DM646x Linux Performance Test Bench

User Guide

Literature Number: SPRUFQ9
JULY 2008

IMPORTANT NOTICE

Texas Instruments Incorporated and its subsidiaries (TI) reserve the right to make corrections, modifications, enhancements, improvements, and other changes to its products and services at any time and to discontinue any product or service without notice. Customers should obtain the latest relevant information before placing orders and should verify that such information is current and complete. All products are sold subject to TI's terms and conditions of sale supplied at the time of order acknowledgment.

TI warrants performance of its hardware products to the specifications applicable at the time of sale in accordance with TI's standard warranty. Testing and other quality control techniques are used to the extent TI deems necessary to support this warranty. Except where mandated by government requirements, testing of all parameters of each product is not necessarily performed.

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

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

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

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

TI products are not authorized for use in safety-critical applications (such as life support) where a failure of the TI product would reasonably be expected to cause severe personal injury or death, unless officers of the parties have executed an agreement specifically governing such use. Buyers represent that they have all necessary expertise in the safety and regulatory ramifications of their applications, and acknowledge and agree that they are solely responsible for all legal, regulatory and safety-related requirements concerning their products and any use of TI products in such safety-critical applications, notwithstanding any applications-related information or support that may be provided by TI. Further, Buyers must fully indemnify TI and its representatives against any damages arising out of the use of TI products in such safety-critical applications.

TI products are neither designed nor intended for use in military/aerospace applications or environments unless the TI products are specifically designated by TI as military-grade or "enhanced plastic." Only products designated by TI as military-grade meet military specifications. Buyers acknowledge and agree that any such use of TI products which TI has not designated as military-grade is solely at the Buyer's risk, and that they are solely responsible for compliance with all legal and regulatory requirements in connection with such use.

TI products are neither designed nor intended for use in automotive applications or environments unless the specific TI products are designated by TI as compliant with ISO/TS 16949 requirements. Buyers acknowledge and agree that, if they use any non-designated products in automotive applications, TI will not be responsible for any failure to meet such requirements.

Following are URLs where you can obtain information on other Texas Instruments products and application solutions:

Products		**Applications**	
Amplifiers	amplifier.ti.com	Audio	www.ti.com/audio
Data Converters	dataconverter.ti.com	Automotive	www.ti.com/automotive
DSP	dsp.ti.com	Broadband	www.ti.com/broadband
Clocks and Timers	www.ti.com/clocks	Digital Control	www.ti.com/digitalcontrol
Interface	interface.ti.com	Medical	www.ti.com/medical
Logic	logic.ti.com	Military	www.ti.com/military
Power Mgmt	power.ti.com	Optical Networking	www.ti.com/opticalnetwork
Microcontrollers	microcontroller.ti.com	Security	www.ti.com/security
RFID	www.ti-rfid.com	Telephony	www.ti.com/telephony
RF/IF and ZigBee® Solutions	www.ti.com/lprf	Video & Imaging	www.ti.com/video
		Wireless	www.ti.com/wireless

Mailing Address: Texas Instruments, Post Office Box 655303, Dallas, Texas 75265

Read This First

About This Manual

This document describes how to install and work with Linux Performance Test Bench for DM646x platform for Linux 2.6.10. The pspTest Package serves to provide application software for performance measurement of Linux Device Drivers. This abstracts the functionality provided by the LSP Package. The product forms the basis for measurement of performance on this platform

This release is based on REL_LSP_01_03_00_070. The Package has been compiled with MV tool chain based on respective EVM LSP release notes.

Intended Audience

This document is intended for the users of LSP package who might want to measure/verify the performance of various device drivers using the Linux Performance Test Bench. This would facilitate the user to identify / rectify performance bottlenecks in their respective systems design.

This document assumes that the user has hands on experience with the Linux platform and some knowledge regarding the LSP device drivers for which performance and CPU load parameters are being measured.

How to Use This Manual

This document includes the following chapters:

- **Chapter 1 – Introduction**, gives the brief introduction about the pspTest tool.
- **Chapter 2 - Installation**, describes the installation procedure for LSP package and pspTest Tool.
- **Chapter 3 - Build**, describes the build procedure for U-Boot, Linux kernel, and pspTest Tool.
- **Chapter 4 - Using pspTest Application**, describes the test setup details, procedure to be followed for running the scripts that are provided as part of pspTest tool, and executing the pspTest through command line.
- **Appendix A - General Setup Details**, provides information about the EVM setting details and output/input console setting details.

- **Appendix B- Adding New Test Case**, provides details on how a test case can be added to PspTest tool.

Before you proceed with the installation, see performance_test_bench_releasenotes.pdf file available in the release package.

Terms and Abbreviations

The following terms and abbreviations are used in this document.

Term/Abbreviation	Description
DUT	Device Under Test
fps	Frames Per Second
TV	Television

If You Need Assistance

For any assistance, send a mail to dsppsp_val@list.ti.com

Trademarks

Code Composer Studio, the DAVINCI Logo, DAVINCI, DSP/BIOS, eXpressDSP, TMS320, TMS320C64x, TMS320C6000, TMS320DM646x, and TMS320C64x+ are trademarks of Texas Instruments.

All trademarks are the property of their respective owners.

Contents

Figures

Tables

Revision History

Date	Author	Comments	Version
March 28, 2008	Surendra Puduru	Initial Draft	0.1.0
March 28, 2008	Surendra Puduru	Updated the sample logs as per the modifications done to pspTest code	0.1.1
May 28, 2008	Som	Updated the user guide for naming conventions	0.1.2
May 29, 2008	Prachi	Updated the user guide video changes	0.2.0
June 27, 2008	Surendra Puduru	Updated for CPU Load measurement	0.2.1

Chapter 1

Introduction

This chapter describes the services, features, limitations and requirements of the Linux Performance Test Bench.

1.1 Overview

Linux Performance Test Bench supports benchmarking of various Linux device drivers supplied as part of the Linux Support Packages (LSP) for TI platforms. The current package support throughput and CPU load measurements for the device driver IO operations. This product can be scaled up to add support for new drivers, new platforms, and additional performance parameters.

1.1.1 Supported Services

Linux Performance Test Bench provides the code to get performance and CPU load parameters for the following device drivers:

- VPIF
- VDCE
- Audio
- EDMA
- NAND
- ATA
- I2C
- SPI
- VLYNQ
- USB ISO Video
- USB ISO Audio
- USB MSC Host

1.1.2 Supported Features

Following are the supported features:

- Linux Performance Test Bench supports throughput measurement for both User Level and Kernel Level device drivers.
- Linux Performance Test Bench supports CPU load measurement for User Level device drivers.
- Using the scripts available in the package for all the device drivers, throughput can be measured with minimal manual effort.
- Using the command line, user can get throughput of all the user level device drivers for various input parameters like different buffer sizes, sampling rates etc.
- Common methods are used for buffer allocation, time measurement for performance calculations.

1.2 Limitations

Following are the limitations:

- Boundary checking for Input parameters given through command line is not taken care. So user should give the input parameters accordingly.
- CPU load measurement for kernel level modules and memory requirements while measuring throughput will be implemented in later phase.
- Directory structure is prone to changes when adding support for new platforms.

1.3 Basic Hardware and Software Requirements

1.3.1 Hardware Requirements

The Hardware required for using the Linux Performance Test Bench is

- DM646x EVM with 5V, 5A Power Supply
- XDS510 USB JTAG Emulator for flashing the U-Boot
- UART and Ethernet Cables

The specific hardware requirements for individual module throughput measurement have been mentioned in Chapter 4.

1.3.2 Software Requirements

Linux Support Package for TI Platforms which includes

- Device drivers required for DM646x
- Source for U-Boot

Chapter 2

Installation

This chapter describes the installation procedure for LSP package and pspTest Tool.

2.1 Release Access

See performance_test_bench_releasenotes.pdf file in the release package for release access details.

2.2 System Requirements

See DM646x_Linux_PSP_UserGuide.pdf file available in LSP Package for system required.

2.3 Installation

The following points provide information on installation:

- See DM646x_Linux_PSP_UserGuide.pdf available in LSP Package for installation of DM646x EVM LSP and Flashing of U-Boot.
- Install linux_performance_test_bench_2.1.0.tar.gz by unzipping the package using the command `tar –xvzf linux_performance_test_bench_2.1.0.tar.gz` in Linux Operating System.

Chapter 3

Build

This chapter describes the build procedure for U-Boot, Linux kernel, and pspTest Tool.

Topic **Page**

3.1 Compiling U-Boot

See DM646x_Linux_PSP_UserGuide.pdf available in LSP Package for Compiling U-Boot.

3.2 Compiling Linux Kernel

See DM646x_Linux_PSP_UserGuide.pdf available in LSP Package for Compiling Linux Kernel.

3.3 Compiling pspTest

1. Export the PSP_TEST_HOME to psp_test_bench directory of the performance test bench installation
2. Refer to the PSP_TEST_HOME/README.txt and PSP_TEST_HOME/HowtoConfigure.txt for configuration and build details
3. Following variables in GENDEFS file needs to be updated:
   a. TOOL_CHAIN- Defines the installation directory of MontaVista Toolchain. By default, this tool chain is assumed to be under /opt. If this installation is in a different location, then this variable needs to be changed.
   b. INSTALL_DIR - Defines the directory where the target binaries and utilities need to be copied. By default, this will refer to TOOL_CHAIN/target/pspTestTarget. Modify this variable to install the pspTest target binaries and utilities at different location.
   c. RELEASE – Defines the LSP installation directory. Modify this variable to point to the location of LSP, which has been configured and built for DM646x.
   d. KERNEL_DIR – Defines the Kernel directory path of the LSP release. Modify this variable to point to the location of LSP, which has been configured and built for DM646x.
   e. CC – GCC compiler name.Modify this variable according to your tool chain.
4. Using the `make` command at //PSP_TEST_HOME/make/target/. This will create the binary executable in //PSP_TEST_HOME/bin directory. The `make` supports the following additional features:
   a. `make clean` - This deletes the pspTest executable and all other object files. It also deletes the INSTALL_DIR/pspTestTarget folder.
   b. `make install` - Copies the pspTest target binaries and utilities to INSTALL_DIR/pspTestTarget.

Chapter 4

Using pspTest Application

This chapter describes test setup details, procedure for running the scripts that are provided as part of pspTest tool and executing the pspTest through the command line.

Topic **Page**

4.1 VPIF

This section provides the steps to execute the VPIF performance tests using the scripts or command line utility. It also provides the performance parameters, test setup information, test environment, and command line arguments. The VPIF performance applications use the V4L2 (Video for Linux 2) Framework.

4.1.1 V4L2 Application Performance Parameters

Following VPIF performance parameters will be obtained using pspTest tool with the V4L2 framework:

a. Capture frame rate for the input stream in fps (frames per sec)

b. Display frame rate for the output stream in fps (frames per sec)

c. Percentage of CPU load

4.1.2 Test Setup

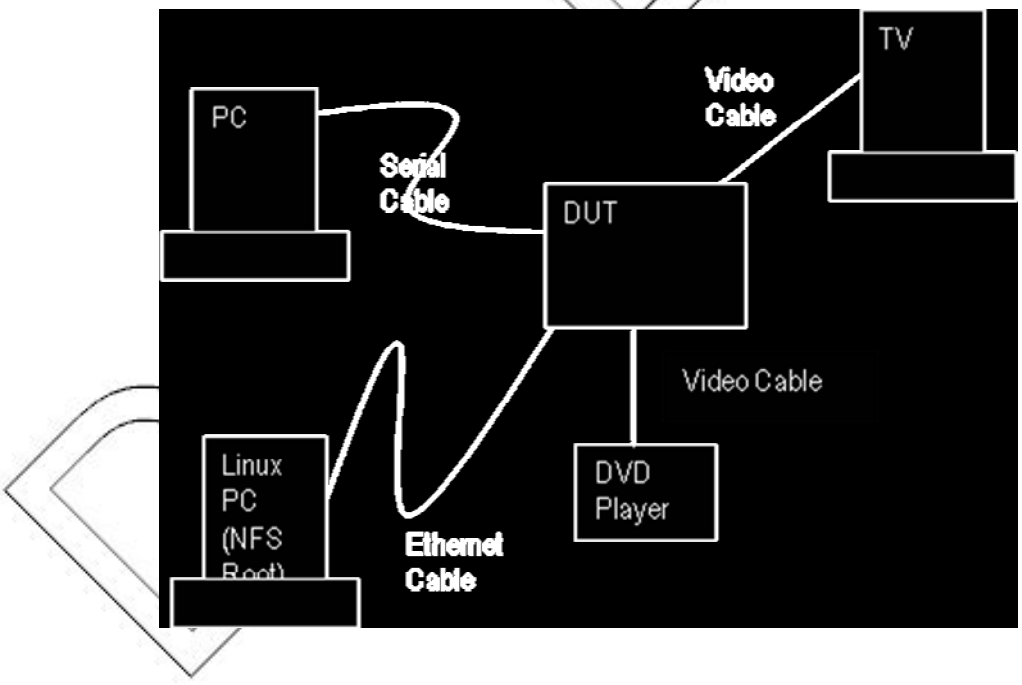

Figure 4-1. VPIF Test Setup

4.1.3 Test Environment

- DUT (Device Under Test), serial console
- DVD Player to give the input stream
- TV to display the output.

4.1.4 Using the V4L2 Capture Application

The V4L2 capture performance measurement application can be run either through the command line or by running the script.

4.1.4.1 V4L2 Capture Using Command Line

This section describes how to run V4L2 capture performance measurement application through the command line.

Go to //./pspTestTarget/bin and run the executable pspTest with the following input parameters:

i. String ThruPut for Throughput performance and Percentage of CPU load.

ii. String `FRv4l2capture` for capture.

iii. Capture device ((for DM646x, can be /dev/video0 in case component/composite interface is used or /dev/video1 in case svideo interface is used).

iv. Number of buffers enqueued (any number from two to eight)

v. Number of frames captured (can be anything less than 10000)

Example: `./pspTest ThruPut FRv4l2capture /dev/video0 4 500`

Note:

To run for more than 10000 frames, change `MAXLOOPCOUNT` in v4l2capture_dm646x.c file located in //./PSP_TEST_HOME/performanceTests/throughput/userlevel/video/v4l2/src directory to the required value.

4.1.4.2 V4L2 Capture Using Script

To use performance measurement application by running the script:

1. See the respective LSP User Guides and flash the EVM. All default settings in EVM should be restored. See section A.1, for the default EVM and the switch settings.
2. In Power switch off mode, connect the DVD Player to the DUT using any one of the interfaces (Composite, S-video, or Component).
3. The input stream can be of the following resolutions/interfaces:
   a. NTSC on Composite interface. Expected fps : 30
   b. PAL on Composite interface. Expected fps : 25
   c. NTSC on Svideo interface. Expected fps : 30
   d. PAL on Svideo interface. Expected fps : 25
   e. 480p on Component interface. Expected fps : 60
   f. 576p on Component interface. Expected fps : 50
   g. 720p@50 on Component interface. Expected fps : 50
   h. 720p@60 on Component interface. Expected fps : 60
   i. 1080i@25 on Component interface. Expected fps : 25
   j. 1080i@30 on Component interface. Expected fps:30

4. Open HyperTerminal/Teraterm (Output/Input Console). Set the required settings for HyperTerminal/TeraTerm (see section A.2). Switch on the power for EVM to boot.

5. See respective LSP User Guides for enabling VPIF, Compiling and Running Linux Kernel.

6. Environment variables needs to be updated as specified in respective LSP User Guides. Press **Enter** to stop booting immediately after bootup prints starts. Use the command `printenv` to see the environment variables and update them if necessary.

7. Setup is now ready to run the capture performance test.

8. In the Output/Input console, run the capture throughput script run_dm646x_v4l2capture_tests.sh available at //./pspTestTarget/scripts/throughput using the command `./run_dm646x_v4l2capture_tests.sh` and press **Enter** on output/input console.

V4L2 Capture Script Details

The run_dm646x_v4l2capture_tests.sh script available at //./pspTestTarget/scripts/throughput will perform the following operations on the target:

1. Performs a capture on /dev/video0 using the component or composite interface for 500 iterations/frames by enqueuing 4 buffers.

2. Running the script provides the frame rate measurements for the capture of the input stream.

4.1.4.3 V4L2 Capture Sample Logs

Following are the log for NTSC:

- v4l2capture_dm646x.c:v4l2capture_perf:554:Running Capture:
- v4l2capture_dm646x.c:v4l2capture_perf:648:Capture frame rate: 30.000051
- v4l2: capture: percentage cpu load: 4.00%

Following are the log for PAL:

- v4l2capture_dm646x.c:v4l2capture_perf:554:Running Capture:
- v4l2capture_dm646x.c:v4l2capture_perf:648:Capture frame rate: 25.051748
- v4l2: capture: percentage cpu load: 4.00%

Following are the log for 480p@60Hz

- v4l2capture_dm646x.c:v4l2capture_perf:554:Running Capture:
- v4l2capture_dm646x.c:v4l2capture_perf:648:Capture frame rate: 59.918156

- v4l2: capture: percentage cpu load: 4.00%

Following are the log for 576p@50Hz:

- v4l2capture_dm646x.c:v4l2capture_perf:554:Running Capture:
- v4l2capture_dm646x.c:v4l2capture_perf:648:Capture frame rate: 50.220013
- v4l2: capture: percentage cpu load: 4.00%

Following are the log for 720p@50Hz:

- v4l2capture_dm646x.c:v4l2capture_perf:554:Running Capture:
- v4l2capture_dm646x.c:v4l2capture_perf:648:Capture frame rate: 50.046112
- v4l2: capture: percentage cpu load: 4.00%

Following are the log for 720p@60Hz:

- v4l2capture_dm646x.c:v4l2capture_perf:554:Running Capture:
- v4l2capture_dm646x.c:v4l2capture_perf:648:Capture frame rate: 60.001610
- v4l2: capture: percentage cpu load: 4.00%

Following are the log for 1080i@50Hz:

- v4l2capture_dm646x.c:v4l2capture_perf:554:Running Capture:
- v4l2capture_dm646x.c:v4l2capture_perf:648:Capture frame rate: 25.028823
- v4l2: capture: percentage cpu load: 4.00%

Following are the log for 1080i@60Hz:

- v4l2capture_dm646x.c:v4l2capture_perf:554:Running Capture:
- v4l2capture_dm646x.c:v4l2capture_perf:648:Capture frame rate: 30.001811
- v4l2: capture: percentage cpu load: 4.00%

4.1.5 Using the V4L2 Display Application

The V4L2 display performance measurement application can be run either through the command line or by running the script.

4.1.5.1 V4L2 Display Using Command Line

This section describes how to run V4L2 display performance measurement application through the command line.

Go to //./pspTestTarget/bin and run the executable pspTest with the following input parameters:

i. String ThruPut for Throughput performance and Percentage of CPU load

ii. String `FRv4l2display` for display.

iii. Display device (for DM646x, can be /dev/video2)

iv. Number of buffers enqueued (any number from three to five)

v. Number of frames to be displayed (can be anything less than 10000)

vi. Display interface (can be COMPOSITE, COMPONENT, SVIDEO)

vii. Display mode (can be NTSC, PAL, 480P-60, 576P-50, 720P-50, 720P-60, 1080I-25, 1080I-30)

Example: `./pspTest ThruPut FRv4l2display /dev/video2 4 500 COMPOSITE NTSC`

Note:

To run for more than 10000 frames, change `MAXLOOPCOUNT` in v4l2display_dm646x.c file located in //./PSP_TEST_HOME/performanceTests/throughput/userlevel/video/v4l2/src directory to the required value.

4.1.5.2 V4L2 Display Using Script

To use display performance measurement application by running the script:

1. See the respective LSP User Guides and flash the EVM. All default settings in EVM should be restored. See section A.1, for the default EVM and the switch settings.
2. In Power switch off mode, connect the TV to the DUT using any one of the interfaces (Composite, S-video, or Component).
3. Open HyperTerminal/Teraterm (Output/Input Console). Set the required settings for HyperTerminal/TeraTerm (see section A.2). Switch on the power for EVM to boot.
4. See respective LSP User Guides for enabling VPIF, Compiling and Running Linux Kernel.
5. Environment variables needs to be updated as specified in respective LSP User Guides. Press **Enter** to stop booting immediately after bootup prints starts. Use the command `printenv` to see the environment variables and update them if necessary.
6. Setup is now ready to run the display performance test.
7. In the Output/Input console, run the capture throughput script run_dm646x_v4l2display_tests.sh available at //./pspTestTarget/scripts/throughput using the command `./run_dm646x_v4l2display_tests.sh` and press **Enter** on output/input console.

V4L2 Display Script Details

The run_dm646x_v4l2display_tests.sh script available at //./pspTestTarget/scripts/throughput will perform the following operations on the target:

1. Performs a display of scrolling color bars on /dev/video2 using the composite interface for 500 iterations/frames in NTSC mode by enqueuing 4 buffers.
2. Running the script provides the frame rate measurements for the display.

4.1.5.3 V4L2 Display Sample Logs

Following are the log for NTSC:

- v4l2display_dm646x.c:v4l2display_perf:554:Running Display:
- v4l2display_dm646x.c:v4l2display_perf:647:Display frame rate: 30.000189
- v4l2: display: percentage cpu load: 4.00%

Following are the log for PAL:

- v4l2display_dm646x.c:v4l2display_perf:554:Running Display:
- v4l2display_dm646x.c:v4l2display_perf:647:Display frame rate: 25.052305
- v4l2: display: percentage cpu load: 4.00%

Following are the log for 480p@60Hz

- v4l2display_dm646x.c:v4l2display_perf:554:Running Display:
- v4l2display_dm646x.c:v4l2display_perf:647:Display frame rate: 59.920175
- v4l2: display: percentage cpu load: 4.00%

Following are the log for 576p@50Hz:

- v4l2display_dm646x.c:v4l2display_perf:554:Running Display:
- v4l2display_dm646x.c:v4l2display_perf:647:Display frame rate: 50.070671
- v4l2: display: percentage cpu load: 4.00%

Following are the log for 720p@50Hz:

- v4l2display_dm646x.c:v4l2display_perf:554:Running Display:
- v4l2display_dm646x.c:v4l2display_perf:647:Display frame rate: 50.047467
- v4l2: display: percentage cpu load: 4.00%

Following are the log for 720p@60Hz:

- v4l2display_dm646x.c:v4l2display_perf:554:Running Display:
- v4l2display_dm646x.c:v4l2display_perf:647:Display frame rate: 60.001877
- v4l2: display: percentage cpu load: 4.00%

Following are the log for 1080i@50Hz:

- v4l2display_dm646x.c:v4l2display_perf:554:Running Display:
- v4l2display_dm646x.c:v4l2display_perf:647:Display frame rate: 25.026109
- v4l2: display: percentage cpu load: 4.00%

Following are the log for 1080i@60Hz:

- v4l2display_dm646x.c:v4l2display_perf:554:Running Display:
- v4l2display_dm646x.c:v4l2display_perf:647:Display frame rate: 30.000077
- v4l2: display: percentage cpu load: 4.00%

4.2 VDCE

This section provides the steps to execute the VDCE performance tests using the scripts or command line utility. It also provides the performance parameters, test setup information, test environment, and command line arguments.

4.2.1 Performance Parameters

Following VDCE performance parameters will be obtained using pspTest tool:

a. Time consumed for resizing operation on standalone YUV files in microseconds and Percentage of CPU load.

b. Time consumed for chrominance conversion operations on standalone YUV files (4:2:2 to 4:2:0 and 4:2:0 to 4:2:2) in microseconds and Percentage of CPU load.

c. Time consumed for blending operation on standalone YUV files in microseconds and Percentage of CPU load.

d. Time consumed for range mapping operation on standalone YUV files in microseconds and Percentage of CPU load.

e. Time consumed for edge padding operation on standalone YUV files in microseconds and Percentage of CPU load.

4.2.2 Test Setup

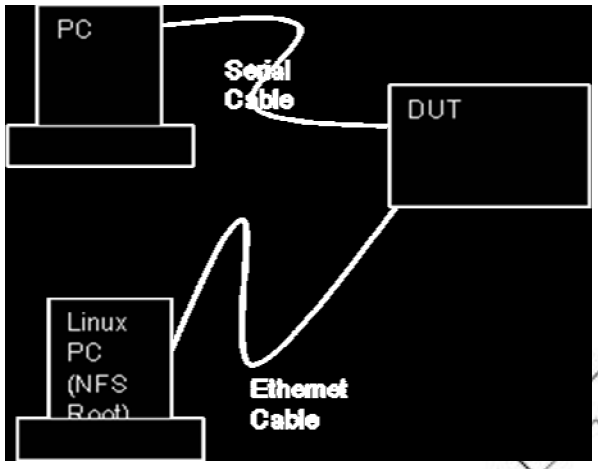

Figure 4-2.VDCE Test Setup

4.2.3 Test Environment

- DUT (Device Under Test), serial console
- YUV images available at target used as input for VDCE operations.

4.2.4 Using the Application

The performance measurement application can be run either through the command line or by running the script.

4.2.4.1 Using Command Line

This section describes how to run performance measurement application through the command line.

Go to //./pspTestTarget/bin and run the executable pspTest with the following input parameters:

1. String ThruPut for Throughput performance and Percentage of CPU load

2. String `FRResize` for performing resize operation.
   i. Device name, that is /dev/DavinciHD_vdce
   ii. input width
   iii. input height
   iv. input file
   v. output width
   vi. output height

   Example: `./pspTest ThruPut FRResize /dev/DavinciHD_vdce 1920 1080 1080i.yuv 720 480`

3. String `FRCCV422to420` for performing chroma conversion operation from YUV422 image to YUV 420 image
   i. Device name, that is /dev/DavinciHD_vdce

ii. input width
iii. input height
iv. input file

Example: `./pspTest ThruPut FRCCV422to420 /dev/DavinciHD_vdce 1920 1080 1080i_422.yuv`

4. String `FRCCV420to422` for performing chroma conversion operation from YUV420 to YUV420.
   i. Device name, that is /dev/DavinciHD_vdce
   ii. input width
   iii. input height
   iv. input file

Example: `./pspTest ThruPut FRCCV420to422 /dev/DavinciHD_vdce 1920 1080 1080i_420.yuv`

5. String `FRblending` for performing blending operation .
   i. Device name, that is /dev/DavinciHD_vdce
   ii. input width
   iii. input height
   iv. input file
   v. bmp image width
   vi. bmp image height
   vii. bmp image pitch
   viii. bmp image horizontal start position
   ix. bmp image vertical start position

Example: `./pspTest ThruPut FRBlending /dev/DavinciHD_vdce 720 480 480p.yuv 256 160 64 120 120`

6. String `FRepad` for performing edge padding operation.
   i. Device name, that is /dev/DavinciHD_vdce
   ii. input width
   iii. input height
   iv. input file

Example: `./pspTest ThruPut FRepad /dev/DavinciHD_vdce 1920 1080 1080i.yuv`

7. String `FRRmap` for performing range mapping operation.
   i. Device name, that is /dev/DavinciHD_vdce
   ii. input width
   iii. input height
   iv. input file

Example: `./pspTest ThruPut FRRmap /dev/DavinciHD_vdce 1920 1080 1080i.yuv`

4.2.4.2 Using Script

To use performance measurement application by running the script:

1. See the respective LSP User Guides and flash the EVM. All default settings (see section A.1) in EVM should be restored.
2. In Power switch off mode, connect the DVD Player and the TV to the DUT using any one of the interfaces (Composite, S-video or Component).
3. The input files can be of the following resolutions:
    a. 1920 x 1080
    b. 1280 x 720
    c. 720 x 480
    d. 720 x 576
    e. For CCV operation, input file should be in YUV 4:2:2 format (4:2:2 to 4:2:0 conversion) or YUV 4:2:0 format (4:2:0 to 4:2:2 conversion).
4. Open HyperTerminal/Teraterm (Output/Input Console). Set the required settings for HyperTerminal/TeraTerm (see section A.2). Switch on the power for EVM to boot.
5. See respective LSP User Guides for enabling VDCE, Compiling and Running Linux Kernel.
6. Environment variables needs to be updated as specified in respective LSP User Guides. Press **Enter** to stop booting immediately after bootup prints starts. Use the command `printenv` to see the environment variables and update them if necessary.
7. Setup is now ready to run the performance test.
8. In the Output/Input console, run the VDCE throughput script run_dm646x_vdce_tests.sh available at //./pspTestTarget/scripts/throughput using the command ./run_dm646x_vdce_tests.sh and press **Enter**.

Script Details

The run_dm646x_vdce_tests.sh script available at //./pspTestTarget/scripts/throughput will perform the following operations on the target:

1. Time taken in microseconds to resize a 1920x1080 image to 720x480 image.
2. Time taken in microseconds to resize a 1920x1080 image to 1280x720 image.
3. Time taken in microseconds to convert a 1920x1080 YUV422 image to 1920x1080 YUV420 image.
4. Time taken in microseconds to convert a 720x480 YUV422 image to 720x480 YUV420 image.

5. Time taken in microseconds to convert a 1920x1080 YUV420 image to 1920x1080 YUV422 image.
6. Time taken in microseconds to convert a 720x480 YUV420 image to 720x480 YUV422 image.
7. Time taken in microseconds to blend a 1920x1080 YUV image with 256x160 bitmap.
8. Time taken in microseconds to blend a 1920x1080 YUV image with 512x160 bitmap.
9. Time taken in microseconds to blend a 720x480 YUV image with 256x160 bitmap.
10. Time taken in microseconds to blend a 720x480 YUV image with 512x160 bitmap.
11. Time taken in microseconds to perform edge padding operation on a 1920x1080 image.
12. Time taken in microseconds to perform edge padding operation on a 720x480 image.
13. Time taken in microseconds to perform range mapping operation on a 1920x1080 image.
14. Time taken in microseconds to perform range mapping operation on a 720x480 image.

4.2.5 Sample Logs

Time for Resize of 1920x1080 image to 720x480 image is 20393 microseconds

vdceResize.c:vdce_resize:207:Resize complete.

vdceResize.c:resize:281:

Time for Resize of 1920x1080 image to 1280x720 image is 29423 microseconds

vdceResize: percentage cpu load for Resize of 1920x1080 image to 1280x720 image is 4.00%

vdceResize.c:vdce_resize:207:Resize complete.

vdce420_422.c:ccv420_422:287:

Time for 1920x1080 image CCV from 420 to 422 is 14115 microseconds

vdce420_422: percentage cpu load for 1920x1080 image CCV from 420 to 422 is 4.00%

vdce420_422.c:vdce_ccv420_422:206:CCV 420 --- 422 complete.

vdce420_422.c:ccv420_422:287:

Time for 720x480 image CCV from 420 to 422 is 2399 microseconds

vdce420_422: percentage cpu load for 720x480 image CCV from 420 to 422 is 4.00%

vdce420_422.c:vdce_ccv420_422:206:CCV 420 --- 422 complete.

vdce422_420.c:ccv422_420:293:CCV from 1920x1080 image 422 to 420 takes 18166 microseconds

vdce422_420: percentage cpu load for 1920x1080 image CCV from 422 to 420 is 4.00%

vdce422_420.c:vdce_ccv422_420:207:CCV 422 --- 420 complete.

vdce422_420.c:ccv422_420:293:CCV from 720x480 image 422 to 420 takes 3087 microseconds

vdce422_420: percentage cpu load for 720x480 image CCV from 422 to 420 is 4.00%

vdce422_420.c:vdce_ccv422_420:207:CCV 422 --- 420 complete.

vdceEpad.c:epad:293:Time for 720x480 Edge padding is 536 microseconds

vdceEpad: percentage cpu load for 720x480 edge padding is 4.00%

vdceEpad.c:vdce_epad:227:Edge Padding complete.

vdceEpad.c:epad:293:Time for 1920x1080 Edege padding is 1148 microseconds

vdceEpad: percentage cpu load for 1920x1080 edge padding is 4.00%

vdceEpad.c:vdce_epad:227:Edge Padding complete.

vdceRmap.c:rmap:275:Time for 720x480 Range mapping is 2903 microseconds

vdceRmap: percentage cpu load for 720x480 range mapping is 4.00%

vdceRmap.c:vdce_rmap:211:Range mapping Completed

vdceRmap.c:rmap:275:Time for 1920x1080 Range mapping is 16798 microseconds

vdceRmap: percentage cpu load for 1920x1080 range mapping is 4.00%

vdceRmap.c:vdce_rmap:211:Range mapping Completed

vdceBlending.c:blending:371:Time for 256x160 bitmap and 720x480 image Blending is 992 microseconds

vdceblending: percentage cpu load for 256x160 bitmap and 720x480 image Blending is 4.00%

vdceBlending.c:vdce_blending:239:Blending complete.

vdceBlending.c:blending:371:Time for 512x160 bitmap and 720x480 image Blending is 1905 microseconds

vdceblending: percentage cpu load for 512x160 bitmap and 720x480 image Blending is 4.00%

vdceBlending.c:vdce_blending:239:Blending complete.

vdceBlending.c:blending:371:Time for 256x160 bitmap and 1920x1080 image Blending is 993 microseconds

vdceblending: percentage cpu load for 256x160 bitmap and 1920x1080 image Blending is 4.00%

vdceBlending.c:vdce_blending:239:Blending complete.

vdceBlending.c:blending:371:Time for 512x160 bitmap and 1920x1080 image Blending is 1908 microseconds

vdceblending: percentage cpu load for 512x160 bitmap and 1920x1080 image Blending is 4.00%

vdceBlending.c:vdce_blending:239:Blending complete.

4.3 Audio

This section discusses the steps to execute the Audio performance tests by scripts and command line utility by providing an overview about the test setup information, command line arguments and the performance parameters.

4.3.1 Performance Parameters

Following Audio performance parameters will be obtained using pspTest tool:

a. Time taken in seconds for read/write of given data size

b. Data rate for read/write in Bytes/sec

c. Percentage of CPU load

4.3.2 Test Setup

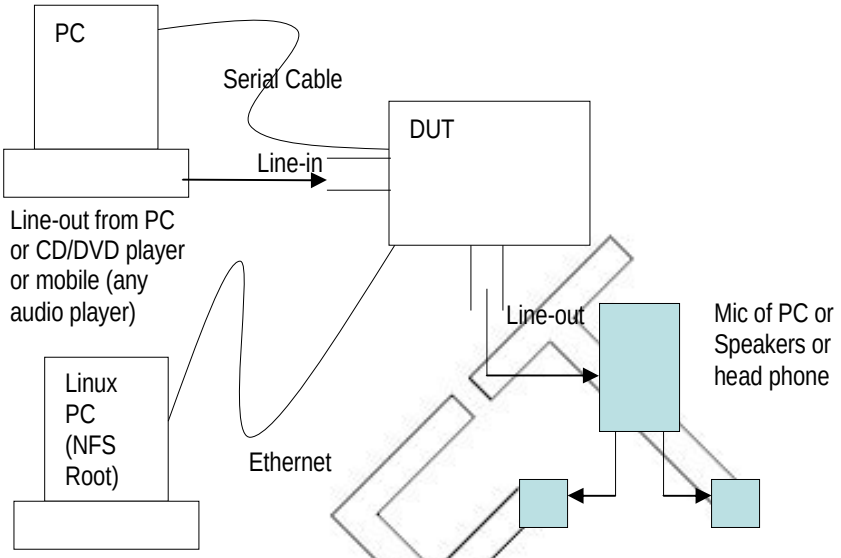

Figure 4-3. Audio Test Setup

4.3.3 Test Environment

- q DUT (Device Under Test), serial console
- q Speaker or Head phone or Microphone port of PC
- q Line-Out of PC or CD/DVD Player or any audio player with coaxial out cable

4.3.4 Using the Application

The performance measurement application can be run either through the command line or by running the script.

4.3.4.1 Using Command Line

The pspTest utility supports following two throughputs for measuring performance of audio.

Audio Throughput

Go to //./pspTestTarget/bin and run the executable pspTest with the following configurable parameters as arguments:

i. String ThruPut for Throughput performance and Percentage of CPU load

ii. String `FRaudiowrite` for write or `FRaudioread` for read

iii. Device node for read or write

iv. Sampling Rate (Hz) for which performance is carried out

v. Application buffer size (Bytes)

vi. Data size (Bytes)

Example: `./pspTest ThruPut  FRaudioread /dev/dsp 8000 4096 5242880`

Audio-File Throughput

Go to //./pspTestTarget/bin and run the executable pspTest with the following configurable parameters as arguments:

i. String ThruPut for Throughput performance

ii. String FRaudiowritefromfile for write or FRaudioreadtofile for read

iii. Absolute path of the file used for writing recorded audio data and reading audio data for playback

iv. Device node for read or write

v. Sampling Rate (Hz) for which performance is carried out

vi. Application buffer size (Bytes)

vii. Data size (Bytes)

Example: `./pspTest ThruPut  FRaudioreadtofile /dev/dsp $PERF_DIR/perf1.txt 8000 4096 5242880`

4.3.4.2 Using Script

To use the performance measurement application by running the script:

1. See the respective LSP User Guides and flash the EVM. All default settings (see section A.1) in EVM should be restored.
2. In Power switch off mode, connect audio input/line-in and output/line-out devices to the EVM. For further details on connecting hardware, refer EVM user manual.
3. Open HyperTerminal/Teraterm (Output/Input Console). Set the required settings for HyperTerminal/TeraTerm (see section A.2). Switch on the power for EVM to boot.
4. See the respective LSP User Guides for enabling Audio, compiling, and running Linux Kernel.
5. Environment variables needs to be updated as specified in respective LSP User Guides. Press **Enter** to stop booting immediately after bootup prints starts. Use the command `printenv` to see the environment variables and update them, if necessary.
6. Setup is now ready to run the performance test.

7. The pspTest utility provides individual scripts for two throughputs for measuring performance of audio. Steps for running the scripts are as follows:

    a. In the output/input console, run the audio throughput run_audio_oss_tests.sh script available at //./pspTestTarget/scripts/throughput using the command ./run_audio_oss_tests.sh and press **Enter**.

    b. In the output/input console, run the audio-file throughput run_audio_oss_filethroughput_tests.sh script available at //./pspTestTarget/scripts/throughput using the command ./run_audio_oss_filethroughput_tests.sh and press **Enter**.

Script Details

The pspTest utility provides individual scripts for two throughputs for measuring performance of audio. Following are the script details for measuring performance of audio with direct throughput and audio to file throughput.

Audio Throughput Script

The run_audio_oss_tests.sh script available at //./pspTestTarget/scripts/throughput will perform the following operations on the target:

1. Performs read/record of data size 5242880 bytes and Application Buffer of size 4096 bytes with sampling rates of various values like 8000, 32000, 44100, and 48000 Hz.

2. Performs write/playback of data size 5242880 bytes and Application Buffer of size 4096 bytes with sampling rates of various values like 8000, 32000, 44100, and 48000 Hz.

Audio-File Throughput Script

The run_audio_oss_filethroughput_tests.sh script available at //./pspTestTarget/scripts/throughput will perform the following operations on the target:

1. Creates a directory /mnt/audio_pspTest on root file system, this is used for writing the recorded audio data to a file and reading the file for audio playback.

2. Performs read/record of data size 5242880 bytes and Application Buffer of size 4096 bytes with sampling rates of various values like 8000, 32000, 44100, and 48000 Hz. The recorded data will be written to a file in /mnt/audio_pspTest directory.

3. Performs write/playback of data size 5242880 bytes and Application Buffer of size 4096 bytes with sampling rates of various values like 8000, 32000, 44100, and 48000 Hz. Audio data used for playback will be read from a file in /mnt/audio_pspTest directory.

4. Deletes the .txt files created while doing read/write.

5. Deletes the directory /mnt/audio_pspTest.

4.3.5 Sample Logs

Following are the logs for read and write:

- audio: read: Word Length in bits: 16
- audio: read: No. of channels per sample: 2
- audio: read: Sampling Rate in Hz: 8000
- audio: read: Duration in sec: 163.888481
- audio: read: No. of bits/sec: 255924
- audio: read: percentage cpu load: 1.3%
- audio: write: Word Length in bits: 16
- audio: write: No. of channels per sample: 2
- audio: write: Sampling Rate in Hz: 8000
- audio: write: Duration in sec: 163.504268
- audio: write: No. of bits/sec: 256526
- audio: write: percentage cpu load: 1.3%

4.4 EDMA

This section discusses the steps to execute the EDMA performance tests by scripts and command line utility by providing an overview about the test setup information, command line arguments and the performance parameters.

4.4.1 Performance Parameters

Following EDMA performance parameters will be obtained using pspTest tool:

a. Time taken for data transfer of 65536 Bytes with different values of A,B, and C Count values in EDMA A/AB sync mode

b. Time taken for data transfer of 65536 Bytes with different values of A,B, and C Count values in QDMA A/AB sync mode

4.4.2 Test Setup

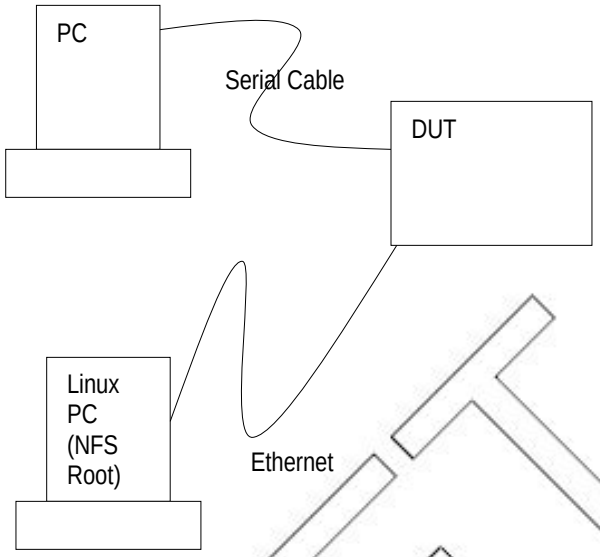

Figure 4-4. EDMA Test Setup

4.4.3 Test Environment

- DUT (Device Under Test), serial console

4.4.4 Using the Application

The performance measurement application can be run either through the command line or by running the script.

4.4.4.1 Using Command Line

This section describes how to run performance measurement application through the command line.

Go to //.\pspTestTarget/bin and,

1. Insert kperfTimer.ko. To do this use the command insmod kperfTimer.ko, You should see Delta 1 , on the teraterm/HyperTerm.
2. Now issue the command insmod edmaAsyncIncr.ko, this will perform the data transfer of 65535 Bytes, with following combinations of A,B,C counts using EDMA channel.

A Count	B Count	C count	Total Size(Bytes)
1024	64	1	65535
4096	16	1	65535
8192	8	1	65535

16384	4	1	65535
32767	2	1	65535
65535	1	1	65535

Table 1 A, B, and C Counts for EDMA A Sync Incremental Mode

3. Displays the time taken to do the transfer.
4. Issue the command `mmod edmaAsyncIncr.ko.`
5. Run the command `insmod edmaABsyncIncr.ko.` This performs the data transfer of 65535 Bytes, with the following combinations of A,B, and C counts using EDMA Channel.

A Count	B Count	C count	Total Size(Bytes)
1024	64	1	65535
4096	16	1	65535
8192	8	1	65535
16384	4	1	65535
32767	2	1	65535
65535	1	1	65535

Table 2 A, B, and C Counts for EDMA AB Sync Incremental Mode

6. Displays the time taken to do the transfer.
7. Issue the command `rmmod edmaABsyncIncr.ko`.
8. Run the command `insmod qdmaAsyncIncr.ko.` This performs the data transfer of 65535 Bytes, with the following combinations of A,B, and C counts using QDMA channels.

A Count	B Count	C count	Total Size(Bytes)
1024	64	1	65535
4096	16	1	65535
8192	8	1	65535
16384	4	1	65535
32767	2	1	65535
65535	1	1	65535

Table 3 A, B, and C Counts for QDMA A Sync Incremental Mode

9. Displays the time taken to do the transfer.

10. Run the command `rmmod qdmaAsyncIncr.ko.`
11. Run the command `insmod qdmaABsyncIncr.ko`. This performs the data transfer of 65535 Bytes, with the following combinations of A,B, and C counts using QDMA channels.

A Count	B Count	C count	Total Size(Bytes)
1024	64	1	65535
4096	16	1	65535
8192	8	1	65535
16384	4	1	65535
32767	2	1	65535
65535	1	1	65535

Table 4 A, B, and C Counts for QDMA AB Sync Incremental mode

12. Displays the time taken to do the transfer.
13. Run the command `rmmod qdmaABsyncIncr.ko`.
14. Finally, run the command `rmmod kperfTimer.ko`.

Example:

```
./bin/ insmod kperfTimer.ko
./bin/ insmod edmaAsyncIncr.ko
/bin/rmmod edmaAsyncIncr.ko
./bin/ insmod edmaABsyncIncr.ko
/bin/rmmod edmaABsyncIncr.ko
./bin/ insmod qdmaAsyncIncr.ko
/bin/rmmod qdmaAsyncIncr.ko
./bin/ insmod qdmaABsyncIncr.ko
/bin/rmmod qdmaABsyncIncr.ko
/bin/rmmod kperfTimer.ko
```

4.4.4.2 Using Script

To use performance measurement application by running the script:

1. See the respective LSP User Guides and flash the EVM. All default settings (see section A.1) in EVM should be restored.
2. Open HyperTerminal/Teraterm (Output/Input Console). Set the required settings for HyperTerminal/TeraTerm (see section A.2). Switch on the power for EVM to boot.

3. See respective LSP User Guides for enabling EDMA device driver, compiling, and running Linux Kernel.

4. Environment variables needs to be updated as specified in respective LSP User Guides. Press **Enter** to stop booting immediately after bootup prints starts. Use the command `printenv` to see the environment variables and update them, if necessary.

5. Setup is now ready to run the performance test.

6. In the Output/Input console, run the EDMA throughput script run_edma_tests.sh available at //./pspTestTarget/scripts/throughput using the command `./run_edma_tests.sh` and press **Enter**.

Script Details

The run_edma_tests.sh script available at //./pspTestTarget/scripts/throughput will perform the following operations on the target:

1. Inserts kperfTimer.ko present in at //./pspTestTarget/bin directory

2. Inserts the Kernel object, edmaAsyncIncr.ko present in //./pspTestTarget/bin directory.

3. Displays the performance figures by EDMA, Async incremental mode to transfer 65535 bytes with different A, B, and C counts.

4. Removes the Kernel object, edmaAsyncIncr.ko.

5. Inserts the Kernel object, edmaABsyncIncr.ko present in //./pspTestTarget/bin directory.

6. Displays the Performance figures by EDMA, ABsync Incremental mode to transfer 65535 bytes with different A, B, and C counts.

7. Removes the Kernel Object, edmaABsyncIncr.ko.

8. Inserts the Kernel object, qdmaAsyncIncr.ko present in //./pspTestTarget/bin directory.

9. Displays the Performance figures by QDMA, Async incremental mode to transfer 65535 bytes with different A, B, and C counts.

10. Removes the Kernel object, qdmaAsyncIncr.ko.

11. Inserts the Kernel object, qdmaABsyncIncr.ko present in //./pspTestTarget/bin directory.

12. Displays the Performance figures by QDMA, ABsync incremental mode to transfer 65535 bytes with different A, B, and C counts.

13. Removes the Kernel object, edmaABsyncIncr.ko.

14. Removes kperfTimer.ko.

4.4.5 Sample Logs

Following are the logs for EDMA:

- edma: async: A Count: 65535
- edma: async: B Count: 1
- edma: async: C Count: 1
- edma: async: Application buffer Size in Kbits: 65536
- edma: async: Time Elapsed in usec: 90
- edma: absync: A Count: 65535
- edma: absync: B Count: 1
- edma: absync: C Count: 1
- edma: absync: Application buffer Size in Kbits: 65536
- edma: absync: Time Elapsed in usec: 87

Following are the logs for QDMA:

- qdma: async: A Count: 65534
- qdma: async: B Count: 1
- qdma: async: C Count: 1
- qdma: async: Application buffer Size in Kbits: 65534
- qdma: async: Time Elapsed in usec: 91
- qdma: absync: A Count: 65535
- qdma: absync: B Count: 1
- qdma: absync: C Count: 1
- qdma: absync: Application buffer Size in Kbits: 65535
- qdma: absync: Time Elapsed in usec: 93

4.5 NAND

This section discusses the steps to execute the NAND performance tests by scripts and command line utility by providing an overview about the test setup information, command line arguments and the performance parameters.

4.5.1 Performance Parameters

Following NAND performance parameters will be obtained using pspTest tool:

a. Time taken in micro seconds for read/write of given data size

b. Data rate for read/write in MBytes/sec

c. Percentage of CPU load

4.5.2 Test Setup

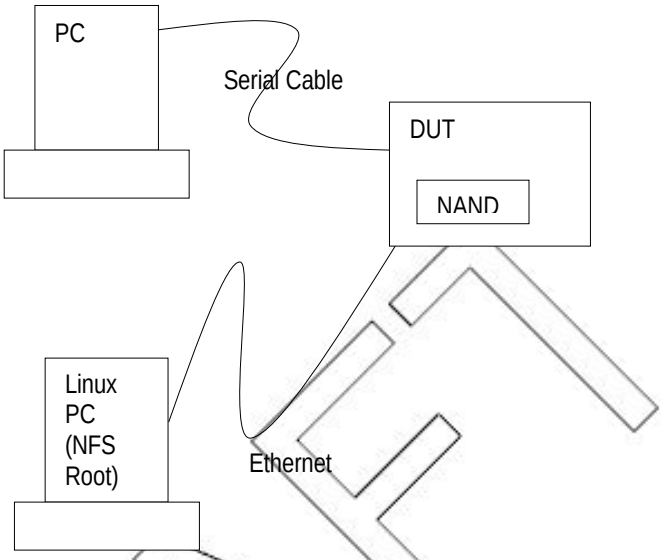

Figure 4-5. NAND Test Setup

4.5.3 Test Environment

q DUT (Device Under Test), serial console

4.5.4 Using the Application

The performance measurement application can be run either through the command line or by running the script.

4.5.4.1 Using Command Line

This section describes how to run performance measurement application through the command line.

Select proper switch settings (see section A.1). Go to //./pspTestTarget/bin and run the executable pspTest with the following configurable parameters as arguments:

i. String ThruPut for Throughput performance and Percentage of CPU load

ii. String MTDBlkWrite for write or MTDBlkRead for read

iii. Absolute path of the file used for I/O

iv. Application buffer size (Bytes)

v. Data size (Bytes)

Example: ./pspTest ThruPut MTDBlkWrite /dev/mtd3 102400 52428800

Note:

- To obtain performance values without cache influence during read operation power cycle the EVM after every write and then read the data written.
- Data size should be less than 123MB (for /dev/mtd3 NAND partition) as NAND available on the DM646x EVM is 128MB.

4.5.4.2 Using Script

To use performance measurement application by running the script:

1. See the respective LSP User Guides and flash the EVM. All default settings in EVM should be restored. Select the NAND through the switch settings. See section A.1, for the default EVM and the switch settings.
2. Open HyperTerminal/Teraterm (Output/Input Console). Set the required settings for HyperTerminal/TeraTerm (see section A.2). Switch on the power for EVM to boot.
3. See the respective LSP User Guides for enabling NAND device driver, compiling, and running Linux Kernel.
4. Environment variables needs to be updated as specified in respective LSP User Guides. Press **Enter** to stop booting immediately after bootup prints starts. Use the command `printenv` to see the environment variables and update them, if necessary.
5. Setup is now ready to run the performance test.
6. In the Output/Input console, run the NAND throughput script run_dm646x_nand_tests.sh available at //./pspTestTarget/scripts/throughput using the command ./run_dm646x_nand_tests.sh and press **Enter**.

Script Details

The run_dm646x_nand_tests.sh script available at //./pspTestTarget/scripts/throughput will perform the following operations on the target:

1. Performs read and write of data size 52428800 bytes with an application buffer of various sizes like 102400, 262144, 524288, 1048576, and 5242880 bytes on /dev/mtd3 partition.
2. Prints the performance measurement for each read write operation in MB/s.

4.5.5 Sample Logs

Following are the logs for read and write:

- filewrite: Buffer Size in bytes: 102400
- filewrite: FileSize in bytes: 52428800
- filewrite: Durartion in usecs: 18171310
- filewrite: Mega Bytes/sec: 2.751788
- filewrite: percentage cpu load: 100.00%

- fileread: Buffer Size in bytes: 102400
- fileread: FileSize in bytes: 52428800
- fileread: Durartion in usecs: 60653471
- fileread: Mega Bytes/Sec: 0.824402
- fileread: percentage cpu load: 100.00%

4.6 ATA

This section discusses the steps to execute the ATA performance tests by scripts and command line utility by providing an overview about the test setup information, command line arguments and the performance parameters.

4.6.1 Performance Parameters

Following ATA performance parameters will be obtained using pspTest tool:

a. Time taken in micro seconds for read/write of given data size
b. Data rate for read/write in MBytes/sec
c. Percentage of CPU load

4.6.2 Test Setup

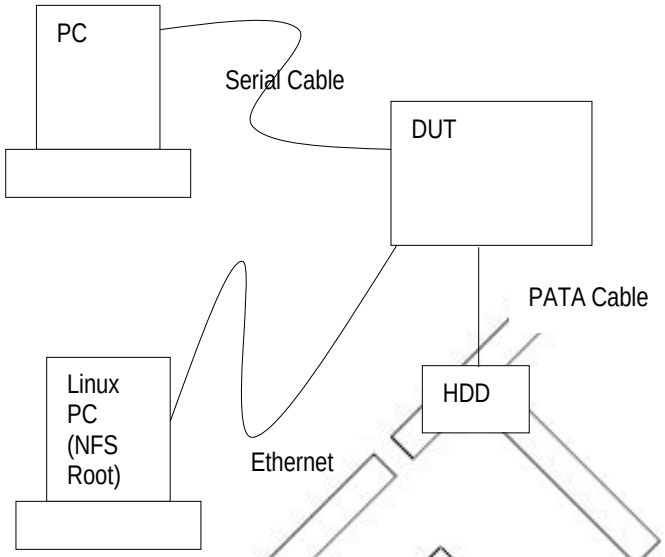

Figure 4-6. ATA Test Setup

4.6.3 Test Environment

- DUT (Device Under Test), serial console
- Hard Disk Drive
- Parallel ATA Cable

4.6.4 Using the Application

The performance measurement application can be run either through the command line or by running the script.

4.6.4.1 Using Command Line

This section describes how to run performance measurement application through the command line.

Go to //../pspTestTarget/bin and run the executable pspTest with the following configurable parameters as arguments.

i. String ThruPut for Throughput performance and Percentage of CPU load

ii. String TPfswrite for write or TPfsread for read

iii. Absolute path of the file used for I/O

iv. Application buffer size (Bytes)

v. Data size (Bytes)

Example: `./pspTest ThruPut TPfswrite /mnt/ata_pspTest/perf.txt 102400 104857600`

> **Note:**
>
> Mount point (/mnt/ata_pspTest) needs to be created before running the application in command line.

4.6.4.2 Using Script

To use performance measurement application by running the script:

1. Refer to respective LSP User Guides and flash the EVM. All default settings in EVM should be restored. See section A.1, for the default EVM and the switch settings.
2. In Power switch off mode, connect the Hard disk to the EVM. For further details on connecting hardware, refer EVM user manual.
3. Open HyperTerminal/Teraterm (Output/Input Console). Set the required settings for HyperTerminal/TeraTerm (see section A.2). Switch on the power for EVM to boot.
4. See the respective LSP User Guides for enabling ATA device driver, compiling, and running Linux Kernel.
5. Environment variables needs to be updated as specified in respective LSP User Guides. Press **Enter** to stop booting immediately after bootup prints starts. Use the command `printenv` to see the environment variables and update them, if necessary.
6. Setup is now ready to run the performance test.
7. Verify the presence of /dev/hda1 device entry.
8. In the Output/Input console, run the ata throughput script run_ ata_tests.sh available at //./pspTestTarget/scripts/throughput using the command `./run_ ata_tests.sh` and press **Enter**. (**Warning:** This will format the hard disk)

Script Details

The run_ata_tests.sh script available at //./pspTestTarget/scripts/throughput will perform the following operations on the target:

1. Formats the hda1 partition with ext2 filesystem.
2. Creates a directory as defined by MOUNT_POINT in the script for mounting.
3. Unmounts the mount directory to make sure it is not mounted already.
4. Mounts /dev/hda1 partition to the location defined by MOUNT_POINT in the script.

5. The hdparm sets the driver to PIO mode 4 (-P4).
6. Performs read and write of data size 104857600 bytes with an application buffer of various sizes like 102400, 262144, 524288, 1048576, and 5242880 bytes.
7. Removes the .txt files created while doing read/write.
8. hdparm sets the driver to MDMA mode 2 (-X34) and UDMA mode 5 (-X69) and steps 4 and 5 are repeated.

4.6.5 Sample Logs

Following are the logs for read and write:

- fileread: Buffer Size in bytes: 102400
- fileread: FileSize in bytes: 104857600
- fileread: Durartion in usecs: 29817255
- fileread: Mega Bytes/sec: 3.516675
- fileread: percentage cpu load: 100.00%
- filewrite: Buffer Size in bytes: 102400
- filewrite: FileSize in bytes: 104857600
- filewrite: Durartion in usecs: 19888754
- filewrite: Mega Bytes/sec: 5.272205
- filewrite: percentage cpu load: 100.00%

4.7 I2C

This section discusses the steps to execute the I2C performance tests by scripts and command line utility by providing an overview about the test setup information, command line arguments and the performance parameters.

4.7.1 Performance Parameters

Following I2C performance parameters will be obtained using pspTest tool:

a. Time taken in micro seconds for read/write of given data size
b. Data rate for read/write in Kbits/sec
c. Percentage of CPU load

4.7.2 Test Setup

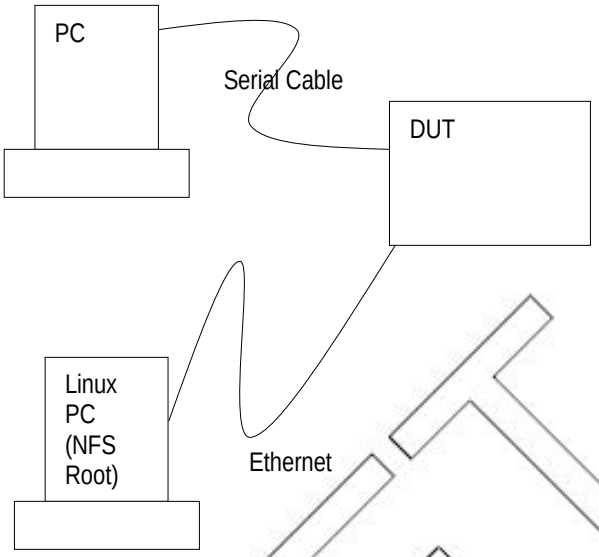

Figure 4-7. I2C Test Setup

4.7.3 Test Environment

q DUT (Device Under Test), serial console

4.7.4 Using the Application

The performance measurement application can be run either through the command line or by running the script.

4.7.4.1 Using Command Line

This section describes how to run performance measurement application through the command line.

Go to //./pspTestTarget/bin and run the executable ‘pspTest’ with the following configurable parameters as arguments.

i. String ThruPut for Throughput performance and Percentage of CPU load

ii. String I2cWrite for write or I2cRead for read

iii. Application buffer size (Bytes)

iv. Total Buffer Size (Bytes)

Example: `../bin/pspTest ThruPut I2cWrite 16 1024`

4.7.4.2 Using Script

To use performance measurement application by running the script:

1. See the respective LSP User Guides and flash the EVM. All default settings in EVM should be restored.
2. Open HyperTerminal/Teraterm (Output/Input Console). Set the required settings for HyperTerminal/TeraTerm (see section A.2). Switch on the power for EVM to boot.
3. See the respective LSP User Guides for enabling I2C device driver, compiling, and running Linux Kernel.
4. Environment variables needs to be updated as specified in respective LSP User Guides. Press **Enter** to stop booting immediately after bootup prints starts. Use the command `printenv`' to see the environment variables and update them, if necessary.
5. Setup is now ready to run the performance test.
6. In the Output/Input console, run the I2C throughput script run_ dm646x_i2c_tests.sh available at //./pspTestTarget/scripts/throughput using the command `./run_ dm646x_i2c_tests.sh` and press **Enter**.

Script Details

The run_dm646x_i2c_tests.sh script available at //./pspTestTarget/scripts/throughput will perform the following operations on the target.

1. Performs Read and Write of data size 1024 bytes with an Application Buffer of various sizes like 16, 32, 64,128, and 1024 bytes.

4.7.5 Sample Logs

Following are the logs for read and write:

- I2C: Write: Application buffer Size in Bytes: 16
- I2C: Write: Total buffer Size in Bytes: 1024
- I2C: Write: Duration in usec: 411565
- I2C: Write: Kbits/sec: 12.585938
- I2C: Write: percentage of cpu load: .20%
- I2C: Read: Application buffer Size in Bytes: 16
- I2C: Read: Total buffer Size in Bytes: 1024
- I2C: Read: Duration in usec: 38886
- I2C: Read: Kbits/sec: 205.468750
- I2C: Read: percentage of cpu load: .20%

4.8 SPI

This section discusses the steps to execute the SPI performance tests by scripts and command line utility by providing an overview about the test setup information, command line arguments and the performance parameters.

4.8.1 Performance Parameters

Following SPI performance parameters will be obtained using pspTest tool:

a. Time taken in micro seconds for read/write of given data size

b. Data rate for read/write in Kbits/sec

c. Percentage of CPU load

4.8.2 Test Setup

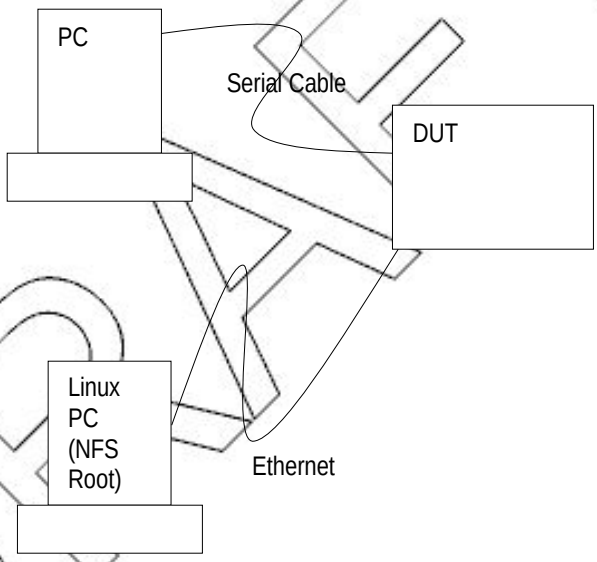

Figure 4-8. SPI Test Setup

4.8.3 Test Environment

q DUT (Device Under Test), serial console

4.8.4 Using the Application

The performance measurement application can be run either through the command line or by running the script.

4.8.4.1 Using Command Line

This section describes how to run performance measurement application through the command line.

Go to //./pspTestTarget/bin and run the executable pspTest with the following configurable parameters as arguments:

i. String ThruPut for Throughput performance and Percentage of CPU load

ii. String SpiWrite for write or SpiRead for read

iii. Application buffer size (Bytes)

iv. Total Buffer Size (Bytes)

Example: `../bin/pspTest ThruPut SpiWrite 16 1024`

4.8.4.2 Using Script

To use performance measurement application by running the script:

1. Refer to respective LSP User Guides and flash the EVM. All default settings in EVM should be restored. See section A.1, for the default EVM and the switch settings.
2. Open HyperTerminal/Teraterm (Output/Input Console). Set the required settings for HyperTerminal/TeraTerm (see section A.2). Switch on the power for EVM to boot.
3. See the respective LSP User Guides for enabling SPI device driver, compiling, and running Linux Kernel.
4. Environment variables needs to be updated as specified in respective LSP User Guides. Press **Enter** to stop booting immediately after bootup prints starts. Use the command `printenv` to see the environment variables and update them, if necessary.
5. Setup is now ready to run the performance test.
6. In the Output/Input console, run the EDMA throughput script run_dm646x_spi_tests.sh available at //./pspTestTarget/scripts/throughput using the command `./run_dm646x_spi_tests.sh` and press **Enter**.

Script Details

The run_dm646x_spi_tests.sh script available at //./pspTestTarget/scripts/throughput will perform the following operations on the target:

1. Performs read and write of data size 1024 bytes with an Application Buffer of various sizes like 16, 32, 64, and 1024 bytes.

4.8.5 Sample Logs

Following are the logs for read and write:

- SPI: read: Application buffer Size in Bytes: 16
- SPI: read: Total buffer Size in Bytes: 1024

- SPI: read: Duration in usec: 350391
- SPI: read: Kbits/sec: 22.656250
- SPI: Read: percentage of cpu load: 10.20%
- SPI: write: Application buffer Size in Bytes: 16
- SPI: write: Total buffer Size in Bytes: 1024
- SPI: write: Duration in usec: 739465
- SPI: write: Kbits/sec: 10.156250
- SPI: Write: percentage of cpu load: 10.20%

4.9 VLYNQ

This section discusses the steps to execute the VLYNQ performance tests by scripts and command line utility by providing an overview about the test setup information, command line arguments and the performance parameters.

4.9.1 Performance Parameters

Following VLYNQ performance parameters will be obtained using pspTest tool:

a. Time taken in micro seconds for read/write of given data size

b. Data rate for read/write in Mbits/sec

4.9.2 Test Setup

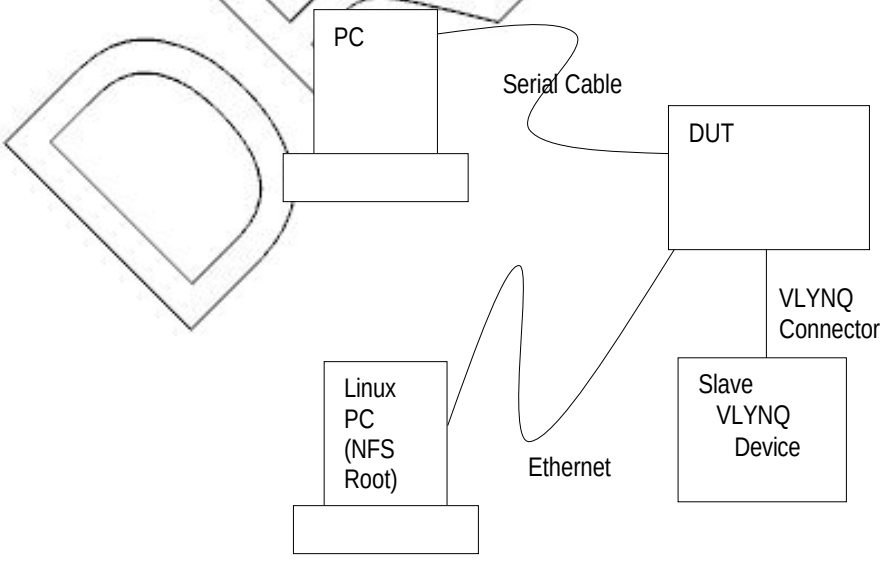

Figure 4-9. VLYNQ Test Setup

4.9.3 Test Environment

- DUT (Device Under Test), serial console
- VLYNQ Daughter Card for Slave Device
- VLYNQ Connector

4.9.4 Using the Application

The performance measurement application can be run either through the command line or by running the script.

4.9.4.1 Using Command Line

This section describes how to run performance measurement application through the command line.

Go to //./pspTestTarget/bin and insert the following modules:

i. Insert the timer module (kperfTimer.ko) using `insmod` command.

ii. Insert the VLYNQ application module (vlynq_cpu_transfer.ko for CPU transfer or vlynq_edma_transfer.ko for EDMA transfer) using `insmod` command.

iii. Remove the Modules inserted using `rmmod` command.

Example: `insmod vlynq_cpu_transfer.ko`

4.9.4.2 Using Script

To use performance measurement application by running the script:

1. See the respective LSP User Guides and flash the EVM. All default settings in EVM should be restored. See section A.1, for the default EVM and the switch settings.
2. Connect the Slave VLYNQ Device with the EVM using the VLYNQ Slave daughter card and the connector. For further details on connecting hardware, see the respective EVM user manual.
3. Open HyperTerminal/Teraterm (Output/Input Console). Set the required settings for HyperTerminal/TeraTerm (see section A.2). Switch on the power for EVM to boot.
4. See the respective LSP User Guides for enabling VLYNQ device driver, compiling and running Linux Kernel.
5. Environment variables needs to be updated as specified in respective LSP User Guides. Press **Enter** to stop booting immediately after bootup prints starts. Use the command `printenv` to see the environment variables and update them, if necessary.
6. Power ON the Slave VLYNQ Device and initialize VLYNQ on the Slave device.
7. Setup is now ready to run the performance test.

8. In the Output/Input console, run the VLYNQ throughput script run_ vlynq_tests.sh available at //./pspTestTarget/scripts/throughput using the command `./run_ vlynq_tests.sh` and press **Enter**.

Script Details

The run_vlynq_tests.sh script available at //./pspTestTarget/scripts/throughput will perform the following operations on the target:

1. Inserts the Timer Module. (kperfTimer.ko)
2. Inserts the VLYNQ application Module used to measure VLYNQ performance with CPU data transfer. (vlynq_cpu_transfer.ko)
3. Removes the VLYNQ application Module used to measure VLYNQ performance with CPU data transfer. (vlynq_cpu_transfer.ko)
4. Inserts the VLYNQ application Module used to measure VLYNQ performance with EDMA data transfer. (vlynq_edma_transfer.ko)
5. Removes the VLYNQ application Module used to measure VLYNQ performance with EDMA data transfer. (vlynq_edma_transfer.ko)
6. Removes the Timer Module. (kperfTimer.ko)

4.9.5 Sample Logs

Following are the logs for read and write:

- VLYNQ: CPU mode write: Buffer Size in Bytes: 1024
- VLYNQ: CPU mode write: Duration in usec: 114
- VLYNQ: CPU mode write: Data Rate in Mbps: 71
- VLYNQ: CPU mode read: Buffer Size in Bytes: 1024
- VLYNQ: CPU mode read: Duration in usec: 468
- VLYNQ: CPU mode read: Data Rate in Mbps: 17
- VLYNQ: EDMA ABSYNC mode write: Buffer Size in Bytes: 1024
- VLYNQ: EDMA ABSYNC mode: Duration in usec: 88
- VLYNQ: EDMA ABSYNC mode: Data rate in Mbps: 93
- VLYNQ: EDMA ABSYNC mode read: Buffer Size in Bytes: 1024
- VLYNQ: EDMA ABSYNC mode: Duration in usec: 144
- VLYNQ: EDMA ABSYNC mode: Data rate in Mbps: 56

4.10 USB ISO Video

This section discusses the steps to execute the USB Isochronous Video performance tests by scripts and command line utility by providing an

overview about the test setup information, command line arguments and the performance parameters.

4.10.1 Performance Parameters

Following USB ISO Video performance parameters will be obtained using pspTest tool:

a. Capture Frame rate of uvcvideo webcam driver for video over USB ISO in fps (Frames per sec).

b. Percentage of CPU load

4.10.2 Test Setup

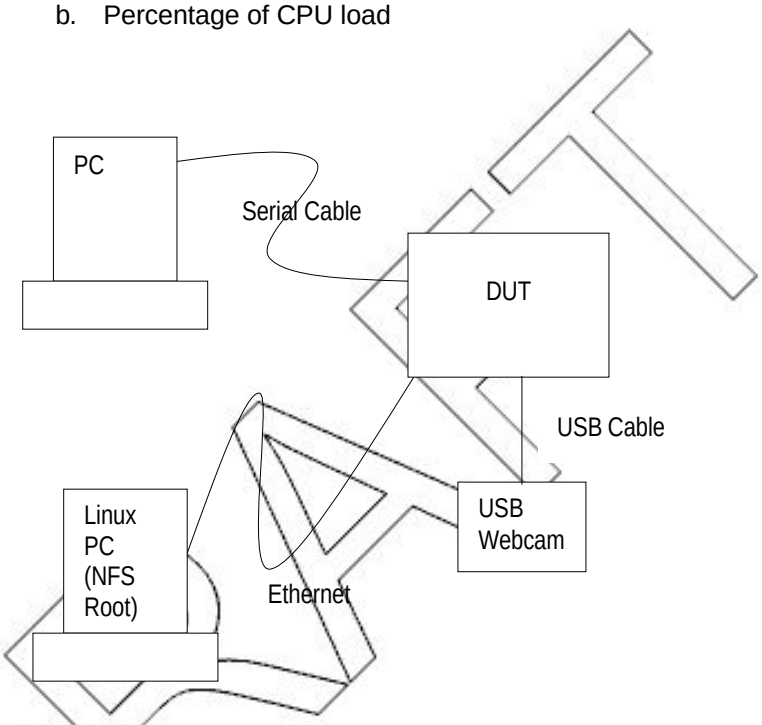

Figure 4-10. USB ISO Video Test Setup

4.10.3 Test Environment

- DUT (Device Under Test), serial console
- USB Webcam with Isochronous support

4.10.4 Using the Application

The performance measurement application can be run either through the command line or by running the script.

4.10.4.1 Using Command Line

This section describes how to run performance measurement application through the command line.

Go to //./pspTestTarget/bin and run the executable pspTest with the following configurable parameters as arguments:

i. String ThruPut for Throughput performance and Percentage of CPU load

ii. String FRusbisovideocapture for capturing video over the USB device

iii. Capture device (USB device)

iv. Number of fames to be captured (must be less than 10000)

v. Number of the frame to be written to a file (must be less than total number of frames to be captured)

vi. Output YUV Filename (E.g. USB.yuv)

Example: `./pspTest ThruPut FRusbisovideocapture /dev/video4 500 100 USB.yuv`

Note:

USB device must be inserted and enumerated before running the application in command line.

4.10.4.2 Using Script

To use performance measurement application by running the script:

1. See the respective LSP User Guides and flash the EVM. All default settings in EVM should be restored. See section A.1, for the default EVM and the switch settings.
2. Open HyperTerminal/Teraterm (Output/Input Console). Set the required settings for HyperTerminal/TeraTerm (see section A.2). Switch on the power for EVM to boot.
3. See the respective LSP User Guides for enabling USB ISO device driver, compiling and running Linux Kernel.
4. Environment variables needs to be updated as specified in respective LSP User Guides. Press **Enter** to stop booting immediately after bootup prints starts. Use the command `printenv` to see the environment variables and update them, if necessary.
5. The open source uvcvideo driver for the webcam Logitech Orbit MP Sphere must be downloaded from the link http://svn.berlios.de/svnroot/repos/linux-uvc/linux-uvc/trunk/.

Note:

For a different webcam, you may need to download a suitable open source driver.

6. This uvcvideo driver must be built on the target using commands make `clean -> make -> make` install. After this reboot the board

and insert the Logitech Orbit MP Sphere Webcam into the USB slot.

7. The USB Webcam must be enumerated as uvcvideo: Found UVC 1.00. This can be checked using command `cat /proc/modules`, which should show uvcvideo module. The device must also be enumerated as /dev/video*.

8. Setup is now ready to run the performance test.

9. In the Output/Input console, run the USB iso video throughput script run_usbisovideo_tests.sh available at //./pspTestTarget/scripts/throughput using the command `./run_usbisovideo_tests.sh` and press **Enter**.

Script Details

The run_usbisovideo_tests.sh script available at //./pspTestTarget/scripts/throughput will perform the following operations on the target:

1. Performs Capture of 500 frames of video over the USB device /dev/video4.

2. Captures the 100th frame and saves it as USB.yuv. This saved frame can be viewed using a YUV player. This captured image is of size 320*240, has the sampling format of YUV422, component order of YUYV, and is in progressive-packed format.

3. Calculates the capture frame rate over 500 frames.

4.10.5 Sample Logs

Following are the logs for capture:

- usbisovideo_perf:281:Capturing frames:
- usbisovideo_perf:351:Capture frame rate: 15.210806
- usbisovideo: capture: percentage cpu load: 4.00%

4.11 USB ISO Audio

This section discusses the steps to execute the USB Isochronous Audio performance tests by scripts and command line utility by providing an overview about the test setup information, command line arguments and the performance parameters.

4.11.1 Performance Parameters

Following USB ISO Audio performance parameters will be obtained using pspTest tool:

a. Time taken in seconds for read/write of given data size

b. Data rate for read/write in Bytes/sec

c. Percentage of CPU load

4.11.2 Test Setup

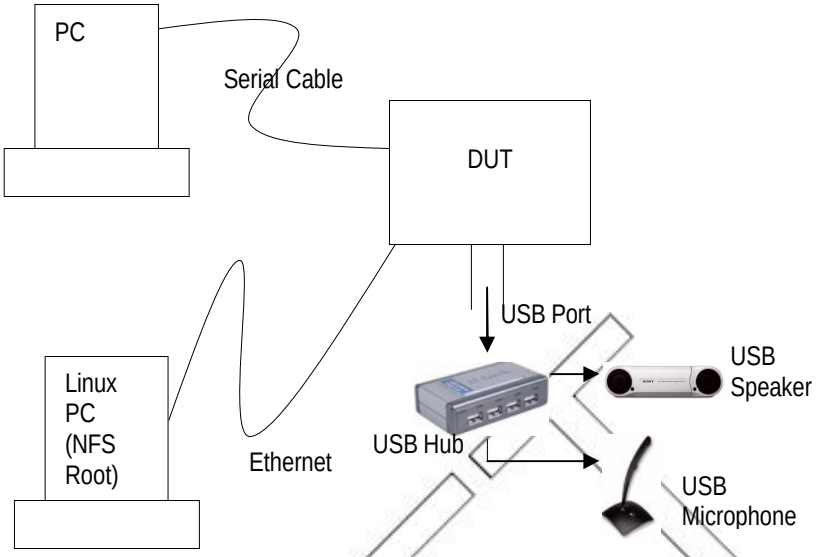

Figure 4-11. USB ISO Audio Test Setup

4.11.3 Test Environment

- DUT (Device Under Test), serial console
- USB Speaker with Isochronous support
- USB Microphone with Isochronous support
- USB Hub

4.11.4 Using the Application

The performance measurement application can be run either through the command line or by running the script.

4.11.4.1 Using Command Line

This section describes how to run performance measurement application through the command line.

Go to //./pspTestTarget/bin and run the executable pspTest with the following configurable parameters as arguments:

i. String ThruPut for Throughput performance and Percentage of CPU load

ii. String FRusbisoaudiowrite for write or FRusbisoaudioread for read

iii. Device node for read or write

iv. Sampling Rate (Hz) for which performance is carried out

v. Application buffer size (Bytes)

vi. Data size (Bytes)

Example: `./pspTest ThruPut  FRusbisoaudioread /dev/dsp_usb_mic 8000 4096 5242880`

> **Note:**
>
> Required to do mknod for USB ISO Audio device before executing pspTest and same device node needs to be used in argument. Command for doing mknod is `mknod <Device Node> c 14 <minor number for USB Device>`

Example: `mknod /dev/dsp_usb_mic c 14 35`

4.11.4.2 Using Script

To use performance measurement application by running the script:

1. See the respective LSP User Guides and flash the EVM. All default settings in EVM should be restored. See section A.1, for the default EVM and the switch settings.
2. Open HyperTerminal/Teraterm (Output/Input Console). Set the required settings for HyperTerminal/TeraTerm (see section A.2). Switch on the power for EVM to boot.
3. See the respective LSP User Guides for enabling USB ISO Audio device driver, compiling and running Linux Kernel.
4. Environment variables needs to be updated as specified in respective LSP User Guides. Press **Enter** to stop booting immediately after bootup prints starts. Use the command `printenv` to see the environment variables and update them, if necessary.
5. Connect USB Speaker and USB Microphone through USB Hub to the EVM. For further details on connecting hardware, refer EVM user manual.
6. Perform mknod of Microphone with a device node name /dev/dsp_usb_mic using the command `mknod /dev/dsp_usb_mic c 14 <minor number for Microphone>.`
7. Perform mknod of Speaker with a device node name /dev/dsp_usb_spk using the command `mknod /dev/dsp_usb_spk c 14 <minor number for Speaker>`.
8. Setup is now ready to run the performance test.
9. In the Output/Input console, run the usb iso audio throughput script run_usbiso_audio_tests.sh available at //./pspTestTarget/scripts/throughput using the command `./run_usbiso_audio_tests.sh` and press **Enter**.

Script Details

The run_usbiso_audio_tests.sh script available at //./pspTestTarget/scripts/throughput will perform the following operations on the target:

1. Performs read/record of data size 5242880 bytes and Application Buffer of size 4096 bytes with sampling rates of various values like 8000, 11025, and 22050 Hz.
2. Performs write/playback of data size 5242880 bytes and Application Buffer of size 4096 bytes with sampling rates of various values like 32000, 44100, and 48000 Hz.

4.11.5 Sample Logs

Following are the logs for read and write:

- audio: read: Word Length in bits: 16
- audio: read: No. of channels per sample: 2
- audio: read: Sampling Rate in Hz: 8000
- audio: read: Duration in sec: 163.861243
- audio: read: No. of bits/sec: 255967
- audio: read: percentage cpu load: 2.00%
- audio: write: Word Length in bits: 16
- audio: write: No. of channels per sample: 2
- audio: write: Sampling Rate in Hz: 32000
- audio: write: Duration in sec: 39.930519
- audio: write: No. of bits/sec: 1050401
- audio: write: percentage cpu load: 2.00%

4.12 USB MSC Host

This section discusses the steps to execute the USB MSC host performance tests by scripts and command line utility by providing an overview about the test setup information, command line arguments and the performance parameters.

4.12.1 Performance Parameters

Following USB MSC Host performance parameters will be obtained using pspTest tool:

a. Time taken In micro seconds for read/write of given data size

b. Data rate for read/write in MBytes/sec

c. Percentage of CPU load

4.12.2 Test Setup

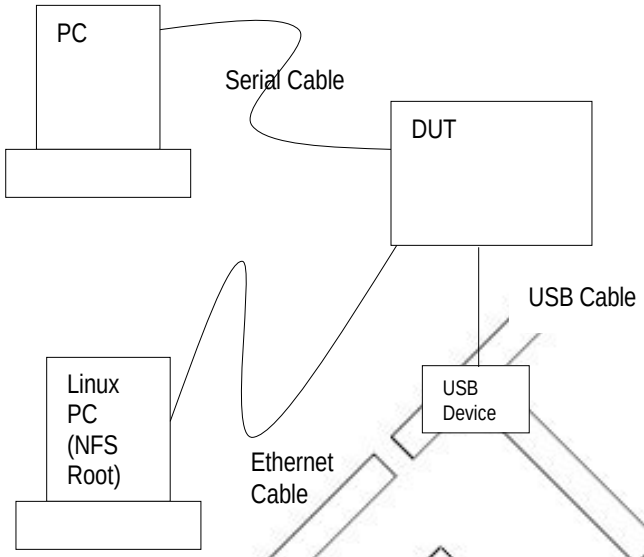

Figure 4-12. USB MSC Host Test Setup

4.12.3 Test Environment

- DUT (Device Under Test), serial console
- USB Device (Example: Pendrive)
- USB Cable

4.12.4 Using the Application

The performance measurement application can be run either through the command line or by running the script.

4.12.4.1 Using Command Line

This section describes how to run performance measurement application through the command line.

Go to //../pspTestTarget/bin and run the executable pspTest with the following configurable parameters as arguments:

1. String ThruPut for Throughput performance and Percentage of CPU load
2. String TPfswrite for write or TPfsread for read
3. Absolute path of the file used for I/O
4. Application buffer size (Bytes)
5. Data size (Bytes)

Example: `./pspTest ThruPut TPfswrite  /mnt/usbmsc/perf.txt 102400 104857600`

> **Note:**
>
> Mount point (/mnt/usbmsc) needs to be created before running the application in command line.

4.12.4.2 Using Script

To use performance measurement application by running the script:

1. See the respective LSP User Guides and flash the EVM. All default settings in EVM should be restored. See section A.1, for the default EVM and the switch settings.
2. In Power switch off mode, connect the USB Device to the EVM via USB Cable. For further details on connecting hardware, refer EVM user manual.
3. Open HyperTerminal/Teraterm (Output/Input Console). Set the required settings for HyperTerminal/TeraTerm (see section A.2). Switch on the power for EVM to boot.
4. See the respective LSP User Guides for enabling ATA device driver, compiling and running Linux Kernel.
5. Environment variables needs to be updated as specified in respective LSP User Guides. Press **Enter** to stop booting immediately after bootup prints starts. Use the command `printenv` to see the environment variables and update them if necessary.
6. Setup is now ready to run the performance test.
7. Verify the presence of /dev/sda1 device entry.
8. In the Output/Input console, run the USB MSC Host throughput script run_usb_msc_host_tests.sh available at //./pspTestTarget/scripts/throughput using the command `./run_usb_msc_host_tests.sh` and press **Enter**. (**Warning:** This will format the USB device)

A.1.1.1.1 Script Details

The run_usb_msc_host_tests.sh script available at //./pspTestTarget/scripts/throughput will perform the following operations on the target:

1. Formats the sda1 partition with ext2 filesystem.
2. Creates a directory as defined by MOUNT_POINT in the script for mounting.
3. Unmounts the mount directory to make sure it is not mounted already.
4. Mounts /dev/sda1 partition to the location defined by MOUNT_POINT in the script.

5. Performs read and write of data size 104857600 bytes with an Application Buffer of various sizes like 102400, 262144, 524288, 1048576, and 5242880 bytes.
6. Removes the .txt files created while doing read/write.

4.12.5 Sample Logs

Following are the logs for read and write:

- filewrite: Buffer Size in bytes: 102400
- filewrite: FileSize in bytes: 104857600
- filewrite: Durartion in usecs: 6797321
- filewrite: Mega Bytes/sec: 15.426312
- filewrite: percentage cpu load: 20.00%
- fileread: Buffer Size in bytes: 102400
- fileread: FileSize in bytes: 104857600
- fileread: Durartion in usecs: 8347341
- fileread: Mega Bytes/sec: 12.56179
- fileread: percentage cpu load: 20.00%

Appendix A

General Setup Details

This Chapter gives EVM setting details and output/input console setting details.

A.1 EVM Settings

SW3: Select the switch settings according to the boot mode used.

BM[0:3]	Boot mode when PCI-OFF	Boot mode when PCI-ON
0000	EMU	
0100	HPI-16	PCI without auto init
1100	HPI-32	PCI with auto init
0010	EMIFA	
0110	I2C	
1110	NAND	
0001	UART0	
1001	Emulation	
0101	VLYNQ	
1101	EMAC	
0111	SPI	
ELSE	NA	

Table A-1. SW3 Switch Settings for various Boot modes

A.2 HyperTerminal/TeraTerm Settings

Serial Port	Go to setup > Serial Port and select the following. Port: COM1 Baud rate: 115200 Data: 8 bit

	Parity: none Stop: 1 bit Flow Control: none
General	Go to setup > General and select the following. Default port: COM1 Language: English

Table A-2. HyperTerminal/TeraTerm Settings

Appendix B

Adding New Test Case

This appendix provides details on how a test case can be added to PspTest Tool.

B.1 Adding a New pspTest Module

PspTest is designed to be extensible in terms of test methodologies. Under tests directory, each module can be added. Each module should cover a particular driver or a specific multi-driver scenario. To add a new module:

1. Create the new test module for user-level module under //./psp_test_bench/performanceTest/throughput/userlevel/tests/ and for kernel level module under //./psp_test_bench/performanceTest/throughput/kernellevel/tests/. Choose a name that reflects the scope of the test module.
2. Update tests/DIRS with the name of the test module so that make is aware of the new test module.
3. Create two files - Makefile and SOURCES in the test module directory. Any existing test modules can be used as an example. The only file that will need to change is the SOURCES file. Changes to SOURCES are described in the section B.2.

B.2 Adding a New pspTest Command

To add a new pspTest command:

1. Write the tests under the targeted test module. The entry point to a test should be a function that has the following signature:

```
int test_name(int, const char **);
```

2. Update SOURCES in the test module to include the new file(s).

> **Note:**
>
> Only .c sources needs to be added here and not headers.

3. Update throughputEngine.c at //./psp_test_bench/main to call the test function.
4. Write the script under //./psp_test_bench/performanceTest/throughput/scripts/ to execute the tests.

B.2.1 Updating throughputEngine.c File

The throughputEngine.c requires the following changes:

1. Declare your test entry function at the top of the file by extending it into throughputEngine.c

2. Get the hash value of the command using the pspTest hash <commandString> command. Define the command in throughputEngine.c file. Add the command to throughputTestArray array with the function pointer pointing to the test entry function. You can use the tests already defined as a sample.