

lwIP TCP/IP Stack and Kinetis SDK Integration User's Guide

1 Overview

This document describes how to compile and run the lwIP TCP/IP stack examples. This document also provides the board-specific information related to the TWR-K64F120M Tower System module and the Freescale Freedom FRDM-K64F platforms.

Contents

1	Overview	1
2	Release scope.....	2
3	Requirements for running lwIP demos	2
4	lwIP code structure.....	3
5	Compiling or running the lwIP stack and demos.....	4
6	Revision history	15

2 Release scope

2.1 Hardware

- Support for TWR-K64F120M and TWR-K60D100M Tower System module and Freescale Freedom FRDM-K64F platform

2.2 Software

- Contains PING, TCP, UDP and HTTP demos
- BM and RTOS are both supported

3 Requirements for running lwIP demos

3.1 Hardware

- TWR-K60D100M
- TWR-K64F120M/ Freescale Freedom FRDM-K64F platform
- TWR-SER and elevator
- USB cable
- Ethernet cable

3.2 Software

- Freescale KSDK release package that includes the lwIP TCP/IP package
- IAR Embedded Workbench for ARM® version 7.20.2
- Keil µVision5 Integrated Development Environment Version 5.11 service pack for Kinetis K60
- Kinetis Design Studio Version: 1.1
- Makefiles support with GCC revision 4.8.3 from ARM Embedded

3.3 Board jumper settings

The Ethernet-related jumper settings are described in this document. For other jumper settings, see board-related user's guide.

By default the lwIP stack uses RMII mode, please follow the below hardware configuration:

- TWR-K60D100M
 - TWR-K60D100 board

- J10 2-3: Use the external clock from the CLOCKIN0 to keep the synchronization with the external PHY on TWR-SER board.
- TWR-K64F120M
 - TWR-K64F120M Tower System module board
 - J32 1-2: Use the external clock from the CLOCKIN0 to keep the synchronization with the external PHY on TWR-SER board.
- TWR-SER
 - J2 3-4: Ethernet PHY Clock Select 50 MHz, RMII mode. Cut off other connections on this jumper.
 - J3 2-3: Route 50 MHz clock to CLOCKIN0. Cut off other connections on this jumper.
 - J12 9-10: Ethernet PHY Configuration, pull-up CONFIG0, RMII select. Cut off other connections on this jumper.
 - Freescale Freedom FRDM-K64F platform
 - No jumper specifications

4 lwIP code structure

The lwIP code is located in the “tcpip/lwip” folder at the root level of the Kinetis_SDK folder.

Name	Date modified	Type
.git	5/30/2014 3:51 PM	File folder
bin	5/29/2014 6:07 PM	File folder
boards	5/5/2014 1:37 PM	File folder
demos	5/28/2014 5:58 PM	File folder
doc	5/28/2014 5:58 PM	File folder
doxygen	5/5/2014 11:16 AM	File folder
filesystem	5/6/2014 10:43 AM	File folder
lib	5/13/2014 2:52 PM	File folder
mk	5/27/2014 10:54 AM	File folder
platform	5/5/2014 1:37 PM	File folder
rtos	5/5/2014 1:37 PM	File folder
tcpip	5/6/2014 10:43 AM	File folder
usb	5/15/2014 6:20 PM	File folder

Figure 4-1 SDK folder structure

The lwIP folder includes the source code. There are two subfolders in the lwIP folder as shown in the figure.

Name	Date modified	Type
port	6/3/2014 2:43 PM	File folder
src	5/5/2014 1:37 PM	File folder

Figure 4-2 lwIP folder structure

- src
This subfolder includes the lwIP 1.4.1 source code which can be downloaded from this link: download.savannah.gnu.org/releases/lwip/
- port
This subfolder includes the adapter files which can make the lwIP stack run on the KSDK and different RTOSes.

5 Compiling or running the lwIP stack and demos

5.1 Configuration

1. ENET driver configuration

This release supports both polling and interrupt mode for frame receiving.

In `<install_dir>/platform/drivers/enet/fsl_enet_driver.h`, set

`#define ENET_RECEIVE_ALL_INTERRUPT 0` to enable polling mode.

Or set

`#define ENET_RECEIVE_ALL_INTERRUPT 1` to enable interrupt mode.

5.2 Step-by-step guide for IAR

This section shows how to compile and run demos in IAR.

1. Open the workspace corresponding to different demos and different boards. For example, the `lwip_ping_demo.eww` on Freescale Freedom FRDM-K64F Platform under `<install_dir>/demos/lwip_ping_demo/ping_bm/iar/frdmk64f120m` or the `lwip_ping_demo_freertos.eww` on Freescale Freedom FRDM-K64F platform under `<install_dir>/demos/lwip_ping_demo/ping_rtos/ping_freertos/iar/frdmk64f120m`. These steps use `lwip_ping_demo.eww` on FRDM-K64F120M as an example.

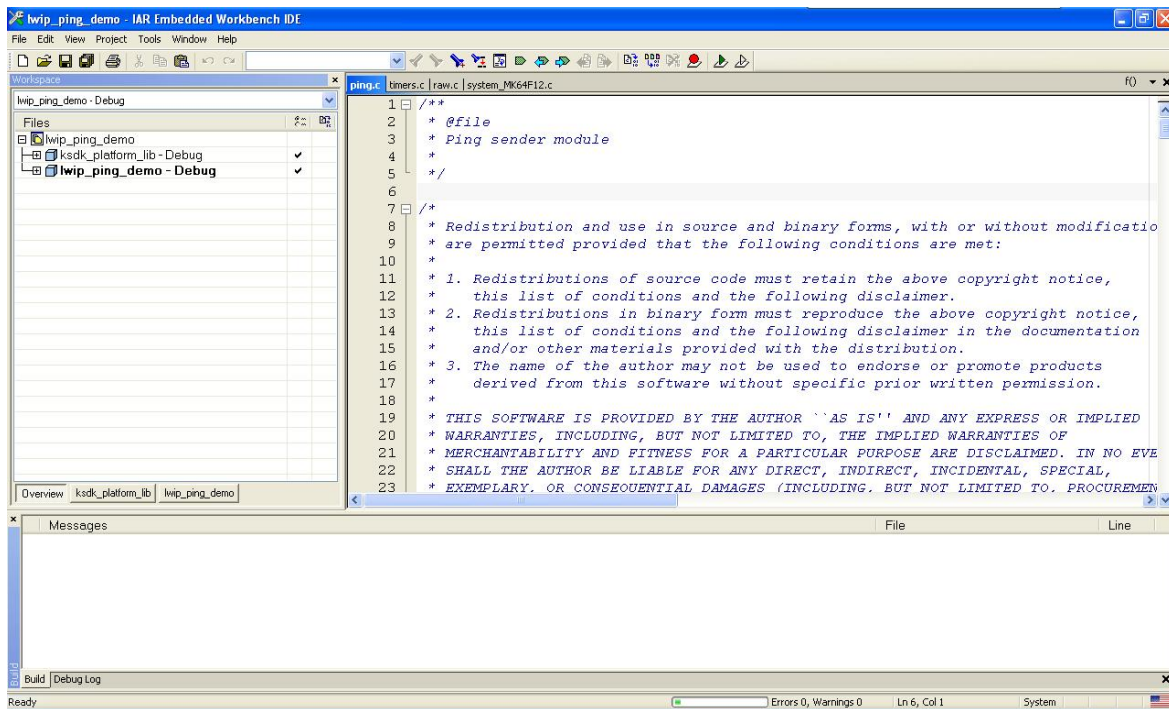


Figure 5-1 Workspace

2. Build the ksdk_platform_lib library.

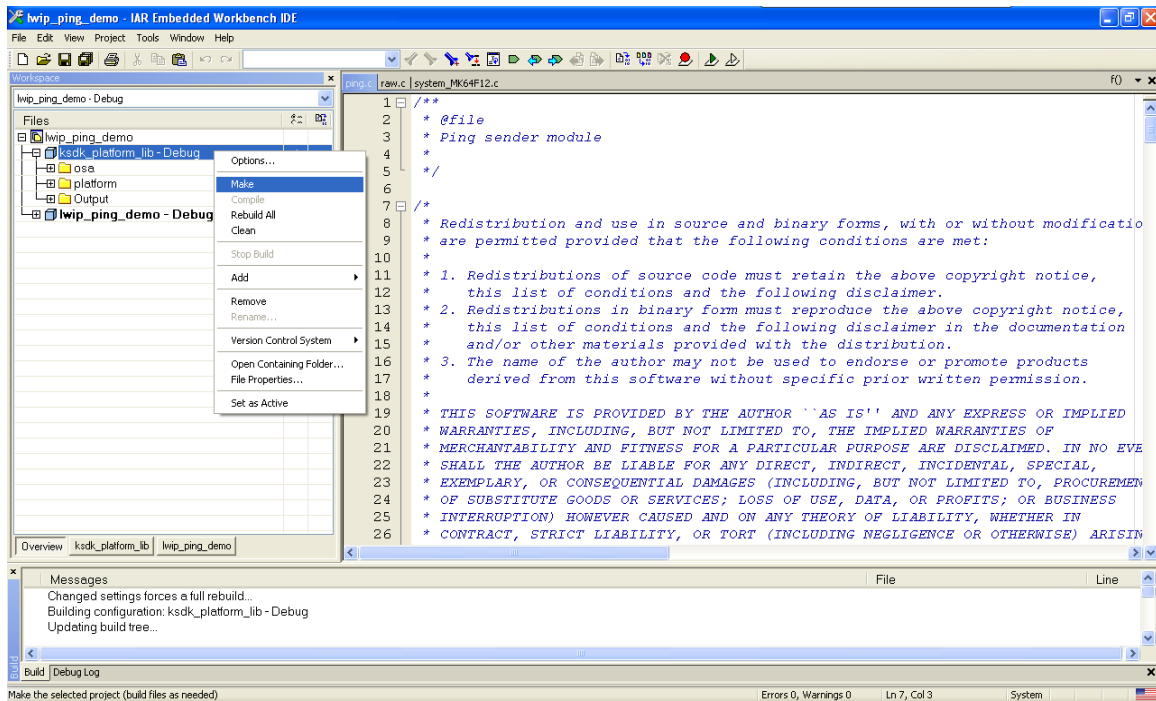


Figure 5-2 ksdk_platform_lib

3. Build the lwip_ping_demo.

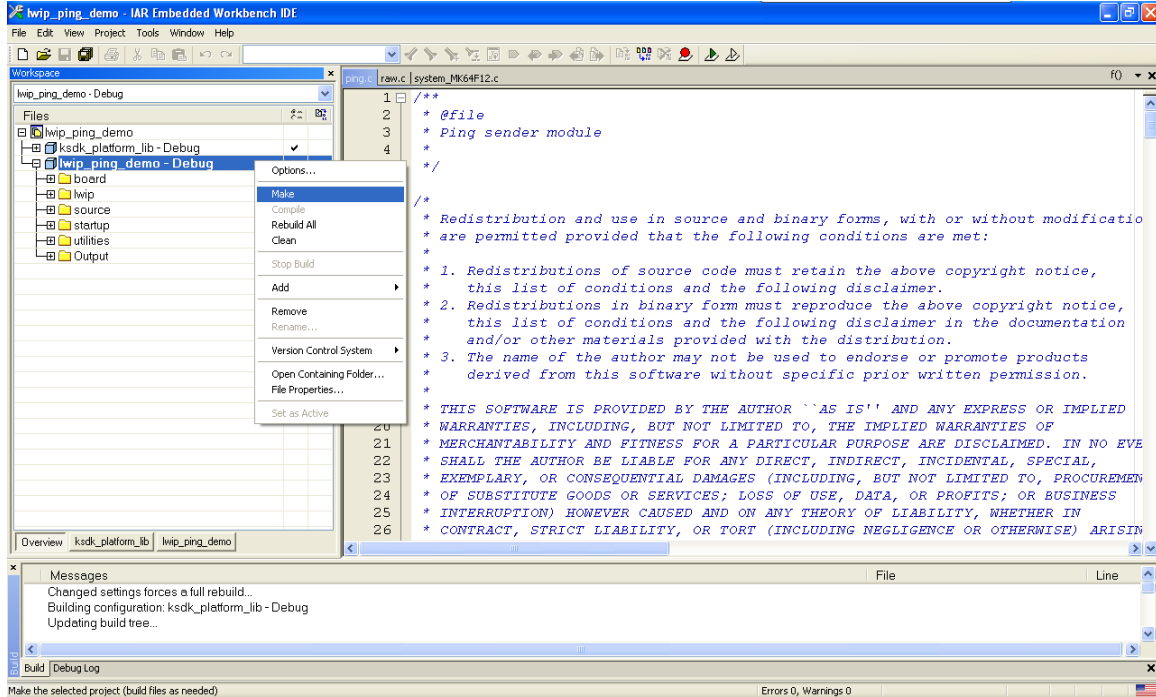



Figure 5-3 lwip_ping_demo



- 
4. Click Download and Debug. Wait for the download to finish.
 5. Click the “Go button” to run the demo.

5.3 Step-by-step guide for Keil

This section shows how to compile and run demos in Keil.

1. Open the workspace corresponding to different demos and different boards. For example, the lwip_ping_demo.uvmpw on Freescale Freedom FRDM-K64F platform under <install_dir>/demos/lwip_ping_demo/ping_bm/uv4/frdmk64f120m or the lwip_ping_demo_freertos.uvmpw on Freescale Freedom FRDM-K64F platform under <install_dir>/demos/lwip_ping_demo/ping_rtos/ping_freertos/uv4/frdmk64f120m. These steps take lwip_ping_demo.uvmpw on Freescale Freedom FRDM-K64F platform for an example.

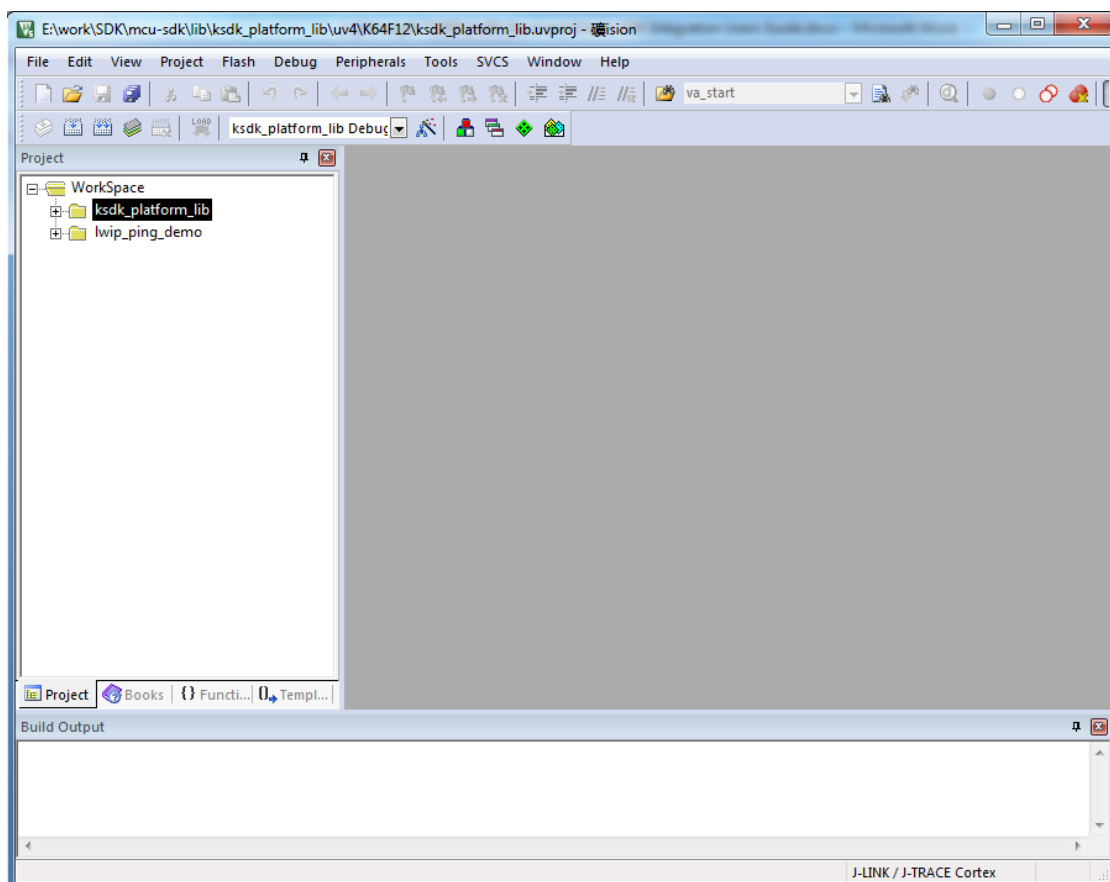


Figure 5-4 Workspace

2. Build the ksdk_platform_lib library.
3. Build the lwip_ping_demo.
4. Click Start/Stop Debug Session. Wait for the download to finish.
5. Click Run to run the demo.

5.4 Step-by-step guide for the Kinetis Design Studio IDE and Atollic

This section shows how to compile and run demos in the Kinetis Design Studio IDE. The steps are identical for Atollic.

1. The Kinetis Design Studio doesn't have a workspace. Create a workspace and import the lwIP demos and the platform/rtos libraries. For example, `ksdk_platform_lib` under `<install_dir>/lib/ksdk_platform_lib/kds/K64F12`, and `.cproject` for `lwip_ping_demo` on Freescale Freedom FRDM-K64F platform under `<install_dir>/demos/lwip_ping_demo/ping_bm/kds/frdmk64f120m`; `ksdk_freertos_lib` under `<install_dir>/lib/ksdk_freertos_lib/kds/K64F12` and `.cproject` for `lwip_ping_demo_freertos` on Freescale Freedom FRDM-K64F platform under `<install_dir>/demos/lwip_ping_demo/ping_rtos/ping_freertos/kds/frdmk64f120m`.

Note

For lwIP and MQX RTOS demos, in addition to the `ksdk_mqx_ib_K64F12` and the demo project, import the `_$(board)` under `<install_dir>/rtos/mqx/mqx/build/kds` and `mqx_stdlib_$(board)` under `<install_dir>/rtos/mqx/mqx_stdlib/build/kds`.

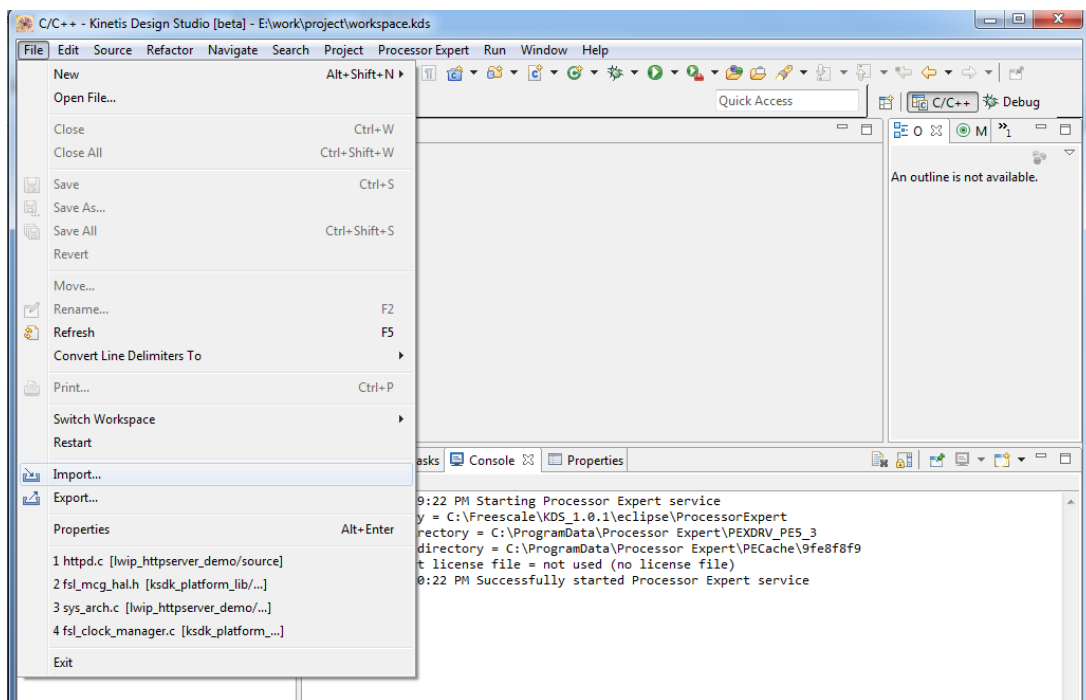


Figure 5-5 Import project -1

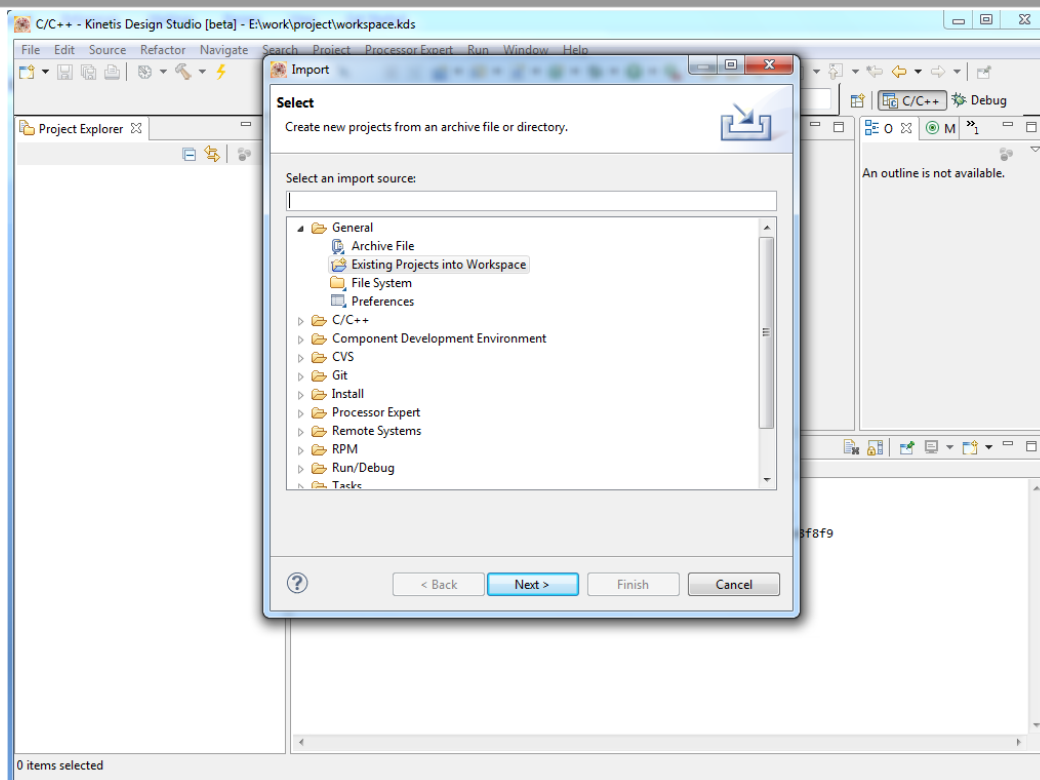


Figure 5-6 Import project - 2

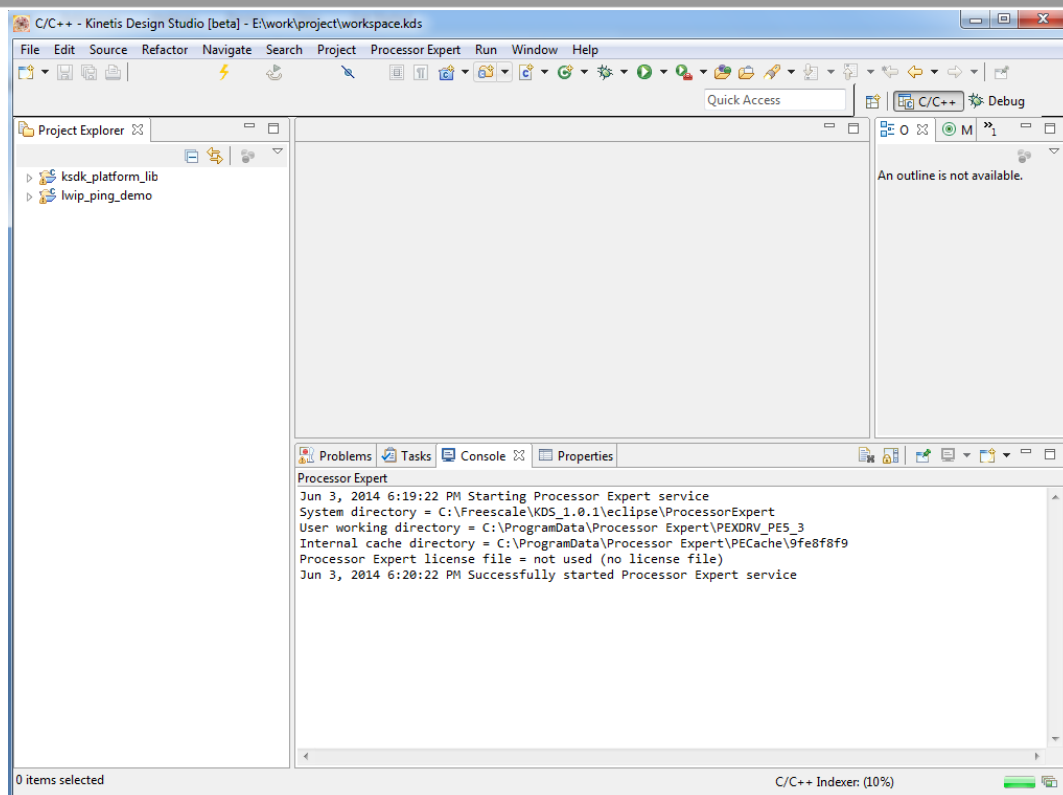


Figure 5-7 Lib project and demo project

2. Build the ksdk_platform_lib library.
3. Build the lwip_ping_demo.
4. Open debug configurations and choose J-Link Debugging.

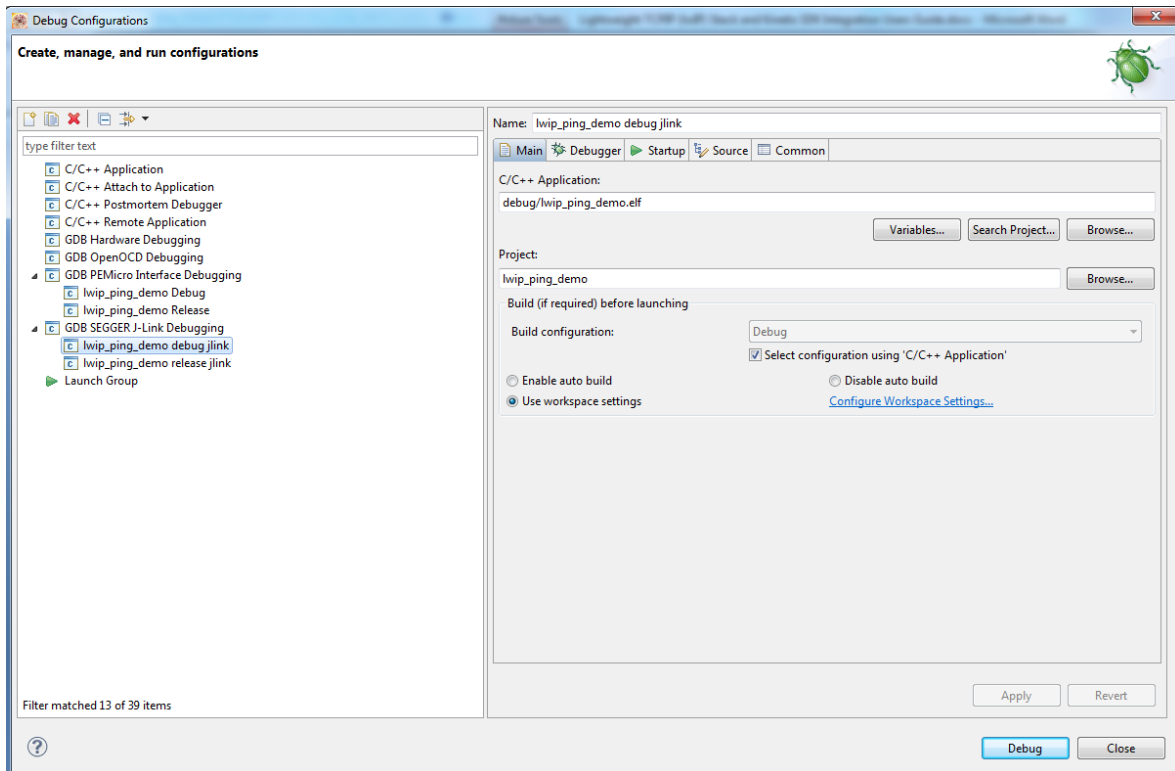


Figure 5-8 Debug Configurations

5. Click the “Debug” button. Wait for the download to finish.
6. Click Resume to run the demo.

5.5 Step-by-step guide for ARMGCC and KDSGCC

1. ARMGCC and KDSGCC both use cMake to generate makefiles. Run the batch file (in Windows®) or sh file (in Linux) to build projects. These steps use ARMGCC as an example.
2. Before building the lwIP demos in the KSDK, the driver library project should be built to generate the library archives:
 - libksdk_platform.a
 - libksdk_platform_freertos.a
 - libksdk_platform_ucosii.a
 - libksdk_platform_ucosii.a
 - libksdk_platform_mqx.a
 - ib_mqx.a
 - lib_mqx_stdlib.a

3. To build the platform library, change the current directory to <install_dir>/lib. libksdk_platform.a for TWR-K64F120M. Under armgcc/K64F12, run build_all.bat to build both debug and release lib.

For lib_mqx.a, change directory to <install_dir>/rtos/mqx/mqx/build/armgcc/mqx_\$(board). Separately run the build_debug.bat and build_release.bat to build debug and release libs.

For lib_mqx_stdlib.a, change directory to

<install_dir>/rtos/mqx/mqx_stdlib/build/armgcc/mqx_stdlib_\$(board) and separately run the build_debug.bat and build_release.bat to build debug and release libs.

4. Change to the demo directory.

For example: <install_dir>/demos/lwip_ping_demo/ping_bm/armgcc/frdmk64f120m

5. Run build_all.bat to build both debug and release projects.

6. Go to the debug/release directory to download and run the elf file using gdb.

6 Revision history

This table summarizes revisions to this document.

Table 1 Revision History		
Revision number	Date	Substantial changes
1.0.0	7/2014	Initial release

How to Reach Us:

Home Page:

freescale.com

Web Support:

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. The ARM is a trademark of ARM Limited (or its subsidiaries) in the EU and/or elsewhere. All rights reserved.

© 2014 Freescale Semiconductor, Inc.

