Application Report
SPRAC56–November 2016

IPC FAQ for KeyStone™ Devices

ABSTRACT

This document is a collection of frequently asked questions (FAQ) about IPC on the KeyStone™ family of devices, along with some useful collateral and software reference links. Project collateral and source code discussed in this application report can be downloaded from the following URL: http://www.ti.com/lit/zip/SPRAC56.

Contents

KeyStone, Code Composer Studio are trademarks of Texas Instruments.
ARM is a registered trademark of ARM Limited.
Linux is a trademark of Linus Torvalds.
Windows is a registered trademark of Microsoft Corporation.

Copyright © 2016, Texas Instruments Incorporated

1 Guide on Building and Running IPC Examples of Processor SDK

The guide in the following link provides step-by-step instructions on how to bring up the target EVMs and how to run the IPC examples of processor SDK on target EVM (see *Guide Keystone II IPC examples*).

The guide provides steps on how to:

- Flash the u-boot and boot u-boot
- Flash the UBIFS image (Linux™ and root file system) into NAND and boot Linux
- Build the IPC package
- Build and run the IPC examples (ex02_MessageQ, ex44_compute, ex45_host, and ex46_graph) on the target EVM
- Real-time debug the DSP side programs using Code Composer Studio™ (CCS)

2 Where do I look for the list of IPC API reference documents?

Go to the *IPC API Reference* page to view the APIs. The IPC product contains the following APIs.

- GateMP (BIOS, Linux, QNX)
- HeapBufMP (BIOS)
- HeapMemMP (BIOS)
- HeapMultiBufMP (BIOS)
- Ipc (BIOS, Linux, QNX)
- ListMP (BIOS)
- MessageQ (BIOS, Linux, QNX)
- MultiProc (BIOS, Linux, QNX)
- NameServer (BIOS, Linux, QNX)
- Notify (BIOS)
- SharedRegion (BIOS)
- IpcPower (BIOS)

Some environments also provide a Multimedia RPC interface. Currently, this is limited to OMAP5 and DRA7XX devices running a high-level operating system (HLOS).

- MmRpc (Linux, QNX)
- MmServiceMgr (BIOS)

Copyright © 2016, Texas Instruments Incorporated

3 How do you rebuild the IPC package and libraries?

The user guide that comes with the IPC package (under the documents folder) is sufficient to build the IPC package. The following steps are from the user guide. For building IPC package, users can use either Cygwin and command-line utility on a Windows® machine or a Linux machine; this cannot be built through CCS.

1. Install the mcsdk_bios_3_xx_xx_xx which will install the IPC package version (ipc_3_xx_xx_xx).
2. Go to the directory where the IPC package is installed.
3. Open the products.mak file and make sure the following parameters are set appropriately.

   For example (for IPC-Linux build):

```
PLATFORM = TCI6638
DESTDIR = /opt/ti/ipc_3_xx_xx_xx/ipc_3_xx_xx_xx_lib

XDC_INSTALL_DIR = /opt/ti/xdctools_3_xx_xx_xx
BIOS_INSTALL_DIR = /opt/ti/bios_6_xx_xx_xx
ti.targets.elf.C66 = /opt/ti/ccsv5/tools/compiler/C6000_7.4.5
```

The following are commands to input to the console to rebuild the package.

```
$cd /opt/ti/ ipc_3_xx_xx_xx
$make distclean
$make -f ipc-linux.mak config
$make
$make install
```

4 Is there any simple example to demonstrate IPC methods like message Q or notify for KeyStone™ II?

Please look at the ex44_compute.zip file in the :~/ti/ipc_3_3x_xx_xx/examples/TCI6638_linux_elf directory.

```
~/ti/ipc_3_3x_xx_xx/examples$ ls
C6472_bios_elf C6A8149_bios_elf DRA7XX_android_elf DRA7XX_linux_elf makefile TCI6638_linux_elf
TI814X_bios_elf C6678_bios_elf dosrc.bat DRA7XX_bios_elf DRA7XX_qnx_elf OMAPL138_linux_elf
TDA3XX_bios_elf
```

Please refer to the readme.txt file to run and build the example according to the target used. For K2, use the cluster ID as 0 instead of 18.

- perl patchExec.pl 0 compute_dspN.xe66 compute_dspN_patched.xe66
- lad_tci6638 -r 8 -n 9 -b 0 -l log.txt

5 For KeyStone™ II, are there any CCS based examples to demonstrate a simple communication between ARM® core to DSP?

No. There are only image processing demos. In MCSDK 3.x, the ARM® core only runs Linux kernel and user Space applications. There is not a Linux application example using CCS; an example like this would use Linux commands.

Copyright © 2016, Texas Instruments Incorporated

6 In IPC packages, there are many test example (sample.c) codes given in the ~\ipc_3_3x_xx_xx\packages\ti\ipc\tests path, but there is only one command line option to build the whole IPC package. No option is available to build the test examples individually. It is time consuming to build the whole IPC package. Customers were asking for a CCS based environment to build and test as individual examples for DSP and ARM® core sides.

The IPC package has been developed to work on multiple platforms such as Linux, Android, QNX, and TI-RTOS (SYS/BIOS), so a command line build that is common across each of these platforms was selected and there is not a CCS based project for these examples.

7 For KeyStone™ II devices, where do I find the source code of the image processing demo and how do I rebuild them (using the ARM® core as a master, DSP cores as slaves)?

The tested version is mcsdk_bios_3_00_03_15. The Image processing demo was tested multiple times with this version and it works. The image processing demo source code can be found in the following path: C:\ti\mcsdk_bios_3_0x_0x_0x\demos\image_processing\ipc (master, slave, and common directories).

7.1 Building Slave Code

The project can be imported into CCS and can be rebuilt.

- K2E: ~\mcsdk_bios_3_0x_0x_0x\demos\image_processing\ipc\evm66ak2e\slave
- K2K: ~\mcsdk_bios_3_0x_0x_0x\demos\image_processing\ipc\evmtci6638k2k\slave
- K2L: ~\mcsdk_bios_3_0x_0x_0x\demos\image_processing\ipc\evmtci6630k2l\slave

7.2 Building Master Code

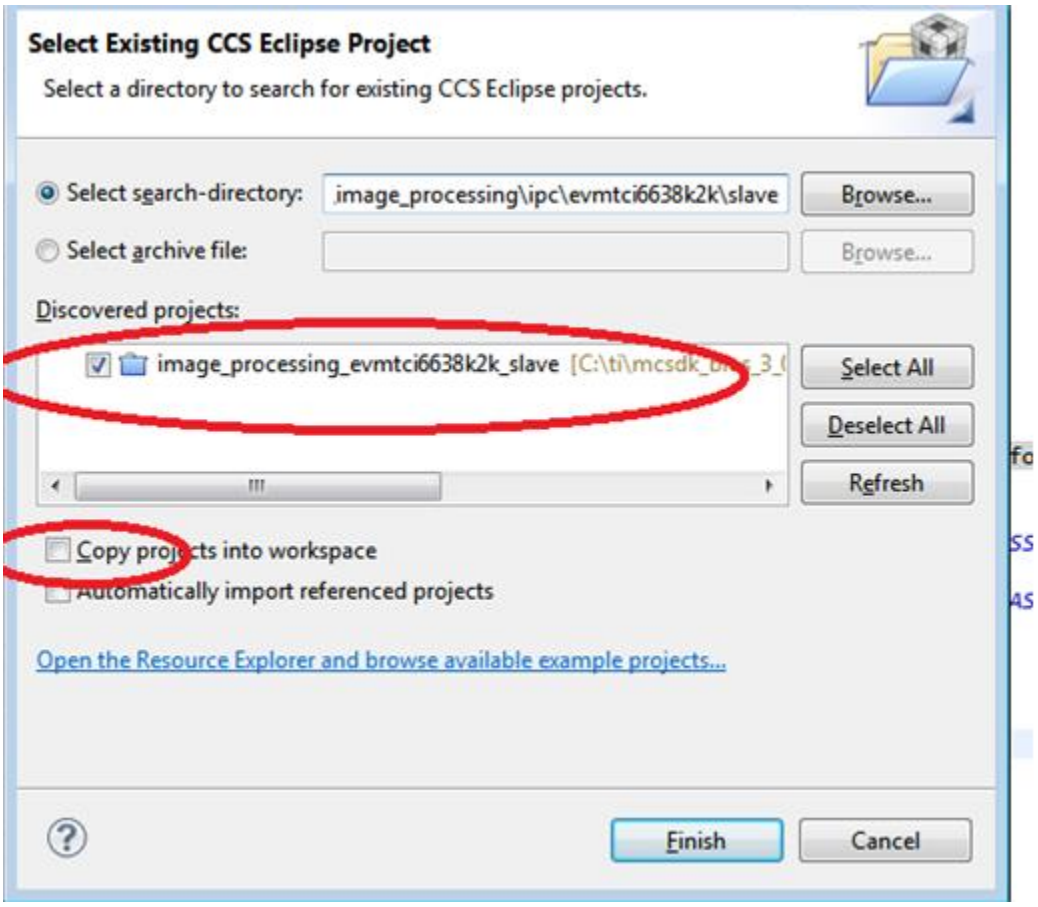
The master code can be built in the Linux environment using the makefile at ~\mcsdk_bios_3_0x_0x_0x\demos\image_processing\ipc\evm66ak2x\master\make which uses the makefile located at ~\mcsdk_bios_3_0x_0x_0x\demos\image_processing\ipc\master\src.

Copyright © 2016, Texas Instruments Incorporated

8 How do I import the slave code of the image processing demo and how do I build it?

Figure 1 shows the correct path that should be provided to pick up the project and its sources. The Copy projects into work space option should be unchecked.

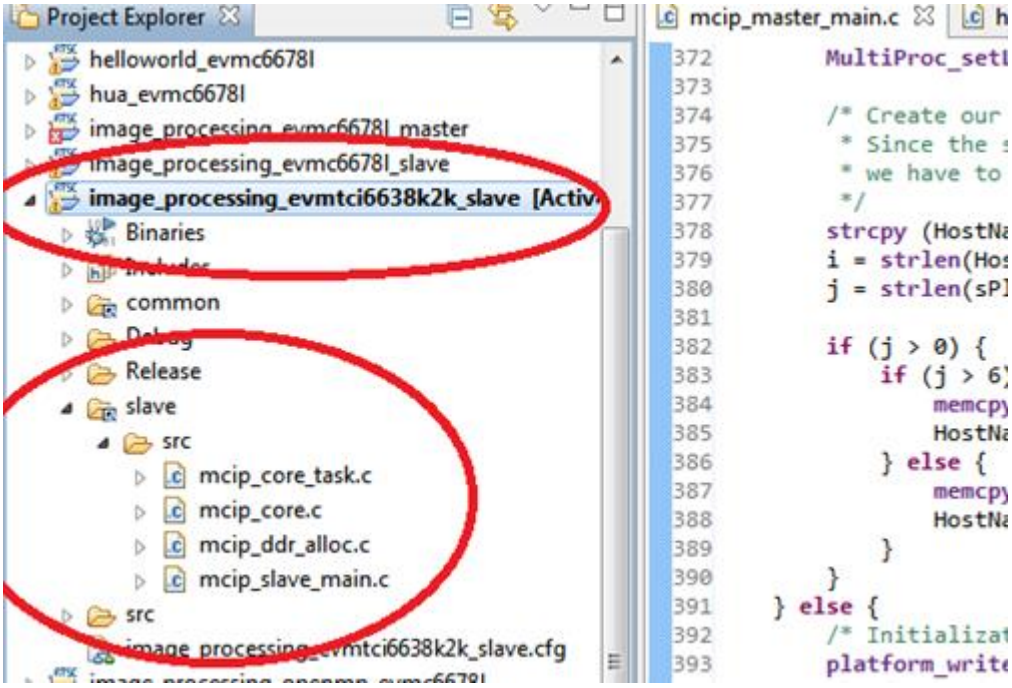
Figure 1. Project and Source File Path



Copyright © 2016, Texas Instruments Incorporated

In the project explorer screen (Figure 2), check whether the slave → src → *.c files can be seen. Right click on the project and give build.

Figure 2. Project Explorer



9 After building the slave code of the image processing demo using CCS, where must it be replaced in the Linux™ file system?

Take the binary file (image_processing_evmtci66xxk2x_slave.out) and replace it in the path target Linux file system (/usr/share/matrix-gui-2.0/apps/demo_imageproc/bin/).

10 While building the ARM® core (master) side code of the image processing demo, I see a compilation error message about Std.h when I make it with or without BUILD_LOCAL=true. How do I resolve this?

Error message:

```
user@ubuntu:~/ti/mcsdk_bios_3_0x_0x_0x/demos/image_processing/ipc/evm66ak2x/master$ make

make[1]: Entering directory
`/home/user/ti/mcsdk_bios_3_0x_0x_0x/demos/image_processing/ipc/master/src'
mcip_mem_mgmt.c:53:24: fatal error: ti/ipc/Std.h: No such file or directory #include
<ti/ipc/Std.h> Compilation terminated.
```

There is an error in the makefile. The makefile must be updated to include the appropriate search path for Std.h. Users can find the path at ../ti/ipc_3_3x_0x_0x/linux/include.

Update the makefile located at mcsdk_bios_3_0x_0x_0x\demos\image_processing\ipc\master\src\makefile as shown in the following code snippet.

```
IPC_INSTALL_DIR := /opt/ti/ipc_3_xx_0x_0x
CFLAGS  := -Wall -I$(COMMON_INC) -I$(MASTER_INC) -I$(IPC_INSTALL_DIR)/linux/include -
I$(IPC_INSTALL_DIR)/packages -D_GNU_SOURCE
```


Copyright © 2016, Texas Instruments Incorporated

11 While building the ARM® core (master) side code of the image processing demo, I see a linker error message. How do I resolve this?

Error message:

```
/usr/bin/ld: skipping incompatible /opt/ti/ipc_3_35_01_07/examples/TCI6638_linux_elf
/ex44_compute_bkp/lib//libtitransportrpmsg.a when searching for -ltitransportrpmsg
/usr/bin/ld: cannot find –ltitransportrpmsg
usr/bin/ld: cannot find -ltiipcutils
collect2: error: ld returned 1 exit status
make: *** [../../master/image_processing_master.out] Error 1
root@e2e:/opt/ti/mcsdk_bios_3_0x_xx_xx/demos/image_processing/ipc/master/src#
```

For these errors, ensure the whole IPC package has been built and the libraries (such as transportrpmsg) have been installed in a destination directory. This destination directory should be given in the makefile to find those libraries.

For example, in the makefile located at mcsdk_bios_3_0x_xx_xx\demos\image_processing\ipc\master\src\makefile, reference the following code snippets.

```
IPC_INSTALL_DIR := /opt/ti/ipc_3_3x_xx_xx

#The location where the libraries are installed after building the IPC package
SIPC_LINUX_DIR := /opt/ti/ipc_3_3x_xx_xx/IPC_Linux_libraries

CROSS_COMPILE ?= arm-linux-gnueabihf-
CC            := $(CROSS_COMPILE)gcc
CFLAGS        := -Wall -I$(COMMON_INC) -I$(MASTER_INC) -I$(IPC_INSTALL_DIR)/linux/include -
I$(IPC_INSTALL_DIR)/packages -D_GNU_SOURCE
LFLAGS        := -lpthread -L$(SIPC_LINUX_DIR)/ -ltitransportrpmsg -L$(SIPC_LINUX_DIR)/ -ltiipc -
L$(SIPC_LINUX_DIR)/ -ltiipcutils
```

12 The image processing demo does not work on the version of MCSDK, V3.0x.xx.x on both the K2H and K2E EVMs. The earlier version of MCSDK works on both the EVMs. Will it be fixed on next version?

Ensure the u-boot environments are set to work on MCSDK 3.x as shown in the following code snippet.

```
u-boot# env default -f -a
u-boot# setenv mem_reserve 1536M
u-boot# saveenv
```

13 How do you build and run the qmssIpcBenchmark on the C6678 EVM?

13.1 Hardware Setup

Set the boot mode dip switch to no boot/EMIF16 mode, then connect the power and emulator to the C6678 EVM.

13.2 Software Setup

Use the following instructions to set up the software.

1. Power on the EVM.
2. Create and launch the target configuration file (.ccxml) for CCS debugging.
3. Group Core 0 and Core 1 in CCS.
4. Connect to both cores through the group.
5. Load the *evmc66xxl.gel* file to initialize the DDR.

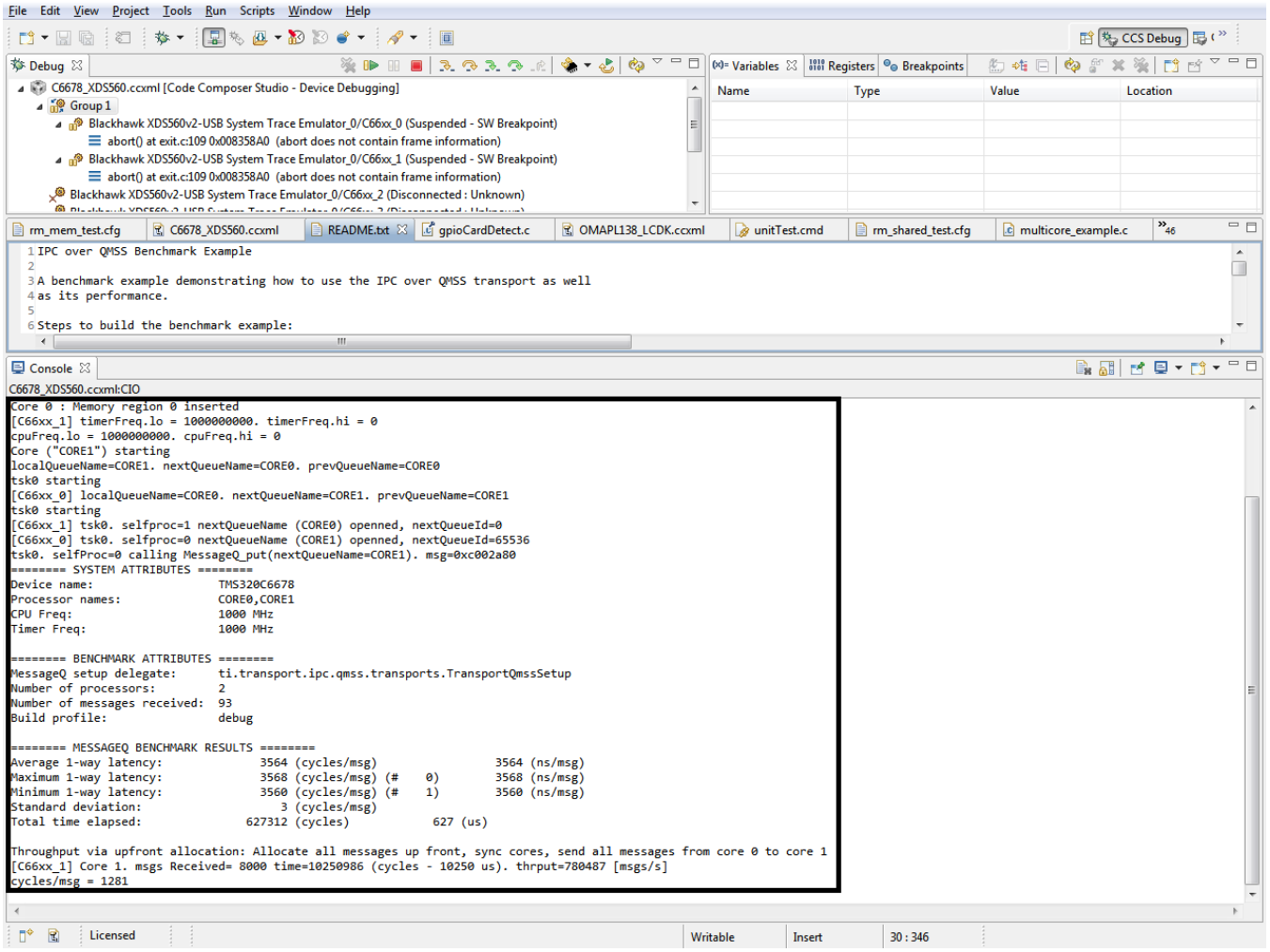
Copyright © 2016, Texas Instruments Incorporated

The GEL can be found in the CCS install dir"\ccsv5\ccs_base_x.x.x.xxxxx\emulation\boards\evmc66xxl\gel directory. Once the file is loaded:

6. Execute the default setup script on each core. In the CCS menu, go to Scripts → EVMC6678L Init Functions → Global_Default_Setup.
7. Highlight the group in the CCS Debug window, then load transport\ipc\examples\qmssIpcBenchmark\Debug\qmssIpcBenchmark_c66xx.out on each core (see Figure 3).
8. Highlight the group in the CCS Debug window, run the program in CCS on both cores simultaneously (see Figure 3).

   qmssIpcBenchmark_c66xx will send messageQ messages between the cores through the QMSS transport. The messages will be used to measure the performance of the transport. The test will be complete after the throughput (msg/s) has been calculated.

Figure 3. Debug Window



Copyright © 2016, Texas Instruments Incorporated

14 How can I build the qmssIpcbenchmark of pdk_C6678_1_1_2_x pdk_C6678_1_1_2_x with the release build configuration?

The option *-mo -o3 -q -k -eo.o* works for building the IPC-QMSS transport library in release mode, and the option *-mo -g -q -k -eo.o* works for building the IPC-QMSS transport library in debug mode.

The common.bld script of the IPC does not create a release folder. By default, this script always creates the debug folder and dumps all of the binaries. By tweaking the common.bld file, a release folder can be made and the IPC-qmssIpcBenchmark project can be built in release mode.

14.1 How do you change the Common.bld file?

Use the following instructions to change the common.bld file.

1. Go to C:\ti\ipc_3_00_xx_xx\packages\ti\sdo\ipc\build\Common.bld.

   NOTE: Go to the IPC version used for building the transport library. Here, this is IPC version 3.00.4.29.

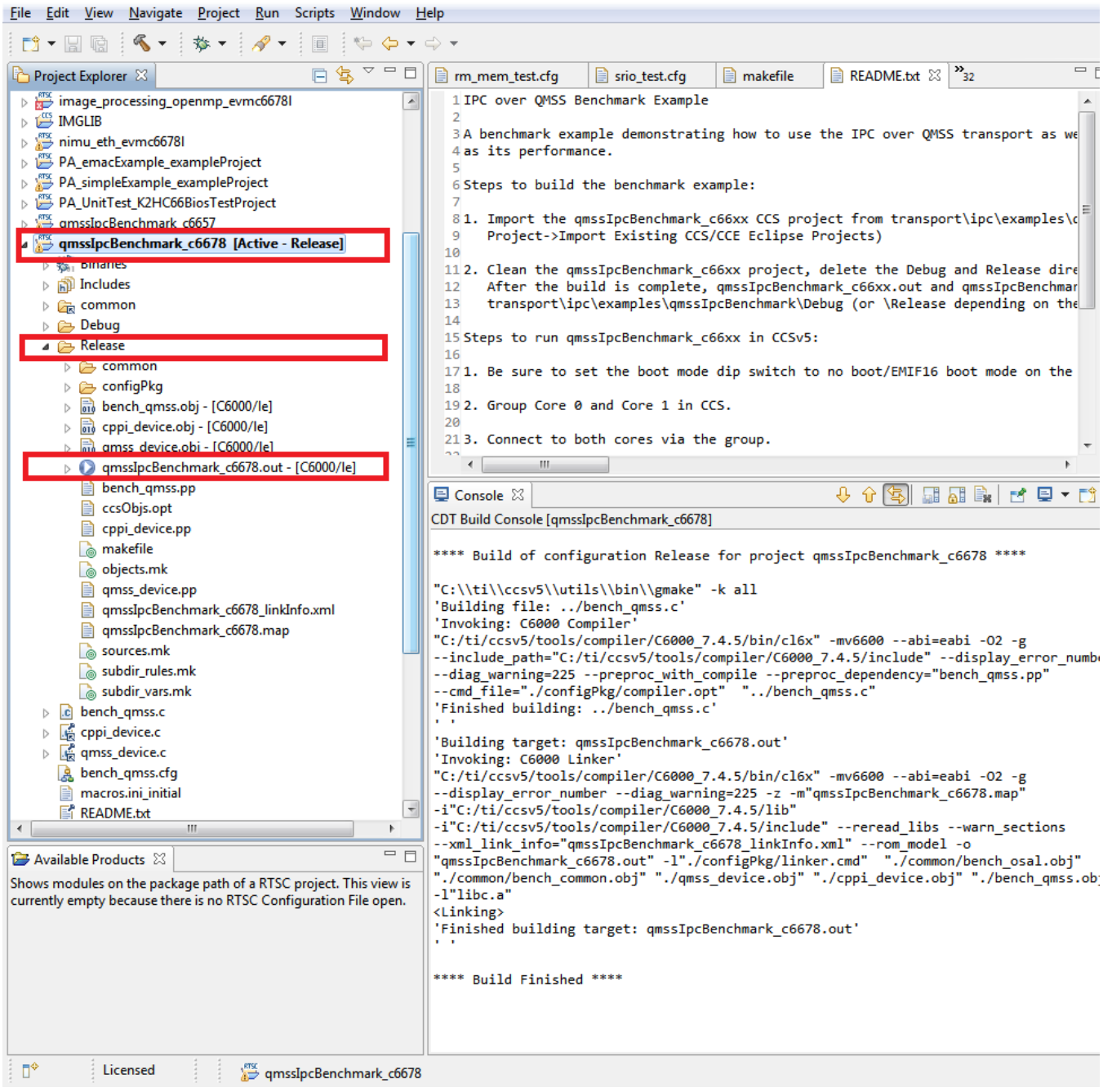
2. Modify the following code.

```
Line No:88 profiles[0] = "release";
Line No: 99 var libPath = "lib/ipc/release/";
```

3. Build with option *-mo -o3 -q -k -eo.o* in the *config.bld* file of the transport library (\ti\pdk_C6678_1_1_2_x\packages\ti\transport\ipc\qmss\transports).
4. Build the qmssIpcBenchmark project. Figure 4 shows a screenshot of the successful build.

Copyright © 2016, Texas Instruments Incorporated

Figure 4. Successful Release Build



Copyright © 2016, Texas Instruments Incorporated

15 How do you rebuild the IPC-QMSS transport library and generate ti.transport.ipc.qmss.transports.ae66?

After installing PDK, go to ~\ti\pdk_C6678_1_1_2_x\packages\ti\transport\ipc\qmss\transports, then use the following steps.

1. (Optional — required for single-step-debugging) Modify the*config.bld* file *C66LE/BE.ccOpts.prefix* of the transports library to remove optimization and add symbolic debug.
   - From: -mo -o3 -q -k -eo.o
   - To: -mo -g -q -k -eo.o
2. Navigate to the pdk\packages\ti\transport\ipc\(qmss or srio) directory by using a command prompt.
3. Configure the XDCPATH environment variable with the BIOS menu and IPC install locations.

   ```
   set XDCPATH=c:\ti\bios_w_xx_yy_zz\packages\
   set XDCPATH=%XDCPATH%;c:\ti\ipc_w_xx_yy_zz\packages\
   ```

4. Configure the XDCCGROOT environment variable with the compiler install path (using CGT 7.2.4 installed as part of CCS as an example).

   ```
   set XDCCGROOT=c:\ti\ccsv5\tools\compiler\c6000_7.2.4
   ```

5. Add the XDC tools to the system PATH.

   ```
   set PATH=%PATH%;c:\ti\xdctools_w_xx_yy_zz\
   ```

6. Clean the transport.

   ```
   >xdc clean -PR .
   ```

7. Build the transport.

   ```
   >xdc -PR .
   ```

Users should be able to build in release mode by using the previous steps.

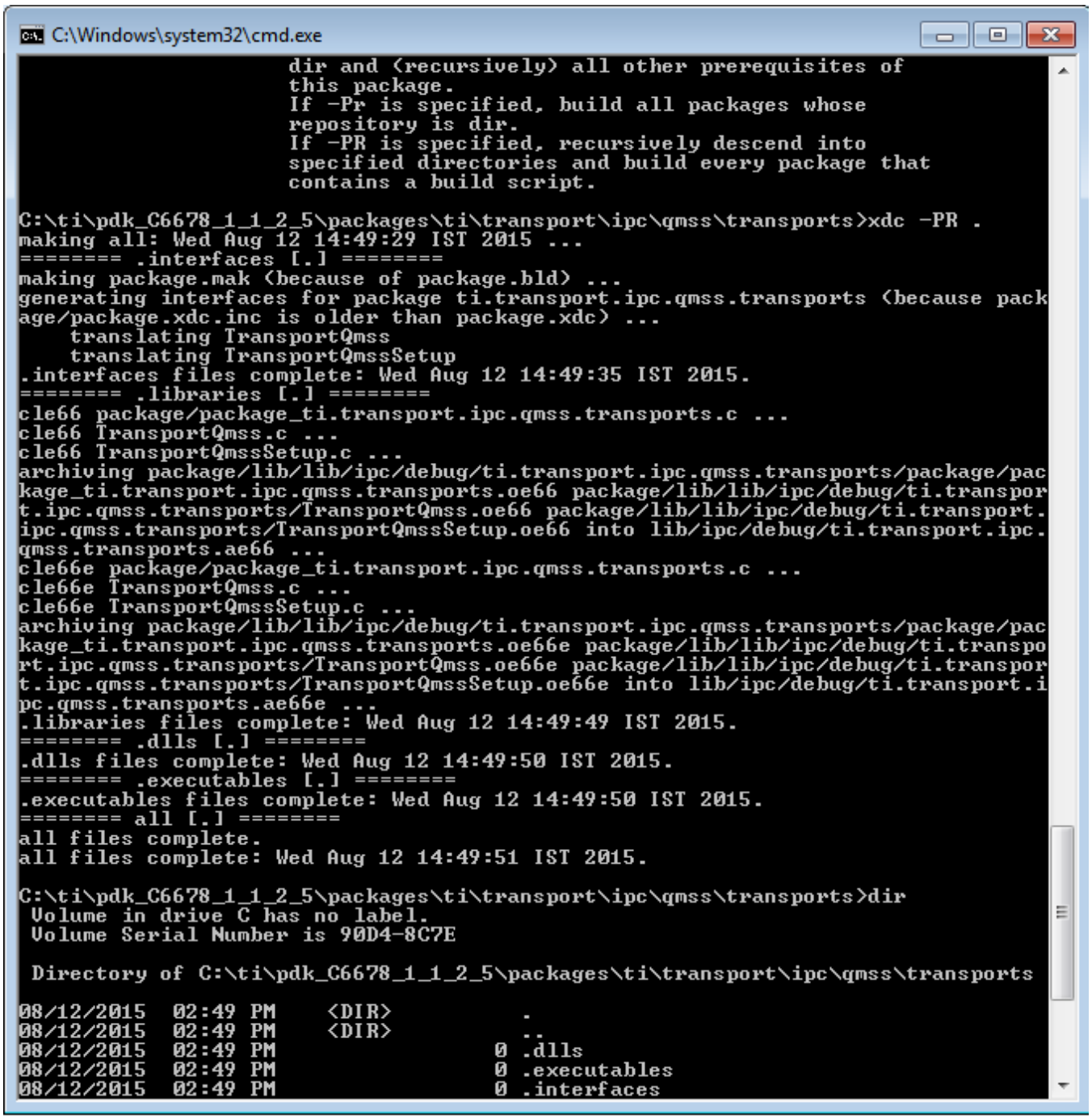
NOTE: To allow single-step debugging of the IPC and BIOS source, rebuild the example projects with the following command added to the .cfg file of the example.

```
BIOS.libType = BIOS.LibType_Debug;
```


Copyright © 2016, Texas Instruments Incorporated

Figure 5 shows the build log.

Figure 5. Build Log



Copyright © 2016, Texas Instruments Incorporated

IMPORTANT NOTICE

Texas Instruments Incorporated and its subsidiaries (TI) reserve the right to make corrections, enhancements, improvements and other changes to its semiconductor products and services per JESD46, latest issue, and to discontinue any product or service per JESD48, latest issue. Buyers should obtain the latest relevant information before placing orders and should verify that such information is current and complete. All semiconductor products (also referred to herein as “components”) are sold subject to TI’s terms and conditions of sale supplied at the time of order acknowledgment.

TI warrants performance of its components to the specifications applicable at the time of sale, in accordance with the warranty in TI’s terms and conditions of sale of semiconductor products. Testing and other quality control techniques are used to the extent TI deems necessary to support this warranty. Except where mandated by applicable law, testing of all parameters of each component is not necessarily performed.

TI assumes no liability for applications assistance or the design of Buyers’ products. Buyers are responsible for their products and applications using TI components. To minimize the risks associated with Buyers’ products and applications, Buyers should provide adequate design and operating safeguards.

TI does not warrant or represent that any license, either express or implied, is granted under any patent right, copyright, mask work right, or other intellectual property right relating to any combination, machine, or process in which TI components or services are used. Information published by TI regarding third-party products or services does not constitute a license to use such products or services or a warranty or endorsement thereof. Use of such information may require a license from a third party under the patents or other intellectual property of the third party, or a license from TI under the patents or other intellectual property of TI.

Reproduction of significant portions of TI information in TI data books or data sheets is permissible only if reproduction is without alteration and is accompanied by all associated warranties, conditions, limitations, and notices. TI is not responsible or liable for such altered documentation. Information of third parties may be subject to additional restrictions.

Resale of TI components or services with statements different from or beyond the parameters stated by TI for that component or service voids all express and any implied warranties for the associated TI component or service and is an unfair and deceptive business practice. TI is not responsible or liable for any such statements.

Buyer acknowledges and agrees that it is solely responsible for compliance with all legal, regulatory and safety-related requirements concerning its products, and any use of TI components in its applications, notwithstanding any applications-related information or support that may be provided by TI. Buyer represents and agrees that it has all the necessary expertise to create and implement safeguards which anticipate dangerous consequences of failures, monitor failures and their consequences, lessen the likelihood of failures that might cause harm and take appropriate remedial actions. Buyer will fully indemnify TI and its representatives against any damages arising out of the use of any TI components in safety-critical applications.

In some cases, TI components may be promoted specifically to facilitate safety-related applications. With such components, TI’s goal is to help enable customers to design and create their own end-product solutions that meet applicable functional safety standards and requirements. Nonetheless, such components are subject to these terms.

No TI components are authorized for use in FDA Class III (or similar life-critical medical equipment) unless authorized officers of the parties have executed a special agreement specifically governing such use.

Only those TI components which TI has specifically designated as military grade or “enhanced plastic” are designed and intended for use in military/aerospace applications or environments. Buyer acknowledges and agrees that any military or aerospace use of TI components which have ***not*** been so designated is solely at the Buyer's risk, and that Buyer is solely responsible for compliance with all legal and regulatory requirements in connection with such use.

TI has specifically designated certain components as meeting ISO/TS16949 requirements, mainly for automotive use. In any case of use of non-designated products, TI will not be responsible for any failure to meet ISO/TS16949.

Products		**Applications**	
Audio	www.ti.com/audio	Automotive and Transportation	www.ti.com/automotive
Amplifiers	amplifier.ti.com	Communications and Telecom	www.ti.com/communications
Data Converters	dataconverter.ti.com	Computers and Peripherals	www.ti.com/computers
DLP® Products	www.dlp.com	Consumer Electronics	www.ti.com/consumer-apps
DSP	dsp.ti.com	Energy and Lighting	www.ti.com/energy
Clocks and Timers	www.ti.com/clocks	Industrial	www.ti.com/industrial
Interface	interface.ti.com	Medical	www.ti.com/medical
Logic	logic.ti.com	Security	www.ti.com/security
Power Mgmt	power.ti.com	Space, Avionics and Defense	www.ti.com/space-avionics-defense
Microcontrollers	microcontroller.ti.com	Video and Imaging	www.ti.com/video
RFID	www.ti-rfid.com		
OMAP Applications Processors	www.ti.com/omap	**TI E2E Community**	e2e.ti.com
Wireless Connectivity	www.ti.com/wirelessconnectivity		

Mailing Address: Texas Instruments, Post Office Box 655303, Dallas, Texas 75265
Copyright © 2016, Texas Instruments Incorporated