

# Integration of the USB Stack and Kinetis SDK

## 1 Overview

This document describes how to compile the USB stack and examples, download a binary image, and run the examples. The TWR-K22F120M Tower System module is used as an example board.

### Contents

1	Overview	1
2	Requirements for Building USB Examples	2
3	USB Code Structure	3
4	Compiling or Running the USB Stack and Examples	4
5	USB Stack Configuration	28
6	Revision History	31

## 2 Requirements for Building USB Examples

The TWR-K22F120M Tower System module is used as an example in this document. Compiling, downloading, and running examples are similar on all other boards.

### 2.1 Hardware

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

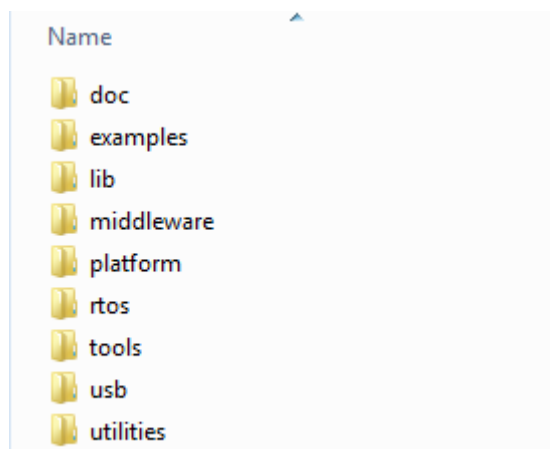
### 2.2 Software

- KSDK release package
- IAR Embedded Workbench for ARM® Version 7.4.3
- Keil µVision5 Integrated Development Environment Version 5.15, available for Kinetis ARM® Cortex®-M4 devices
- Kinetis Design Studio IDE v3.0.0
- Atollic® TrueSTUDIO® v5.3.1
- Makefiles support with GCC revision 4.9.2015q1-update from ARM Embedded

### 3 USB Code Structure

The USB code is located in the folder:

`<install_dir>/usb`



**Figure-1 Kinetis SDK folder structure**

The USB folder includes the source code, projects, and tools.

Name	Date modified	Type	Size
adapter	3/26/2015 2:31 PM	File folder	
facility	3/26/2015 2:31 PM	File folder	
usb_core	3/26/2015 2:31 PM	File folder	

**Figure-2 USB Folder Structure**

The USB folder includes three subfolders:

- adapter

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

- usb\_core

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

- facility

This subfolder includes the interfaces to accommodate various peripherals to which the USB refers to by, such as UART.

## 4 Compiling or Running the USB Stack and Examples

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

This section shows how to use IAR. Open IAR as shown in this figure:

1. Open the workspace corresponding to different examples.

For example, the workspace file is located at:

*<install\_dir>/examples/twrk22f120m/demo\_apps/usb/host/hid/mouse/bm/iar/host\_hid\_mouse\_b  
m.eww.*

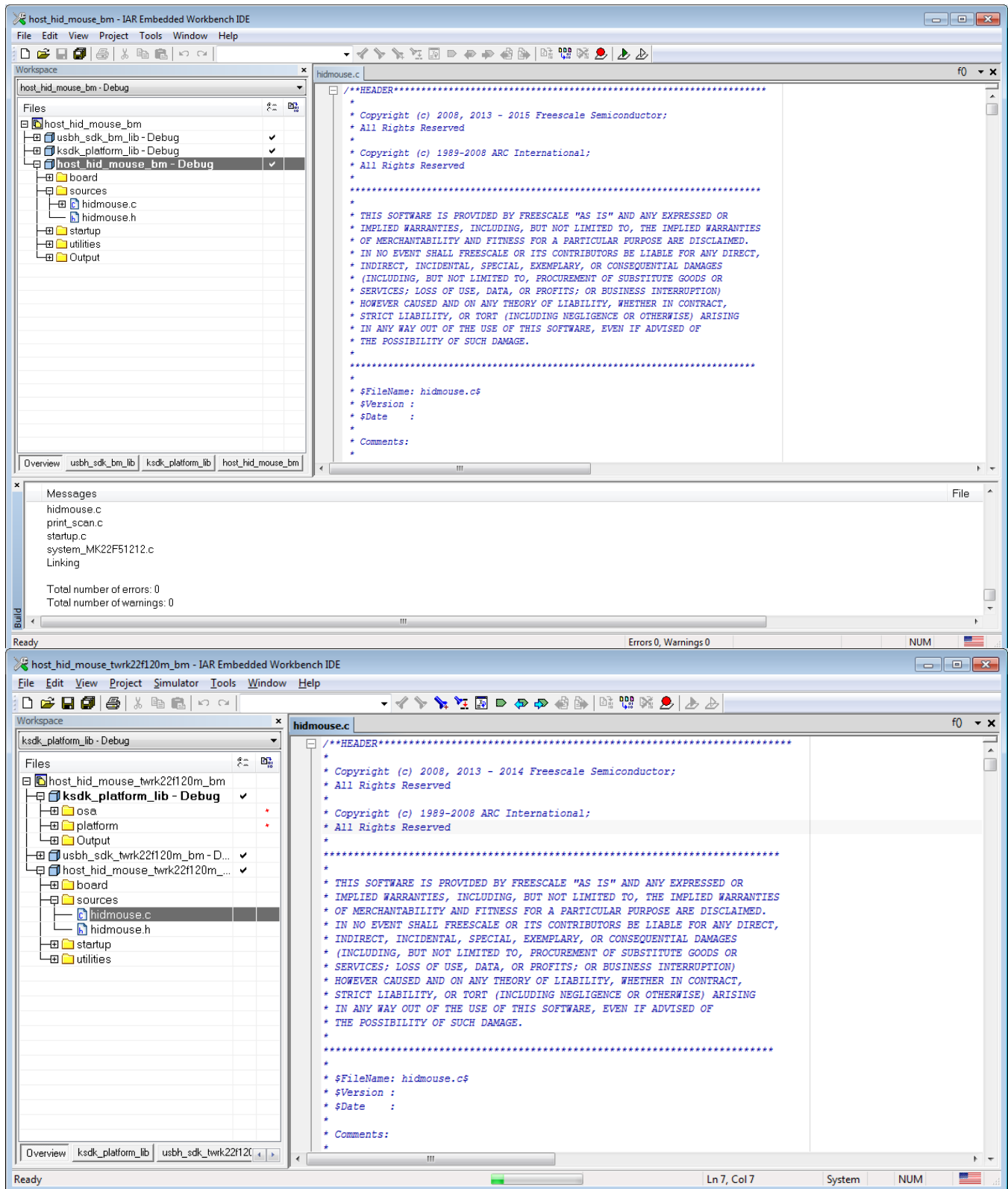


Figure-3 IAR workspace

2. Build the ksdk\_platform\_lib library.
3. Build the usbh\_sdk\_bm\_lib library.
4. Check the USB library build result.
5. After the USB library is built, the generated library binary file is located in  
`<install_dir>/usb/usb_core/host/lib/bm/iar/MK22F51212/debug/libusbh_bm.a`.
6. Build the host\_hid\_mouse\_bm example.
7. Connect the micro USB cable from a PC to the J25 of the TWR-K22F120M Tower System module to power on the board.
8. Click the “Download and Debug” button. Wait for the download to complete.
9. Click the “Go” button to run the example.
10. See the example-specific readme.pdf for more test information.

## 4.2 Step-by-step guide for Keil $\mu$ Vision5

This section shows how to use Keil  $\mu$ Vision5. Open Keil  $\mu$ Vision5 as shown in this figure:

1. Open the workspace corresponding to different examples.

For example, the workspace file is located in

`<install_dir>/examples/twrk22f120m/demo_apps/usb/host/hid/mouse/bm/mdk/host_hid_mouse_bm.uvmpw`

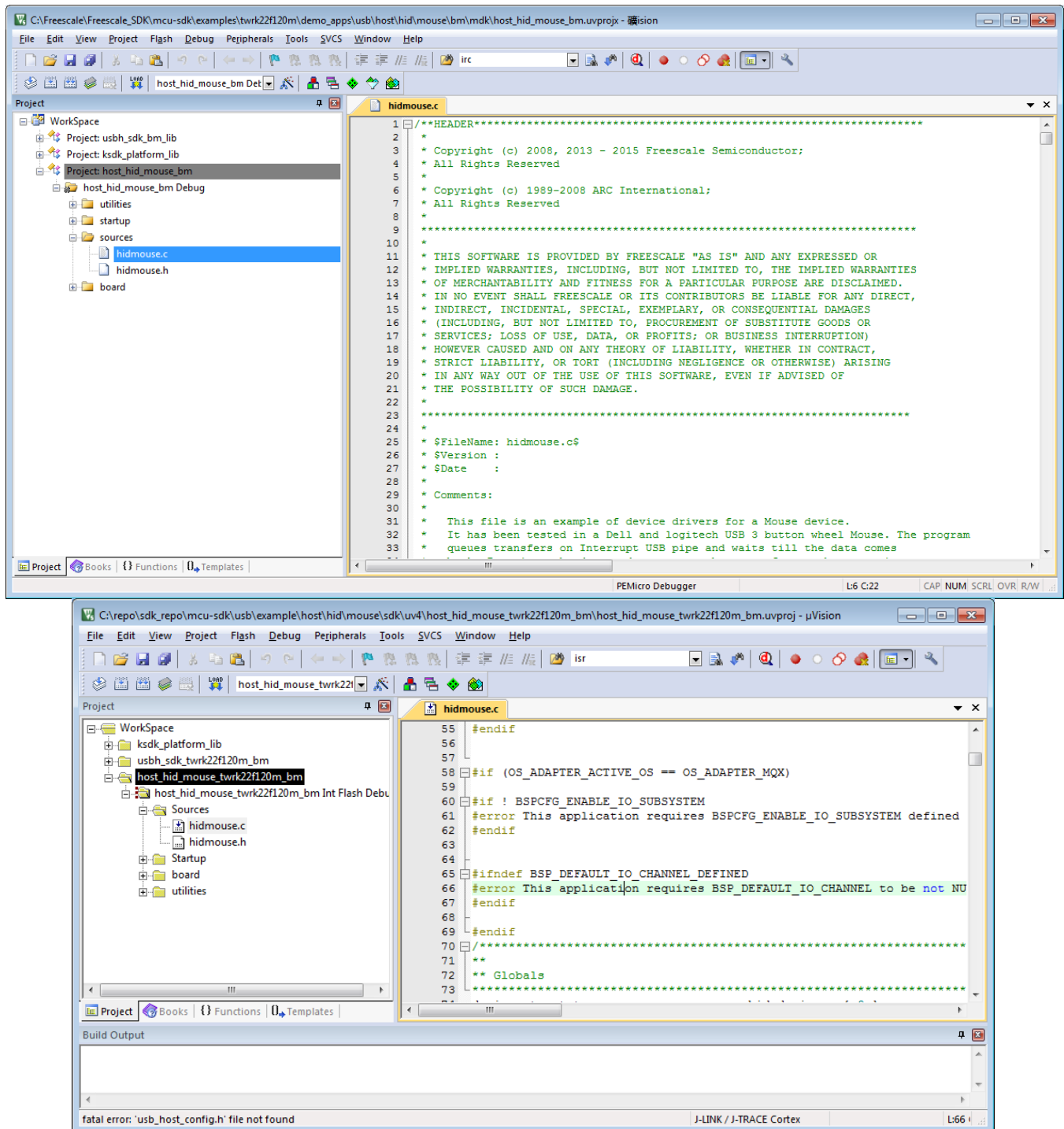


Figure-4 Keil uVision5 Workspace

2. Build the ksdk\_platform\_lib library.
3. Build the usbh\_sdk\_bm\_lib library.
4. Build the host\_hid\_mouse\_bm example.

5. Click the “Start/Stop” debug session button. Wait for the download to complete.
6. Click the “Go” button to run the example.
7. See the example-specific readme.pdf for more test information.

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

1. Unlike IAR or Keil, the Kinetis Design Studio 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.

These are the USB library paths.

Host:

`<install_dir>/usb/usb_core/host/lib/<OS_NAME>/kds/<SOC_NAME>`

Device:

`<install_dir>/usb/usb_core/device/lib/<OS_NAME>/kds/<SOC_NAME>`

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.



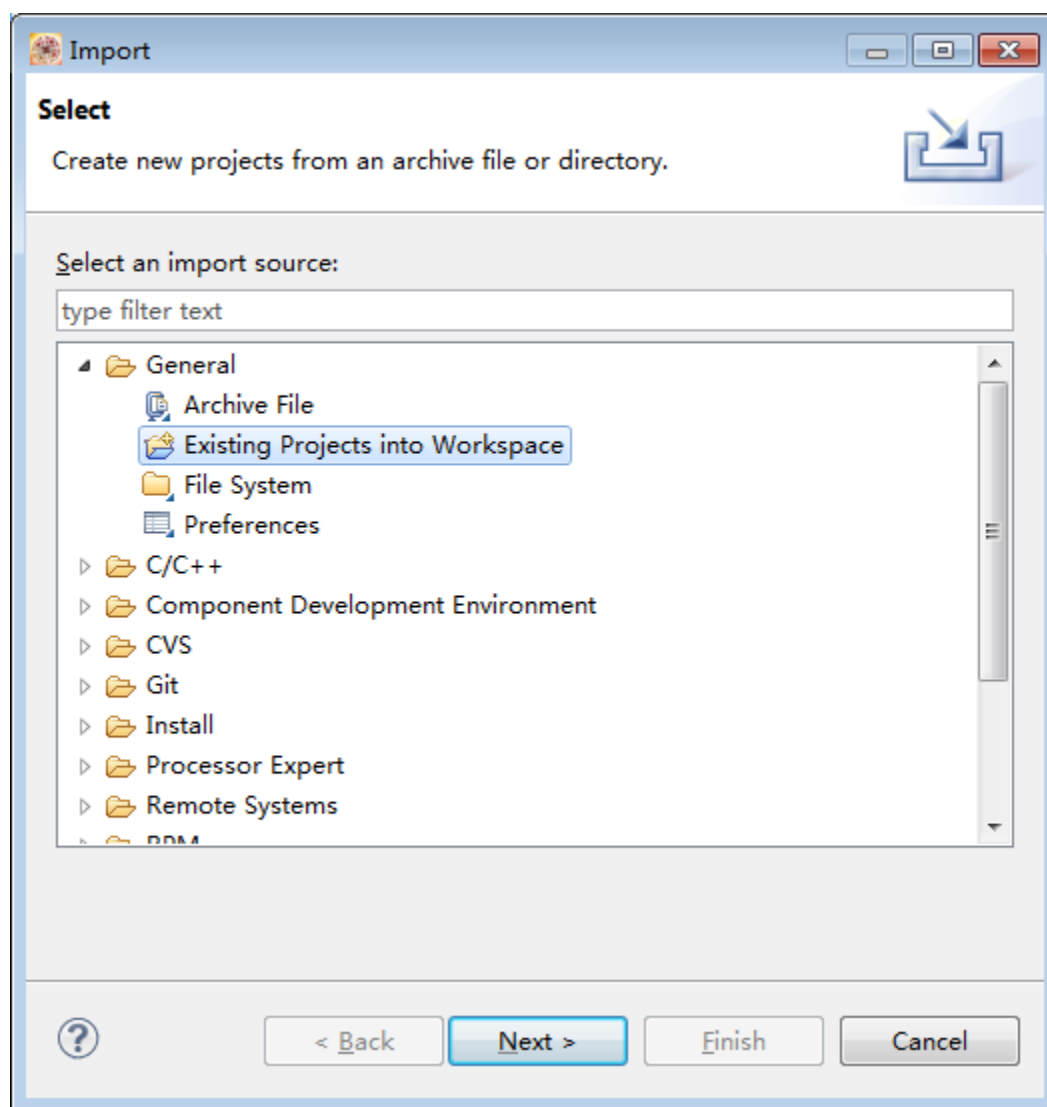


Figure-5 Selection of the correct import type in KDS IDE

4. Point the KDS IDE to the *ksdk\_platform\_lib* project in the K22. The import projects directory selection window should resemble this figure.

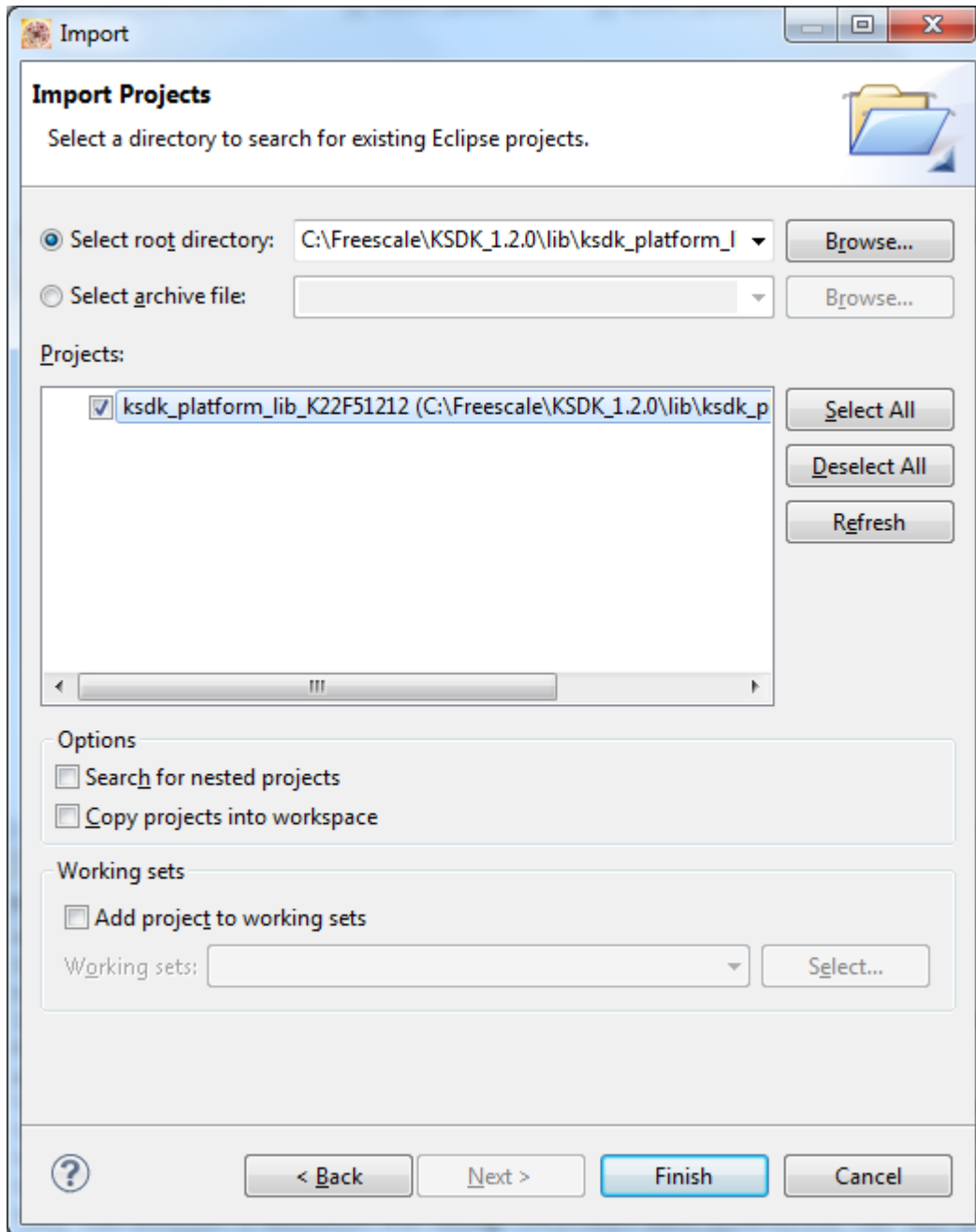
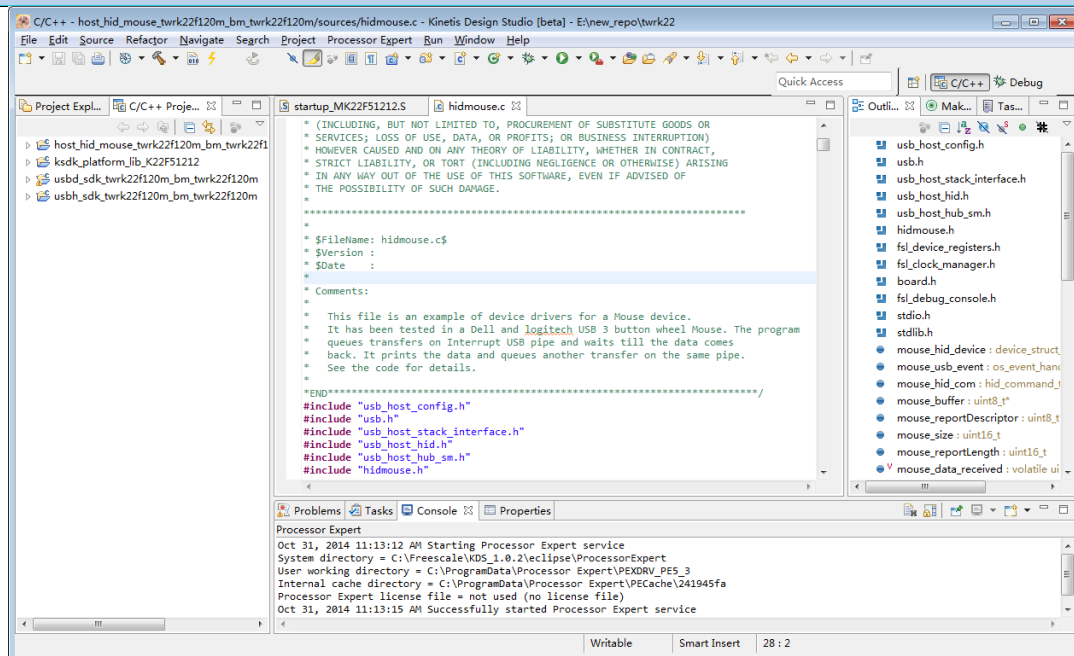
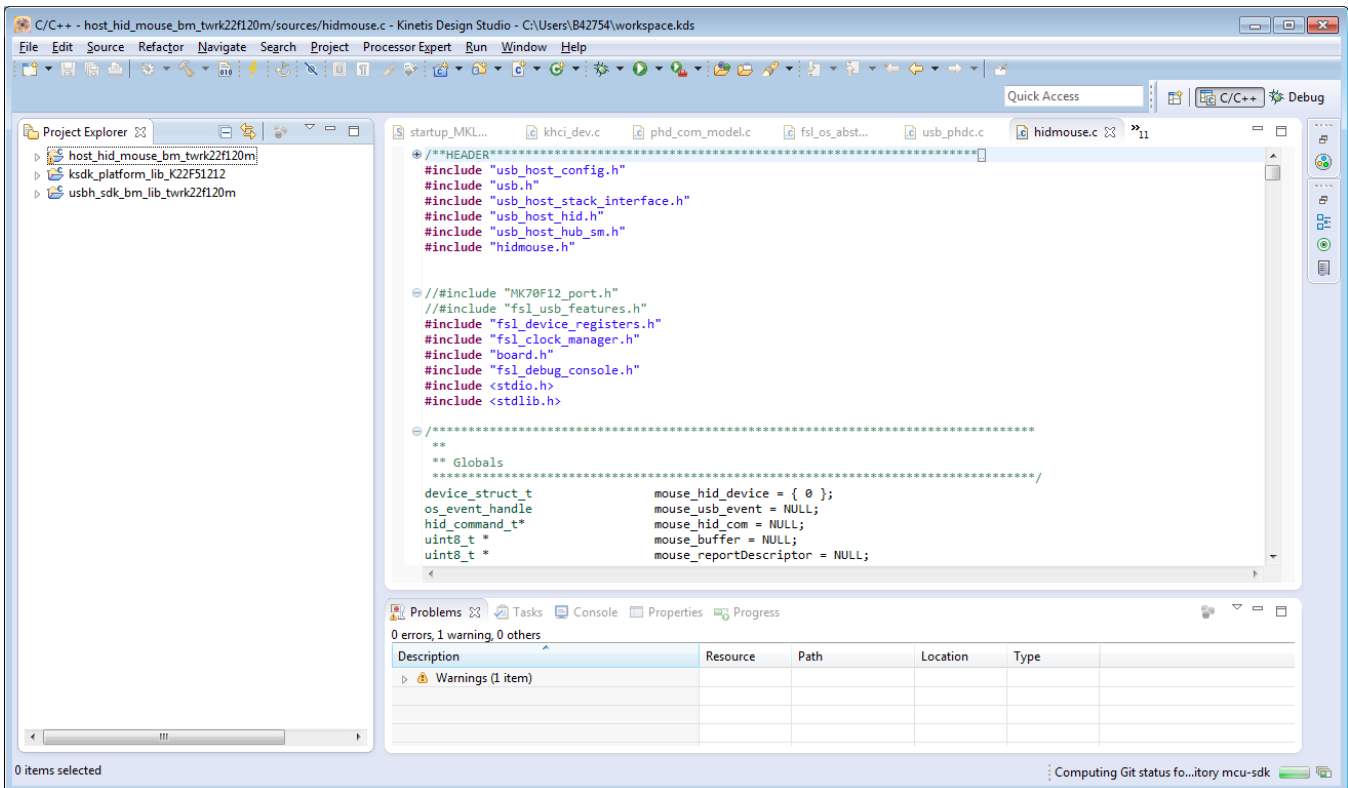


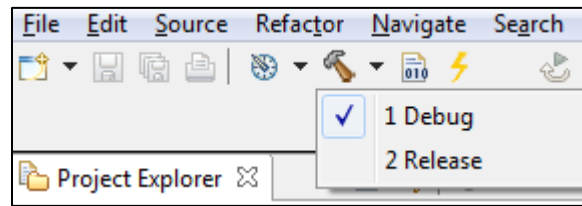
Figure-6 Selection of the K22 ksdk\_platform\_lib project

- Follow the same steps to import the USB device/host library and the USB example. After importing, the window should look like this.



**Figure-7 The USB projects workspace**

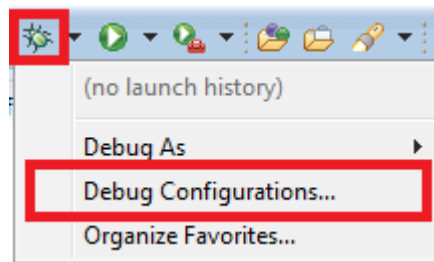
6. Choose the appropriate build target: “Debug” or “Release” by left-clicking the arrow next to the hammer icon as shown here.



**Figure-8 The hammer button**

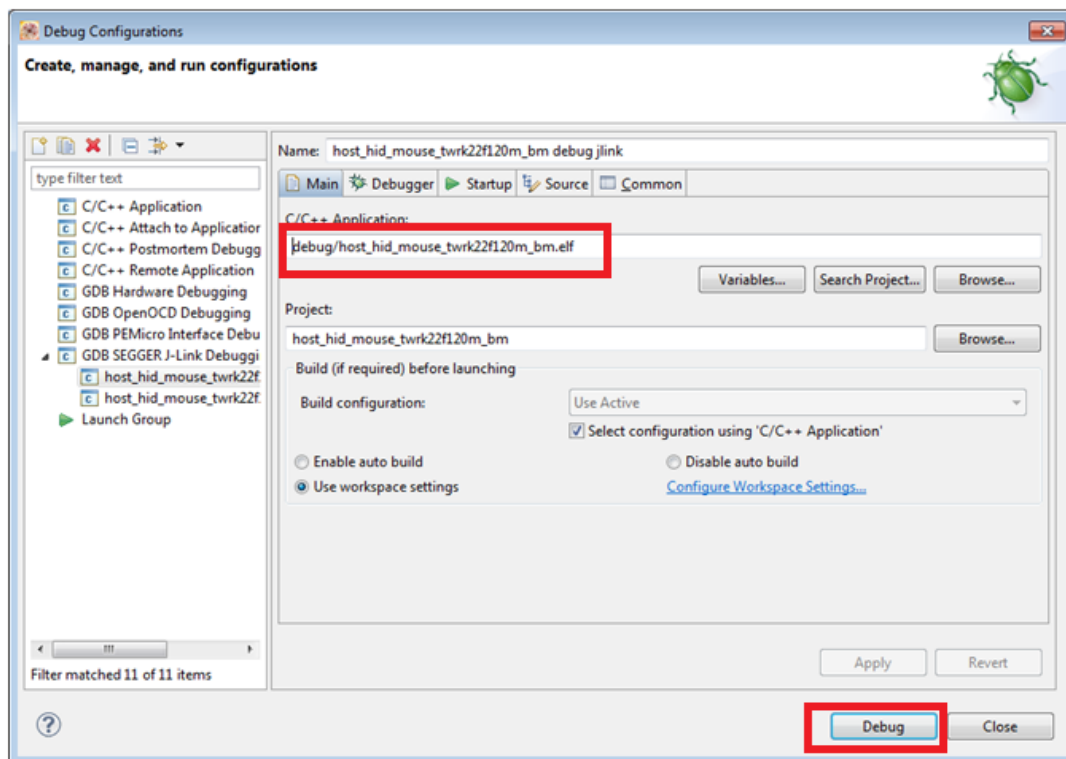
7. If the library build does not begin after selecting the desired target, left-click the hammer icon to start the build.
8. Follow the same steps to build the `ksdk_platform_lib` library, the `usbh_sdk_twrk22f120m_bm` library and the `host_hid_mouse_twrk22f120m_bm` example.

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



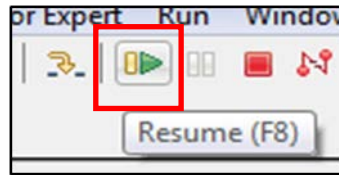
**Figure-9 Debug configurations**

10. After verifying that the debugger configurations are correct, click the “Debug” button.



**Figure-10 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:



**Figure-11 Resume button**

13. See the example-specific readme.pdf for more test information.

## 4.4 Step-by-step guide for the Atollic TrueSTUDIO

1. Unlike IAR or Keil, the Atollic TrueSTUDIO does not have a workspace. As a result, create a workspace and import Atollic TrueSTUDIO USB examples, platform libraries, and the USB stack library.

These are the USB library paths.

Host:

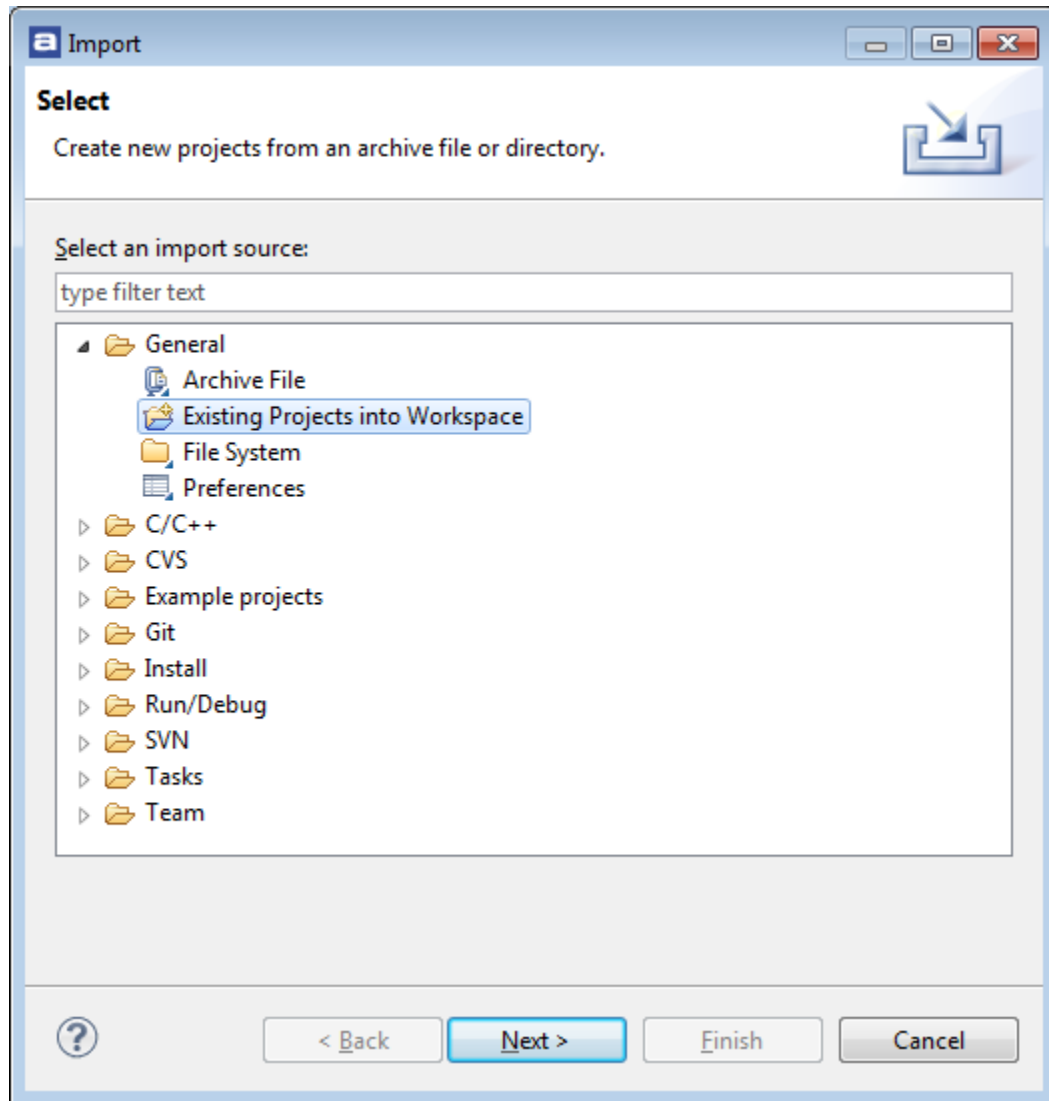
`<install_dir>/usb/usb_core/host/lib/<OS_NAME>/atl/<SOC_NAME>`

Device:

`<install_dir>/usb/usb_core/device/lib/<OS_NAME>/atl/<SOC_NAME>`

2. Select “File” then “Import...” from the Atollic TrueSTUDIO IDE Eclipse menu.

3. Expand the General folder and select “Existing Projects into Workspace”. Then, click the “Next” button.



**Figure-12 Selection of the correct import type in Atollic TrueSTUDIO IDE**

4. Point the Atollic TrueSTUDIO IDE to the *ksdk\_platform\_lib* project in the K22. The import projects directory selection window should resemble this figure.

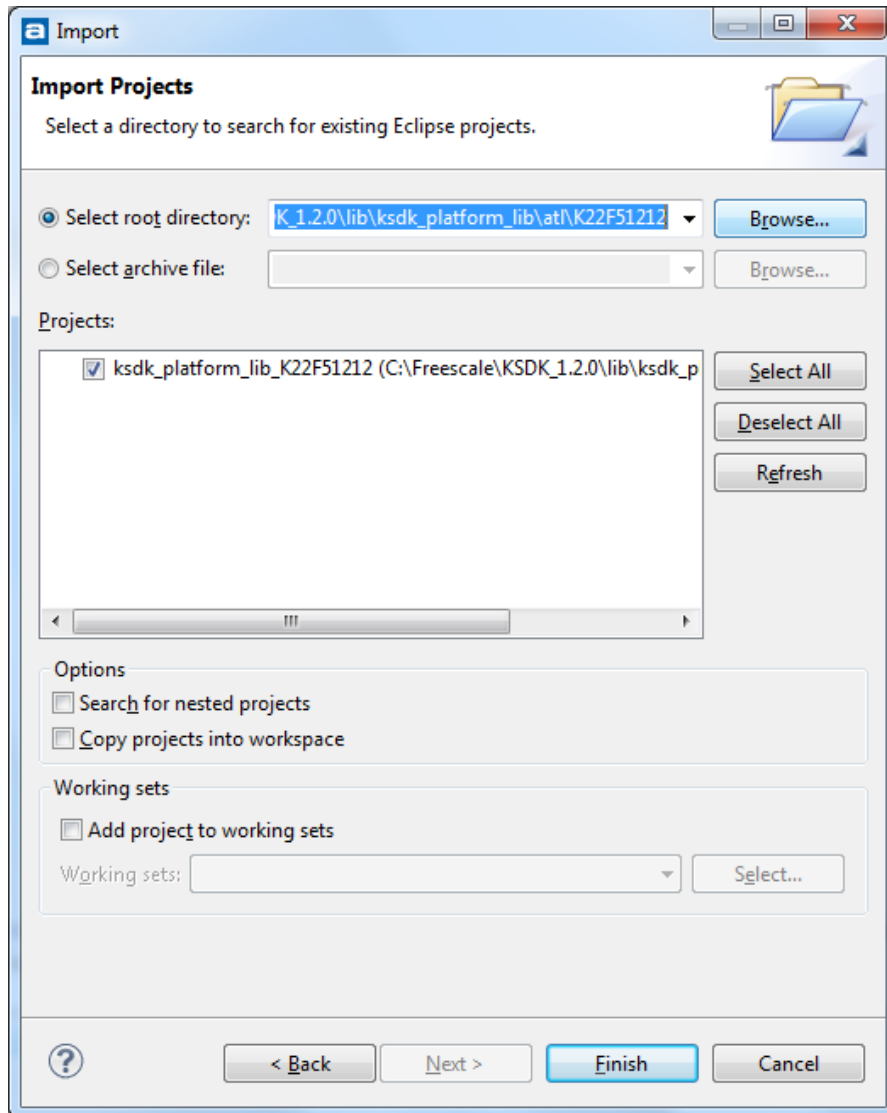


Figure-13 Selection of the K22 ksdk\_platform\_lib project



- Follow the same steps to import the USB device/host library and the USB example. After importing, the window should like this.

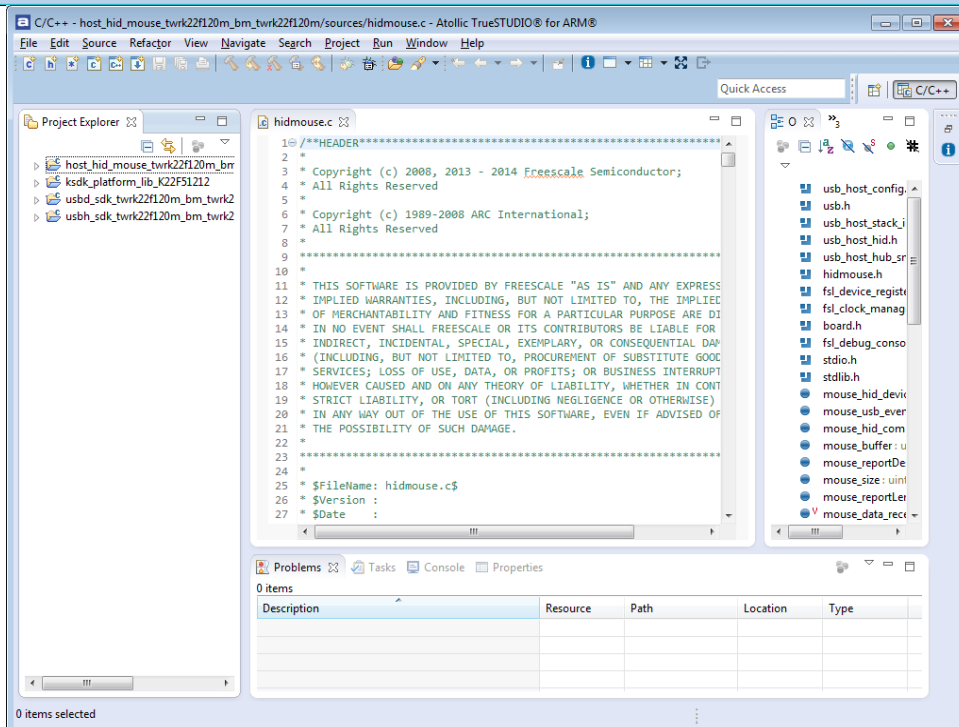
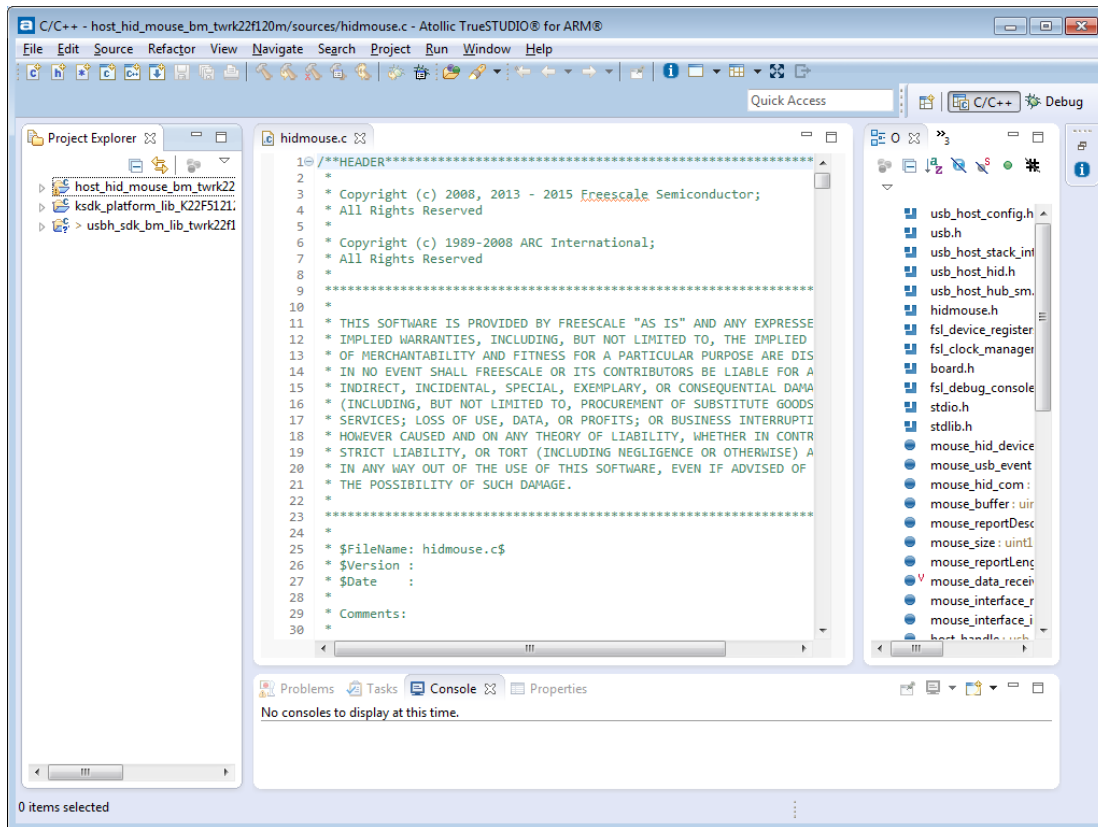
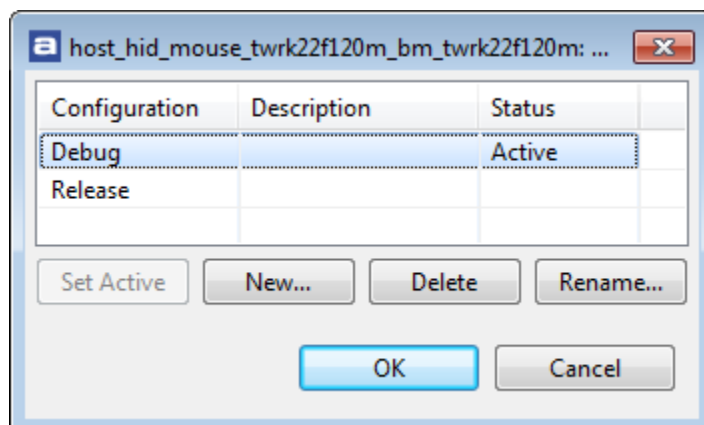


Figure-14 The USB projects workspace

6. Choose the appropriate build target: “Debug” or “Release” by left-clicking the build configuration icon as shown here.



**Figure-15 Manage build configuration button**



**Figure-16 Set build configuration**

7. If the library build does not begin after selecting the desired target, left-click the build icon to start the build.



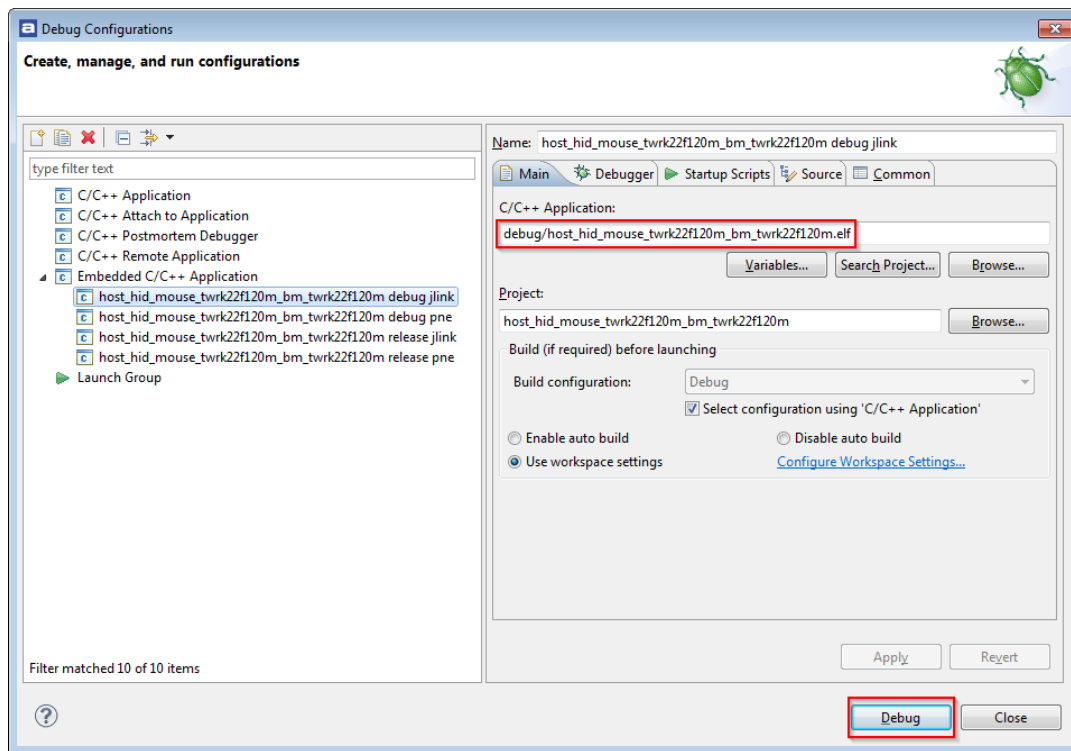
**Figure-17 Build project button**

8. Follow the same steps to build the ksdk\_platform\_lib library, the usbh\_sdk\_twrk22f120m\_bm library and the host\_hid\_mouse\_twrk22f120m\_bm example.
9. To check the debugger configurations, click the “Configure Debug” button.



**Figure-18 Configure debug button**

10. After verifying that the debugger configurations are correct, click the “Debug” button.



**Figure-19 Atollic TrueSTUDIO 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:



**Figure-20 The resume button**

13. See the example-specific readme.pdf for more test information.

## 4.5 Step-by-step guide for the ARM 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](http://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”.

MinGW Installation Manager					
Installation Package Settings					
Basic Setup	Package	Class	Installed...	Repository Version	Description
All Packages	<input type="checkbox"/> mingw-developer-toolkit	bin		2013072300	An MSYS Installation for MinGW Developers (meta)
	<input checked="" type="checkbox"/> mingw32-base	bin		2013072200	A Basic MinGW Installation
	<input type="checkbox"/> mingw32-gcc-ada	bin		4.8.1-4	The GNU Ada Compiler
	<input type="checkbox"/> mingw32-gcc-fortran	bin		4.8.1-4	The GNU FORTRAN Compiler
	<input type="checkbox"/> mingw32-gcc-g++	bin		4.8.1-4	The GNU C++ Compiler
	<input type="checkbox"/> mingw32-gcc-objc	bin		4.8.1-4	The GNU Objective-C Compiler
	<input checked="" type="checkbox"/> msys-base	bin		2013072300	A Basic MSYS Installation (meta)

**Figure 21: 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.

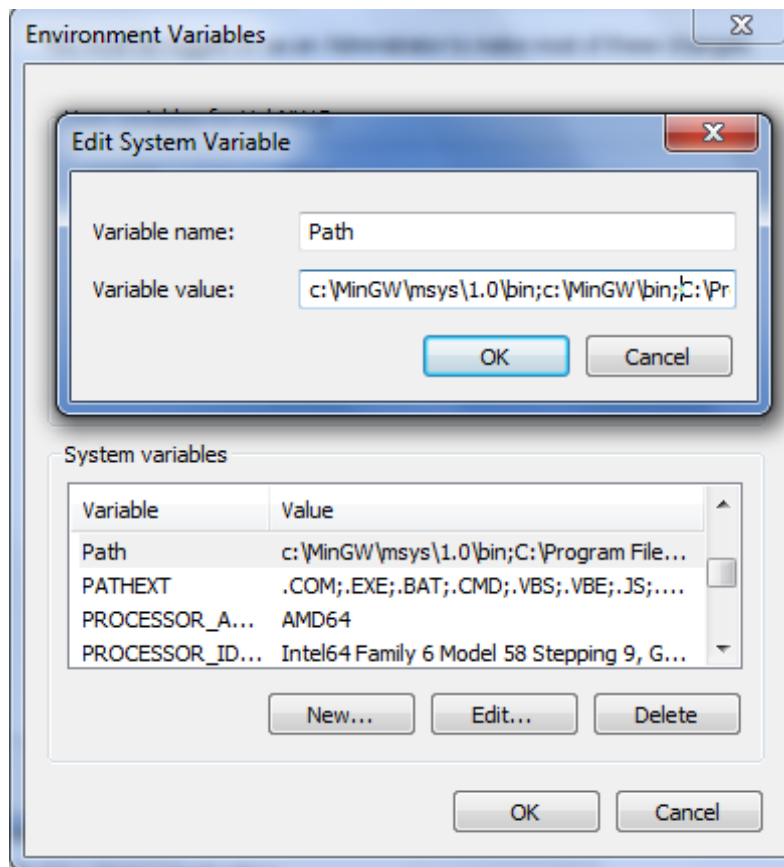


Figure 22: Add Path to systems environment

#### 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.

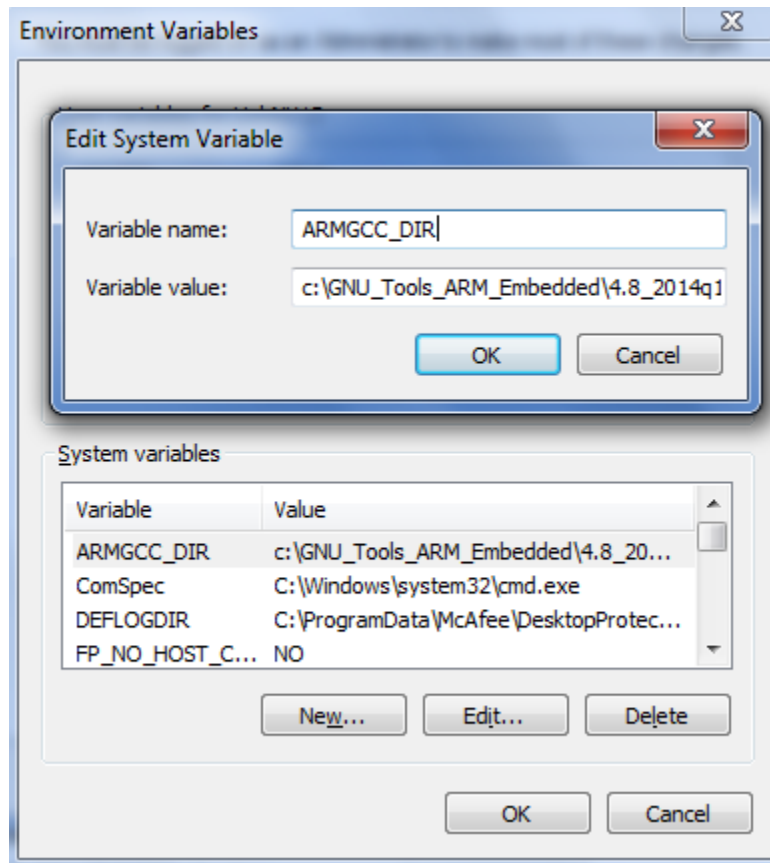


Figure 23: Add ARMGCC\_DIR system variable

#### 4.5.1.4 Install CMake

1. Download CMake 3.0.1 from [www.cmake.org/cmake/resources/software.html](http://www.cmake.org/cmake/resources/software.html).
2. Install Cmake 3.0.1 (ensure that the option "Add CMake to system PATH" is selected when installing).

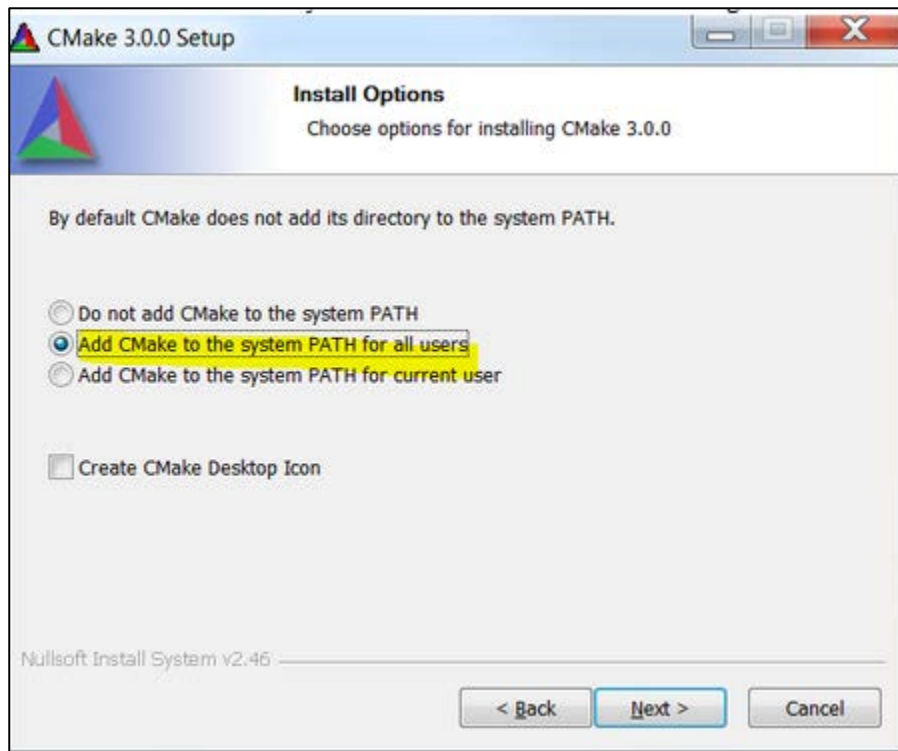


Figure 24: Install CMake

#### 4.5.2 Build the platform driver library

To build the platform library, follow these instructions:

1. Open a GCC ARM Embedded tool chain command window.
2. Change the directory of the command window to the platform lib directory in the KSDK:  
`<install_dir>/lib/ksdk_platform_lib/armgcc/<device_name>/`
3. Type "build\_all".

```

[ 95%] [ 96%] Building C object CMakeFiles/KsdkPlatformLib.dir/C:/Freescale/Freescale_SDK/mcu-sdk/platform/drivers/src/uart/fsl_uart_driver.c.obj
Building C object CMakeFiles/KsdkPlatformLib.dir/C:/Freescale/Freescale_SDK/mcu-sdk/platform/drivers/src/uart/fsl_uart_dma_driver.c.obj
[ 97%] [ 98%] Building C object CMakeFiles/KsdkPlatformLib.dir/C:/Freescale/Freescale_SDK/mcu-sdk/platform/drivers/src/wdog/fsl_wdog_driver.c.obj
Building C object CMakeFiles/KsdkPlatformLib.dir/C:/Freescale/Freescale_SDK/mcu-sdk/platform/hal/src/wdog/fsl_wdog_hal.c.obj
[100%] Building C object CMakeFiles/KsdkPlatformLib.dir/C:/Freescale/Freescale_SDK/mcu-sdk/platform/drivers/src/wdog/fsl_wdog_common.c.obj
Linking C static library release\libksdk_platform.a
[100%] Built target KsdkPlatformLib
c:\Freescale\Freescale_SDK\mcu-sdk\lib\ksdk_platform_lib\armgcc\K22F51212>pause
Press any key to continue . . .

```

Figure 25: ksdk\_platform\_lib build successfully

- The library (ksdk\_platform\_lib.a) is generated in these directories according to the build target:

*<install\_dir>/lib/ksdk\_platform\_lib/armgcc/<device\_name>/Debug*

*<install\_dir>/lib/ksdk\_platform\_lib/armgcc/<device\_name>/Release*

### 4.5.3 Build the USB host/device library

- Change the directory to the project directory:

*<install\_dir>/usb/usb\_core/host/lib/bm/armgcc/MK22F51212*

- Run the build\_all.bat as described in steps 3 and 4 of section 5.5.2. The build output is shown in this figure:

```

[ 85%] [ 90%] [ 95%] Building C object CMakeFiles/KsdkUsbHostPlatformLib.dir/C:/Freescale/Freescale_SDK/mcu-sdk/usb/usb_core/host/sources/controller/usb_host_dev_mng.c.obj
Building C object CMakeFiles/KsdkUsbHostPlatformLib.dir/C:/Freescale/Freescale_SDK/mcu-sdk/usb/usb_core/host/sources/controller/khci/khci.c.obj
Building C object CMakeFiles/KsdkUsbHostPlatformLib.dir/C:/Freescale/Freescale_SDK/mcu-sdk/usb/usb_core/host/sources/bsp/twrk22f120m/usb_host_bsp.c.obj
Linking C static library release\libusbh_bm.a
[100%] Built target KsdkUsbHostPlatformLib
[ 89%] [ 94%] Building C object CMakeFiles/KsdkUsbHostPlatformLib.dir/C:/Freescale/Freescale_SDK/mcu-sdk/usb/usb_core/host/sources/controller/usb_host_dev_mng.c.obj
Building C object CMakeFiles/KsdkUsbHostPlatformLib.dir/C:/Freescale/Freescale_SDK/mcu-sdk/usb/usb_core/host/sources/controller/khci/khci.c.obj
[100%] Building C object CMakeFiles/KsdkUsbHostPlatformLib.dir/C:/Freescale/Freescale_SDK/mcu-sdk/usb/usb_core/host/sources/bsp/twrk22f120m/usb_host_bsp.c.obj
Linking C static library release\libusbh_bm.a
[100%] Built target KsdkUsbHostPlatformLib
c:\Freescale\Freescale_SDK\mcu-sdk\usb\usb_core\host\build\armgcc\usbh_sdk_twrk22f120m_bm>pause
Press any key to continue . . .

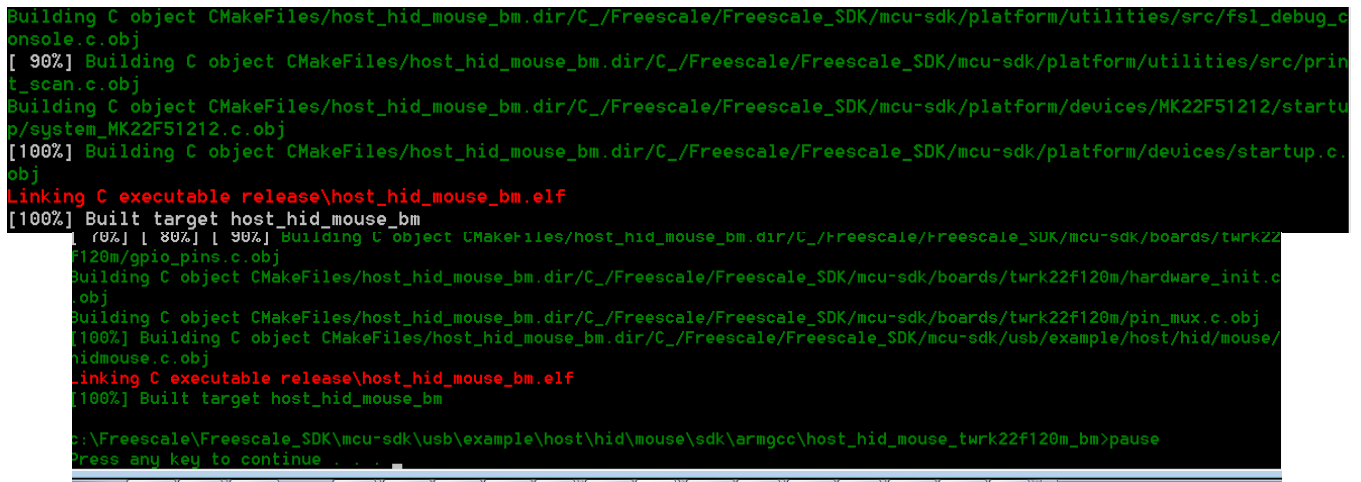
```

Figure 26: USB host library build successfully



## 4.5.4 Build the USB demo

1. Change the directory to the project directory:
2. `<install_dir>/examples/twrk22f120m/demo_apps/usb/host/hid/mouse/bm/armgcc` Run the `build_all.bat` as described in steps 3 and 4 of section 5.5.2. The build output is shown in this figure:



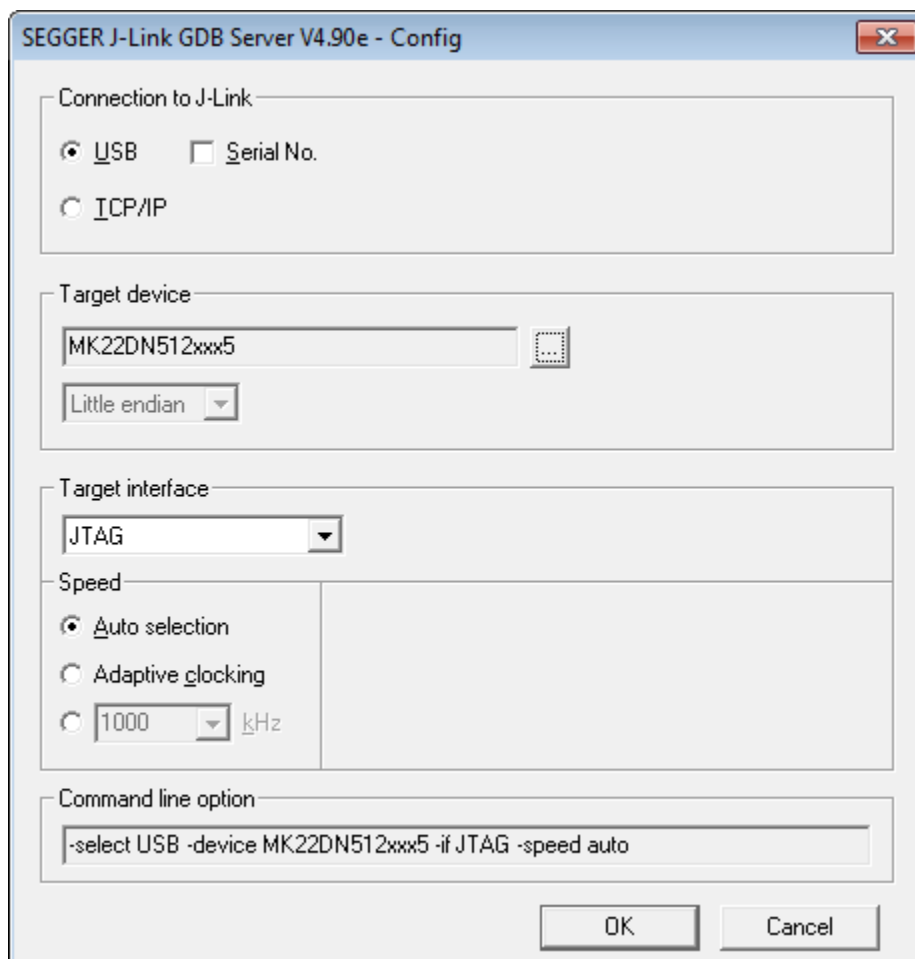
```
Building C object CMakeFiles/host_hid_mouse_bm.dir/C:/Freescale/Freescale_SDK/mcu-sdk/platform/utilities/src/fsl_debug_console.c.obj
[ 90%] Building C object CMakeFiles/host_hid_mouse_bm.dir/C:/Freescale/Freescale_SDK/mcu-sdk/platform/utilities/src/print_scan.c.obj
Building C object CMakeFiles/host_hid_mouse_bm.dir/C:/Freescale/Freescale_SDK/mcu-sdk/platform/devices/MK22F51212/startup/system_MK22F51212.c.obj
[100%] Building C object CMakeFiles/host_hid_mouse_bm.dir/C:/Freescale/Freescale_SDK/mcu-sdk/platform/devices/startup.c.obj
Linking C executable release\host_hid_mouse_bm.elf
[100%] Built target host_hid_mouse_bm
[ 80%] [ 80%] [ 90%] Building C object CMakeFiles/host_hid_mouse_bm.dir/C:/Freescale/Freescale_SDK/mcu-sdk/boards/twrk22f120m/gpio_pins.c.obj
Building C object CMakeFiles/host_hid_mouse_bm.dir/C:/Freescale/Freescale_SDK/mcu-sdk/boards/twrk22f120m/hardware_init.c.obj
Building C object CMakeFiles/host_hid_mouse_bm.dir/C:/Freescale/Freescale_SDK/mcu-sdk/boards/twrk22f120m/pin_mux.c.obj
[100%] Building C object CMakeFiles/host_hid_mouse_bm.dir/C:/Freescale/Freescale_SDK/mcu-sdk/usb/example/host/hid/mouse/hidmouse.c.obj
Linking C executable release\host_hid_mouse_bm.elf
[100%] Built target host_hid_mouse_bm
C:/Freescale/Freescale_SDK/mcu-sdk/usb/example/host/hid/mouse/sdk/armgcc/host_hid_mouse_twrk22f120m_bm>pause
Press any key to continue . . .
```

Figure 27: USB host demo build successfully

### 4.5.5 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 port 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.

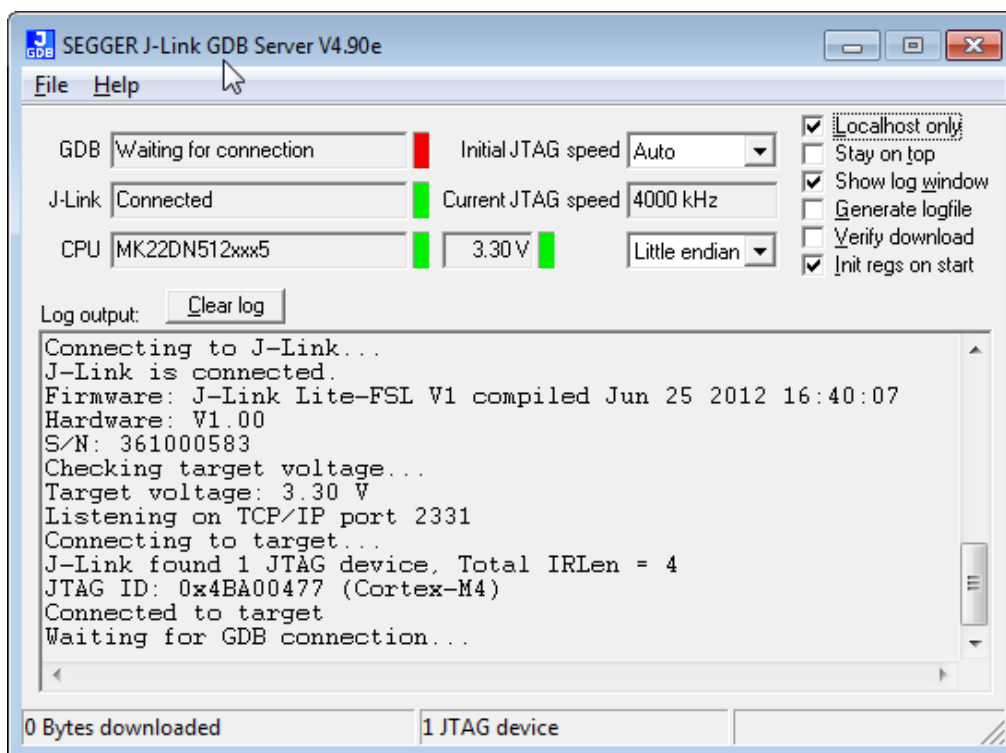


**Figure 28: SEGGER J-Link GDB Server configuration**

#### **Note**

The target device selection should be MK22FN512xxx12. The target interface should be SWD.

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



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

#### **Note**

The CPU selection should be CPU to: MK22FN512xxx12.

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>/examples/twrk22f120m/demo_apps/usb/host/hid/mouse/bm/armgcc/debug`
5. Run the command "arm-none-eabi-gdb.exe <DEMO\_NAME>.elf". Run these commands:
  - a. "target remote localhost: 2331"
  - b. "monitor reset"
  - c. "monitor halt"
  - d. "load"
  - e. "monitor reset"
6. The application is downloaded and connected. Execute the "monitor go" command to start the demo application.
7. See the example-specific readme.pdf for more test information.

## 5 USB Stack Configuration

### 5.1 Device configuration

All device configurations are listed in this file:

`<install_dir>/usb/usb_core/device/include/<SOC_NAME>/usb_device_config.h`

Replace SOC\_NAME with the name of the SoC.

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 device library and the example projects.

### 5.2 Host configuration

All the host configurations are listed in this file:

`<install_dir>/usb/usb_core/host/include/<SOC_NAME>/usb_host_config.h`

Replace SOC\_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 Host stack configuration is changed, rebuild both the USB Host 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):
  - J4 1-2
  - J27 1-2

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

In `<install_dir>/examples/<BOARD_NAME>/board.h`,

`USB_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):

- J4 1-2
- J27 1-2
- 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:

In `<install_dir>/examples/<BOARD_NAME>/board.h`,

```
USB_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:

```
<Installer_dir>/usb/usb_core/otg/include/<SOC_NAME>/usb_otg_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 OTG 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:

- J4 1-2
- J27 1-2
- 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:

```
USB_HOST_PORT_NATIVE    0
```

#### NOTE

1. If the USB mini port (J14) on the TWR-SER 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 FRDM-K64F Freescale Freedom Development Platform is a USB device when no OpenSDA power is supplied, add a 0-ohm resistor on R61 to power on the P3V3\_SDA.

## 6 Revision History

This table summarizes revisions to this document.

Revision History		
Revision number	Date	Substantial changes
0	12/2014	Kinetis SDK 1.1.0
1	04/2015	Kinetis SDK 1.2.0
2	09/2015	Section 4 and all its subsections, Section 5 and all its subsections

#### **How to Reach Us:**

##### **Home Page:**

[www.freescale.com](http://www.freescale.com)

##### **Web Support:**

[www.freescale.com/support](http://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](http://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.

