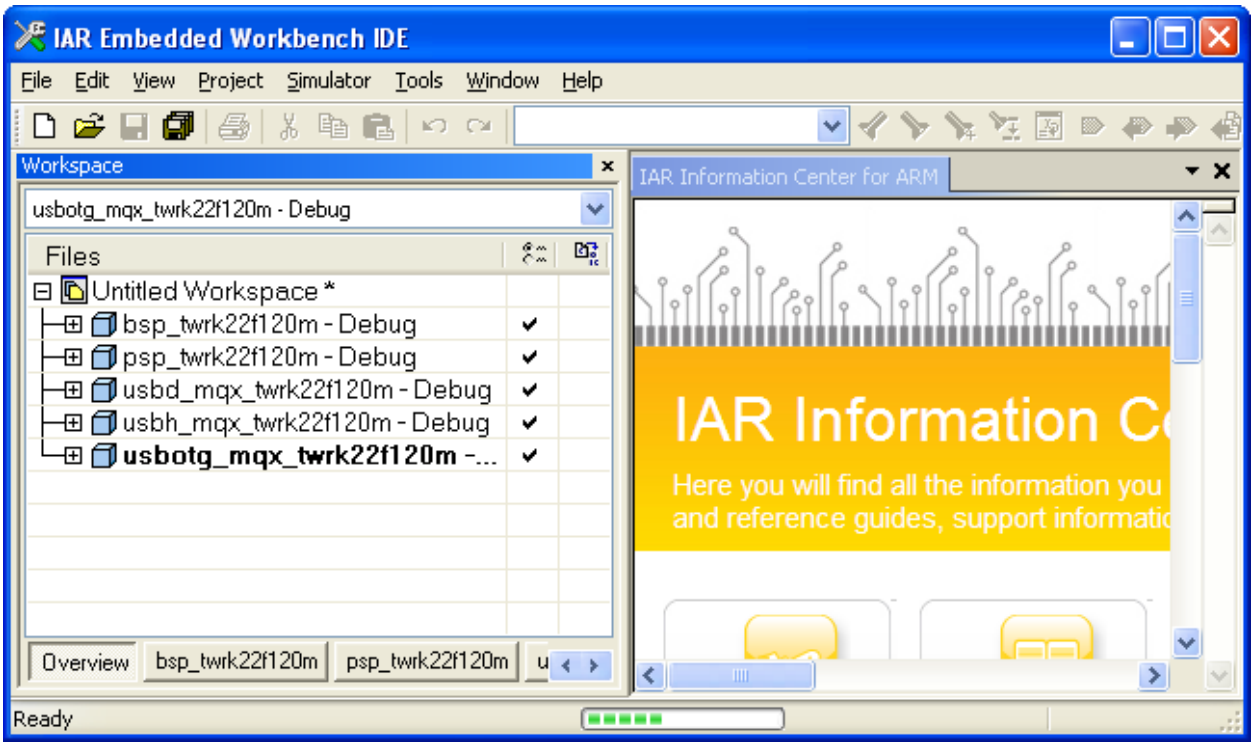


Figure-5 Folder name of USB device stack

4. Add a USB example project.

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

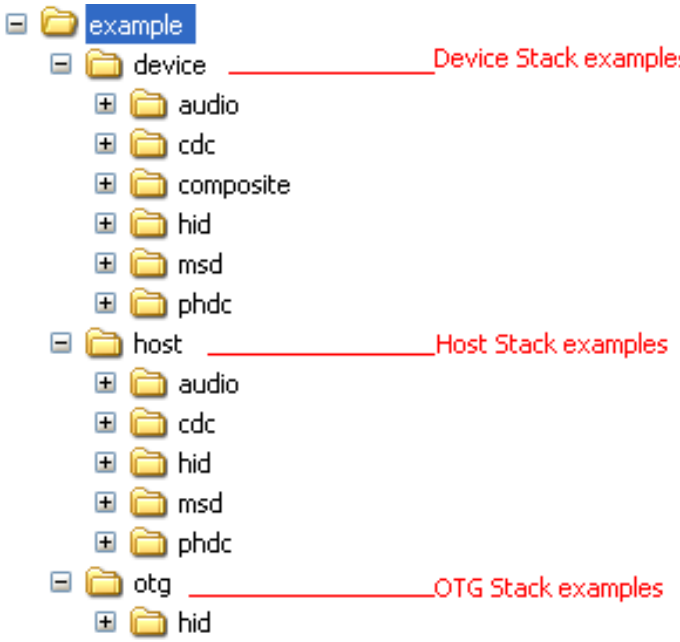


Figure-6 Example Folder Structure

This guide adds the USB device HID mouse example.

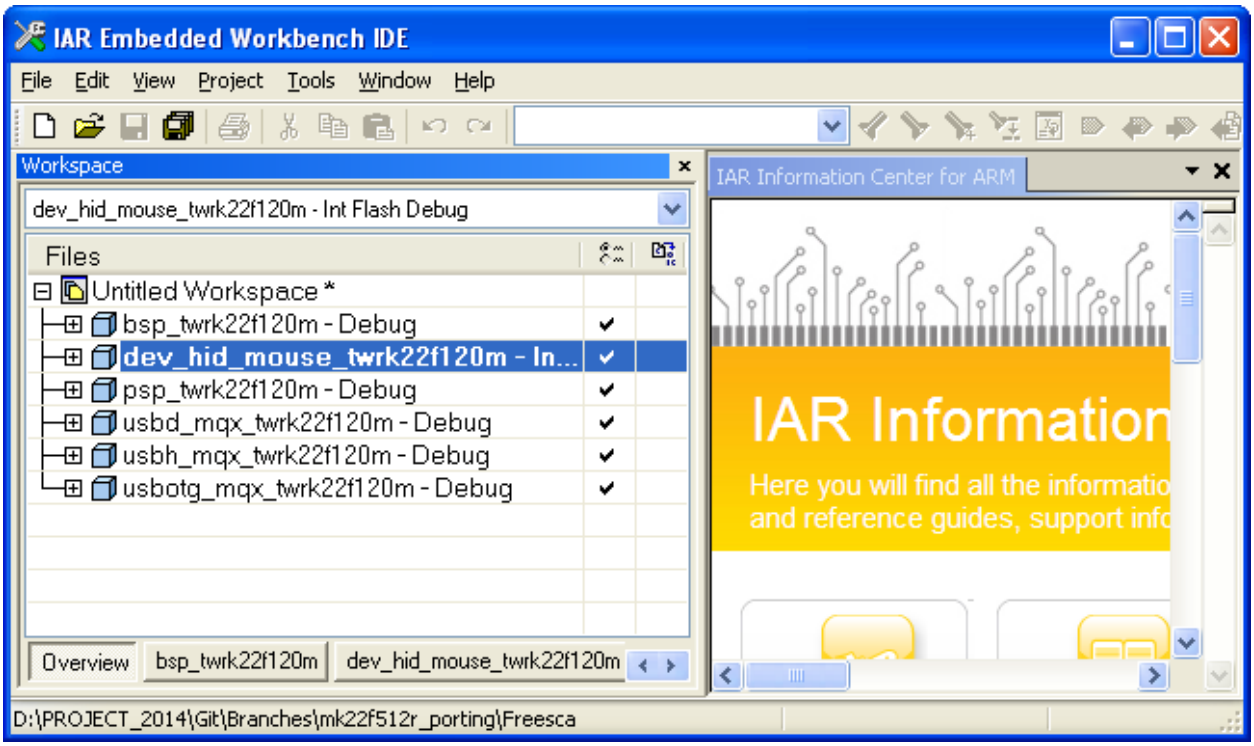


Figure-7 Adding a USB example project

5. Build the bsp and psp libraries.

A dialog box appears to ask you to save a workspace. Save the workspace and proceed with the builds.

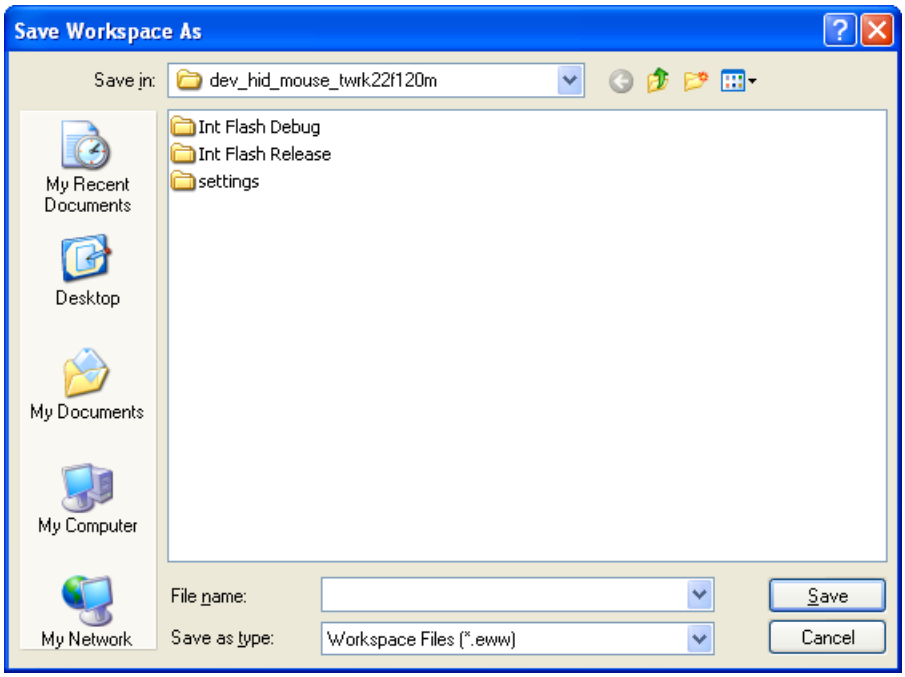
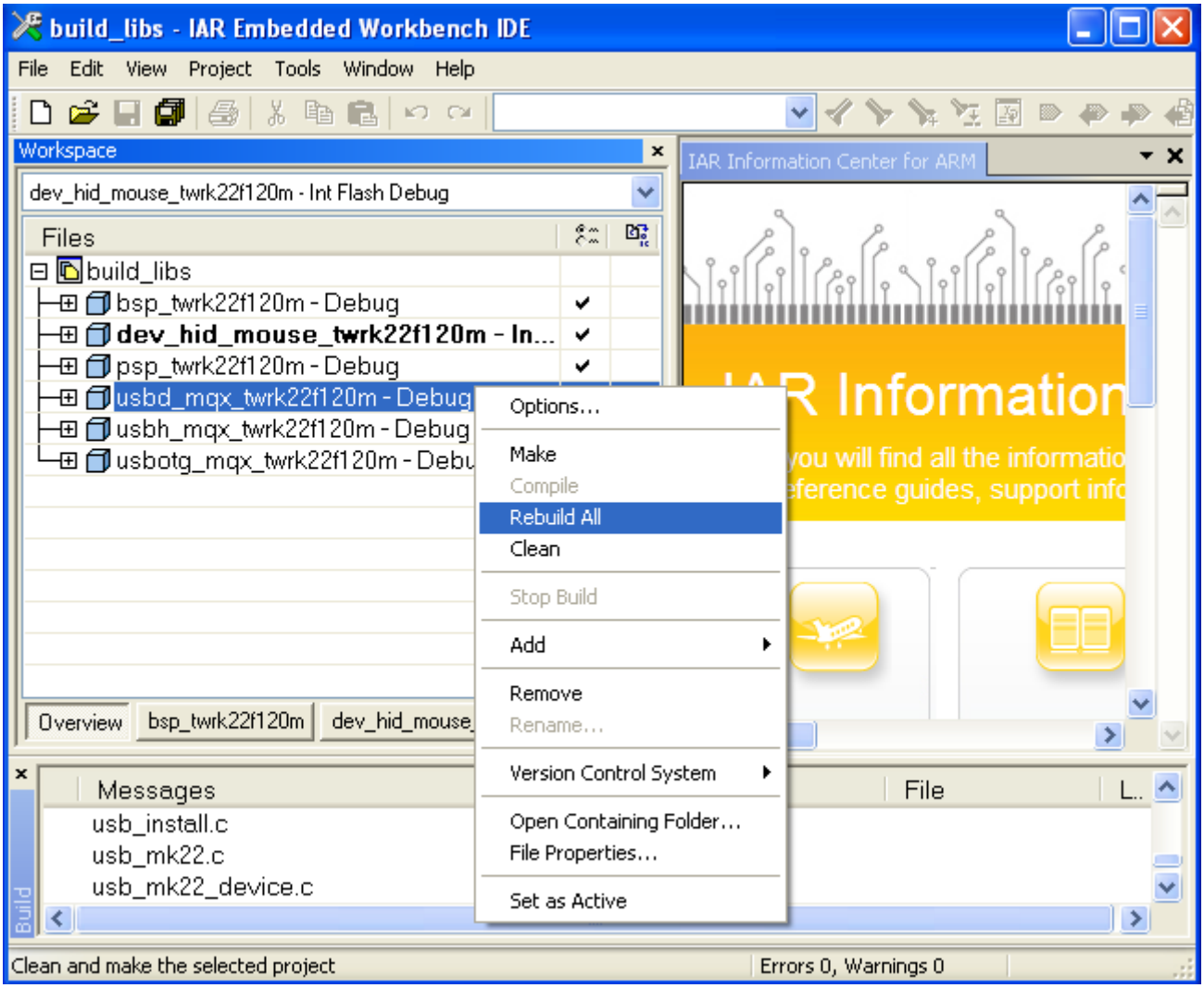


Figure-8 Save Workspace As dialog box

6. Build a USB stack library.



7. Check the USB library build result.
   - After the USB library is built, you can find the generated library binary file (usbd.a) under `<install_dir>/usb_v2/output/twrk22f120m.iar/debug/usbd/mqx/`.
   - In addition, all the USB related public header files are copied to this folder.
8. Build the USB device HID mouse example.

   Notice that the build for the example may fail if the USB library fails to compile properly in the previous step.

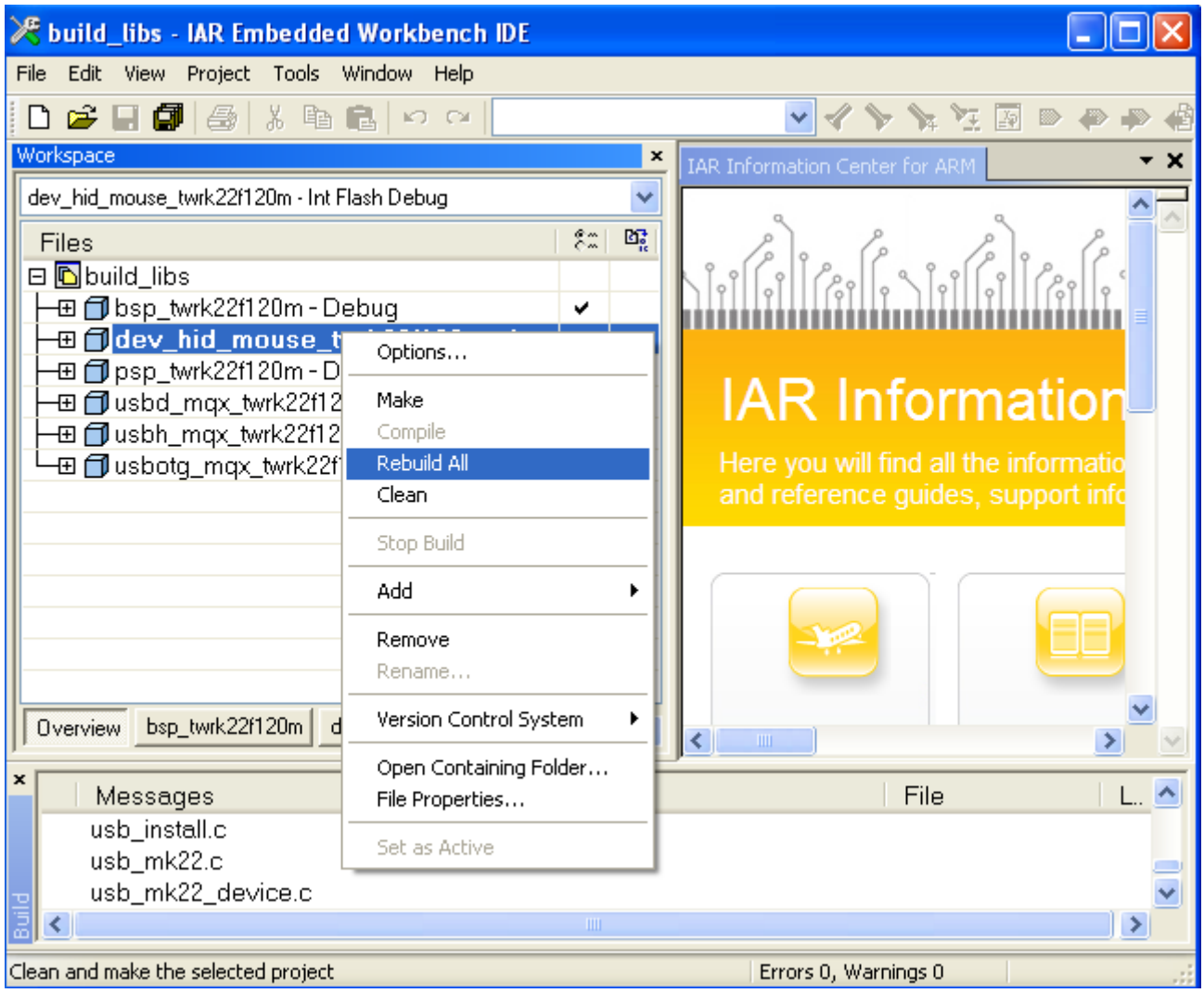


Figure-9 Building the USB device HID mouse example

9. Connect the J-Link to the JTAG port (J31) on TWR-K22F120M.
10. Connect the micro USB cable from a PC to J25 of TWR-K22F120M to power on the board.
11. Click **Download and Debug**. Wait for the downloading to be finished.
12. Click **Go** to run the example.
13. 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 will be 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.

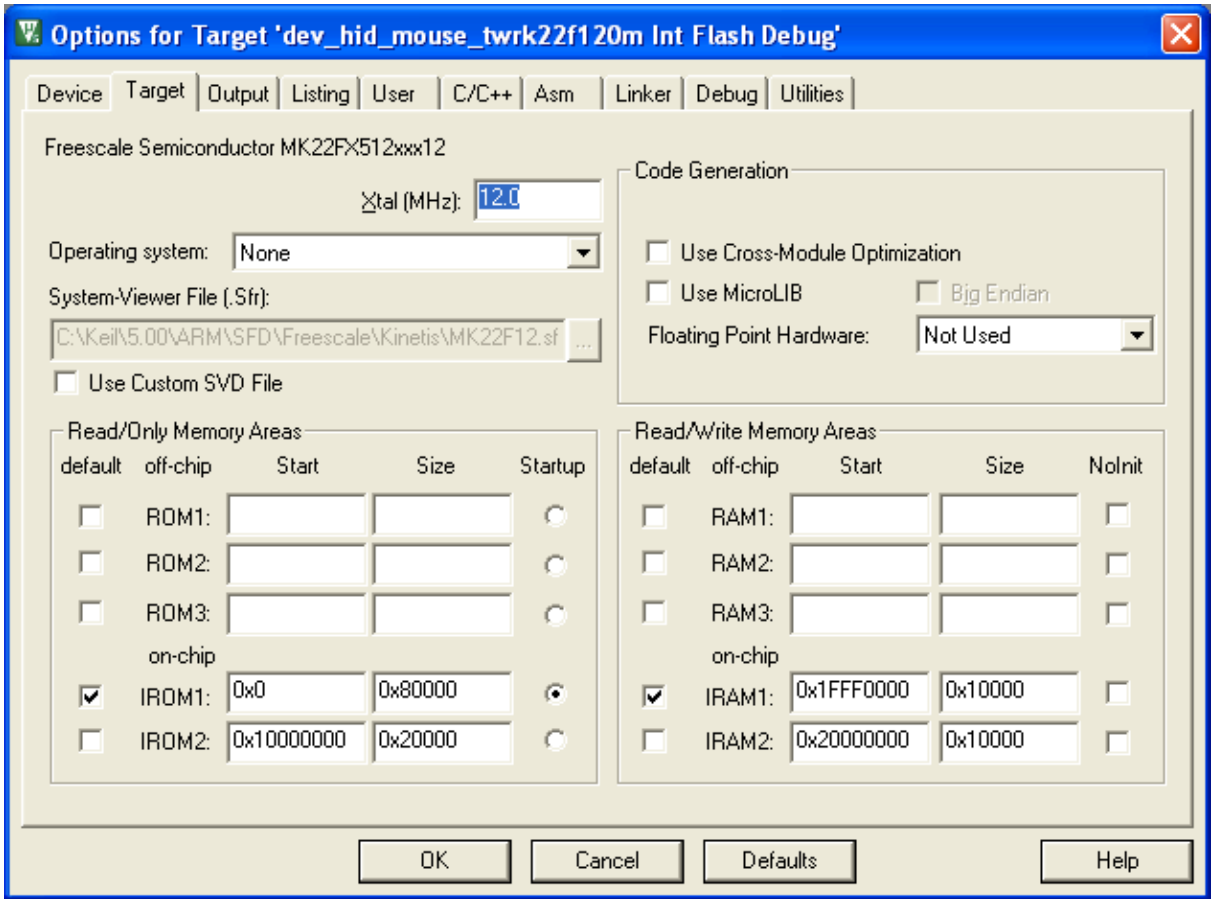


Figure-10 Options for the target project

2. Click the **Debug** tab.

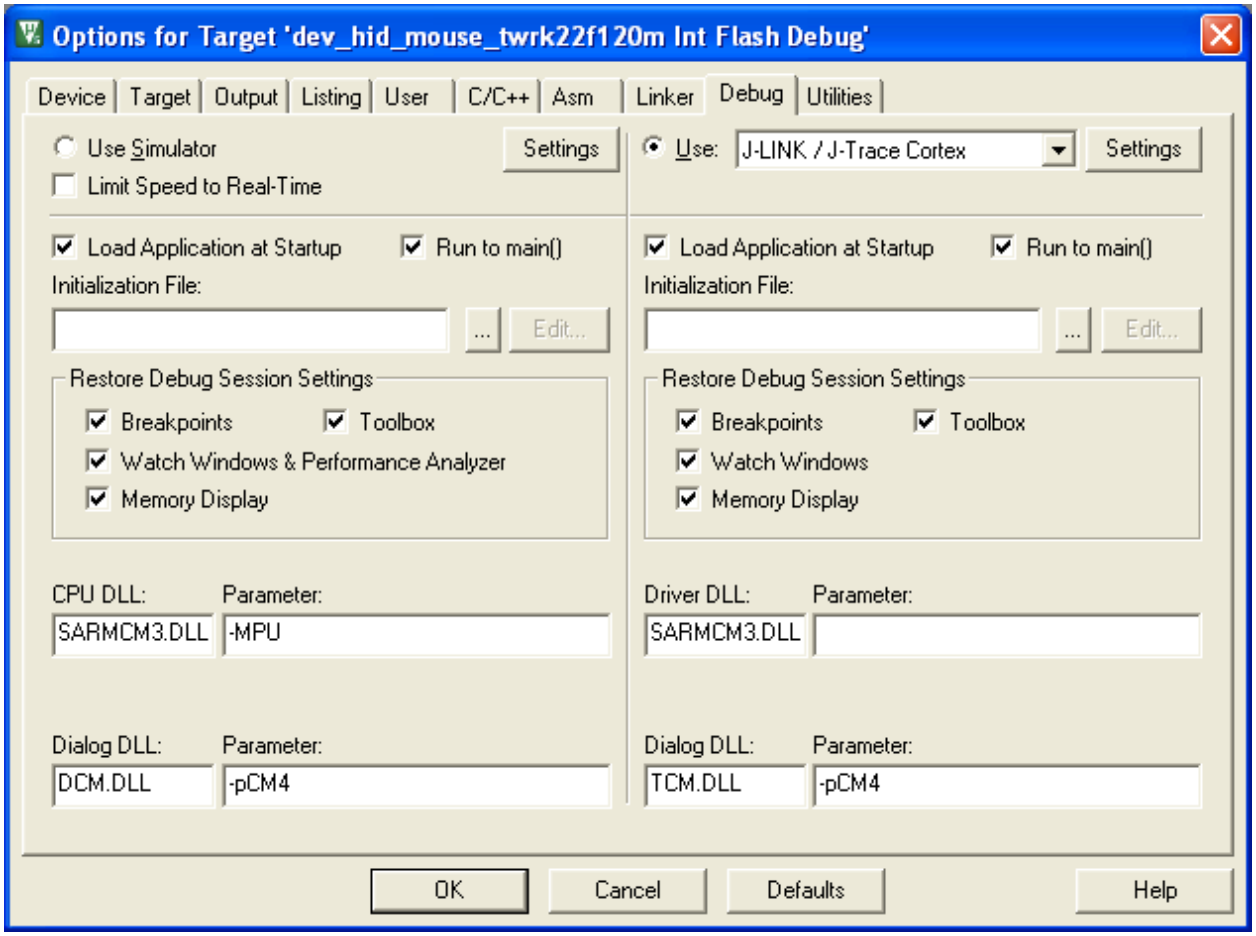


Figure-11 Debug tab

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

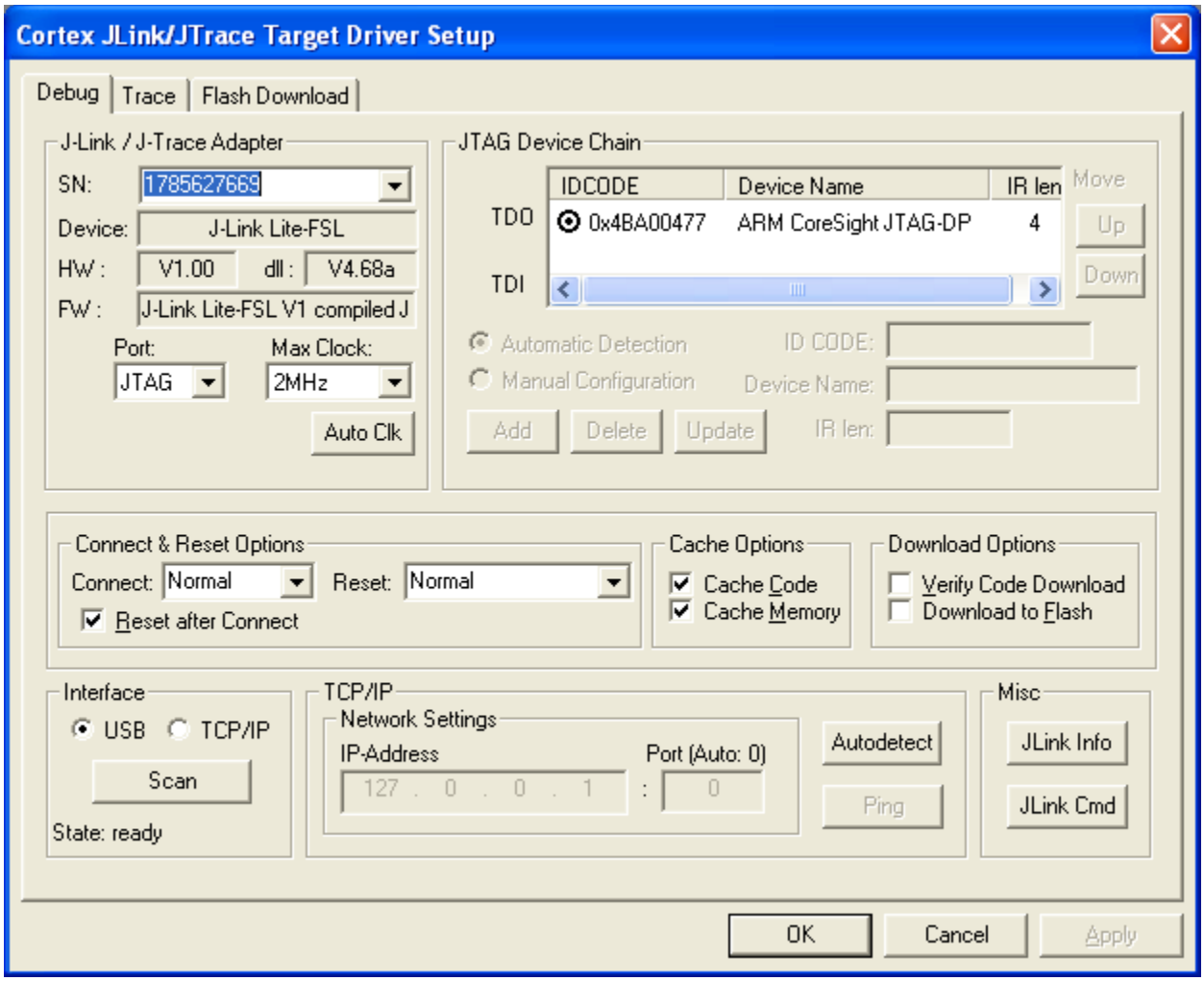


Figure-12 Cortex JLink/JTrace Target Driver Setup dialog box

4. Click the **Flash Download** tab.

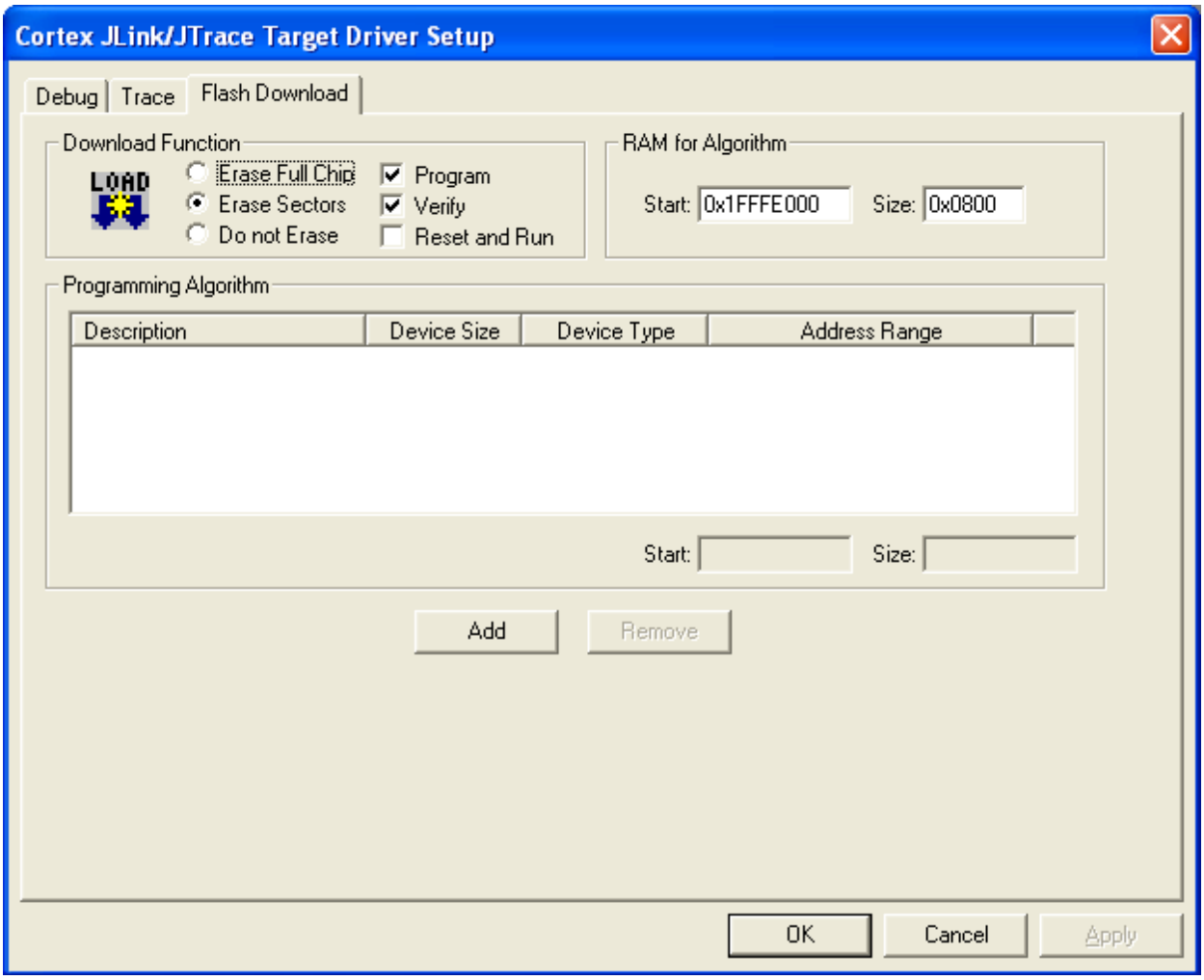


Figure-13 Cortex JLink/JTrace Target Driver Setup dialog box - Flash Download tab

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

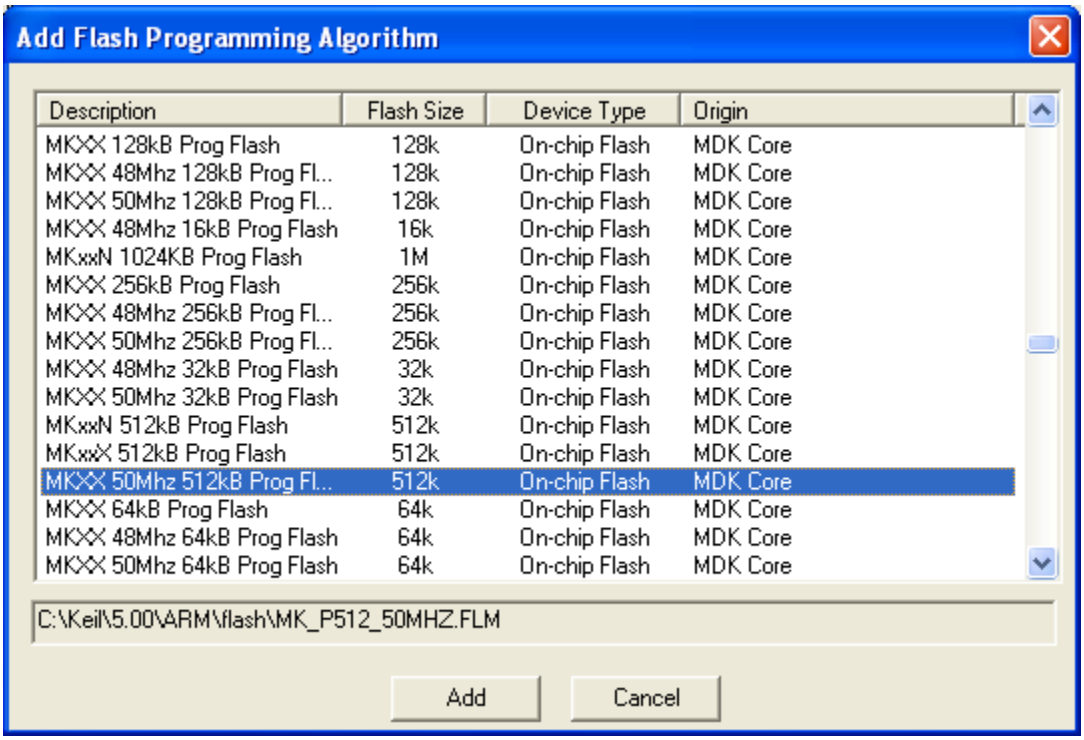


Figure-14 Selecting MKXX 50Mhz 512kB programming flash

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 in command line or just run the corresponding batch file "build_gcc_arm.bat" for each example.

Note: Before gcc arm compiler can work, it has to make sure the GCC_TOOLCHAIN_DIR in build\common\make\global.mak is valid, by default GCC_TOOLCHAIN_DIR is not configured at all. 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, so 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 has been added into the system path (`MINGW\bin`, `MINGW\msys\1.0\bin`).

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**.

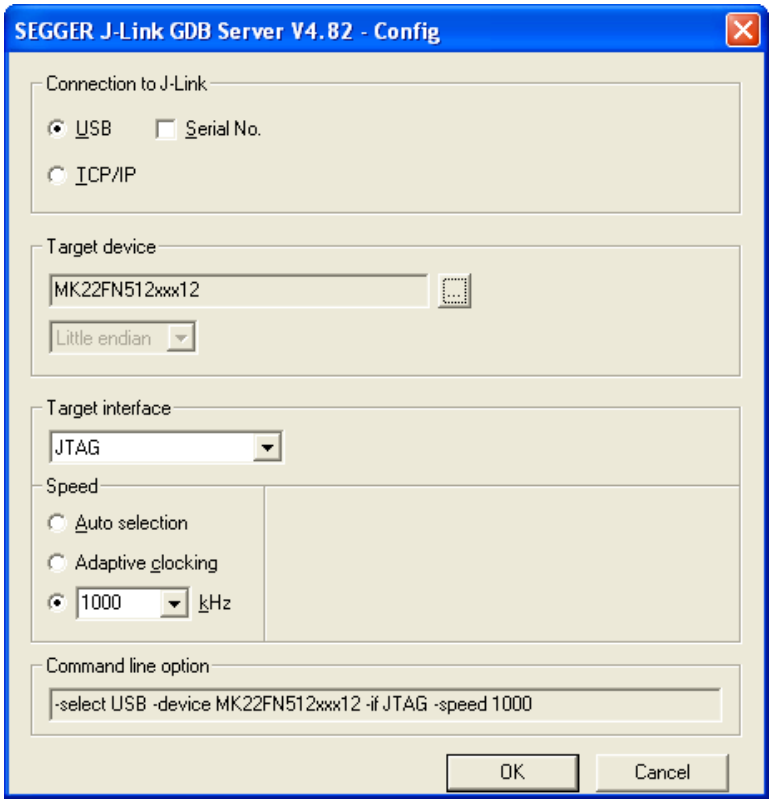


Figure-15 J-Link GDB Server Configuration dialog box

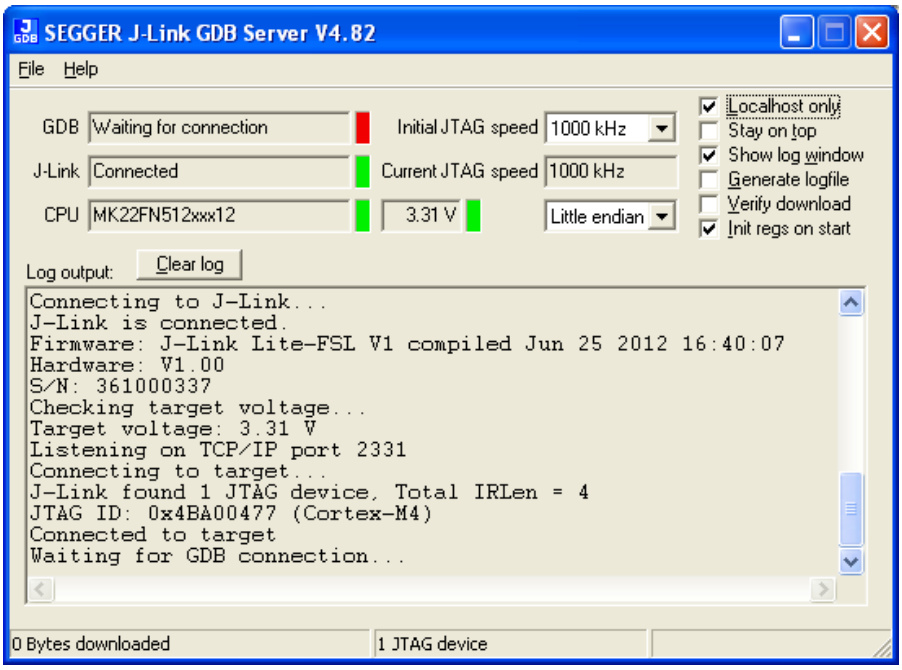


Figure-16 J-Link GDB server configuration result

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:

   `<install_dir>/usb_v2/example/device/hid/hid_mouse/mqx/make/dev_hid_mouse_twrk22f120m/gcc_arm/intflash_release`

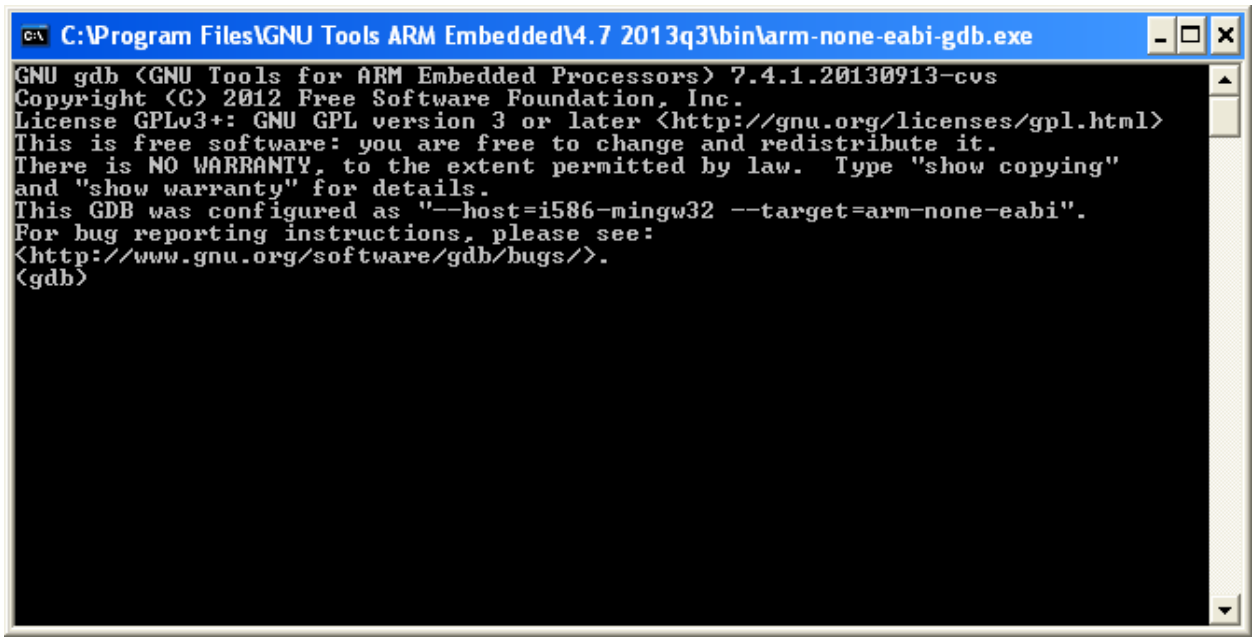


Figure-17 arm-none-eabi-gdb folder

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:

```
<install_dir>/usb_v2/usb_core/device/include/twrk22f120m/usb_device_config.h
```

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:

```
<install_dir>/usb_v2/usb_core/host/include/twrk22f120m/usb_host_config.h
```

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 and 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 and 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:

```
<install_dir>/usb_v2/usb_core/host/include/twrk22f120m/usb_device_config.h
<install_dir>/usb_v2usb_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 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 and J27 2-3 for Rev. B
- 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

How to Reach Us:

Home Page:
www.freescale.com

Web Support:
www.freescale.com/support

Document Number: MQXTWRK22512RUG
Rev. 1.0
04/2014