



OpenAMP Framework Getting Started Guide

**© 2011-2014 Mentor Graphics Corporation
All rights reserved.**

This document contains information that is proprietary to Mentor Graphics Corporation. The original recipient of this document may duplicate this document in whole or in part for internal business purposes only, provided that this entire notice appears in all copies. In duplicating any part of this document, the recipient agrees to make every reasonable effort to prevent the unauthorized use and distribution of the proprietary information.

This document is for information and instruction purposes. Mentor Graphics reserves the right to make changes in specifications and other information contained in this publication without prior notice, and the reader should, in all cases, consult Mentor Graphics to determine whether any changes have been made.

The terms and conditions governing the sale and licensing of Mentor Graphics products are set forth in written agreements between Mentor Graphics and its customers. No representation or other affirmation of fact contained in this publication shall be deemed to be a warranty or give rise to any liability of Mentor Graphics whatsoever.

MENTOR GRAPHICS MAKES NO WARRANTY OF ANY KIND WITH REGARD TO THIS MATERIAL INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.

MENTOR GRAPHICS SHALL NOT BE LIABLE FOR ANY INCIDENTAL, INDIRECT, SPECIAL, OR CONSEQUENTIAL DAMAGES WHATSOEVER (INCLUDING BUT NOT LIMITED TO LOST PROFITS) ARISING OUT OF OR RELATED TO THIS PUBLICATION OR THE INFORMATION CONTAINED IN IT, EVEN IF MENTOR GRAPHICS HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

U.S. GOVERNMENT LICENSE RIGHTS: The software and documentation were developed entirely at private expense and are commercial computer software and commercial computer software documentation within the meaning of the applicable acquisition regulations. Accordingly, pursuant to FAR 48 CFR 12.212 and DFARS 48 CFR 227.7202, use, duplication and disclosure by or for the U.S. Government or a U.S. Government subcontractor is subject solely to the terms and conditions set forth in the license agreement provided with the software, except for provisions which are contrary to applicable mandatory federal laws.

TRADEMARKS: The trademarks, logos and service marks ("Marks") used herein are the property of Mentor Graphics Corporation or other parties. No one is permitted to use these Marks without the prior written consent of Mentor Graphics or the owner of the Mark, as applicable. The use herein of a third-party Mark is not an attempt to indicate Mentor Graphics as a source of a product, but is intended to indicate a product from, or associated with, a particular third party. A current list of Mentor Graphics' trademarks may be viewed at: www.mentor.com/trademarks.

The registered trademark Linux[®] is used pursuant to a sublicense from LMI, the exclusive licensee of Linus Torvalds, owner of the mark on a world-wide basis.

Mentor Graphics Corporation
8005 S.W. Boeckman Road, Wilsonville, Oregon 97070-7777
Telephone: 503.685.7000
Toll-Free Telephone: 800.592.2210
Website: www.mentor.com
SupportNet: supportnet.mentor.com/

Send Feedback on Documentation: supportnet.mentor.com/doc_feedback_form

Table of Contents

Chapter 1	
OpenAMP Framework Overview	7
OpenAMP Framework Configuration	7
Supported Environment	8
Directory Structure	9
Chapter 2	
OpenAMP Framework Applications and Tests	11
Linux Master/Nucleus Remote, and Linux Master/Baremetal Remote Configurations . . .	11
Nucleus Master/Nucleus Remote, and Nucleus Master/Baremetal Remote Configurations	13
Nucleus Master/Linux Remote and Baremetal Master /Linux Remote Configurations . . .	16
Chapter 3	
Environment Setup, Build, and Execution of Applications and Test Binaries	19
OpenAMP Framework Environment Setup	19
Setting up the PetaLinux Environment	20
Setting up the Hardware	23
Build and Execute Applications for Linux Master/Nucleus Remote and Linux Master/Baremetal Remote Configurations	24
Building the OpenAMP Framework Library, Applications, and Tests	24
Building Linux Master Applications and Tests	25
Executing Applications and Tests for Linux Master/ Nucleus Remote and Linux Master/Baremetal Remote Configurations	28
Executing the Matrix Multiply Application	28
Executing the Echo Test Application	32
Executing the Proxy Application	35
Build and Execute Applications for Nucleus Master/Nucleus Remote and Nucleus Master/Baremetal Remote Configurations	38
Building the OpenAMP Framework Library, Applications, and Tests	38
Executing Applications and Tests for Nucleus Master/Nucleus Remote and Nucleus Master/Baremetal Remote Configurations	39
Executing the Matrix Multiply Application	40
Executing the Echo Test Application	42
Executing the Functional Test Suite	44
Build and Execute Applications for Nucleus Master/Linux Remote and Baremetal Master/ Linux Remote Configurations	50
Building a Linux Remote Kernel Image	50
Building the OpenAMP Framework Library, Applications and Tests	54
Executing Applications and Tests for Nucleus Master/Linux Remote Configuration . . .	54
Executing the Matrix Multiply Application for Nucleus Master/Linux Remote	55
Executing the Echo Test Application	62
Executing the Functional Test Suites	68

Executing Applications and Tests for Baremetal Master/Linux Remote Configuration ..	70
Executing the Matrix Multiply Application for Baremetal Master/Linux Remote	71
Executing the Echo Test Application	77
Executing the Functional Test Suites	81

Third-Party Information

Embedded Software and Hardware License Agreement

List of Figures

List of Tables

Table 1-1. OpenAMP Framework Configurations	7
Table 1-2. Supported Environments	8
Table 3-1. OpenAMP Framework Environment	19
Table 3-2. Ubuntu Tools and Libraries	21

Chapter 1

OpenAMP Framework Overview

This document presents the steps to setup the environment, build and execute the sample applications, tests, and firmware provided within the OpenAMP Framework package.

This overview contains the following information for getting started with the OpenAMP Framework:

OpenAMP Framework Configuration	7
Supported Environment	8
Directory Structure	9

OpenAMP Framework Configuration

The OpenAMP Framework package works with multiple configurations.

[Table 1-1](#) shows a summary of the these configurations and the capabilities showcased in these configurations:

Table 1-1. OpenAMP Framework Configurations

Config	Master	Remote	Capabilities showcased using OpenAMP Framework
1	Linux ^{®1}	Nucleus	<ul style="list-style-type: none">• Ability to load a remote Nucleus RTOS context from the Linux master.• Ability to perform IPC from the Linux master to the Nucleus remote.
2	Linux	Baremetal	<ul style="list-style-type: none">• Ability to load a remote baremetal context from the Linux master.• Ability to perform IPC from the Linux master to the baremetal remote.
3	Nucleus	Nucleus	<ul style="list-style-type: none">• Ability to load a remote Nucleus RTOS context from the Nucleus RTOS master.• Ability to perform IPC from the Nucleus master to the Nucleus remote.
4	Nucleus	Baremetal	<ul style="list-style-type: none">• Ability to load a remote baremetal context from the Nucleus RTOS master.• Ability to perform IPC from the Nucleus master to the baremetal remote.

Table 1-1. OpenAMP Framework Configurations (cont.)

Config	Master	Remote	Capabilities showcased using OpenAMP Framework
5	Nucleus	Linux	<ul style="list-style-type: none"> • Ability to load a remote Linux context from the Nucleus RTOS master. • Ability to perform IPC from the Linux rpmsg master to the Nucleus rpmsg remote.
6	Baremetal	Linux	<ul style="list-style-type: none"> • Ability to load a remote Linux context from the baremetal master. • Ability to perform IPC from the Linux rpmsg master to the baremetal rpmsg remote.

1. "Linux® is a registered trademark of Linus Torvalds in the U.S. and other countries.

Related Topics

[Supported Environment](#)

[Directory Structure](#)

Supported Environment

The OpenAMP Framework is supported for the following environment:

[Table 1-2](#) shows the hardware platform, Nucleus RTOS version, Linux version, and the toolset.

Table 1-2. Supported Environments

Hardware Platform	Xilinx Zynq-7000 All Programmable SoC ZC702 Evaluation Kit
Nucleus RTOS	Based on Nucleus ReadyStart 2013.08.1 with Zynq ZC702evk BSP (Pre-build Nucleus libraries are provided with the OpenAMP Framework and the user is not required to setup a Nucleus environment).
Linux	PetaLinux v2013.10
Toolset	<p>The OpenAMP Framework SW configurations were tested with two tool chains:</p> <ul style="list-style-type: none"> • Mentor Graphics CodeSourcery GNU Lite tools: <ul style="list-style-type: none"> • Target: arm-non-eabi, gcc version 4.8.1 (Sourcery CodeBench Lite 2013.11-24) • Xilinx GCC tools: <ul style="list-style-type: none"> • Target: arm-xilinx-eabi, gcc version 4.7.3 (Sourcery CodeBench Lite 2013.05-39)

Related Topics

[OpenAMP Framework Configuration](#)

[Supported Environment](#)

Directory Structure

The directory structure looks like below:

```

+ ---- apps                -----> OpenAMP Framework applications
|   + ---- samples        -----> OpenAMP Framework Samples
|     + ---- master       -----> Master sample applications
|     + ---- remote       -----> Remote sample applications
|   + ---- tests          -----> OpenAMP Framework Tests
|     + ---- master       -----> Master test applications
|     + ---- remote       -----> Remote test applications
+ ---- common             -----> Common OpenAMP Framework components
+ ---- docs               -----> Documentation
+ ---- include            -----> External include files
|   + ---- open_amp.h
+ ---- libs               -----> OpenAMP Framework lib and System libs for Nucleus
                             RTOS and BMEs.
+ ---- porting
|   + ---- config         -----> Interface to define system level configuration
                             information like-remote firmware info.
|   + ---- env            -----> Software environment interface (OS or baremetal)
|   + ---- hil            -----> Hardware interface layer
+ ---- proxy              -----> Proxy infrastructure for Linux master and Nucleus
                             RTOS and baremetal remotes
+ ---- remoteproc        -----> remoteproc sub-component
+ ---- rpmsg              -----> rpmsg sub-component
+ ---- virtio             -----> virtio sub-component

```

Related Topics

[OpenAMP Framework Configuration](#)

[Supported Environment](#)

Chapter 2

OpenAMP Framework Applications and Tests

This chapter briefly describes the sample applications and tests provided within the OpenAMP Framework package for each supported system configuration.

Linux Master/Nucleus Remote, and Linux Master/Baremetal Remote Configurations

This demonstration showcases the Life Cycle Management (LCM) of the remote processor and its software context and the Inter Processor Communication (IPC) with remote software context.

This section is for the Linux Master/Nucleus or baremetal remote configurations.

Matrix Multiplication Demonstration

The Linux kernels space application on the master processor showcases IPC with remote Nucleus RTOS and baremetal based firmware using rpmsg. The `zynq_remoteproc_driver` is used to load the appropriate remote firmware image.

The Linux user space application performs the same matrix multiplication functionality by accessing the `rpmsg` character device created and handled by the `rpmsg_user_dev_driver`.

1. Linux master boots remote Nucleus or baremetal based firmware using the `zynq_remoteproc_driver`.
2. Linux master application transmits two random matrices to remote firmware using `rpmsg`.
3. Nucleus RTOS or baremetal based firmware performs multiplication and transmits results back to master using `rpmsg`.
4. Linux master application prints the results of the matrix multiplication

Application	Location
Linux Master (kernel space)	<i>apps/samples/master/linux/kernelspace/rpmsg_mat_mul_kern_app</i>
Linux Master (user space)	<i>apps/samples/master/linux/userspace/matrix_multiply</i>
Nucleus remote firmware	<i>apps/samples/remote/nucleus/matrix_multiply</i>

Application	Location
Baremetal remote firmware	<i>apps/samples/remote/baremetal/matrix_multiply</i>

Echo Test Application

Description

This application tests data integrity of the payload transmitted. The Linux master loads Nucleus/baremetal based firmware first, executes the echo test and performs shutdown.

Echo test between Linux master<-> Nucleus/baremetal remote based firmware

1. Linux master boots remote Nucleus/baremetal based firmware using remoteproc.
2. Linux master application transmits echo payloads to remote firmware using rpmsg.
3. Nucleus/baremetal remote based firmware echoes back the payload received to the master using rpmsg.
4. Linux master application verifies integrity of payload received and prints results.

Application	Location
Linux master (kernel space)	<i>apps/tests/master/linux/kernelspace/rpmsg_echo_test_kern_app</i>
Nucleus remote	<i>apps/tests/remote/nucleus/echo_test</i>
Linux master (user space)	<i>apps/tests/master/linux/userspace/echo_test</i>
Baremetal remote	<i>apps/tests/remote/baremetal/echo_test</i>

Proxy Application

Description

This application allows the firmware on the remote core to use console and execute file I/O on master by communicating with a proxy application running on the Linux master.

Demonstration to showcase proxy infrastructure for Linux master <-> Nucleus/baremetal remote configuration.

1. The Linux master boots Nucleus/baremetal based remote firmware using proxy_app.

2. Remote firmware does File I/O on Linux FS on master processor, and uses master console to receive input from the user and display output.

Application	Location
Linux master (kernel space driver)	<i>proxy/master/linux/kernelspace</i>
Linux master (user space application)	<i>proxy/master/linux/userspace</i>
Nucleus remote	<i>apps/samples/remote/nucleus/rpc_demo</i>
Baremetal remote	<i>apps/samples/remote/baremetal/rpc_demo</i>

Related Topics

[Nucleus Master/Nucleus Remote, and Nucleus Master/Baremetal Remote Configurations](#)

[Nucleus Master/Linux Remote and Baremetal Master /Linux Remote Configurations](#)

Nucleus Master/Nucleus Remote, and Nucleus Master/Baremetal Remote Configurations

This demonstration showcases the Life Cycle Management (LCM) of the remote processor and its software context, and the Inter Processor Communication (IPC) with remote software context.

This section is for the Nucleus Master/Nucleus or baremetal remote configurations.

Matrix Multiplication Demonstration

Description

The Nucleus master application loads Nucleus based firmware first, demonstrates functionality, performs shutdown, and then boots baremetal based firmware next to demonstrate the same functionality, and performs shutdown.

To demonstrate LCM and IPC between Nucleus master <-> Nucleus remote based firmware

- 1.) The Nucleus master application boots Nucleus remote based firmware using `remoteproc`.
- 2.) The Nucleus master application transmits random matrices to remote firmware using `rpmsg`.
- 3.) The Nucleus remote firmware performs multiplications of matrices and transmits results back to master using `rpmsg`.
- 4.) The Nucleus master application prints results of matrix multiplication.

To demonstrate LCM and IPC between Nucleus master<-> baremetal remote based firmware

- 5.) The Nucleus master application boots baremetal remote based firmware using remoteproc.
- 6.) The Nucleus master application transmits random matrices to remote firmware using rpmsg.
- 7.) The baremetal remote firmware performs multiplications and transmits results back to master using rpmsg.
- 8.) The Nucleus master application prints the results of matrix multiplication.

Application	Location
Nucleus master	<i>apps/samples/master/nucleus/matrix_multiply/nucleus_nucleusbm</i>
Nucleus remote	<i>apps/samples/remote/nucleus/matrix_multiply</i>
Baremetal remote	<i>apps/samples/remote/baremetal/matrix_multiply</i>

Echo Test Application

Description

This application tests data integrity of the payload transmitted. The Nucleus master application loads Nucleus based firmware first, executes the echo test, performs shutdown, and then boots baremetal based firmware next to execute the same test, and performs shutdown.

Echo test between Nucleus master<-> Nucleus remote based firmware

- 1.) The Nucleus master application boots the Nucleus based remote firmware using remoteproc.
- 2.) The Nucleus master application transmits echo payloads to the remote firmware using rpmsg.
- 3.) The Nucleus firmware echoes back the payload received to master using rpmsg.
- 4.) The Nucleus master application verifies integrity of payload received and prints the results.

Echo test between Nucleus master<-> baremetal remote based firmware

- 5.) The Nucleus master application boots baremetal based remote firmware using remoteproc.
- 6.) The Nucleus master application transmits echo payloads to remote firmware using rpmsg.
- 7.) The baremetal firmware echoes back the payload received to master using rpmsg.

8.) The Nucleus master application verifies integrity of payload received and prints results.

Application	Location
Nucleus master	<i>apps/tests/master/nucleus/echo_test/nucleus_nucleusbm</i>
Nucleus remote	<i>apps/tests/remote/nucleus/echo_test</i>
Baremetal remote	<i>apps/tests/remote/baremetal/echo_test</i>

Function Test Suite Application

Description

This application performs functional tests on various features and capabilities supported by the OpenAMP Framework. The following capabilities are tested:

- rpmsg send, and send_off channel
- remote channel deletion
- endpoint creation and deletion
- Multiple remote firmware life cycles from a single master run-time instance

The Nucleus master application loads Nucleus based firmware first, executes functional tests, performs shutdown, and then boots baremetal based firmware next to execute the same test, and performs shutdown.

Functional tests between Nucleus master<-> Nucleus remote based firmware

- 1.) The Nucleus application boots remote Nucleus based functional test firmware using remoteproc.
- 2.) The Nucleus master based application conducts functional tests with remote firmware and prints the results.

Functional tests between Nucleus master<-> baremetal remote based firmware

- 3.) The Nucleus master application boots remote baremetal based functional test firmware using remoteproc.
- 4.) The Nucleus master application conducts functional tests with remote firmware and prints the results

Application	Location
Nucleus master	<i>apps/tests/master/nucleus/func_test_suite/nucleus_nucleusbm</i>
Nucleus remote	<i>apps/tests/remote/nucleus/func_test_suite</i>
Baremetal remote	<i>apps/tests/remote/baremetal/func_test_suite</i>

Related Topics

[Linux Master/Nucleus Remote, and Linux Master/Baremetal Remote Configurations](#)

[Nucleus Master/Linux Remote and Baremetal Master /Linux Remote Configurations](#)

Nucleus Master/Linux Remote and Baremetal Master /Linux Remote Configurations

This demonstration showcases the Life Cycle Management (LCM) of the remote processor and its software context and the Inter Processor Communication (IPC) with remote software context.

This section is for the Nucleus/baremetal Master/Linux remote configurations.

Matrix Multiplication Demonstration

Description

Nucleus/baremetal based master application uses remoteproc to boot and shut down Linux on a remote processor. Once Linux is booted on the remote processor, the matrix multiplication demo can be executed that showcases IPC between remote and master.

To demonstrate LCM and IPC between Nucleus/baremetal master<->Linux remote based firmware

1. The Nucleus/baremetal based master application boots Linux on remote processor using remoteproc.
2. The Linux application on remote processor acts as rpmsg master and transmits random matrices to Nucleus/baremetal using rpmsg.
3. Nucleus/baremetal applications on master core performs multiplications and transmits results back to Linux using rpmsg.
4. Linux application on remote core prints the results of matrix multiplication.
5. User selects quit from Linux application, Linux sends a shutdown message to Nucleus/baremetal master.
6. Linux executes a system halt call to gracefully shutdown itself.
7. Nucleus/baremetal on master core shutdown the remote core

Application	Location
Nucleus master	<i>apps/samples/master/nucleus/matrix_multiply/nucleus_linux</i>
Baremetal master	<i>apps/samples/master/baremetal/matrix_multiply</i>
Linux rpmsg master (kernel space)	<i>apps/samples/master/linux/kernelspace/rpmsg_mat_mul_kern_app</i>

Application	Location
Linux rpmsg master (user space)	<i>apps/samples/master/linux/userspace/matrix_multiply</i>

Echo Test Application

Description

This application tests the data integrity of the payload transmitted. The Nucleus/baremetal master application loads Linux based firmware first, executes the echo test and performs shutdown.

Echo test between Nucleus/baremetal master<->Linux remote based firmware

1. The Nucleus/baremetal master based application boots Linux remote based firmware using remoteproc.
2. Linux remote firmware acts as rpmsg master and transmits echo payloads to Nucleus/baremetal using rpmsg.
3. Nucleus/baremetal echoes back the payload received to Linux using rpmsg.
4. Linux based application verifies integrity of payload received and prints the results.
5. User selects quit from the Linux application, Linux sends a shutdown message to remoteproc Nucleus/baremetal master.
6. Linux executes a system halt call to gracefully shutdown itself.
7. Nucleus/baremetal on master core shutdown the remote core.

Application	Location
Nucleus master	<i>apps/tests/master/nucleus/echo_test/nucleus_linux</i>
Baremetal master	<i>apps/tests/master/baremetal/echo_test</i>
Linux rpmsg master (kernel space)	<i>apps/tests/master/linux/kernelspace/rpmsg_echo_test_kern_app</i>
Linux rpmsg master (user space)	<i>apps/tests/master/linux/userspace/echo_test</i>

Function Test Suite Application

Description

This application performs functional tests on various features and capabilities supported by the OpenAMP Framework. The following capabilities are tested:

- rpmsg send, and send_off channel

- remote channel deletion
- endpoint creation and deletion
- multiple remote firmware life cycles from a single master run-time instance

The Nucleus/baremetal master application loads Linux based firmware first, executes functional tests and performs shutdown.

Functional tests between Nucleus/baremetal master<-> Linux remote based firmware

1. The Nucleus/baremetal master based application boots Linux remote based functional test firmware using remoteproc.
2. The Nucleus/baremetal master based application conducts functional tests with Linux firmware and sends results to Linux which prints the results on console

Application	Location
Nucleus master	<i>apps/tests/master/nucleus/func_test_suite/nucleus_linux</i>
Baremetal master	<i>apps/tests/master/baremetal/func_test_suite</i>
Linux rpmsg master (kernel space)	<i>apps/tests/master/linux/kernel-space/rpmsg_func_test_kern_app</i>

Related Topics

[Linux Master/Nucleus Remote, and Linux Master/Baremetal Remote Configurations](#)

[Nucleus Master/Nucleus Remote, and Nucleus Master/Baremetal Remote Configurations](#)

Chapter 3

Environment Setup, Build, and Execution of Applications and Test Binaries

This chapter contains instructions for environment setup, building and executing applications and test binaries.

OpenAMP Framework Environment Setup

Here you will find a description of the toolset, build environment, installation, and running the setup script.

Toolset

The SW configurations were tested with the tool-chains listed in [Table 3-1](#):

Table 3-1. OpenAMP Framework Environment

Supported Tool chains	Notes
Mentor Graphics CodeSourcery Lite tools Target: arm-none-eabi, gcc version 4.8.1 (Sourcery CodeBench Lite 2013.11-24)	A free version of Sourcery CodeBench tools can be obtained from http://www.mentor.com/embedded-software/sourcery-tools/sourcery-codebench/editions/lite-edition/
Xilinx GCC Tools Target: arm_xilinx-eabi, gcc version 4.7.3 (Sourcery CodeBench Lite 2013.05-39)	

OpenAMP Framework Environment

Assume that the OpenAMP Framework package is installed at location `<open_amp_root>`.

To setup the environment for Mentor Graphics CodeSourcery GNU Lite tools - Target: arm-none-eabi, gcc version 4.8.1 (Sourcery CodeBench Lite 2013.11-24).

```
> $ PATH=<Mentor Graphics Sourcery CodeBench Lite for ARM EABI  
Installation Path>/bin:${PATH}  
> cd <open_amp_root>
```

Modify the CROSS parameter in *Makefile.common*s to look like this

```
CROSS := arm-none-eabi-
```

To setup the environment for Xilinx GCC tools - Xilinx GCC tools - Target: arm-xilinx-eabi, gcc version 4.7.3 (Sourcery CodeBench Lite 2013.05-39)

```
> $ PATH="<Xilinx ARM EABI Installation Path>/bin:${PATH}"  
> cd <open_amp_root>
```

Modify CROSS parameter *Makefile.common*s to look like this

```
CROSS := arm-xilinx-eabi-
```

Environment variable

- Change directory to *<open_amp_root>*

```
$ export OPENAMP=$PWD
```

Note



The scope of the variable \$OPENAMP is local to the current shell. You must define it whenever you start working with a new shell terminal.

Related Topics

[Setting up the PetaLinux Environment](#)

Setting up the PetaLinux Environment

In order to become familiar with PetaLinux workflows and the environment setup guidelines, you are advised to go through the following documents:

[PetaLinux SDK User Guide: Installation Guide \(ver 2013.10\)](#)

[PetaLinux SDK User Guide: Zynq AM Linux FreeRTOS \(ver 2013.10\)](#)

[PetaLinux SDK User Guide: Application Development Guide \(ver 2013.10\)](#)

[PetaLinux SDK User Guide: Getting Started Guide \(ver 2013.10\)](#)


The following section walks you through the steps required to setup a PetaLinux environment.

Prerequisites

The environment must satisfy the following requirements as per PetaLinux SDK User Guide, Installation Guide:

- Minimum workstation requirements:
 - 2GB RAM (recommended minimum for Xilinx tools)
 - Pentium 4 2GHz CPU clock or equivalent

- 5 GB free HDD space
- Supported OS:
 - RHEL 5 (32-bit or 64-bit)
 - RHEL 6 (32-bit or 64-bit)
 - SUSE Enterprise 11 (32-bit or 64-bit)
- PetaLinux release package downloaded.
- Valid PetaLinux license.

Note  You can also setup a PetaLinux environment on Ubuntu 12.04 LTS using a bash shell.

The following documentation also assumes that you are familiar with basic Linux environment, commands and shell routines.

For Ubuntu 12.04LTS, please make sure that the following tools and libraries have been installed:

Table 3-2. Ubuntu Tools and Libraries

Tools/Library	ATP Package for Ubuntu
dos2unix	tofrodos
ip	iproute
gawk	gawk
gcc	gcc
git	git-core
gpg	gnupg
make	Make
netstat	net-tools
ncurses	ncurses-dev
tftp server	tftpd
zlib	zlibg-dev
flex	flex
bison	bison

Procedure

1. Please refer to the to

[PetaLinux SDK User Guide: Installation Guide \(ver 2013.10\)](#)

for PetaLinux and license installation.

2. Invoke the Petalinux installer with the following command:

```
$ ./petalinux-v2013.10-final-installer.run <installation root>
```

A successful execution installs the PetaLinux in a directory named *petalinux-v2013.10-final* within the current working directory.

3. Run Setup Script

You must source the appropriate setup script.

- For bash shell:

```
$ source <path-to-installed-PetaLinux>/settings.sh
```

- For C shell:

```
source <path-to-installed-PetaLinux>/settings.csh
```

Results

Executing these commands displays an output similar to the following:

```
PetaLinux environment set to '<installation path>/petalinux-v2013.10-final'  
INFO: Finalising PetaLinux installation  
INFO: Checking free disk space  
INFO: Checking installed tools  
INFO: Checking installed development libraries  
INFO: Checking network and other service
```

Verify that the PetaLinux environment has been correctly setup:

```
$ echo $PETALINUX  
<installation path>/petalinux-v2013.10-final
```

Note



You are required to run a setup script each time a shell terminal is started for PetaLinux build.

Note



In certain cases, the shell could be linked to bin/dash instead of bin/bash as default and will result in error messages.

To resolve this issue, link bin/sh to bin/bash:

```
$ sudo rm /bin/sh
```

```
$ sudo ln -s /bin/bash /bin/sh
```

Related Topics

[OpenAMP Framework Environment Setup](#)

Setting up the Hardware

This section details the hardware setup to execute an OpenAMP Framework application. Execute these steps and repeat the necessary steps (copying images to SD card) in order to execute the applications as described in the following sections.

Procedure

1. Power off the board.
2. Connect the serial port on the board (USB UART Connector J17) to your workstation using a USB mini-B cable.
3. Set the switches on the board to boot from the SD card:

SW16.1	SW16.2	SW16.3	SW16.4	SW16.5
0	0	1	1	0
4. Copy the U-Boot and application binaries (as described in the following sections) on an FAT32 formatted SD card.
5. Place the SD card into the J64 slot.
6. Power up the board.
7. Open a serial terminal on your workstation with the following settings:
 baud rate = 115,200
 data bits = 8
 stop bits = 1
 flow control = none
8. Wait for the U-Boot console to appear on the serial terminal.

Note



It is recommended to format the SD card with a single FAT32 partition 0.

Related Topics

[OpenAMP Framework Environment Setup](#)

Build and Execute Applications for Linux Master/Nucleus Remote and Linux Master/Baremetal Remote Configurations

In order to build and run applications in this configuration, you need to first build remote Nucleus and remote baremetal applications and then install these firmware images within the Linux master kernel root file system.

Building the OpenAMP Framework Library, Applications, and Tests	24
Building Linux Master Applications and Tests	25
Executing Applications and Tests for Linux Master/ Nucleus Remote and Linux Master/ Baremetal Remote Configurations	28
Executing the Matrix Multiply Application	28
Executing the Echo Test Application	32
Executing the Proxy Application	35

Building the OpenAMP Framework Library, Applications, and Tests

Follow these steps to build the OpenAMP Framework library, applications and/or tests.

Prerequisites

- OpenAMP Framework library installed at location \$OPENAMP

Procedure

1. Change directory to \$OPENAMP

```
$ cd $OPENAMP
```

2. Execute the build script. This operation will build remote applications for Nucleus and Baremetal environments.

Note: The build operation will generate application outputs in the respective application directories.

```
$ source open_amp_build.sh
```

3. If you need to clean all previous builds, you can execute the script with the following option:


```
$ source open_amp_build.sh - c
```

Related Topics

[Building Linux Master Applications and Tests](#) [Executing Applications and Tests for Linux Master/ Nucleus Remote and Linux Master/Baremetal Remote Configurations](#)

Building Linux Master Applications and Tests

The following steps are required to build PetaLinux and Linux applications for master Linux kernel.

Procedure

1. Create a project for the Linux Master:
 - a. Change the directory to the location where the Linux Master project needs to be created and execute the following command:
2. Navigate to *<master-root>* and enable the following configuration options in kernel.
 - a. A user needs to setup PetaLinux so that the Linux kernel starts at address 0x10000000.

```
>petalinux-create -t project -n <master_name> --template zynq
```

<master_name> is the user defined name of the Linux Master project. The location of project will be referred as *<master-root>* from this point onwards.

```
$ cd <master-root>  
$ petalinux-config
```

- i. Change the Kernel base address value to 0x10000000 if it is not already set to this value.

```
*** linux Components Selection ***  
...  
(0x10000000) Kernel base address  
PetaLinux Kernel Configuration
```

- ii. Save the configuration changes.
- b. Execute the following command while in the *<master-root>* directory

```
$ petalinux-config -c kernel
```

- i. Select Enable loadable module support.

```
Kernel Configuration --->  
[*] Enable loadable module support --->
```

- ii. Select High Memory Support within Kernel Features
- iii. Select 2G/2G user/kernel split as Memory split within Kernel Features

```
Kernel Configuration --->
  Kernel Features --->
    Memory split (2G/2G user/kernel split) --->
      [*] High Memory Support
```

- iv. Enable Userspace firmware loading support

```
Kernel Configuration --->
  Device Drivers --->
    Generic Driver Options --->
      <*> Userspace firmware loading support
      [ ] Include in-kernel firmware blobs in kernel binary
      ( ) External firmware blobs to build into the kernel
        binary
```

- v. Enable Remoteproc Driver

```
Kernel Configuration --->
  Device Drivers --->
    Remoteproc drivers (EXPERIMENTAL) --->
      <M> Support ZYNQ remote proc
```

Note



Enable the driver as a Module <M>

3. Setup Device TreeSource (DTS)

- a. Open DTS file with a text editor:

```
<master_root>/subsystems/linux/hw-description/system.dts
```

- b. Add the following device node for the Zynq remoteproc driver so that the driver can be probed.

```
test: remoteproc-test@0 {
compatible = "xlnx,zynq_remoteproc";
reg = < 0x0 0x10000000 >;
interrupt-parent = <&ps7_scugic_0>;
interrupts = < 0 37 4 0 38 4 >;
firmware = "firmware";
ipino = <6>;
vring0 = <2>;
vring1 = <3>;
} ;
```

4. Execute the automation scripts provided within \$OPENAMP to create the necessary application projects, kernel driver projects, and firmware projects for master Linux project. The script creates projects, moves sources and make files from \$OPENAMP as needed, enables the project to be included/enabled in the PetaLinux initramfs.

```
>cd <master_root>
```

Note



OpenAMP Framework library and applications should be created using instructions under “[Building the OpenAMP Framework Library, Applications, and Tests](#)” on page 24 before proceeding beyond this point.

```
>source $OPENAMP/libs/system/zc702evk/linux/scripts/  
open_amp_create_projects.sh master
```

where `<master_root>` is the directory location of the master Linux project.

It will generate the following user-space application and kernel modules:

1. `<master_root>/components/apps/mat_mul_demo`
2. `<master_root>/components/apps/echo_test`
3. `<master_root>/components/apps/proxy_app`
4. `<master_root>/components/modules/zynq_remoteproc_driver`
5. `<master_root>/components/modules/rpmsg_mat_mul_kern_app`
6. `<master_root>/components/modules/rpmsg_echo_test_kern_app`
7. `<master_root>/components/modules/rpmsg_user_dev_driver`
8. `<master_root>/components/modules/rpmsg_proxy_dev_driver`
9. `<master_root>/components/apps/mat_mul_nucleus_fw`
10. `<master_root>/components/apps/echo_test_nucleus_fw`
11. `<master_root>/components/apps/rpc_demo_nucleus_fw`
12. `<master_root>/components/apps/mat_mul_baremetal_fw`
13. `<master_root>/components/apps/echo_test_baremetal_fw`
14. `<master_root>/components/apps/rpc_demo_baremetal_fw`

5. Build PetaLinux

- a. Change into the `<master_root>`

```
$ cd <master_root>
```

- b. Execute the following command:

```
$ petalinux-build
```

Results

A successful build will generate a Linux FIT image (*image.ub*) at:

```
<master_root>/images/linux
```

Related Topics

[Building the OpenAMP Framework Library, Applications, and Tests](#)

[Executing Applications and Tests for Linux Master/ Nucleus Remote and Linux Master/ Baremetal Remote Configurations](#)

Executing Applications and Tests for Linux Master/ Nucleus Remote and Linux Master/ Baremetal Remote Configurations

This is a procedure for moving the U-Boot, application and test binaries to boot medium (SD Card).

Procedure

1. Place the U-Boot image for ZC702EVK(*<master_root>/pre-built/linux/images/BOOT.bin*) and PetaLinux FIT image (*<master_root>/images/linux/image.ub*) in an SDCARD formatted for FAT.
2. Insert the SDCARD in the SDIO slot on ZC702EVK.
3. Power up board to boot U-Boot
4. In the U-Boot console:
 - `fatload mmc <dev>:<part> 0x1000000 image.ub`
 - `bootm 0x1000000`
5. PetaLinux boots up
6. Username:root, Password:root

NOTE:

`<dev>` refers to the relevant MMC device number. The list of available devices can be retrieved from the U-boot prompt by entering - U-Boot-PetaLinux> `mmc list`

`<part>` refers to the relevant partition on the current mmc device. It can be retrieved from the U-Boot prompt by entering - U-Boot-PetaLinux> `mmc part`

Results

Once PetaLinux is up and you have performed the root login, you are ready to load and execute one of the firmwares present in the Linux root FS.

Related Topics

[Building the OpenAMP Framework Library, Applications, and Tests](#), [Building Linux Master Applications and Tests](#)

Executing the Matrix Multiply Application

This section explains how one can load and execute samples and test applications built into the Linux FS.

On completion of execution of a given application or test, follow the instructions provided to shutdown and unload the relevant kernel drivers before executing the next application.

Procedure

1. Execute the Kernel Space Matrix Multiplication sample application.
 - a. Load Zynq platform remoteproc module

- i. For Nucleus based remote matrix multiply firmware:

```
modprobe zynq_remoteproc_driver firmware=  
"/lib/firmware/zc702evk/nucleus/matrix_multiply/firmware"
```

- ii. For Bare metal based remote matrix multiply firmware:

```
modprobe zynq_remoteproc_driver firmware=  
"/lib/firmware/zc702evk/baremetal/matrix_multiply/firmware"
```

- b. Load `rpmsg_mat_mul_kern_app` module to execute the application

```
root@Xilinx-ZC702-2013_3:~# modprobe rpmsg_mat_mul_kern_app
```

The matrix multiply kernel-space application output should look similar to:

```
Demo Start - Demo rpmsg driver got probed  
since the rpmsg device associated with driver was found !  
  
Create endpoint and register rx callback  
  
Master : Linux : Generating random matrices  
  
Master : Linux : Input matrix 0  
  
1 0 0 5 4 8  
0 0 4 7 4 3  
5 4 7 2 5 4  
1 9 4 5 4 9  
2 8 3 7 5 5  
6 3 6 2 5 3  
  
Master : Linux : Input matrix 1  
  
6 8 8 4 9 1  
7 6 0 8 2 2  
7 5 4 4 5 3  
2 9 6 8 8 9  
9 2 8 9 7 1  
2 6 0 3 1 2  
  
Master : Linux : Sent 296 bytes of data over rpmsg channel to  
remote  
  
Master : Linux : Received 148 bytes of data over rpmsg channel from  
remote  
  
Master : Linux : Printing results
```

```
root@Xilinx-ZC702-2013_3:~# 68 109 70 104 85 66
84 109 90 117 107 85
164 151 120 153 143 65
161 189 86 195 124 98
158 182 110 200 145 105
154 142 124 142 144 59
virtio_rpmsg_bus virtio0: destroying channel rpmsg-openamp-demo-
channel addr 0x1
```

You can now shutdown the firmware using the following set of commands:

```
root@Xilinx-ZC702-2013_3:~# modprobe -r rpmsg_mat_mul_kern_app
root@Xilinx-ZC702-2013_3:~# modprobe -r virtio_rpmsg_bus
root@Xilinx-ZC702-2013_3:~# modprobe -r zynq_remoteproc_driver
```

2. Execute the User Space Matrix Multiplication Application.

a. Load Zynq platform remoteproc module

i. For Nucleus based remote matrix multiply firmware:

```
modprobe zynq_remoteproc_driver firmware=
"/lib/firmware/zc702evk/nucleus/matrix_multiply/firmware"
```

ii. For Bare metal based remote matrix multiply firmware:

```
modprobe zynq_remoteproc_driver firmware=
"/lib/firmware/zc702evk/baremetal/matrix_multiply/firmware"
```

b. Load the rpmsg user device driver using the following command:

```
root@Xilinx-ZC702-2013_3:~# modprobe rpmsg_user_dev_driver
```

c. Start matrix_multiply user-space application

```
root@Xilinx-ZC702-2013_3:~# mat_mul_demo
```

The matrix multiply user-space application output should look similar to:

```
Matrix multiplication demo start

Open rpmsg dev!

Query internal info ..
rpmsg kernel fifo size = 2048
rpmsg kernel fifo free space = 2048

Creating ui_thread and compute_thread ...

*****
Please enter command and press enter key
*****
1 - Generates random 6x6 matrices and transmits them to remote core
over rpmsg ..
2 - Quit this application ..
CMD>1
```

```
Compute thread unblocked ..  
The compute thread is now blocking on a read() from rpmsg device
```

```
Generating random matrices now ...
```

```
Master : Linux : Input matrix 0
```

```
2 7 7 5 3 0  
7 1 9 4 5 3  
4 6 5 8 5 7  
4 1 8 0 1 9  
8 6 0 6 4 7  
5 9 4 2 4 0
```

```
Master : Linux : Input matrix 1
```

```
4 1 1 3 7 6  
8 3 4 6 3 0  
3 8 1 4 8 5  
3 8 1 3 6 8  
2 1 7 7 5 3  
7 9 6 0 4 4
```

```
Writing generated matrices to rpmsg device, 296 bytes written ..
```

```
Received results! - 148 bytes from rpmsg device (transmitted from  
remote context)
```

```
Master : Linux : Printing results
```

```
106 122 63 112 136 96  
106 146 77 110 185 146  
162 194 118 127 187 156  
113 153 77 57 136 103  
155 141 108 106 158 136  
118 84 75 119 126 78
```

```
*****
```

```
Please enter command and press enter key
```

```
*****
```

```
1 - Generates random 6x6 matrices and transmits them to remote core  
over rpmsg ..
```

```
2 - Quit this application ..
```

```
CMD>
```

3. You can shutdown the firmware using the following set of commands:

```
root@Xilinx-ZC702-2013_3:~# modprobe -r rpmsg_user_dev_driver  
root@Xilinx-ZC702-2013_3:~# modprobe -r virtio_rpmsg_bus  
root@Xilinx-ZC702-2013_3:~# modprobe -r zynq_remoteproc_driver
```

Related Topics

[Executing Applications and Tests for Linux Master/ Nucleus Remote and Linux Master/ Baremetal Remote Configurations](#)

[Executing the Echo Test Application](#)

Executing the Echo Test Application

This procedure outlines the steps to execute the Linux user space or kernel space applications to perform echo test on one of the firmwares at a time. In order to execute the test on another firmware shutdown the first firmware and repeat the steps.

Procedure

1. Execute the Kernel Space Echo Application.

- a. Load Zynq platform remoteproc module

- i. For Nucleus based remote echo firmware:

```
modprobe zynq_remoteproc_driver firmware=  
"/lib/firmware/zc702evk/nucleus/echo_test/firmware"
```

- ii. For Bare metal based remote echo firmware:

```
modprobe zynq_remoteproc_driver firmware=  
"/lib/firmware/zc702evk/baremetal/echo_test/firmware"
```

- b. Load rpmsg_echo_test_kern_app module to execute the application

```
root@Xilinx-ZC702-2013_3:~# modprobe virtio_rpmsg_bus  
root@Xilinx-ZC702-2013_3:~# modprobe rpmsg_echo_test_kern_app
```

The echo test kernel-space application output should look similar to:

```
root@Xilinx-ZC702-AMP-2013_3:/# modprobe rpmsg_echo_test_kern_app  
remoteproc0: powering up 0.remoteproc-test  
remoteproc0: Booting fw image firmware, size 230628  
remoteproc0: remote processor 0.remoteproc-test is now up  
virtio_rpmsg_bus virtio0: rpmsg host is online  
virtio_rpmsg_bus virtio0: creating channel rpmsg-openamp-demo-  
channel addr 0x1  
Echo Test Start!  
Master : Linux Kernal Space : Sending payload num 0 of size 9  
Master : Linux Kernal Space : Received payload num 0 of size 9  
Master : Linux Kernal Space : Sending payload num 1 of size 10  
Master : Linux Kernal Space : Received payload num 1 of size 10  
Master : Linux Kernal Space : Sending payload num 2 of size 11  
Master : Linux Kernal Space : Received payload num 2 of size 11  
Master : Linux Kernal Space : Sending payload num 3 of size 12  
Master : Linux Kernal Space : Received payload num 3 of size 12  
Master : Linux Kernal Space : Sending payload num 4 of size 13  
Master : Linux Kernal Space : Received payload num 4 of size 13  
Master : Linux Kernal Space : Sending payload num 5 of size 14  
Master : Linux Kernal Space : Received payload num 5 of size 14  
  
...  
...  
  
Master : Linux Kernal Space : Received payload num 483 of size 492  
Master : Linux Kernal Space : Sending payload num 484 of size 493  
Master : Linux Kernal Space : Received payload num 484 of size 493
```



```
Master : Linux Kernel Space : Sending payload num 485 of size 494
Master : Linux Kernel Space : Received payload num 485 of size 494
Master : Linux Kernel Space : Sending payload num 486 of size 495
Master : Linux Kernel Space : Received payload num 486 of size 495
Master : Linux Kernel Space : Sending payload num 487 of size 496
Master : Linux Kernel Space : Received payload num 487 of size 496
*****
Echo Test Results: Error count = 0
*****
root@Xilinx-ZC702-AMP-14_7:~# virtio_rpmsg_bus virtio0: destroying
channel rpmsg-openamp-demo-channel addr 0x1
```

You can shutdown the firmware using the following set of commands:

```
root@Xilinx-ZC702-2013_3:~# modprobe -r rpmsg_echo_test_kern_app
root@Xilinx-ZC702-2013_3:~# modprobe -r virtio_rpmsg_bus
root@Xilinx-ZC702-2013_3:~# modprobe -r zynq_remoteproc_driver
```

2. Execute the User Space Echo Application.

a. Load Zynq platform remoteproc module

i. For Nucleus based remote echo firmware:

```
modprobe zynq_remoteproc_driver firmware=
"/lib/firmware/zc702evk/nucleus/echo_test/firmware"
```

ii. For Bare metal based remote echo firmware:

```
modprobe zynq_remoteproc_driver firmware=
"/lib/firmware/zc702evk/baremetal/echo_test/firmware"
```

b. Load the rpmsg user device driver using the following command:

```
root@Xilinx-ZC702-2013_3:~# modprobe rpmsg_user_dev_driver
```

c. Start echo_test user-space application

```
root@Xilinx-ZC702-2013_3:~# echo_test
```

The echo test user-space application output should look similar to:

```
root@Xilinx-ZC702-AMP-2013_3:~# modprobe rpmsg_user_dev_driver
rpmsg_user_dev_driver rpmsg0: new channel: 0x400 -> 0x1!
root@Xilinx-ZC702-AMP-2013_3:~# echo_test
Echo test start
Open rpmsg dev!
Query internal info ..
rpmsg kernel fifo size = 2048
rpmsg kernel fifo free space = 2048
*****
Please enter command and press enter key
*****
1 - Send data to remote core, retrieve the echo and validate its
integrity ..
2 - Quit this application ..
CMD>1
sending payload number 0 of size 9
```

```
received payload number 0 of size 9
sending payload number 2 of size 10
received payload number 2 of size 10
sending payload number 3 of size 11
received payload number 3 of size 11
sending payload number 4 of size 12
received payload number 4 of size 12
sending payload number 5 of size 13
received payload number 5 of size 13
sending payload number 6 of size 14
received payload number 6 of size 14
...
...
...

sending payload number 478 of size 486
received payload number 478 of size 486
sending payload number 479 of size 487
received payload number 479 of size 487
sending payload number 480 of size 488
received payload number 480 of size 488
sending payload number 481 of size 489
received payload number 481 of size 489
sending payload number 482 of size 490
received payload number 482 of size 490
sending payload number 483 of size 491
received payload number 483 of size 491
sending payload number 484 of size 492
received payload number 484 of size 492
sending payload number 485 of size 493
received payload number 485 of size 493
sending payload number 486 of size 494
received payload number 486 of size 494
sending payload number 487 of size 495
received payload number 487 of size 495
*****
Test Results: Error count = 0
*****
*****
Please enter command and press enter key
*****
1 - Send data to remote core, retrieve the echo and validate its
integrity ..
2 - Quit this application ..
CMD>
```

3. You can shutdown the firmware by using the following set of commands:

```
root@Xilinx-ZC702-2013_3:~# modprobe -r rpmsg_user_dev_driver
root@Xilinx-ZC702-2013_3:~# modprobe -r virtio_rpmsg_bus
root@Xilinx-ZC702-2013_3:~# modprobe -r zynq_remoteproc_driver
```

Related Topics

[Executing Applications and Tests for Linux Master/ Nucleus Remote and Linux Master/ Baremetal Remote Configurations](#)

[Executing the Proxy Application](#)

Executing the Proxy Application

This procedure describes the usage of the Linux user space proxy application, which provides an interface to the OpenAMP Framework proxy infrastructure.

Procedure

1. The proxy application help can be accessed as shown below:

- to display proxy application help, execute:

```
proxy_app -h
```

Output:

```
Linux proxy application.  
-r      Displays proxy application version.  
-f      Accepts path of firmware to load on remote core.  
-h      Displays this help message.
```

2. In order to bring up RPC demo firmware on the remote core, execute:

- for Nucleus Remote:

```
proxy_app -f /lib/firmware/zc702evk/nucleus/rpc_demo/firmware
```

- for baremetal remote:

```
proxy_app -f /lib/firmware/zc702evk/baremetal/rpc_demo/firmware
```

Note



Once you have selected 'no' for repeat demo option and application quits, you are required to enter CTRL+C to quit the proxy application.

The RPC demo output should look similar to:

```
root@Xilinx-ZC702-AMP-14_7:~# proxy_app -f  
lib/firmware/zc702evk/nucleus/rpc_demo/firmware  
  
Master>Loading remote firmware  
cp: can't stat 'lib/firmware/zc702evk/nucleus/rpc_demo/firmware': No such  
file or directory  
CPU1: shutdown  
remoteproc0: 0.remoteproc-test is available  
remoteproc0: Note: remoteproc is still under development and considered  
experimental.
```

```
remoteproc0: THE BINARY FORMAT IS NOT YET FINALIZED, and backward
compatibility isn't yet guaranteed.
```

```
Master>Create rpmsg proxy device
```

```
Master>Opening rpmsg proxy device
```

```
remoteproc0: powering up 0.remoteproc-test
remoteproc0: Booting fw image firmware, size 263444
remoteproc0: remote processor 0.remoteproc-test is now up
virtio_rpmsg_bus virtio0: rpmsg host is online
remoteproc0: registered virtio0 (type 7)
virtio_rpmsg_bus virtio0: creating channel rpmsg-openamp-demo-channel addr
0x1
rpmsg_proxy_dev_driver rpmsg0: new channel: 0x400 -> 0x1!
```

```
Master>RPC service started !!
```

```
Remote>*****
```

```
Remote><Nucleus RTOS or Bare metal comes here> Remote Procedure Call (RPC)
Demonstration
```

```
Remote>*****
```

```
Remote>Rpmsg based retargetting to proxy initialized..
```

```
Remote>FileIO demo using open, write, read, close CRTL calls ..
```

```
Remote>Creating a file on master and writing to it..
```

```
Remote>Opened file 'remote.file' with fd = 4
```

```
Remote>Wrote to fd = 4, size = 45 bytes
content = This is a test string being written to file..
```

```
Remote>Closed fd = 4
```

```
Remote>Reading a file on master and displaying its contents..
```

```
Remote>Opened file 'remote.file' with fd = 4
```

```
Remote>Read from fd = 4, size = 45 bytes
content = This is a test string being written to file..
```

```
Remote>Closed fd = 4
```

```
Remote>Remote firmware using scanf and printf ..
```

```
Remote>Scanning user input from master..
```

```
Remote>Enter name
bob
```

```
Remote>Enter age
33
```

```
Remote>Enter value for pi
3.14
```

```
Remote>User name = 'bob'

Remote>User age = '33'

Remote>User entered value of pi = '3.140000'

Remote>Repeat demo ? (enter yes or no)
yes

Remote>Remote firmware using scanf and printf ..

Remote>Scanning user input from master..

Remote>Enter name
zack

Remote>Enter age
33

Remote>Enter value for pi
3.14

Remote>User name = 'zack'

Remote>User age = '33'

Remote>User entered value of pi = '3.140000'

Remote>Repeat demo ? (enter yes or no)
no

Remote>RPC retargetting quitting ...

Remote> Firmware's rpmsg-openamp-demovirtio_rpmsg_bus virtio0: destroying
channel rpmsg-openamp-demo-channel addr 0x1
-channel going down!

Master>RPC service exiting !!
remoteproc0: stopped remote processor 0.remoteproc-test
zynq_remoteproc 0.remoteproc-test: zynq_remoteproc_remove
zynq_remoteproc 0.remoteproc-test: Deleting the irq_list
remoteproc0: releasing 0.remoteproc-test
CPU1: Booted secondary processor
```

3. Verify the file created by remote context.

```
root@Xilinx-ZC702-AMP-14_7:~# cat remote.file
```

This is a test string being written to file.

Related Topics

[Executing Applications and Tests for Linux Master/ Nucleus Remote and Linux Master/ Baremetal Remote Configurations](#)

[Executing the Matrix Multiply Application](#)

Build and Execute Applications for Nucleus Master/Nucleus Remote and Nucleus Master/Baremetal Remote Configurations

The following procedures generate remote firmware images for both Nucleus and baremetal remote environments. They also generate Nucleus master binaries which can load these firmware images on the remote core.

Building the OpenAMP Framework Library, Applications, and Tests	38
Executing Applications and Tests for Nucleus Master/Nucleus Remote and Nucleus Master/Baremetal Remote Configurations.	39
Executing the Matrix Multiply Application	40
Executing the Echo Test Application	42
Executing the Functional Test Suite	44

Building the OpenAMP Framework Library, Applications, and Tests

Follow these steps to build the OpenAMP Framework library, applications and/or tests.

Prerequisites

- The OpenAMP Framework library is installed at location \$OPENAMP

Procedure

1. Change directory to \$OPENAMP

```
$ cd $OPENAMP
```

2. Execute the build script. This operation will build remote applications for Nucleus and Baremetal environments as well as master applications for Nucleus.

Note: The build operation will generate application outputs in the respective application directories.

```
$ source open_amp_build.sh
```

3. In order to clean all previous builds, you can execute the script with the following option:

```
$ source open_amp_build.sh -c
```

Related Topics

[Build and Execute Applications for Nucleus Master/Nucleus Remote and Nucleus Master/Baremetal Remote Configurations](#)

[Executing Applications and Tests for Nucleus Master/Nucleus Remote and Nucleus Master /Baremetal Remote Configurations](#)

Executing Applications and Tests for Nucleus Master/Nucleus Remote and Nucleus Master /Baremetal Remote Configurations

The following is a procedure for moving U-Boot, application and test binaries to a boot medium (SD Card).

Prerequisites

- The U-Boot binary image (*BOOT.bin*), and OpenAMP Framework application and test binaries are built and available.

Procedure

1. Format a SDCARD and partition for FAT.
2. Insert the SDCARD in the host machine and place the generated U-Boot image (*BOOT.bin*) for ZC702EVK, OpenAMP Framework binary images available at locations listed below should be moved to SDCARD as well.
 - *apps/samples/master/nucleus/matrix_multiply/nucleus_nucleusbm/matrix_multiply.bin*
 - *apps/tests/master/nucleus/echo_test/nucleus_nucleusbm/echo_test.bin*
 - *apps/tests/master/nucleus/func_test_suite/nucleus_nucleusbm/func_test_suite.bin*
3. Insert the SDCARD in the SDIO slot J64 on ZC702EVK
4. Power up board to boot U-Boot
5. Once U-Boot is up, In the U-Boot console, enter

```
U-Boot-PetaLinux> dcache off;fatload mmc  
<dev>:<part> 0x10000000 <nucleus app>.bin; go 0x10000000
```

Note



<dev> - refers to the relevant MMC device number. The list of available devices can be retrieved from u-boot prompt by entering -U-Boot-PetaLinux>mmc list.

<part> - refers to the relevant partition on the current mmc device. It can be retrieved from the U-Boot prompt by entering -U-Boot-PetaLinux>mmc part.

Note



The following sections assume that the relevant <dev>:<part> value is 0:0. The actual values may vary and the user is advised to confirm the device and partition numbers first.

Related Topics

[Executing the Matrix Multiply Application](#)

[Executing the Echo Test Application](#)

Executing the Matrix Multiply Application

This procedure outlines the steps to execute a matrix multiply sample application. Once the test suites have finished execution, you need to provide a power reset to the board in order to execute another application.

Procedure

1. Start the matrix multiply application from the U-Boot prompt.

```
U-Boot-PetaLinux> dcache off;
fatload mmc 0:0 0x10000000 matrix_multiply.bin;
go 0x10000000
```

Results

Matrix Multiplication Application

This sample application will offload matrix multiplication operations from Nucleus master to remote cores

```
*****
*****
Loading remote context : firmware1
*****
*****
*****
BOOTING NUCLEUS REMOTE FIRMWARE
*****
*****
Starting application for first firmware ..
*****
*****
Please enter command and press enter key
*****
1 - Generates random 6x6 matrices and transmits them to remote core over
rpmsg ..
2 - Quit this application ..
CMD>
Generating random matrices now ...
Master : Nucleus : Input matrix 0
0 9 1 1 9 4
```



```
2 8 5 2 1 3
9 2 6 2 6 2
0 3 3 4 0 1
0 2 3 1 3 7
4 3 2 5 7 7
Master : Nucleus : Input matrix 1
1 4 0 2 6 6
4 7 0 5 5 6
0 1 6 8 6 5
6 6 9 0 1 6
3 3 0 1 4 0
7 9 4 6 1 5
Writing generated matrices to rpmsg device, 296 bytes written ..
Received results! - 148 bytes from rpmsg device (transmitted from remote
context)
Master : Nucleus : Printing results
97 133 31 86 92 85
70 111 60 103 91 112
61 104 62 94 128 118
43 57 58 45 38 62
72 95 55 79 48 68
116 153 85 88 91 117
*****
Please enter command and press enter key
*****
1 - Generates random 6x6 matrices and transmits them to remote core over
rpmsg ..
2 - Quit this application ..
CMD>
Quitting application ..
Matrix multiplication demo end
*****
*****
Shutting down and deinitializing remote context : firmware1
*****
WARNING rx_vq: freeing non-empty virtqueue
WARNING tx_vq: freeing non-empty virtqueue

*****
BOOTING BAREMETAL REMOTE FIRMWARE
*****
*****
Starting application for second firmware ..
*****
*****
Please enter command and press enter key
*****
1 - Generates random 6x6 matrices and transmits them to remote core over
rpmsg ..
2 - Quit this application ..
CMD>
*****
Starting application for first firmware ..
*****
*****
*****
Please enter command and press enter key
*****
```

```
1 - Generates random 6x6 matrices and transmits them to remote core over
rpmsg ..
2 - Quit this application ..
CMD>
Generating random matrices now ...
Master : Nucleus : Input matrix 0
0 9 1 1 9 4
2 8 5 2 1 3
9 2 6 2 6 2
0 3 3 4 0 1
0 2 3 1 3 7
4 3 2 5 7 7
Master : Nucleus : Input matrix 1
1 4 0 2 6 6
4 7 0 5 5 6
0 1 6 8 6 5
6 6 9 0 1 6
3 3 0 1 4 0
7 9 4 6 1 5
Writing generated matrices to rpmsg device, 296 bytes written ..
Received results! - 148 bytes from rpmsg device (transmitted from remote
context)
Master : Nucleus : Printing results
97 133 31 86 92 85
70 111 60 103 91 112
61 104 62 94 128 118
43 57 58 45 38 62
72 95 55 79 48 68
116 153 85 88 91 117
*****
Please enter command and press enter key
*****
1 - Generates random 6x6 matrices and transmits them to remote core over
rpmsg ..
2 - Quit this application ..
CMD>
Quitting application ..
Matrix multiplication demo end
```

Related Topics

[Executing Applications and Tests for Nucleus Master/Nucleus Remote and Nucleus Master /Baremetal Remote Configurations](#) [Executing the Echo Test Application](#)

Executing the Echo Test Application

This procedure outlines the instructions to execute the echo test application. Once it is complete, you need to provide a power reset to the board in order to execute another application.

Procedure

1. Start the Echo Application from the U-Boot prompt:

```
U-Boot-PetaLinux> dcache off;  
                  fatload mmc 0:0 0x10000000 echo_test.bin;  
                  go 0x10000000
```

Results

```
*****  
                        Echo Test Application  
*****  
  
This test application will send variable length data  
packets to remote cores  
*****  
  
*****  
Loading remote context : firmware1  
*****  
*****  
BOOTING NUCLEUS REMOTE FIRMWARE  
*****  
  
*****  
Starting echo test on : firmware1  
*****  
  
Sending data packet of size: 1 to remote firmware : firmware1  
Received data packet of size: 1 from remote successfully  
Sending data packet of size: 2 to remote firmware : firmware1  
Received data packet of size: 2 from remote successfully  
Sending data packet of size: 3 to remote firmware : firmware1  
Received data packet of size: 3 from remote successfully  
Sending data packet of size: 4 to remote firmware : firmware1  
Received data packet of size: 4 from remote successfully  
Sending data packet of size: 5 to remote firmware : firmware1  
Received data packet of size: 5 from remote successfully  
Sending data packet of size: 6 to remote firmware : firmware1  
Received data packet of size: 6 from remote successfully  
.....  
.....  
Sending data packet of size: 486 to remote firmware : firmware1  
Received data packet of size: 486 from remote successfully  
Sending data packet of size: 487 to remote firmware : firmware1  
Received data packet of size: 487 from remote successfully  
Sending data packet of size: 488 to remote firmware : firmware1  
Received data packet of size: 488 from remote successfully  
*****  
*****  
Removing remote context : firmware1  
*****  
  
WARNING rx_vq: freeing non-empty virtqueue  
WARNING tx_vq: freeing non-empty virtqueue  
  
*****  
Loading remote context : firmware2  
*****
```

```
*****
BOOTING BAREMETAL REMOTE FIRMWARE
*****

*****
Starting echo test on : firmware2
*****

Sending data packet of size: 1 to remote firmware : firmware2
Received data packet of size: 1 from remote successfully
Sending data packet of size: 2 to remote firmware : firmware2
Received data packet of size: 2 from remote successfully
Sending data packet of size: 3 to remote firmware : firmware2
Received data packet of size: 3 from remote successfully
Sending data packet of size: 4 to remote firmware : firmware2
Received data packet of size: 4 from remote successfully
Sending data packet of size: 5 to remote firmware : firmware2
Received data packet of size: 5 from remote successfully
Sending data packet of size: 6 to remote firmware : firmware2
Received data packet of size: 6 from remote successfully
...
...
Sending data packet of size: 487 to remote firmware : firmware2
Received data packet of size: 487 from remote successfully
Sending data packet of size: 488 to remote firmware : firmware2
Received data packet of size: 488 from remote successfully
Removing remote context : firmware2
WARNING rx_vq: freeing non-empty virtqueue
WARNING tx_vq: freeing non-empty virtqueue
Test Completed
```

Related Topics

[Executing Applications and Tests for Nucleus Master/Nucleus Remote and Nucleus Master /Baremetal Remote Configurations](#) [Executing the Functional Test Suite](#)

Executing the Functional Test Suite

This procedure outlines the steps to execute functional test suites. Once the test suites have finished execution, you need to provide a power reset to the board in order to execute another application.

Procedure

1. Start the Functional Test Suite application from the U-Boot prompt:

```
U-Boot-PetaLinux> dcache off;
fatload mmc 0:0 0x10000000 func_test_suite.bin;
go 0x10000000
```



```
Transmitting payloads to remote
.....
.....
.....
.....
.....
.....
.....
*****
```

```
.....
.....
```

```
Boot remote context : firmware1
*****
```

```
Transmitting payloads to remote
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
*****
```

```
Shutdown remote context : firmware1
*****
```

```
WARNING rx_vq: freeing non-empty virtqueue
WARNING tx_vq: freeing non-empty virtqueue
```

```
RESULT: PASSED
FUNCTIONAL TEST SUITE COMPLETED EXECUTION!
```

```
Boot remote context : firmware2
*****
```

```
REMOTE FIRMWARE NAME : baremetal-fn-test-suite-remote-firmware
```

```
*****
This test case will test rpmsg_send API
*****
```

```
Transmitting payloads to remote
.....
.....
.....
.....
.....
.....
.....
```

.....
.....
RESULT: PASSED

This test case will test rpmsg_send_offchannel API

.....
.....
.....
.....
.....
.....
.....
.....
.....

RESULT: PASSED

This test case will test rpmsg_create_ept API

Creating an endpoint at address: 59
Testing payloads on endpoint at address: 59

.....
.....
.....
.....
.....
.....
.....

Creating an endpoint at address: 60
Testing payloads on endpoint at address: 60

.....
.....
.....
.....
.....
.....
.....

Creating an endpoint at address: 61
Testing payloads on endpoint at address: 61

.....
.....
.....
.....
.....
.....
.....

Creating an endpoint at address: 62
Testing payloads on endpoint at address: 62

.....
.....
.....

.....
.....
.....

RESULT: PASSED

This test case will test channel deletion

RESULT: PASSED

Shutdown remote context : firmware2

WARNING rx_vq: freeing non-empty virtqueue
WARNING tx_vq: freeing non-empty virtqueue

This test case will test multiple lifecycles for firmware: firmware2

Boot remote context : firmware2

Transmitting payloads to remote

.....
.....
.....
.....
.....
.....
.....

Shutdown remote context : firmware2

WARNING rx_vq: freeing non-empty virtqueue
WARNING tx_vq: freeing non-empty virtqueue

.....
.....

Boot remote context : firmware2

Transmitting payloads to remote

.....
.....

```
.....  
.....  
.....  
.....  
.....  
.....  
.....  
*****  
  
Shutdown remote context : firmware2  
*****  
  
WARNING rx_vq: freeing non-empty virtqueue  
WARNING tx_vq: freeing non-empty virtqueue  
.  
RESULT:      PASSED  
FUNCTIONAL TEST SUITE COMPLETED EXECUTION!
```

Related Topics

[Executing Applications and Tests for Nucleus Master/Nucleus Remote and Nucleus Master/Baremetal Remote Configurations](#) [Executing the Echo Test Application](#)

Build and Execute Applications for Nucleus Master/Linux Remote and Baremetal Master/Linux Remote Configurations

The following steps will generate a Linux remote kernel image using PetaLinux workflow. Then an OpenAMP Framework build will generate Nucleus and master Baremetal master applications which will be able to boot this Linux image.

Building a Linux Remote Kernel Image	50
Building the OpenAMP Framework Library, Applications and Tests	54
Executing Applications and Tests for Nucleus Master/Linux Remote Configuration	54
Executing the Matrix Multiply Application for Nucleus Master/Linux Remote	55
Executing the Echo Test Application	62
Executing the Functional Test Suites	68
Executing Applications and Tests for Baremetal Master/Linux Remote Configuration	70
Executing the Matrix Multiply Application for Baremetal Master/Linux Remote	71
Executing the Echo Test Application	77
Executing the Functional Test Suites	81

Building a Linux Remote Kernel Image

The following steps generate a Linux kernel image, using the PetaLinux workflow. Then an OpenAMP Framework build will generate Nucleus and Baremetal master applications which will be able to boot this Linux image.

Note



The provided procedure will patch the PetaLinux source code for a unicore configuration. If you want to retain an original copy of the Linux kernel source, it is recommended that you perform another installation of the PetaLinux for remote kernel, go through the environment setup and build the remote kernel image from this new installation.

Procedure

1. Patch Linux Source

- o Patch the gic handler within PetaLinux kernel source tree as:

```
$ patch $PETALINUX/components/linux-kernel/xlnx-  
3.8/arch/arm/common/gic.c <  
$OPENAMP/libs/system/zc702evk/linux/patches/linux/petalinux2013.10/  
gic.patch
```

- o Patch the devtree component as:

```
$patch $PETALINUX/components/linux-kernel/xlnx-  
3.8/arch/arm/kernel/devtree.c <  
$OPENAMP/libs/system/zc702evk/linux/patches/linux/petalinux2013.10/  
devtree.patch
```

Note



This patch modifies the Linux kernel source to enable it to co-operatively operate in a remote context with Nucleus RTOS/Bare-metal based master.

Once this patch is applied, in order to rebuild the SMP Linux kernel project for master context, this patch should be removed.

2. Create a project for the remote Linux unicore kernel:

Change directory to the location where remote Linux unicore kernel project needs to be created and execute the following:

```
>petalinux-create -t project -n <remote_name> --template zynq
```

where *<remote_name>* is the user defined name of the remote Linux unicore kernel project.

Note



This command creates a project within a folder named *<remote name>* under the present working directory. Here forth, *<remote_root>* will refer to this location of remote unicore project.

The structure of this project should be similar to:

```
<remote_root>
  "hw-description"
  "subsystems"
    "linux"
      "hw-decscription"
        Systems.dts
  "config.project"
```

3. Create Linux Kernel-Space and User-Space Applications

Assuming that the remote project has been created successfully and \$OPENAMP is set to the root of OpenAMP Framework source tree, execute the following command:

```
$ source $OPENAMP/libs/system/zc702evk/linux/scripts/
  open_amp_create_projects.sh remote <remote_root>
```

where *<remote_root>* is the path of the remote Linux project.

It will generate the following user space application and modules:

```
1. <remote_root>/components/apps/mat_mul_demo
2. <remote_root>/components/apps/echo_test
3. <remote_root>/components/modules/zynq_rpmsg_driver
4. <remote_root>/components/modules/rpmsg_mat_mul_kern_app
5. <remote_root>/components/modules/rpmsg_echo_test_kern_app
6. <remote_root>/components/modules/rpmsg_user_dev_driver
7. <remote_root>/components/modules/rpmsg_func_test_kern_app
```

4. Configure the System Memory Map

- a. Setup PetaLinux so that the Linux kernel address starts at address 0x00000000. Execute:

```
$ cd <remote_root>
$ petalinux-config
```

- b. Change the Kernel base address value to 0x00000000 if it is not already set to this value.

```
*** linux Components Selection ***
...
(0x00000000) Kernel base address
```

5. Configure the PetaLinux kernel

Ensure the following kernel configurations:

- a. Execute the following command while in the *<remote_root>* directory

```
$ petalinux-config -c kernel
```

- b. Select Enable loadable module support.

```
Kernel Configuration --->
[*] Enable loadable module support --->
```

- c. Disable Symmetric Multi-Processing within Kernel Features

```
Kernel Configuration --->
  Kernel Features --->
    [ ] Symmetric Multi-Processing
```

d. Select High Memory Support within Kernel Features

Select 2G/2G user/kernel split as Memory split within Kernel Features

```
Kernel Configuration --->
  Kernel Features --->
    Memory split (2G/2G user/kernel split) --->
      [*] High Memory Support
```

e. Enable Userspace firmware loading support

```
Kernel Configuration --->
Device Drivers --->
Generic Driver Options --->
<*> Userspace firmware loading support
[ ] Include in-kernel firmware blobs in kernel binary
() External firmware blobs to build into the kernel
Binary
```

f. Enable Remoteproc Driver

```
Kernel Configuration --->
Device Drivers --->
Remoteproc drivers (EXPERIMENTAL) --->
<M> Support ZYNQ remote proc
```

Note



Enable the driver as a Module <M>

6. Build PetaLinux

```
$ cd <remote_root>
$ petalinux-build
```

7. Copy the PetaLinux image to OPENAMP

Copy the PetaLinux image:

```
<remote_root>/images/linux/image.ub
```

to the following OPENAMP directory

```
$OPENAMP/libs/system/zc702evk/linux
```

using the following command

```
$ cp <remote_root>/images/linux/image.ub
$OPENAMP/libs/system/zc702evk/linux/image.ub
```

Related Topics

[Building the OpenAMP Framework Library, Applications and Tests](#)

Building the OpenAMP Framework Library, Applications and Tests

An OpenAMP Framework build generates Nucleus and master Baremetal master applications which can boot the Linux image.

Prerequisites

- The OpenAMP Framework library is installed at location \$OPENAMP.

Procedure

1. Change directory to \$OPENAMP

```
$ cd $OPENAMP
```

2. Execute the build script. This operation builds remote applications for Nucleus and Baremetal environments as well as master applications for Nucleus.

Note



The build operation generates application outputs in the respective application directories.

```
$ source open_amp_build.sh
```

Related Topics

[Building a Linux Remote Kernel Image](#)

Executing Applications and Tests for Nucleus Master/Linux Remote Configuration

This is a procedure for moving the U-Boot, application and test binaries to the boot medium (SD Card).

Prerequisites

- The U-Boot binary image (*BOOT.bin*), and OpenAMP Framework application and test binaries are built and available.

Procedure

1. Format a SDCARD and partition for FAT.
2. Insert the SDCARD in the host machine and place the generated U-Boot image (*BOOT.bin*) for ZC702EVK, OpenAMP Framework binary images available at locations listed below should be moved to SDCARD as well.
 - *apps/samples/master/nucleus/matrix_multiply/nucleus_linux/matrix_multiply.bin*
 - *apps/tests/master/nucleus/echo_test/nucleus_linux/echo_test.bin*
 - *apps/tests/master/nucleus/func_test_suite/nucleus_linux/func_test_suite.bin*
3. Insert the SDCARD in the SDIO slot J64 on ZC702EVK.
4. Power up board to boot U-Boot
5. Once U-Boot is up, in the U-Boot console, enter:

```
U-Boot-PetaLinux> dcache off;  
fatload mmc <dev>:<part> 0x10000000 <nucleus app>.bin;  
go 0x10000000
```

Note



<dev> refers to the relevant MMC device number. The list of available devices can be retrieved from u-boot prompt by entering - U-Boot-PetaLinux> mmc list.
<part> refers to the relevant partition on the current mmc device. It can be retrieved from u-boot prompt by entering - U-Boot-PetaLinux> mmc part.

Note



The following sections assume that the relevant <dev>:<part> value is 0:0. The actual values may vary and the user is advised to confirm the device and partition numbers first.

Related Topics

[Executing Applications and Tests for Baremetal Master/Linux Remote Configuration](#)

[Executing the Matrix Multiply Application for Baremetal Master/Linux Remote](#)

Executing the Matrix Multiply Application for Nucleus Master/Linux Remote

This procedure outlines the instructions to execute the matrix multiplication demo application. You can execute one of the two Linux remote applications (kernel-space or user-space). In order to execute the other application, you need to perform a power reset and restart the Nucleus master matrix_multiply application.

Procedure

1. Start the matrix multiply application from the U-Boot prompt:

```
U-Boot-PetaLinux> dcache off;fatload mmc 0:0 0x10000000  
matrix_multiply.bin; go 0x10000000
```

It starts the Nucleus application on the primary core which will on its own boot the Linux kernel on the remote core. The Linux kernel will boot and capture the console.

2. Once the Linux kernel prompt is up, enter root login and password:

```
Xilinx-ZC702-2013_3 login: root  
Password:  
login[776]: root login on `ttyPS0`
```

3. Load Zynq rpmsg driver module

```
root@Xilinx-ZC702-2013_3:~# modprobe zynq_rpmsg_driver
```

- User Space

- Load the rpmsg user device driver using the following command:

```
root@Xilinx-ZC702-2013_3:~# modprobe rpmsg_user_dev_driver
```

- Start matrix_multiply user-space application

```
root@Xilinx-ZC702-2013_3:~# mat_mul_demo
```

- Kernel Space

- Load the matrix multiply application kernel driver using the following command:

```
root@Xilinx-ZC702-2013_3:~# modprobe rpmsg_mat_mul_kern_app
```

Results

User space

```
*****
```

```
Matrix Multiplication Application
```

```
This sample application will boot a remote Linux firmware
```

```
This sample application will boot a remote Linux firmware
```

```
and handle offloaded matrix multiplication operations from
```

```
Linux RPMSG master to Nuclues RPMS Remote
```

```
*****
```

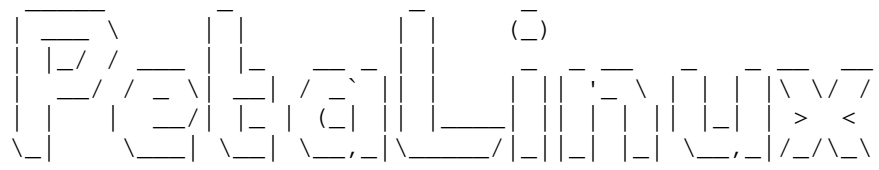
```
Loading remote context : firmware1
```



```
*****
*****
BOOTING LINUX REMOTE FIRMWARE
*****

*****
OpenAMP Linux Bootstrap.
*****

Linux Bootstrap: Locating Linux Kernel and DTB from FIT image.
Linux Bootstrap: Kernel image is compressed. Starting decompression
process. It may take a while...
Linux Bootstrap: Linux image decompression complete.
Linux Bootstrap: Linux kernel image has been loaded into memory.
Linux Bootstrap: Loaded DTB.
Linux Bootstrap: Booting Linux.
Booting Linux on physical CPU 0x0
Linux version 3.8.11 (gcc version 4.7.3 (Sourcery CodeBench Lite 2013.05-
40) ) #99 PREEMPT Fri May 2 15:40:46 PKT 2014
CPU: ARMv7 Processor [413fc090] revision 0 (ARMv7), cr=18c5387d
CPU: PIPT / VIPT nonaliasing data cache, VIPT aliasing instruction cache
Machine: Xilinx Zynq Platform, model: .
Memory policy: ECC disabled, Data cache writeback
...
...
...
starting Busybox inet Daemon: inetd... done.
Starting uWeb server:
INIT: Entering runlevel: 5
Stopping Bootlog daemon: bootlogd.
```



```
PetaLinux v2013.10 (Yocto 1.4) Xilinx-ZC702-2013_3 ttyPS0

Xilinx-ZC702-2013_3 login: root
Password:
login[776]: root login on `ttyPS0'

root@Xilinx-ZC702-2013_3:~#
root@Xilinx-ZC702-2013_3:~# modprobe zynq_rpmsg_driver
zynq_rpmsg_driver 8000000.zynq-rpmsg_driver: virtio device registered
root@Xilinx-ZC702-2013_3:~# modprobe rpmsg_user_dev_driver
virtio_rpmsg_bus virtio0: rpmsg host is online
virtio_rpmsg_bus virtio0: creating channel rpmsg-openamp-demo-channel addr
0x1
rpmsg_user_dev_driver rpmsg0: new channel: 0x400 -> 0x1!
root@Xilinx-ZC702-2013_3:~# mat_mul_demo

Matrix multiplication demo start

Open rpmsg dev!
```

```
Query internal info ..
rpmsg kernel fifo size = 2048
rpmsg kernel fifo free space = 2048

Creating ui_thread and compute_thread ...

*****
Please enter command and press enter key
*****
1 - Generates random 6x6 matrices and transmits them to remote core over
rpmsg ..
2 - Quit this application ..
CMD>1

Compute thread unblocked ..
The compute thread is now blocking on a read() from rpmsg device

Generating random matrices now ...

Master : Linux : Input matrix 0

0 8 2 6 4 3
6 6 9 2 9 0
6 4 5 0 5 8
7 2 5 1 7 5
8 1 6 3 7 0
2 7 8 4 6 4

Master : Linux : Input matrix 1

7 4 1 6 6 0
8 2 6 5 2 2
5 1 4 3 4 4
0 2 7 6 5 4
6 9 4 4 4 0
8 1 4 1 0 0

Writing generated matrices to rpmsg device, 296 bytes written ..

Received results! - 148 bytes from rpmsg device (transmitted from remote
context)

Master : Linux : Printing results
122 69 126 101 70 48
189 130 128 141 130 56
193 90 102 99 84 28
172 107 94 106 99 28
136 109 87 117 117 38
178 96 144 123 102 62

*****
Please enter command and press enter key
*****
1 - Generates random 6x6 matrices and transmits them to remote core over
rpmsg ..
2 - Quit this application ..
CMD>1
```

```
Compute thread unblocked ..  
The compute thread is now blocking on a read() from rpmsg device
```

```
Generating random matrices now ...
```

```
Master : Linux : Input matrix 0
```

```
0 2 3 6 3 9  
3 0 5 6 4 1  
4 6 3 6 2 9  
6 7 2 6 1 5  
1 9 1 5 6 7  
8 8 2 1 5 5
```

```
Master : Linux : Input matrix 1
```

```
2 0 7 0 9 4  
1 3 2 6 1 4  
7 0 3 9 6 4  
7 7 3 0 5 1  
0 3 0 2 7 7  
9 9 9 8 9 8
```

```
Writing generated matrices to rpmsg device, 296 bytes written ..
```

```
Received results! - 148 bytes from rpmsg device (transmitted from remote context)
```

```
Master : Linux : Printing results
```

```
146 138 112 117 152 119  
92 63 63 61 124 74  
158 147 148 139 185 144  
120 111 125 102 155 113  
116 143 106 131 154 147  
90 91 126 116 177 148
```

```
*****  
Please enter command and press enter key  
*****
```

```
1 - Generates random 6x6 matrices and transmits them to remote core over  
rpmsg ..
```

```
2 - Quit this application ..
```

```
CMD>2
```

```
Quitting application ..
```

```
Matrix multiplication demo end
```

```
INIT: Sending processes the TERM signal
```

```
Broadcast message from root@Xilinx-ZC702-2013_3 (Thu Jan 1 00:00:39  
1970):
```

```
The system is going down for system halt NOW!
```

```
INIT: Sending processes the KILL signal
```

```
Deconfiguring network interfaces... done.
```

```
Sending all processes the TERM signal...
```

```
Sending all processes the KILL signal...
```

```
Unmounting remote filesystems...
```

```
Deactivating swap...
```

```
Unmounting local filesystems...
System halted.
```

Kernel space

```
*****

                Matrix Multiplication Application

This sample application will boot a remote Linux firmware

This sample application will boot a remote Linux firmware
and handle offloaded matrix multiplication operations from
Linux RPMSG master to Nuclues RPMS Remote

*****

Loading remote context : firmware1

*****
*****
*****
BOOTING LINUX REMOTE FIRMWARE
*****
*****
OpenAMP Linux Bootstrap.
*****

Linux Bootstrap: Locating Linux Kernel and DTB from FIT image.
Linux Bootstrap: Kernel image is compressed. Starting decompression
process. It may take a while...
Linux Bootstrap: Linux image decompression complete.
Linux Bootstrap: Linux kernel image has been loaded into memory.
Linux Bootstrap: Loaded DTB.
Linux Bootstrap: Booting Linux.

Booting Linux on physical CPU 0x0
Linux version 3.8.11 (gcc version 4.7.3 (Sourcery CodeBench Lite 2013.05-
40) ) #4 PREEMPT Wed May 7 18:46:05 PKT 2014
CPU: ARMv7 Processor [413fc090] revision 0 (ARMv7), cr=18c5387d
CPU: PIPT / VIPT nonaliasing data cache, VIPT aliasing instruction cache
Machine: Xilinx Zynq Platform, model: .
Memory policy: ECC disabled, Data cache writeback
...
...
...
done.
starting Busybox inet Daemon: inetd... done.
Starting uWeb server:
INIT: Entering runlevel: 5
Stopping Bootlog daemon: bootlogd.
```

```
| _ _ _ \   | |   | |   | |   | |
```



```
PetaLinux v2013.10 (Yocto 1.4) Xilinx-ZC702-2013_3 ttyPS0
```

```
Xilinx-ZC702-2013_3 login: root
Password:
login[776]: root login on `ttyPS0'
```

```
root@Xilinx-ZC702-2013_3:~# modprobe zynq_rpmsg_driver
zynq_rpmsg_driver 8000000.zynq-rpmsg_driver: virtio device registered
root@Xilinx-ZC702-2013_3:~#
root@Xilinx-ZC702-2013_3:~# modprobe rpmsg_mat_mul_kern_app
virtio_rpmsg_bus virtio0: rpmsg host is online
virtio_rpmsg_bus virtio0: creating channel rpmsg-openamp-demo-channel addr
0x1
rpmsg_mat_mul_kern_app: module license 'unspecified' taints kernel.
Disabling lock debugging due to kernel taint
```

```
Demo Start - Demo rpmsg driver got probed
since the rpmsg device associated with driver was found !
```

```
Create endpoint and register rx callback
```

```
Master : Linux : Generating random matrices
```

```
Master : Linux : Input matrix 0
```

```
5 0 9 3 1 5
0 9 6 1 9 0
2 2 0 4 4 5
6 2 5 6 3 2
3 5 1 1 2 4
7 1 3 3 2 7
```

```
Master : Linux : Input matrix 1
```

```
3 7 7 6 7 9
9 1 4 4 5 5
4 5 4 3 6 1
7 0 9 2 2 7
5 8 5 0 7 2
1 9 3 1 9 2
```

```
Master : Linux : Sent 296 bytes of data over rpmsg channel to remote
```

```
Master : Linux : Received 148 bytes of data over rpmsg channel from
remote
```

```
root@Xilinx-ZC702-2013_3:~#
Master : Linux : Printing results
82 133 118 68 147 87
157 111 114 56 146 76
77 93 93 33 105 74
115 111 145 73 133 121
79 83 76 47 104 72
80 144 123 68 155 110
```

```
Master : Linux : Received 4 bytes of data over rpmsg channel from remote
INIT: Sending processes the TERM signal
```

```
Broadcast message from root@Xilinx-ZC702-2013_3 (Thu Jan  1 00:02:41
1970):
```

```
The system is going down for system halt NOW!
```

```
INIT: Sending processes the KILL signal
```

```
Deconfiguring network interfaces... done.
```

```
Sending all processes the TERM signal...
```

```
Sending all processes the KILL signal...
```

```
Unmounting remote filesystems...
```

```
Deactivating swap...
```

```
Unmounting local filesystems...
```

```
System halted.
```

Related Topics

[Executing Applications and Tests for Nucleus Master/Linux Remote Configuration](#) [Executing the Echo Test Application](#)

Executing the Echo Test Application

This procedure outlines the instructions to execute the echo test application. You can execute one of the two Linux remote applications (kernel space or user space). In order to execute the other application, you need to perform a power reset and restart the Nucleus master echo test application.

Procedure

1. Start the echo test application from the U-Boot prompt:

```
U-Boot-PetaLinux> dcache off;fatload mmc 0:0 0x10000000
echo_test.bin; go 0x10000000
```

It starts the Nucleus application on the primary core which will on its own boot the Linux kernel on remote core. The Linux kernel will boot and capture the console.

2. Once the Linux kernel prompt is up, enter root login and password:

```
Xilinx-ZC702-2013_3 login: root
Password:
login[776]: root login on `ttyPS0'
```

3. Load Zynq rpmsg driver module

```
root@Xilinx-ZC702-2013_3:~# modprobe zynq_rpmsg_driver
```

- User Space

- i. Load the rpmsg user device driver using the following command:

```
root@Xilinx-ZC702-2013_3:~# modprobe rpmsg_user_dev_driver
```

ii. Start matrix_multiply user-space application

```
root@Xilinx-ZC702-2013_3:~# echo_test
```

- Kernel Space

- Load the echo test driver using the following commands:

```
root@Xilinx-ZC702-2013_3:~# modprobe virtio_rpmsg_bus
root@Xilinx-ZC702-2013_3:~# modprobe rpmsg_echo_test_kern_app
```

Results

User space

```
*****
                          Echo Test Application
*****

This test application will start a linux kernel on remote core and will
act
as an RPMSG remote itself to echo back any data that it receives from
Linux

*****
*****
Loading remote context : firmware1

*****
*****
BOOTING LINUX REMOTE FIRMWARE
*****

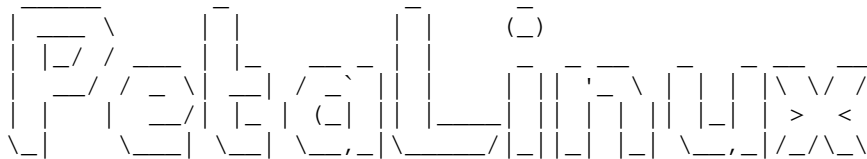
*****
OpenAMP Linux Bootstrap.
*****

Linux Bootstrap: Locating Linux Kernel and DTB from FIT image.
Linux Bootstrap: Kernel image is compressed. Starting decompression
process. It may take a while...
Linux Bootstrap: Linux image decompression complete.
Linux Bootstrap: Linux kernel image has been loaded into memory.
Linux Bootstrap: Loaded DTB.
Linux Bootstrap: Booting Linux.
Booting Linux on physical CPU 0x0
Linux version 3.8.11 (gcc version 4.7.3 (Sourcery CodeBench Lite 2013.05-
40) ) #99 PREEMPT Fri May 2 15:40:46 PKT 2014
CPU: ARMv7 Processor [413fc090] revision 0 (ARMv7), cr=18c5387d
CPU: PIPT / VIPT nonaliasing data cache, VIPT aliasing instruction cache
Machine: Xilinx Zynq Platform, model: .
Memory policy: ECC disabled, Data cache writeback
...
...
...
Starting Bootlog daemon: bootlogd.
Creating /dev/flash/* device nodes
Configuring network interfaces... udhcpc (v1.20.2) started
```

```

Sending discover...
Sending discover...
xemacps e000b000.ps7-ethernet: Set clk to 124999998 Hz
xemacps e000b000.ps7-ethernet: link up (1000/FULL)
IPv6: ADDRCONF(NETDEV_CHANGE): eth0: link becomes ready
Sending discover...
Sending select for 137.202.156.251...
Lease of 137.202.156.251 obtained, lease time 1800
/etc/udhcpc.d/50default: Adding DNS 137.202.187.16
/etc/udhcpc.d/50default: Adding DNS 137.202.23.16
/etc/udhcpc.d/50default: Adding DNS 147.34.2.16
done.
starting Busybox inet Daemon: inetd... done.
Starting uWeb server:
INIT: Entering runlevel: 5
Stopping Bootlog daemon: bootlogd.

```



```
PetaLinux v2013.10 (Yocto 1.4) Xilinx-ZC702-2013_3 ttyPS0
```

```

Xilinx-ZC702-2013_3 login: root
Password:
login[776]: root login on `ttyPS0'

```

```

root@Xilinx-ZC702-2013_3:~# modprobe zynq_rpmsg_driver
zynq_rpmsg_driver 8000000.zynq-rpmsg_driver: virtio device registered
root@Xilinx-ZC702-2013_3:~# modprobe rpmsg_user_dev_driver
virtio_rpmsg_bus virtio0: rpmsg host is online
virtio_rpmsg_bus virtio0: creating channel rpmsg-openamp-demo-channel addr
0x1
rpmsg_user_dev_driver rpmsg0: new channel: 0x400 -> 0x1!
root@Xilinx-ZC702-2013_3:~# echo_test

```

Echo test start

Open rpmsg dev!

```

Query internal info ..
rpmsg kernel fifo size = 2048
rpmsg kernel fifo free space = 2048

```

```

*****
Please enter command and press enter key
*****
1 - Send data to remote core, retrieve the echo and validate its
integrity ..
2 - Quit this application ..
CMD>1

```

```

sending payload number 0 of size 9
received payload number 0 of size 9

```



```
sending payload number 2 of size 10
received payload number 2 of size 10
```

```
sending payload number 3 of size 11
received payload number 3 of size 11
```

```
sending payload number 4 of size 12
received payload number 4 of size 12
```

```
sending payload number 5 of size 13
received payload number 5 of size 13
```

```
...
...
```

```
sending payload number 484 of size 492
received payload number 484 of size 492
```

```
sending payload number 485 of size 493
received payload number 485 of size 493
```

```
sending payload number 486 of size 494
received payload number 486 of size 494
```

```
sending payload number 487 of size 495
received payload number 487 of size 495
```

```
*****
```

```
Test Results: Error count = 0
```

```
*****
```

```
*****
```

```
Please enter command and press enter key
*****
```

```
1 - Send data to remote core, retrieve the echo and validate its
integrity ..
```

```
2 - Quit this application ..
```

```
CMD>2
```

```
INIT: Sending processes the TERM signal
```

```
Broadcast message from root@Xilinx-ZC702-2013_3 (Thu Jan 1 00:00:49
1970):
```

```
The system is going down for system halt NOW!
```

```
INIT: Sending processes the KILL signal
```

```
Deconfiguring network interfaces... done.
```

```
Sending all processes the TERM signal...
```

```
Sending all processes the KILL signal...
```

```
Unmounting remote filesystems...
```

```
Deactivating swap...
```

```
Unmounting local filesystems...
```

```
System halted.
```

Kernel space

```
*****  
*****
```

Echo Test Application

This test application will start a linux kernel on remote core and will act

as an RPMSG remote itself to echo back any data that it receives from Linux

```
*****  
*****
```

```
*****  
*****
```

Loading remote context : firmware1

```
*****  
*****
```

```
*****  
BOOTING LINUX REMOTE FIRMWARE  
*****  
*****
```

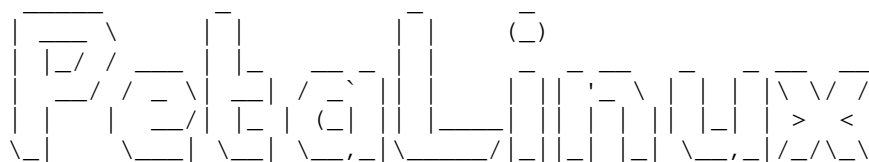
```
OpenAMP Linux Bootstrap.  
*****
```

```
Linux Bootstrap: Locating Linux Kernel and DTB from FIT image.  
Linux Bootstrap: Kernel image is compressed. Starting decompression  
process. It may take a while...  
Linux Bootstrap: Linux image decompression complete.  
Linux Bootstrap: Linux kernel image has been loaded into memory.  
Linux Bootstrap: Loaded DTB.  
Linux Bootstrap: Booting Linux.
```

```
Booting Linux on physical CPU 0x0  
Linux version 3.8.11 (awais@awais-VirtualBox) (gcc version 4.7.3 (Sourcery  
CodeBench Lite 2013.05-40) ) #4 PREEMPT Wed May 7 18:46:05 PKT 2014  
CPU: ARMv7 Processor [413fc090] revision 0 (ARMv7), cr=18c5387d  
CPU: PIPT / VIPT nonaliasing data cache, VIPT aliasing instruction cache  
Machine: Xilinx Zynq Platform, model: .  
Memory policy: ECC disabled, Data cache writeback
```

```
...  
...  
...
```

```
starting Busybox inet Daemon: inetd... done.  
Starting uWeb server:  
INIT: Entering runlevel: 5  
Stopping Bootlog daemon: bootlogd.
```



PetaLinux v2013.10 (Yocto 1.4) Xilinx-ZC702-2013_3 ttyPS0

```
Xilinx-ZC702-2013_3 login: root
Password:
login[776]: root login on `ttyPS0'

root@Xilinx-ZC702-2013_3:~# modprobe zynq_rpmsg_driver
zynq_rpmsg_driver 8000000.zynq-rpmsg_driver: virtio device registered
root@Xilinx-ZC702-2013_3:~# modprobe rpmsg_echo_test_kern_app
virtio_rpmsg_bus virtio0: rpmsg host is online
virtio_rpmsg_bus virtio0: creating channel rpmsg-openamp-demo-channel addr
0x1
rpmsg_echo_test_kern_app: module license 'unspecified' taints kernel.
Disabling lock debugging due to kernel taint

Echo Test Start!

Master : Linux Kernal Space : Sending payload num 0 of size 9
Master : Linux Kernal Space : Received payload num 0 of size 9
Master : Linux Kernal Space : Sending payload num 1 of size 10
Master : Linux Kernal Space : Received payload num 1 of size 10
...
...
Master : Linux Kernal Space : Sending payload num 486 of size 495
Master : Linux Kernal Space : Received payload num 486 of size 495
Master : Linux Kernal Space : Sending payload num 487 of size 496
Master : Linux Kernal Space : Received payload num 487 of size 496
*****

Echo Test Results: Error count = 0

*****

Master : Linux Kernal Space : Received payload num -279534246 of size 4
root@Xilinx-ZC702-2013_3:~#
Broadcast message from root@Xilin +Eëdown for system halt NOW!
INIT: Sending processes the TERM signal
INIT: Sending processes the KILL signal
Deconfiguring network interfaces... done.
Sending all processes the TERM signal...
Sending all processes the KILL signal...
Unmounting remote filesystems...
Deactivating swap...
Unmounting local filesystems...
```

Related Topics

[Executing Applications and Tests for Nucleus Master/Linux Remote Configuration](#) [Executing the Functional Test Suites](#)

Executing the Functional Test Suites

This procedure outlines the instructions to execute the functional test suite.

Note



The target needs to be power cycled after running this application in order to run another application.

Procedure

1. Start the functional test suites application from the U-Boot prompt:

```
U-Boot-PetaLinux> dcache off;fatload mmc 0:0 0x10000000  
func_test_suite.bin; go 0x10000000
```

It starts the Nucleus application on the primary core which will on its own boot the Linux kernel on remote core. The Linux kernel will boot and capture the console.

2. Once the Linux kernel prompt is up, enter root login and password:

```
Xilinx-ZC702-2013_3 login: root  
Password:  
login[776]: root login on `ttyPS0'
```

3. Load Zynq rpmsg driver module

```
root@Xilinx-ZC702-2013_3:~# modprobe zynq_rpmsg_driver
```

4. Load the functional test suite driver using the following command:

```
root@Xilinx-ZC702-2013_3:~# modprobe rpmsg_func_test_kern_app
```

Once the test suite is completed, Linux will shutdown.

Results

```
*****  
OpenAMP Test Suite  
This test suite will execute multiple test cases for rpmsg and rempteproc  
APIs available within OpenAMP  
  
*****  
*****  
Boot remote context : firmware1  
*****  
*****  
OpenAMP Linux Bootstrap.  
*****  
Linux Bootstrap: Locating Linux Kernel and DTB from FIT image.  
Linux Bootstrap: Kernel image is compressed. Starting decompression  
process. It may take a while...  
Linux Bootstrap: Linux image decompression complete.  
Linux Bootstrap: Linux kernel image has been loaded into memory.  
Linux Bootstrap: Loaded DTB.
```


Related Topics

[Executing Applications and Tests for Nucleus Master/Linux Remote Configuration](#) [Executing the Echo Test Application](#)

Executing Applications and Tests for Baremetal Master/Linux Remote Configuration

This procedure shows the steps for moving U-Boot, application, and test binaries to boot medium (SD Card).

Prerequisites

- The U-Boot binary image (*BOOT.bin*), and OpenAMP Framework application and test binaries are built and available.

Procedure

1. Format an SDCARD and partition for FAT.
2. Insert the SDCARD in the host machine and place the generated U-Boot image (*BOOT.bin*) for ZC702EVK, OpenAMP Framework binary images available at locations listed below should be moved to SDCARD as well.
 - *apps/samples/master/baremetal/matrix_multiply/matrix_multiply.bin*
 - *apps/tests/master/baremetal/echo_test/echo_test.bin*
 - *apps/tests/master/baremetal/func_test_suite/func_test_suite.bin*
3. Insert the SDCARD in the SDIO slot J64 on ZC702EVK.
4. Power up the board to boot U-Boot
5. Once U-Boot is up, in the U-Boot console, enter

```
U-Boot-PetaLinux> dcache off;  
fatload mmc <dev>:<part> 0x10000000 <baremetal app>.bin;  
go 0x10000000
```

Note



<dev> refers to the relevant MMC device number. The list of available devices can be retrieved from the U-Boot prompt by entering - U-Boot-PetaLinux> mmc list.
<part> refers to the relevant partition on the current mmc device. It can be retrieved from the U-Boot prompt by entering - U-Boot-PetaLinux> mmc part.

Note



The following sections assume that the relevant <dev>:<part> value is 0:0. The actual values may vary and you are advised to confirm the device and partition numbers first.

Related Topics

[Executing Applications and Tests for Baremetal Master/Linux Remote Configuration](#)

[Executing the Matrix Multiply Application for Baremetal Master/Linux Remote](#)

Executing the Matrix Multiply Application for Baremetal Master/Linux Remote

The following procedure outlines the instructions to execute the matrix multiplication demo application. You can execute one of the two Linux remote applications (kernelspace or userspace). In order to execute the other application, you need to perform a power reset and restart the baremetal master matrix_multiply application.

Procedure

1. Start the matrix multiply application from the U-Boot prompt:

```
U-Boot-PetaLinux> dcache off;
fatload mmc 0:0 0x10000000 matrix_multiply.bin;
go 0x10000000
```

It starts the baremetal application on the primary core which will on its own boot the Linux kernel on remote core. The Linux kernel will boot and capture the console.

2. Once the Linux kernel prompt is up, enter root login and password:

```
Xilinx-ZC702-2013_3 login: root
Password:
login[776]: root login on `ttyPS0'
```

3. Load Zynq rpmsg driver module

```
root@Xilinx-ZC702-2013_3:~# modprobe zynq_rpmsg_driver
```

- User Space

- i. Load the rpmsg user device driver using the following command:

```
root@Xilinx-ZC702-2013_3:~# modprobe rpmsg_user_dev_driver
```

- ii. Start matrix_multiply user-space application

```
root@Xilinx-ZC702-2013_3:~# mat_mul_demo
```

- Kernel Space


```
root@Xilinx-ZC702-2013_3:~# mat_mul_demo

Matrix multiplication demo start

Open rpmsg dev!

Query internal info ..
rpmsg kernel fifo size = 2048
rpmsg kernel fifo free space = 2048

Creating ui_thread and compute_thread ...

*****
Please enter command and press enter key
*****
1 - Generates random 6x6 matrices and transmits them to remote core over
rpmsg ..
2 - Quit this application ..
CMD>1

Compute thread unblocked ..
The compute thread is now blocking on a read() from rpmsg device

Generating random matrices now ...

Master : Linux : Input matrix 0

0 5 0 0 1 6
8 0 1 6 5 2
7 0 1 2 2 2
3 7 7 1 3 7
4 5 9 2 8 9
3 1 4 5 3 8

Master : Linux : Input matrix 1

1 1 8 2 8 4
6 7 6 7 2 8
2 7 7 9 8 2
8 2 8 9 7 8
0 2 9 7 7 5
5 0 6 3 2 6

Writing generated matrices to rpmsg device, 296 bytes written ..

Received results! - 148 bytes from rpmsg device (transmitted from remote
context)

Master : Linux : Printing results
60 37 75 60 29 81
68 37 176 120 153 119
35 22 109 61 96 68
102 109 192 169 136 147
113 122 267 225 202 184
97 54 173 139 130 131

*****
Please enter command and press enter key
```

```
*****
1 - Generates random 6x6 matrices and transmits them to remote core over
rpmsg ..
2 - Quit this application ..
CMD>1

Compute thread unblocked ..
The compute thread is now blocking on a read() from rpmsg device

Generating random matrices now ...

Master : Linux : Input matrix 0

0 0 2 1 5 7
0 5 2 4 2 3
8 5 5 6 5 3
9 1 5 0 3 5
3 2 5 7 2 4
7 4 5 1 5 2

Master : Linux : Input matrix 1

8 8 7 2 2 0
5 3 7 2 9 2
5 0 4 1 0 7
6 5 0 3 2 4
0 9 8 5 0 5
9 0 3 6 2 6

Writing generated matrices to rpmsg device, 296 bytes written ..

Received results! - 148 bytes from rpmsg device (transmitted from remote
context)

Master : Linux : Printing results
79 50 69 72 16 85
86 53 68 52 59 68
177 154 160 92 79 112
147 102 129 70 37 82
137 83 83 70 46 101
125 118 143 67 56 84

*****
Please enter command and press enter key
*****
1 - Generates random 6x6 matrices and transmits them to remote core over
rpmsg ..
2 - Quit this application ..
CMD>2

Quitting application ..
Matrix multiplication demo end
INIT: Sending processes the TERM signal

Broadcast message from root@Xilinx-ZC702-2013_3 (Thu Jan 1 00:01:03
1970):
The system is going down for system halt NOW!
INIT: Sending processes the KILL signal
```



```
virtio_rpmsg_bus virtio0: creating channel rpmsg-openamp-demo-channel addr 0x1
rpmsg_mat_mul_kern_app: module license 'unspecified' taints kernel.
Disabling lock debugging due to kernel taint
```

```
Demo Start - Demo rpmsg driver got probed
since the rpmsg device associated with driver was found !
```

```
Create endpoint and register rx callback
```

```
Master : Linux : Generating random matrices
```

```
Master : Linux : Input matrix 0
```

```
5 0 9 3 1 5
0 9 6 1 9 0
2 2 0 4 4 5
6 2 5 6 3 2
3 5 1 1 2 4
7 1 3 3 2 7
```

```
Master : Linux : Input matrix 1
```

```
3 7 7 6 7 9
9 1 4 4 5 5
4 5 4 3 6 1
7 0 9 2 2 7
5 8 5 0 7 2
1 9 3 1 9 2
```

```
Master : Linux : Sent 296 bytes of data over rpmsg channel to remote
```

```
Master : Linux : Received 148 bytes of data over rpmsg channel from remote
```

```
root@Xilinx-ZC702-2013_3:~#
```

```
Master : Linux : Printing results
```

```
82 133 118 68 147 87
157 111 114 56 146 76
77 93 93 33 105 74
115 111 145 73 133 121
79 83 76 47 104 72
80 144 123 68 155 110
```

```
Master : Linux : Received 4 bytes of data over rpmsg channel from remote
INIT: Sending processes the TERM signal
```

```
Broadcast message from root@Xilinx-ZC702-2013_3 (Thu Jan 1 00:02:41 1970):
```

```
The system is going down for system halt NOW!
```

```
INIT: Sending processes the KILL signal
```

```
Deconfiguring network interfaces... done.
```

```
Sending all processes the TERM signal...
```

```
Sending all processes the KILL signal...
```

```
Unmounting remote filesystems...
```

```
Deactivating swap...
```

```
Unmounting local filesystems...
```

```
System halted.
```

Related Topics

[Executing Applications and Tests for Baremetal Master/Linux Remote Configuration](#)

[Executing the Echo Test Application](#)

Executing the Echo Test Application

The following procedure outlines the instructions to execute the echo test application. You can execute one of the two Linux remote applications (kernelspace or userspace). In order to execute the other application, you need to perform a power reset and restart the baremetal master echo test application.

Procedure

1. Start the echo test application from the U-Boot prompt:

```
U-Boot-PetaLinux> dcache off;fatload mmc 0:0 0x10000000  
echo_test.bin; go 0x10000000
```

It starts the baremetal application on the primary core which will on its own boot the Linux kernel on remote core. The Linux kernel will boot and capture the console.

2. Once the Linux kernel prompt is up, enter root login and password:

```
Xilinx-ZC702-2013_3 login: root  
Password:  
login[776]: root login on `ttyPS0`
```

3. Load Zynq rpmsg driver module

```
root@Xilinx-ZC702-2013_3:~# modprobe zynq_rpmsg_driver
```

- User Space

- i. Load the rpmsg user device driver using the following command:

```
root@Xilinx-ZC702-2013_3:~# modprobe rpmsg_user_dev_driver
```

- ii. Start matrix_multiply user-space application

```
root@Xilinx-ZC702-2013_3:~# echo_test
```

- Kernel Space

- Load the echo test driver using the following commands:

```
root@Xilinx-ZC702-2013_3:~# modprobe virtio_rpmsg_bus  
root@Xilinx-ZC702-2013_3:~# modprobe rpmsg_echo_test_kern_app
```

Results

User space

```
## Starting application at 0x10000000 ...

*****
OpenAMP Linux Bootstrap.
*****

Linux Bootstrap: Locating Linux Kernel and DTB from FIT image.

Linux Bootstrap: Kernel image is compressed. Starting decompression
process. It may take a while...

Linux Bootstrap: Linux image decompression complete.

Linux Bootstrap: Linux kernel image has been loaded into memory.

Linux Bootstrap: Loaded DTB.

Linux Bootstrap: Booting Linux.
Booting Linux on physical CPU 0x0
Linux version 3.8.11 (gcc version 4.7.3 (Sourcery CodeBench Lite 2013.05-
40) ) #99 PREEMPT Fri May 2 15:40:46 PKT 2014
CPU: ARMv7 Processor [413fc090] revision 0 (ARMv7), cr=18c5387d
CPU: PIPT / VIPT nonaliasing data cache, VIPT aliasing instruction cache
Machine: Xilinx Zynq Platform, model: .
Memory policy: ECC disabled, Data cache writeback
...
...
...

/etc/udhcpc.d/50default: Adding DNS 137.202.187.16
/etc/udhcpc.d/50default: Adding DNS 137.202.23.16
/etc/udhcpc.d/50default: Adding DNS 147.34.2.16
done.
starting Busybox inet Daemon: inetd... done.
Starting uWeb server:
INIT: Entering runlevel: 5
Stopping Bootlog daemon: bootlogd.

  _  _  _  _  _  _  _  _  _  _  _  _  _  _  _  _  _  _  _  _  _  _  _  _
 | |_| | | |_| | | |_| | | |_| | | |_| | | |_| | | |_| | | |_| | | |_| |
 | |_| | | |_| | | |_| | | |_| | | |_| | | |_| | | |_| | | |_| | | |_| |
 |_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|
 |_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|
 |_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|
 |_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|
 |_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|
 |_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|
 |_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|
 |_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|

PetaLinux v2013.10 (Yocto 1.4) Xilinx-ZC702-2013_3 ttyPS0

Xilinx-ZC702-2013_3 login: root
Password:
login[776]: root login on `ttyPS0'

root@Xilinx-ZC702-2013_3:~# modprobe zynq_rpmsg_driver
zynq_rpmsg_driver 8000000.zynq-rpmsg_driver: virtio device registered
root@Xilinx-ZC702-2013_3:~#
root@Xilinx-ZC702-2013_3:~# modprobe rpmsg_user_dev_driver
virtio_rpmsg_bus virtio0: rpmsg host is online
virtio_rpmsg_bus virtio0: creating channel rpmsg-openamp-demo-channel addr
0x1
```

```
rpmsg_user_dev_driver rpmsg0: new channel: 0x400 -> 0x1!
root@Xilinx-ZC702-2013_3:~# echo_test

Echo test start

Open rpmsg dev!

Query internal info ..
rpmsg kernel fifo size = 2048
rpmsg kernel fifo avail data = 4096
rpmsg kernel fifo free space = 2048

*****
Please enter command and press enter key
*****
1 - Send data to remote core, retrieve the echo and validate its
integrity ..
2 - Quit this application ..
CMD>1

sending payload number 0 of size 9
received payload number 0 of size 9

sending payload number 2 of size 10
received payload number 2 of size 10

sending payload number 3 of size 11
received payload number 3 of size 11

...
...
...

sending payload number 485 of size 493
received payload number 485 of size 493

sending payload number 486 of size 494
received payload number 486 of size 494

sending payload number 487 of size 495
received payload number 487 of size 495

*****

Test Results: Error count = 0

*****

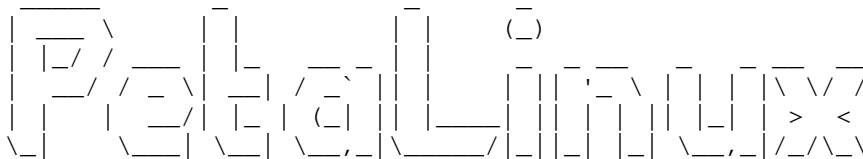
*****
Please enter command and press enter key
*****
1 - Send data to remote core, retrieve the echo and validate its
integrity ..
2 - Quit this application ..
CMD>2
INIT: Sending processes the TERM signal
```

```
Broadcast message from root@Xilinx-ZC702-2013_3 (Thu Jan  1 00:01:23
1970):
The system is going down for system halt NOW!
INIT: Sending processes the KILL signal
Deconfiguring network interfaces... done.
Sending all processes the TERM signal...
Sending all processes the KILL signal...
Unmounting remote filesystems...
Deactivating swap...
Unmounting local filesystems...
System halted.
```

Kernel space

```
*****
OpenAMP Linux Bootstrap.
*****
Linux Bootstrap: Locating Linux Kernel and DTB from FIT image.
Linux Bootstrap: Kernel image is compressed. Starting decompression
process. It may take a while...
Linux Bootstrap: Linux image decompression complete.
Linux Bootstrap: Linux kernel image has been loaded into memory.
Linux Bootstrap: Loaded DTB.
Linux Bootstrap: Booting Linux.

Booting Linux on physical CPU 0x0
Linux version 3.8.11 (gcc version 4.7.3 (Sourcery CodeBench Lite 2013.05-
40) ) #4 PREEMPT Wed May 7 18:46:05 PKT 2014
CPU: ARMv7 Processor [413fc090] revision 0 (ARMv7), cr=18c5387d
CPU: PIPT / VIPT nonaliasing data cache, VIPT aliasing instruction cache
Machine: Xilinx Zynq Platform, model: .
Memory policy: ECC disabled, Data cache writeback
...
...
...
starting Busybox inet Daemon: inetd... done.
Starting uWeb server:
INIT: Entering runlevel: 5
Stopping Bootlog daemon: bootlogd.
```



```
PetaLinux v2013.10 (Yocto 1.4) Xilinx-ZC702-2013_3 ttyPS0
```

```
Xilinx-ZC702-2013_3 login: root
Password:
login[776]: root login on `ttyPS0`
```

```
root@Xilinx-ZC702-2013_3:~# modprobe zynq_rpmsg_driver
zynq_rpmsg_driver 8000000.zynq-rpmsg_driver: virtio device registered
root@Xilinx-ZC702-2013_3:~# modprobe rpmsg_echo_test_kern_app
virtio_rpmsg_bus virtio0: rpmsg host is online
```



```
virtio_rpmsg_bus virtio0: creating channel rpmsg-openamp-demo-channel addr
0x1
rpmsg_echo_test_kern_app: module license 'unspecified' taints kernel.
Disabling lock debugging due to kernel taint

Echo Test Start!

Master : Linux Kernal Space : Sending payload num 0 of size 9
Master : Linux Kernal Space : Received payload num 0 of size 9
Master : Linux Kernal Space : Sending payload num 1 of size 10
Master : Linux Kernal Space : Received payload num 1 of size 10
...
...
...
Master : Linux Kernal Space : Sending payload num 486 of size 495
Master : Linux Kernal Space : Received payload num 486 of size 495
Master : Linux Kernal Space : Sending payload num 487 of size 496
Master : Linux Kernal Space : Received payload num 487 of size 496

*****

Echo Test Results: Error count = 0

*****

Master : Linux Kernal Space : Received payload num -279534246 of size 4
root@Xilinx-ZC702-2013_3:~#
Broadcast message from root@Xilin +Ëëdown for system halt NOW!
INIT: Sending processes the TERM signal
INIT: Sending processes the KILL signal
Deconfiguring network interfaces... done.
Sending all processes the TERM signal...
Sending all processes the KILL signal...
Unmounting remote filesystems...
Deactivating swap...
Unmounting local filesystems...
```

Related Topics

[Executing Applications and Tests for Baremetal Master/Linux Remote Configuration](#)

[Executing the Functional Test Suites](#)

Executing the Functional Test Suites

The following procedure outlines the instructions to execute the functional test suite application.

Note



The target needs to be power cycled after running this application in order to run another application.

Procedure

1. Start the functional test suites application from the U-Boot prompt:

```
U-Boot-PetaLinux> dcache off;fatload mmc 0:0 0x10000000  
func_test_suite.bin; go 0x10000000
```

It starts the baremetal application on the primary core which will on its own boot the Linux kernel on remote core. The Linux kernel will boot and capture the console.

2. Once the Linux kernel prompt is up, enter root login and password:

```
Xilinx-ZC702-2013_3 login: root  
Password:  
login[776]: root login on `ttyPS0'
```

3. Load the Zynq rpmsg driver module

```
root@Xilinx-ZC702-2013_3:~# modprobe zynq_rpmsg_driver
```

4. Load the functional test suite driver using the following command:

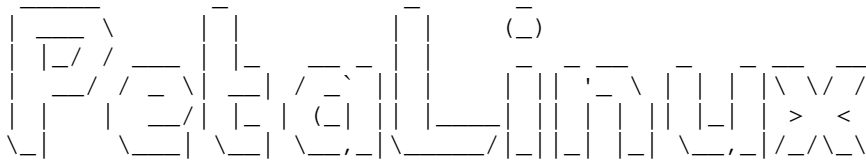
```
root@Xilinx-ZC702-2013_3:~# modprobe rpmsg_func_test_kern_app
```

Once the test suite is completed, Linux will shutdown.

Results

```
## Starting application at 0x10000000 ...  
  
*****  
OpenAMP Linux Bootstrap.  
*****  
  
Linux Bootstrap: Locating Linux Kernel and DTB from FIT image.  
Linux Bootstrap: Kernel image is compressed. Starting decompression  
process. It may take a while...  
Linux Bootstrap: Linux image decompression complete.  
Linux Bootstrap: Linux kernel image has been loaded into memory.  
Linux Bootstrap: Loaded DTB.  
  
Linux Bootstrap: Booting Linux.  
Booting Linux on physical CPU 0x0  
Linux version 3.8.11 (gcc version 4.7.3 (Sourcery CodeBench Lite 2013.05-  
40) ) #99 PREEMPT Fri May 2 15:40:46 PKT 2014  
CPU: ARMv7 Processor [413fc090] revision 0 (ARMv7), cr=18c5387d  
CPU: PIPT / VIPT nonaliasing data cache, VIPT aliasing instruction cache  
Machine: Xilinx Zynq Platform, model: .  
Memory policy: ECC disabled, Data cache writeback  
...
```

```
...
starting Busybox inet Daemon: inetd... done.
Starting uWeb server:
INIT: Entering runlevel: 5
Stopping Bootlog daemon: bootlogd.
```



```
PetaLinux v2013.10 (Yocto 1.4) Xilinx-ZC702-2013_3 ttyPS0
```

```
Xilinx-ZC702-2013_3 login: root
Password:
login[776]: root login on `ttyPS0`
```

```
root@Xilinx-ZC702-2013_3:~# modprobe zynq_rpmsg_driver
zynq_rpmsg_driver 8000000.zynq-rpmsg_driver: virtio device registered
root@Xilinx-ZC702-2013_3:~# modprobe rpmsg_func_test_kern_app
virtio_rpmsg_bus virtio0: rpmsg host is online
virtio_rpmsg_bus virtio0: creating channel rpmsg-openamp-demo-channel addr
0x1
rpmsg_func_test_kern_app: module license 'unspecified' taints kernel.
Disabling lock debugging due to kernel taint
```

```
Func Test Suite Start!
```

```
RPMSG Send Test: Passed
```

```
RPMSG Send Offchannel Test: Passed
```

```
RPMSG Create EPT Test: Passed
```

```
Channel Deletion. Shutdown would be next
root@Xilinx-ZC702-2013_3:~#
Broadcast message from root@Xilinoing down for system halt NOW!
INIT: Sending processes the TERM signal
INIT: Sending processes the KILL signal
Deconfiguring network interfaces... done.
Sending all processes the TERM signal...
Sending all processes the KILL signal...
Unmounting remote filesystems...
Deactivating swap...
Unmounting local filesystems...
System halted.
```

Related Topics

[Executing Applications and Tests for Baremetal Master/Linux Remote Configuration](#)

[Executing the Matrix Multiply Application for Baremetal Master/Linux Remote](#)

Third-Party Information

This software application may include zlib version 1.2.5 third-party software, which is distributed on an "AS IS" basis, WITHOUT WARRANTY OF ANY KIND, either express or implied.

This software application may include libfdt version 17 third-party software, which is distributed on an "AS IS" basis, WITHOUT WARRANTY OF ANY KIND, either express or implied. libfdt version 17 may be subject to the following copyrights:

For the below copyright notice Mentor elects to distribute libfdt under the terms of the BSD license.

© 2006 David Gibson, IBM Corporation.

© 2012 Kim Phillips, Freescale Semiconductor.

libfdt is dual licensed: you can use it either under the terms of the GPL, or the BSD license, at your option.

a) This library is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

This library is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this library; if not, write to the Free Software Foundation, Inc., 51 Franklin St, Fifth Floor, Boston, MA 02110-1301 USA

Alternatively,

b) Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Embedded Software and Hardware License Agreement

The latest version of the Embedded Software and Hardware License Agreement is available on-line at:
www.mentor.com/eshla

IMPORTANT INFORMATION

USE OF ALL PRODUCTS IS SUBJECT TO LICENSE RESTRICTIONS. CAREFULLY READ THIS LICENSE AGREEMENT BEFORE USING THE PRODUCTS. USE OF PRODUCTS INDICATES CUSTOMER'S COMPLETE AND UNCONDITIONAL ACCEPTANCE OF THE TERMS AND CONDITIONS SET FORTH IN THIS AGREEMENT. ANY ADDITIONAL OR DIFFERENT PURCHASE ORDER TERMS AND CONDITIONS SHALL NOT APPLY.

EMBEDDED SOFTWARE AND HARDWARE LICENSE AGREEMENT (“Agreement”)

This is a legal agreement concerning the use of Products (as defined in Section 1) between the company acquiring the Products (“Customer”), and the Mentor Graphics entity that issued the corresponding quotation or, if no quotation was issued, the applicable local Mentor Graphics entity (“Mentor Graphics”). Except for license agreements related to the subject matter of this license agreement which are physically signed by Customer and an authorized representative of Mentor Graphics, this Agreement and the applicable quotation contain the parties' entire understanding relating to the subject matter and supersede all prior or contemporaneous agreements. If Customer does not agree to these terms and conditions, promptly return or, in the case of Products received electronically, certify destruction of Products and all accompanying items within five days after receipt of such Products and receive a full refund of any license fee paid.

1. **Definitions.** As used in this Agreement and any applicable quotation, supplement, attachment and/or addendum (“Addenda”), these terms shall have the following meanings:
 - 1.1. “Customer’s Product” means Customer’s end-user product identified by a unique SKU (including any Related SKUs) in an applicable Addenda that is developed, manufactured, branded and shipped solely by Customer or an authorized manufacturer or subcontractor on behalf of Customer to end-users or consumers;
 - 1.2. “Developer” means a unique user, as identified by a unique user identification number, with access to Embedded Software at an authorized Development Location. A unique user is an individual who works directly with the embedded software in source code form, or creates, modifies or compiles software that ultimately links to the Embedded Software in Object Code form and is embedded into Customer’s Product at the point of manufacture;
 - 1.3. “Development Location” means the location where Products may be used as authorized in the applicable Addenda;
 - 1.4. “Development Tools” means the software that may be used by Customer for building, editing, compiling, debugging or prototyping Customer’s Product;
 - 1.5. “Embedded Software” means Software that is embeddable;
 - 1.6. “End-User” means Customer’s customer;
 - 1.7. “Executable Code” means a compiled program translated into a machine-readable format that can be loaded into memory and run by a certain processor;
 - 1.8. “Hardware” means a physically tangible electro-mechanical system or sub-system and associated documentation;
 - 1.9. “Linkable Object Code” or “Object Code” means linkable code resulting from the translation, processing, or compiling of Source Code by a computer into machine-readable format;
 - 1.10. “Mentor Embedded Linux” or “MEL” means Mentor Graphics' tools, source code, and recipes for building Linux systems;
 - 1.11. “Open Source Software” or “OSS” means software subject to an open source license which requires as a condition for redistribution of such software, including modifications thereto, that the: (i) redistribution be in source code form or be made available in source code form; (ii) redistributed software be licensed to allow the making of derivative works; or (iii) redistribution be at no charge;
 - 1.12. “Processor” means the specific microprocessor to be used with Software and implemented in Customer’s Product;
 - 1.13. “Products” means Software, Term-Licensed Products and/or Hardware;
 - 1.14. “Proprietary Components” means the components of the Products that are owned and/or licensed by Mentor Graphics and are not subject to an Open Source Software license, as more fully set forth in the product documentation provided with the Products;

- 1.15. “Redistributable Components” means those components that are intended to be incorporated or linked into Customer’s Linkable Object Code developed with the Software, as more fully set forth in the documentation provided with the Products;
- 1.16. “Related SKU” means two or more Customer Products identified by logically-related SKUs, where there is no difference or change in the electrical hardware or software content between such Customer Products;
- 1.17. “Software” means software programs, Embedded Software and/or Development Tools, including any updates, modifications, revisions, copies, documentation and design data that are licensed under this Agreement;
- 1.18. “Source Code” means software in a form in which the program logic is readily understandable by a human being;
- 1.19. “Sourcery CodeBench Software” means Mentor Graphics’ Development Tool for C/C++ embedded application development;
- 1.20. “Sourcery VSIPL++” is Software providing C++ classes and functions for writing embedded signal processing applications designed to run on one or more processors;
- 1.21. “Stock Keeping Unit” or “SKU” is a unique number or code used to identify each distinct product, item or service available for purchase;
- 1.22. “Subsidiary” means any corporation more than 50% owned by Customer, excluding Mentor Graphics competitors. Customer agrees to fulfill the obligations of such Subsidiary in the event of default. To the extent Mentor Graphics authorizes any Subsidiary’s use of Products under this Agreement, Customer agrees to ensure such Subsidiary’s compliance with the terms of this Agreement and will be liable for any breach by a Subsidiary; and
- 1.23. “Term-Licensed Products” means Products licensed to Customer for a limited time period (“Term”).

2. Orders, Fees and Payment.

- 2.1. To the extent Customer (or if agreed by Mentor Graphics, Customer’s appointed third party buying agent) places and Mentor Graphics accepts purchase orders pursuant to this Agreement (“Order(s)”), each Order will constitute a contract between Customer and Mentor Graphics, which shall be governed solely and exclusively by the terms and conditions of this Agreement and any applicable Addenda, whether or not these documents are referenced on the Order. Any additional or conflicting terms and conditions appearing on an Order will not be effective unless agreed in writing by an authorized representative of Customer and Mentor Graphics.
- 2.2. Amounts invoiced will be paid, in the currency specified on the applicable invoice, within 30 days from the date of such invoice. All invoices will be sent electronically to Customer on the date stated on the invoice unless otherwise specified in an Addendum. Any past due invoices will be subject to the imposition of interest charges in the amount of one and one-half percent per month or the applicable legal rate currently in effect, whichever is lower. Prices do not include freight, insurance, customs duties, taxes or other similar charges, which Mentor Graphics will state separately in the applicable invoice(s). Unless timely provided with a valid certificate of exemption or other evidence that items are not taxable, Mentor Graphics will invoice Customer for all applicable taxes including, but not limited to, VAT, GST, sales tax, consumption tax and service tax. Customer will make all payments free and clear of, and without reduction for, any withholding or other taxes; any such taxes imposed on payments by Customer hereunder will be Customer’s sole responsibility. If Customer appoints a third party to place purchase orders and/or make payments on Customer’s behalf, Customer shall be liable for payment under Orders placed by such third party in the event of default.
- 2.3. All Products are delivered FCA factory (Incoterms 2010), freight prepaid and invoiced to Customer, except Software delivered electronically, which shall be deemed delivered when made available to Customer for download. Mentor Graphics’ delivery of Software by electronic means is subject to Customer’s provision of both a primary and an alternate e-mail address.

3. Grant of License.

- 3.1. The Products installed, downloaded, or otherwise acquired by Customer under this Agreement constitute or contain copyrighted, trade secret, proprietary and confidential information of Mentor Graphics or its licensors, who maintain exclusive title to all Software and retain all rights not expressly granted by this Agreement. Mentor Graphics grants to Customer, subject to payment of applicable license fees, a nontransferable, nonexclusive license to use Software as described in the applicable Addenda. The limited licenses granted under the applicable Addenda shall continue until the expiration date of Term-Licensed Products or termination in accordance with Section 12 below, whichever occurs first. **Mentor Graphics does NOT grant Customer any right to (a) sublicense or (b) use Software beyond the scope of this Section without first signing a separate agreement or Addenda with Mentor Graphics for such purpose.**
- 3.2. License Type. The license type shall be identified in the applicable Addenda.
 - 3.2.1. Development License: During the Term, if any, Customer may modify, compile, assemble and convert the applicable Embedded Software Source Code into Linkable Object Code and/or Executable Code form by the number of Developers specified, for the Processor(s), Customer’s Product(s) and at the Development Location(s) identified in the applicable Addenda.

- 3.2.2. End-User Product License: During the Term, if any, and unless otherwise specified in the applicable Addenda, Customer may incorporate or embed an Executable Code version of the Embedded Software into the specified number of copies of Customer's Product(s), using the Processor Unit(s), and at the Development Location(s) identified in the applicable Addenda. Customer may manufacture, brand and distribute such Customer's Product(s) worldwide to its End-Users.
- 3.2.3. Internal Tool License: During the Term, if any, Customer may use the Development Tools solely: (a) for internal business purposes and (b) on the specified number of computer work stations and sites. Development Tools are licensed on a per-seat or floating basis, as specified in the applicable Addenda, and shall not be distributed to others or delivered in Customer's Product(s) unless specifically authorized in an applicable Addenda.
- 3.2.4. Sourcery CodeBench Professional Edition License: During the Term specified in the applicable Addenda, Customer may (a) install and use the Proprietary Components of the Software (i) if the license is a node-locked license, by a single user who uses the Software on up to two machines provided that only one copy of the Software is in use at any one time, or (ii) if the license is a floating license, by the authorized number of concurrent users on one or more machines provided that only the authorized number of copies of the Software are in use at any one time, and (b) distribute the Redistributable Components of the Software in Executable Code form only and only as part of Customer's Object Code developed with the Software that provides substantially different functionality than the Redistributable Component(s) alone.
- 3.2.5. Sourcery CodeBench Standard Edition License: During the Term specified in the applicable Addenda, Customer may (a) install and use the Proprietary Components of the Software by a single user who uses the Software on up to two machines provided that only one copy of the Software is in use at any one time, and (b) distribute the Redistributable Component(s) of the Software in Executable Code form only and only as part of Customer's Object Code developed with the Software that provides substantially different functionality than the Redistributable Component(s) alone.
- 3.2.6. Sourcery CodeBench Personal Edition License: During the Term specified in the applicable Addenda, Customer may (a) install and use the Proprietary Components of the Software by a single user who uses the Software on one machine, and (b) distribute the Redistributable Component(s) of the Software in Executable Code form only and only as part of Customer Object Code developed with the Software that provides substantially different functionality than the Redistributable Component(s) alone.
- 3.2.7. Sourcery CodeBench Academic Edition License: During the Term specified in the applicable Addenda, Customer may (a) install and use the Proprietary Components of the Software for non-commercial, academic purposes only by a single user who uses the Software on one machine, and (b) distribute the Redistributable Component(s) of the Software in Executable Code form only and only as part of Customer Object Code developed with the Software that provides substantially different functionality than the Redistributable Component(s) alone.
- 3.3. Mentor Graphics may from time to time, in its sole discretion, lend Products to Customer. For each loan, Mentor Graphics will identify in writing the quantity and description of Software loaned, the authorized location and the Term of the loan. Mentor Graphics will grant to Customer a temporary license to use the loaned Software solely for Customer's internal evaluation in a non-production environment. Customer shall return to Mentor Graphics or delete and destroy loaned Software on or before the expiration of the loan Term. Customer will sign a certification of such deletion or destruction if requested by Mentor Graphics.

4. **Beta Code.**

- 4.1. Portions or all of certain Products may contain code for experimental testing and evaluation ("Beta Code"), which may not be used without Mentor Graphics' explicit authorization. Upon Mentor Graphics' authorization, Mentor Graphics grants to Customer a temporary, nontransferable, nonexclusive license for experimental use to test and evaluate the Beta Code without charge for a limited period of time specified by Mentor Graphics. This grant and Customer's use of the Beta Code shall not be construed as marketing or offering to sell a license to the Beta Code, which Mentor Graphics may choose not to release commercially in any form.
- 4.2. If Mentor Graphics authorizes Customer to use the Beta Code, Customer agrees to evaluate and test the Beta Code under normal conditions as directed by Mentor Graphics. Customer will contact Mentor Graphics periodically during Customer's use of the Beta Code to discuss any malfunctions or suggested improvements. Upon completion of Customer's evaluation and testing, Customer will send to Mentor Graphics a written evaluation of the Beta Code, including its strengths, weaknesses and recommended improvements.
- 4.3. Customer agrees to maintain Beta Code in confidence and shall restrict access to the Beta Code, including the methods and concepts utilized therein, solely to those employees and Customer location(s) authorized by Mentor Graphics to perform beta testing. Customer agrees that any written evaluations and all inventions, product improvements, modifications or developments that Mentor Graphics conceived or made during or subsequent to this Agreement, including those based partly or wholly on Customer's feedback, will be the exclusive property of Mentor Graphics. Mentor Graphics will have exclusive rights, title and interest in all such property. The provisions of this Subsection 4.3 shall survive termination of this Agreement.

5. **Restrictions on Use.**

- 5.1. Customer may copy Software only as reasonably necessary to support the authorized use, including archival and backup purposes. Each copy must include all notices and legends embedded in Software and affixed to its medium and container as received from Mentor Graphics. All copies shall remain the property of Mentor Graphics or its licensors. Except where embedded in Executable Code form in Customer's Product, Customer shall maintain a record of the number and location of all copies of Software, including copies merged with other software and products, and shall make those records available to Mentor Graphics upon request. Customer shall not make Products available in any form to any person other than Customer's employees, authorized manufacturers or authorized contractors, excluding Mentor Graphics competitors, whose job performance requires access and who are under obligations of confidentiality. Customer shall take appropriate action to protect the confidentiality of Products and ensure that any person permitted access does not disclose or use Products except as permitted by this Agreement. Customer shall give Mentor Graphics immediate written notice of any unauthorized disclosure or use of the Products as soon as Customer learns or becomes aware of such unauthorized disclosure or use.
 - 5.2. Customer acknowledges that the Products provided hereunder may contain Source Code which is proprietary and its confidentiality is of the highest importance and value to Mentor Graphics. Customer acknowledges that Mentor Graphics may be seriously harmed if such Source Code is disclosed in violation of this Agreement. Except as otherwise permitted for purposes of interoperability as specified by applicable and mandatory local law, Customer shall not reverse-assemble, reverse-compile, reverse-engineer or in any way derive any Source Code from Products that are not provided in Source Code form. Except as embedded in Executable Code in Customer's Product and distributed in the ordinary course of business, in no event shall Customer provide Products to Mentor Graphics competitors. Log files, data files, rule files and script files generated by or for the Software (collectively "Files") constitute and/or include confidential information of Mentor Graphics. Customer may share Files with third parties, excluding Mentor Graphics competitors, provided that the confidentiality of such Files is protected by written agreement at least as well as Customer protects other information of a similar nature or importance, but in any case with at least reasonable care. Under no circumstances shall Customer use Products or allow their use for the purpose of developing, enhancing or marketing any product that is in any way competitive with Products, or disclose to any third party the results of, or information pertaining to, any benchmark.
 - 5.3. Customer may not assign this Agreement or the rights and duties under it, or relocate, sublicense or otherwise transfer the Products, whether by operation of law or otherwise ("Attempted Transfer"), without Mentor Graphics' prior written consent, which shall not be unreasonably withheld, and payment of Mentor Graphics' then-current applicable relocation and/or transfer fees. Any Attempted Transfer without Mentor Graphics' prior written consent shall be a material breach of this Agreement and may, at Mentor Graphics' option, result in the immediate termination of the Agreement and/or the licenses granted under this Agreement. The terms of this Agreement, including without limitation the licensing and assignment provisions, shall be binding upon Customer's permitted successors in interest and assigns.
 - 5.4. Notwithstanding any provision in an OSS license agreement applicable to a component of the Sourcery CodeBench Software that permits the redistribution of such component to a third party in Source Code or binary form, Customer may not use any Mentor Graphics trademark, whether registered or unregistered, in connection with such distribution, and may not recompile the Open Source Software components with the --with-pkgversion or --with-bugurl configuration options that embed Mentor Graphics' trademarks in the resulting binary.
 - 5.5. The provisions of this Section 5 shall survive the termination of this Agreement.
6. **Support Services.**
- 6.1. Except as described in Sections 6.2, 6.3 and 6.4 below, and unless otherwise specified in any applicable Addenda to this Agreement, to the extent Customer purchases support services, Mentor Graphics will provide Customer updates and technical support for the number of Developers at the Development Location(s) for which support is purchased in accordance with Mentor Graphics' then-current End-User Software Support Terms located at <http://supportnet.mentor.com/about/legal/>.
 - 6.2. To the extent Customer purchases support services for Sourcery CodeBench Software, support will be provided solely in accordance with the provisions of this Section 6.2. Mentor Graphics shall provide updates and technical support to Customer as described herein only on the condition that Customer uses the Executable Code form of the Sourcery CodeBench Software for internal use only and/or distributes the Redistributable Components in Executable Code form only (except as provided in a separate redistribution agreement with Mentor Graphics or as required by the applicable Open Source license). Any other distribution by Customer of the Sourcery CodeBench Software (or any component thereof) in any form, including distribution permitted by the applicable Open Source license, shall automatically terminate any remaining support term. Subject to the foregoing and the payment of support fees, Mentor Graphics will provide Customer updates and technical support for the number of Developers at the Development Location(s) for which support is purchased in accordance with Mentor Graphics' then-current Sourcery CodeBench Software Support Terms located at <http://www.mentor.com/codebench-support-legal>.
 - 6.3. To the extent Customer purchases support services for Sourcery VSIPL++, Mentor Graphics will provide Customer updates and technical support for the number of Developers at the Development Location(s) for which support is purchased solely in accordance with Mentor Graphics' then-current Sourcery VSIPL++ Support Terms located at <http://www.mentor.com/vsipl-support-legal>.
 - 6.4. To the extent Customer purchases support services for Mentor Embedded Linux, Mentor Graphics will provide Customer updates and technical support for the number of Developers at the Development Location(s) for which support is purchased solely in accordance with Mentor Graphics' then-current Mentor Embedded Linux Support Terms located at <http://www.mentor.com/mel-support-legal>.

7. **Third Party and Open Source Software.** Products may contain Open Source Software or code distributed under a proprietary third party license agreement. Please see applicable Products documentation, including but not limited to license notice files, header files or source code for further details. Please see the applicable Open Source Software license(s) for additional rights and obligations governing your use and distribution of Open Source Software. Customer agrees that it shall not subject any Product provided by Mentor Graphics under this Agreement to any Open Source Software license that does not otherwise apply to such Product. In the event of conflict between the terms of this Agreement, any Addenda and an applicable OSS or proprietary third party agreement, the OSS or proprietary third party agreement will control solely with respect to the OSS or proprietary third party software component. The provisions of this Section 7 shall survive the termination of this Agreement.
8. **Limited Warranty.**
 - 8.1. Mentor Graphics warrants that during the warranty period its standard, generally supported Products, when properly installed, will substantially conform to the functional specifications set forth in the applicable user manual and/or specification. Mentor Graphics does not warrant that Products will meet Customer's requirements or that operation of Products will be uninterrupted or error free. The warranty period is 90 days starting on the 15th day after delivery or upon installation, whichever first occurs. Customer must notify Mentor Graphics in writing of any nonconformity within the warranty period. For the avoidance of doubt, this warranty applies only to the initial shipment of Products under an Order and does not renew or reset, for example, with the delivery of (a) Software updates or (b) authorization codes. This warranty shall not be valid if Products have been subject to misuse, unauthorized modification or improper installation. MENTOR GRAPHICS' ENTIRE LIABILITY AND CUSTOMER'S EXCLUSIVE REMEDY SHALL BE, AT MENTOR GRAPHICS' OPTION, EITHER (A) REFUND OF THE PRICE PAID UPON RETURN OF THE PRODUCTS TO MENTOR GRAPHICS OR (B) MODIFICATION OR REPLACEMENT OF THE PRODUCTS THAT DO NOT MEET THIS LIMITED WARRANTY, PROVIDED CUSTOMER HAS OTHERWISE COMPLIED WITH THIS AGREEMENT. MENTOR GRAPHICS MAKES NO WARRANTIES WITH RESPECT TO: (A) SERVICES; OR (B) PRODUCTS PROVIDED AT NO CHARGE, WHICH ARE PROVIDED "AS IS" UNLESS OTHERWISE AGREED IN WRITING.
 - 8.2. THE WARRANTIES SET FORTH IN THIS SECTION 8 ARE EXCLUSIVE TO CUSTOMER AND DO NOT APPLY TO ANY END-USER. NEITHER MENTOR GRAPHICS NOR ITS LICENSORS MAKE ANY OTHER WARRANTIES, EXPRESS, IMPLIED, OR STATUTORY, WITH RESPECT TO PRODUCTS OR OTHER MATERIAL PROVIDED UNDER THIS AGREEMENT. MENTOR GRAPHICS AND ITS LICENSORS SPECIFICALLY DISCLAIM ALL IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NON-INFRINGEMENT OF INTELLECTUAL PROPERTY.
9. **LIMITATION OF LIABILITY.** EXCEPT WHERE THIS EXCLUSION OR RESTRICTION OF LIABILITY WOULD BE VOID OR INEFFECTIVE UNDER APPLICABLE LAW, AND EXCEPT FOR EITHER PARTY'S BREACH OF ITS CONFIDENTIALITY OBLIGATIONS, CUSTOMER'S BREACH OF LICENSING TERMS OR CUSTOMER'S OBLIGATIONS UNDER SECTION 10, IN NO EVENT SHALL: (A) EITHER PARTY OR ITS RESPECTIVE LICENSORS BE LIABLE FOR INDIRECT, SPECIAL, INCIDENTAL, OR CONSEQUENTIAL DAMAGES (INCLUDING LOST PROFITS OR SAVINGS) WHETHER BASED ON CONTRACT, TORT OR ANY OTHER LEGAL THEORY, EVEN IF SUCH PARTY OR ITS LICENSORS HAVE BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES; AND (B) EITHER PARTY OR ITS RESPECTIVE LICENSORS' LIABILITY UNDER THIS AGREEMENT, INCLUDING, FOR THE AVOIDANCE OF DOUBT, LIABILITY FOR ATTORNEYS' FEES OR COSTS, EXCEED THE GREATER OF THE FEES PAID OR OWING TO MENTOR GRAPHICS FOR THE PRODUCT OR SERVICE GIVING RISE TO THE CLAIM OR \$500,000 (FIVE HUNDRED THOUSAND U.S. DOLLARS). NOTWITHSTANDING THE FOREGOING, IN THE CASE WHERE NO AMOUNT WAS PAID, MENTOR GRAPHICS AND ITS LICENSORS SHALL HAVE NO LIABILITY FOR ANY DAMAGES WHATSOEVER. THE PROVISIONS OF THIS SECTION 9 SHALL SURVIVE THE TERMINATION OF THIS AGREEMENT.
10. **Hazardous Applications.**
 - 10.1. Customer agrees that Mentor Graphics has no control over Customer's testing or the specific applications and use that Customer will make of Products. Mentor Graphics Products are not specifically designed for use in the operation of nuclear facilities, aircraft navigation or communications systems, air traffic control, life support systems, medical devices or other applications in which the failure of Mentor Graphics Products could lead to death, personal injury, or severe physical or environmental damage ("Hazardous Applications").
 - 10.2. CUSTOMER ACKNOWLEDGES IT IS SOLELY RESPONSIBLE FOR TESTING PRODUCTS USED IN HAZARDOUS APPLICATIONS AND SHALL BE SOLELY LIABLE FOR ANY DAMAGES RESULTING FROM SUCH USE. NEITHER MENTOR GRAPHICS NOR ITS LICENSORS SHALL BE LIABLE FOR ANY DAMAGES RESULTING FROM OR IN CONNECTION WITH THE USE OF PRODUCTS IN ANY HAZARDOUS APPLICATIONS.
 - 10.3. CUSTOMER AGREES TO INDEMNIFY AND HOLD HARMLESS MENTOR GRAPHICS AND ITS LICENSORS FROM ANY CLAIMS, LOSS, COST, DAMAGE, EXPENSE OR LIABILITY, INCLUDING REASONABLE ATTORNEYS' FEES, ARISING OUT OF OR IN CONNECTION WITH THE USE OF PRODUCTS AS DESCRIBED IN SECTION 10.1.
 - 10.4. THE PROVISIONS OF THIS SECTION 10 SHALL SURVIVE THE TERMINATION OF THIS AGREEMENT.
11. **Infringement.**

- 11.1. Mentor Graphics will defend or settle, at its option and expense, any action brought against Customer in the United States, Canada, Japan, or member state of the European Union which alleges that any standard, generally supported Product acquired by Customer hereunder infringes a patent or copyright or misappropriates a trade secret in such jurisdiction. Mentor Graphics will pay any costs and damages finally awarded against Customer that are attributable to the action. Customer understands and agrees that as conditions to Mentor Graphics' obligations under this section Customer must: (a) notify Mentor Graphics promptly in writing of the action; (b) provide Mentor Graphics all reasonable information and assistance to settle or defend the action; and (c) grant Mentor Graphics sole authority and control of the defense or settlement of the action.
 - 11.2. If a claim is made under Subsection 11.1 Mentor Graphics may, at its option and expense, and in addition to its obligations under Section 11.1, either (a) replace or modify the Product so that it becomes noninfringing; or (b) procure for Customer the right to continue using the Product. If Mentor Graphics determines that neither of those alternatives is financially practical or otherwise reasonably available, Mentor Graphics may require the return of the Product and refund to Customer any purchase price or license fee(s) paid.
 - 11.3. Mentor Graphics has no liability to Customer if the claim is based upon: (a) the combination of the Product with any product not furnished by Mentor Graphics, where the Product itself is not infringing; (b) the modification of the Product other than by Mentor Graphics or as directed by Mentor Graphics, where the unmodified Product would not infringe; (c) the use of the infringing Product when Mentor Graphics has provided Customer with a current unaltered release of a non-infringing Product of substantially similar functionality in accordance with Subsection 11.2(a); (d) the use of the Product as part of an infringing process; (e) a product that Customer makes, uses, or sells, where the Product itself is not infringing; (f) any Product provided at no charge; (g) any software provided by Mentor Graphics' licensors who do not provide such indemnification to Mentor Graphics' customers; (h) Open Source Software, except to the extent that the infringement is directly caused by Mentor Graphics' modifications to such Open Source Software; or (i) infringement by Customer that is deemed willful. In the case of (i), Customer shall reimburse Mentor Graphics for its reasonable attorneys' fees and other costs related to the action.
 - 11.4. THIS SECTION 11 IS SUBJECT TO SECTION 9 ABOVE AND STATES: (A) THE ENTIRE LIABILITY OF MENTOR GRAPHICS AND ITS LICENSORS AND (B) CUSTOMER'S SOLE AND EXCLUSIVE REMEDY, WITH RESPECT TO ANY ALLEGED PATENT OR COPYRIGHT INFRINGEMENT OR TRADE SECRET MISAPPROPRIATION BY ANY PRODUCT PROVIDED UNDER THIS AGREEMENT.
12. **Termination and Effect of Termination.** If a Software license was provided for limited term use, such license will automatically terminate at the end of the authorized Term.
- 12.1. Termination for Breach. This Agreement shall remain in effect until terminated in accordance with its terms. Mentor Graphics may terminate this Agreement and/or any licenses granted under this Agreement, and Customer will immediately discontinue use and distribution of Products, if Customer (a) commits any material breach of any provision of this Agreement and fails to cure such breach upon 30-days prior written notice; or (b) becomes insolvent, files a bankruptcy petition, institutes proceedings for liquidation or winding up or enters into an agreement to assign its assets for the benefit of creditors. Termination of this Agreement or any license granted hereunder will not affect Customer's obligation to pay for Products shipped or licenses granted prior to the termination, which amounts shall be payable immediately upon the date of termination. For the avoidance of doubt, nothing in this Section 12 shall be construed to prevent Mentor Graphics from seeking immediate injunctive relief in the event of any threatened or actual breach of Customer's obligations hereunder.
 - 12.2. Effect of Termination. Upon termination of this Agreement, the rights and obligations of the parties shall cease except as expressly set forth in this Agreement. Upon termination or expiration of the Term, Customer will discontinue use and/or distribution of Products, and shall return Hardware and either return to Mentor Graphics or destroy Software in Customer's possession, including all copies and documentation, and certify in writing to Mentor Graphics within ten business days of the termination date that Customer no longer possesses any of the affected Products or copies of Software in any form, except to the extent an Open Source Software license conflicts with this Section 12.2 and permits Customer's continued use of any Open Source Software portion or component of a Product. Upon termination for Customer's breach, an End-User may continue its use and/or distribution of Customer's Product so long as: (a) the End-User was licensed according to the terms of this Agreement, if applicable to such End-User, and (b) such End-User is not in breach of its agreement, if applicable, nor a party to Customer's breach.
13. **Export.** The Products provided hereunder are subject to regulation by local laws and United States government agencies, which prohibit export or diversion of certain products, information about the products, and direct or indirect products thereof, to certain countries and certain persons. Customer agrees that it will not export Products in any manner without first obtaining all necessary approval from appropriate local and United States government agencies. Customer acknowledges that the regulation of product export is in continuous modification by local governments and/or the United States Congress and administrative agencies. Customer agrees to complete all documents and to meet all requirements arising out of such modifications.
14. **U.S. Government License Rights.** Software was developed entirely at private expense. All Software is commercial computer software within the meaning of the applicable acquisition regulations. Accordingly, pursuant to US FAR 48 CFR 12.212 and DFAR 48 CFR 227.7202, use, duplication and disclosure of the Software by or for the U.S. Government or a U.S. Government subcontractor is subject solely to the terms and conditions set forth in this Agreement, except for provisions which are contrary to applicable mandatory federal laws.
15. **Third Party Beneficiary.** For any Products licensed under this Agreement and provided by Customer to End-Users, Mentor Graphics or the applicable licensor is a third party beneficiary of the agreement between Customer and End-User. Mentor

Graphics Corporation, Mentor Graphics (Ireland) Limited, and other licensors may be third party beneficiaries of this Agreement with the right to enforce the obligations set forth herein.

16. **Review of License Usage.** Customer will monitor the access to and use of Software. With prior written notice, during Customer's normal business hours, and no more frequently than once per calendar year, Mentor Graphics may engage an internationally recognized accounting firm to review Customer's software monitoring system, records, accounts and sublicensing documents deemed relevant by the internationally recognized accounting firm to confirm Customer's compliance with the terms of this Agreement or U.S. or other local export laws. Such review may include FlexNet (or successor product) report log files that Customer shall capture and provide at Mentor Graphics' request. Customer shall make records available in electronic format and shall fully cooperate with data gathering to support the license review. Mentor Graphics shall bear the expense of any such review unless a material non-compliance is revealed. Mentor Graphics shall treat as confidential information all Customer information gained as a result of any request or review and shall only use or disclose such information as required by law or to enforce its rights under this Agreement. Such license review shall be at Mentor Graphics' expense unless it reveals a material underpayment of fees of five percent or more, in which case Customer shall reimburse Mentor Graphics for the costs of such license review. Customer shall promptly pay any such fees. If the license review reveals that Customer has made an overpayment, Mentor Graphics has the option to either provide the Customer with a refund or credit the amount overpaid to Customer's next payment. The provisions of this Section 16 shall survive the termination of this Agreement.
17. **Controlling Law, Jurisdiction and Dispute Resolution.** This Agreement shall be governed by and construed under the laws of the State of California, USA, excluding choice of law rules. All disputes arising out of or in relation to this Agreement shall be submitted to the exclusive jurisdiction of the state and federal courts of California, USA. Nothing in this section shall restrict Mentor Graphics' right to bring an action (including for example a motion for injunctive relief) against Customer or its Subsidiary in the jurisdiction where Customer's or its Subsidiary's place of business is located. The United Nations Convention on Contracts for the International Sale of Goods does not apply to this Agreement.
18. **Severability.** If any provision of this Agreement is held by a court of competent jurisdiction to be void, invalid, unenforceable or illegal, such provision shall be severed from this Agreement and the remaining provisions will remain in full force and effect.
19. **Miscellaneous.** This Agreement contains the parties' entire understanding relating to its subject matter and supersedes all prior or contemporaneous agreements, including but not limited to any purchase order terms and conditions. This Agreement may only be modified in writing, signed by an authorized representative of each party. Waiver of terms or excuse of breach must be in writing and shall not constitute subsequent consent, waiver or excuse.