

Serial RapidIO DMA Demonstration with Two T4240QDS Boards using U-boot

1 Introduction

The T4240 QorIQ multicore processor features a 1x/4x serial RapidIO interface to communicate with other RapidIO devices. This application note is provided to assist those engineers wishing to quickly setup and configure two T4240QDS development platforms via serial RapidIO, and run a basic DMA transfer from DDR memory on Board A to DDR memory on Board B.

This document provides step by step instructions on how to set up the hardware environment, software environment, physical connections needed to boot the board, and switch settings for the T4240QDS platforms.

The SRIO DMA demonstration will run using U-boot scripts that will set up the serial RapidIO inbound and outbound windows, configure the DMA, and perform the DMA transfers.

The T4240QDS is installed in a chassis on delivery, however, for this demonstration it will need to be removed from the chassis to provide access to the riser card slots.

Contents

1	Introduction.....	1
2	Required equipment and visual set-up.....	2
3	Procedure.....	4
A	Text versions of the U-boot scripts.....	9
B	Revision history.....	10






2 Required equipment and visual set-up

This section includes a required equipment and resource list you'll need for the demonstration. It also contains two pictures to help you with the connections and setup.

2.1 Required equipment

Use the equipment and references in the lists below to perform the demonstration.

Table 1. Required hardware

Hardware item:	Notes:	
T4240QDS x 2, power supply with cables	Removed from chassis for access to riser card slots	
PCI Express Gen 2 x8 Host Interface Board (256), Host	Manufactured by: One StopSystems,part number: OSS-PCIE-HIB25-X8-H-E1.1	
PCI Express Gen 2 x8 Host Interface Board (256), Target	Manufactured by: One StopSystems,part number: OSS-PCIE-HIB25-X8-T-E1	
CABLE, PCIe x8, 2 Meters	Manufactured by: One StopSystems,part number: CABLE_PCI-x8-2M	
Serial cable x 2	Additionally, a USB to serial converter may be required because two serial cable connectors are not typically located on desktops	

Required software package:

- Srio1_windows_script.img
- dma_SRIO1.img
- RCW binary

Pre-installed software already on board:

- U-boot
- TeraTerm x2

Suggested reference:

- T4204QDS Reference Manual

2.2 T4204QDS connections

The image below depicts a basic visual set-up of the hardware and connections for the demonstration. Use it to check your overall set-up.

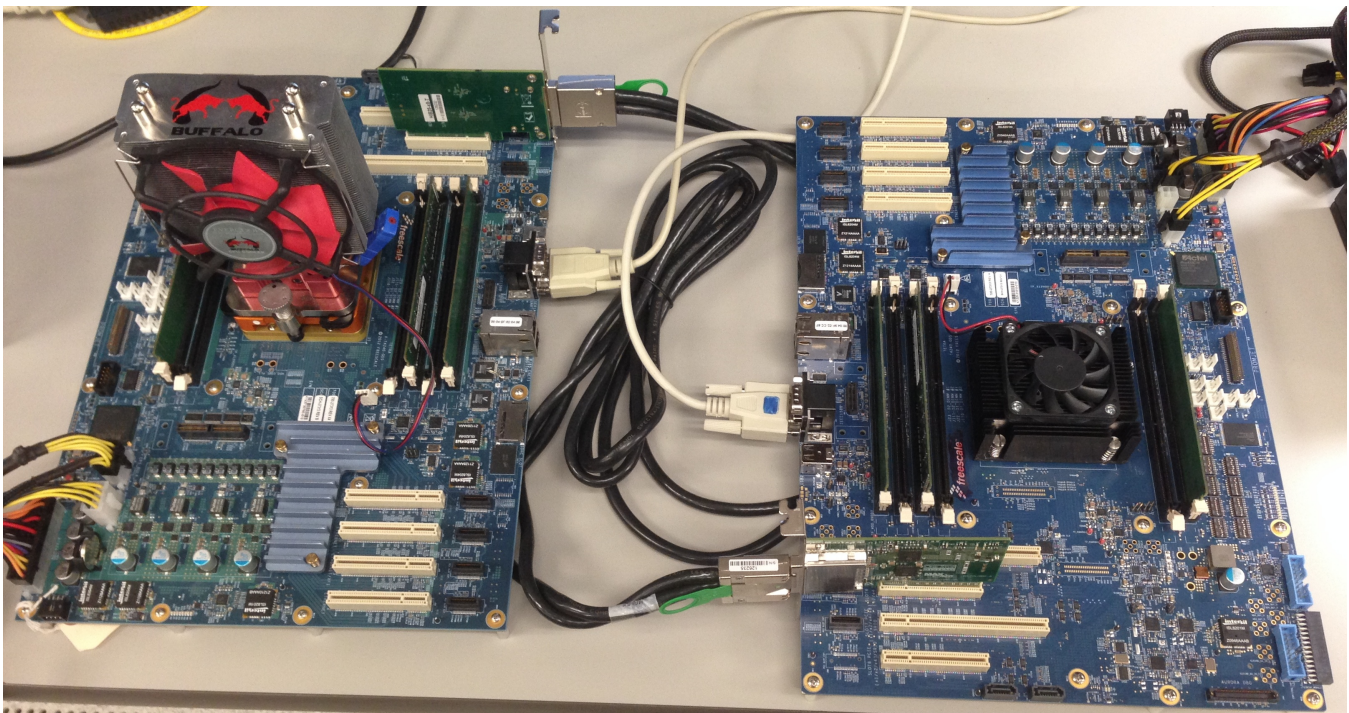


Figure 1. SRIIO hardware connection between two T4204QDS Boards

Procedure

Refer to the figure below to confirm the specific locations of the power supply connectors, UART connectors, SRIO 1 port slot 6, and power pushbuttons on the T4240QDS board.

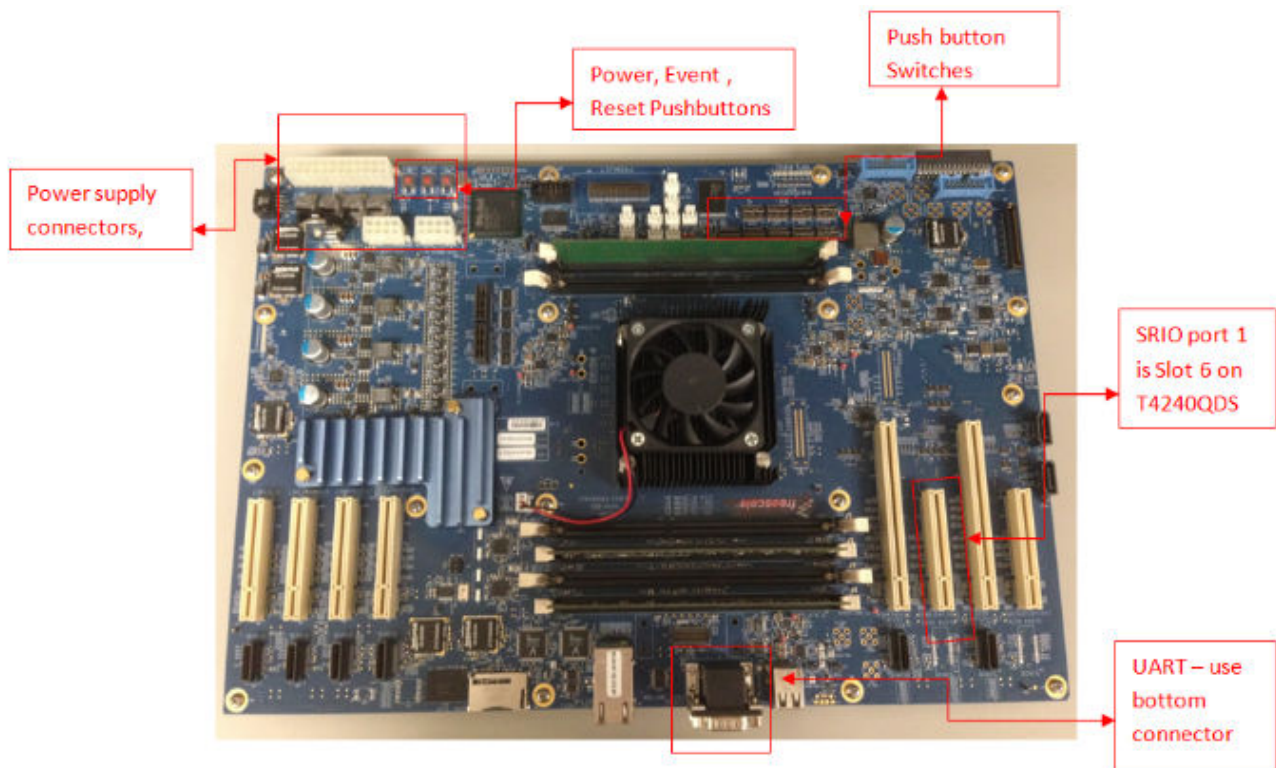


Figure 2. T4240QDS board showing connections

3 Procedure

Use the following procedure in this section to perform the SRIO DMA demonstration. The procedure is broken into four sequential parts:

- Part 1: Set-up the hardware
- Part 2: Set-up the software
- Part 3: Reconfigure the push button switches
- Part 4: Run the test

3.1 Part 1: Set-up the hardware

1. Connect 750W Power Supply as shown in [Figure 1](#).
 - Both boards should be powered down at initiation.
2. Connect each serial cable to bottom UART connector on both boards as shown in [Figure 2](#).
3. Attach cable to both ends PCIe Gen 2 riser cards.
4. Place the PCIe Gen 2 riser cards in slot 6 on both boards as per shown in [Figure 1](#). SRIO 1 port is on slot 6 on the T4240QDS.
5. Open a TeraTerm window for Board A, select the correct COMM port for your serial cable, baud rate should be 115000. The figure below shows the port setup.

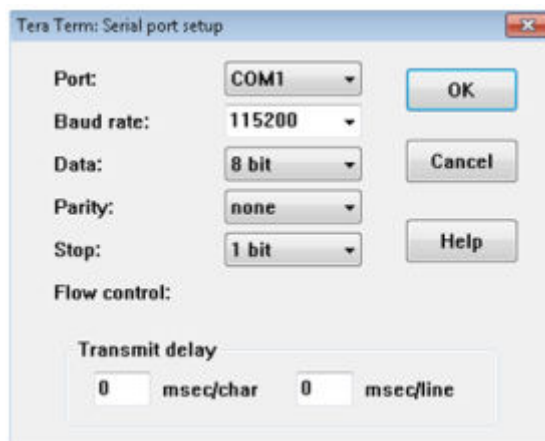


Figure 3. TeraTerm Configuration

6. Open a second TeraTerm window for Board B, select the correct COMM port for your serial cable connected to Board B and configure as in [Figure 3](#).
7. Set switch 6 on T4204QDS to “00001111”. [Figure 2](#) shows the location of the switches on the board.

3.2 Part 2: Set-up the software

1. Power up Board A and immediately catch the U-boot prompt by pressing the space bar on your keyboard.
 - There is a countdown from 10
 - A similar U-boot configuration displays on your TeraTerm screen for Board A.

The next steps [2](#) and [3](#) show you how to configure U-boot as illustrated below:

```

U-Boot 2013.01-00004-g88177d7-dirty (Nov 06 2013 - 09:26:17)

CPU0: T4240E, Version: 2.0, (0x82480020)
Core: E6500, Version: 2.0, (0x80400120)
Error: Unknown FMan2 clock select!
Clock Configuration:
CPU0:1666.667 MHz, CPU1:1666.667 MHz, CPU2:1666.667 MHz, CPU3:1666.667 MHz,
CPU4:1666.667 MHz, CPU5:1666.667 MHz, CPU6:1666.667 MHz, CPU7:1666.667 MHz,
CPU8:1666.667 MHz, CPU9:1666.667 MHz, CPU10:1666.667 MHz, CPU11:1666.667 MHz,
CCB:733.333 MHz,
DDR:233.333 MHz (466.667 MT/s data rate) (Asynchronous), IFC:183.333 MHz
FMAN1: 733.333 MHz
FMAN2: 366.667 MHz
QMAN: 366.667 MHz
PME: 533.333 MHz
L1: D-cache 32 kB enabled
I-cache 32 kB enabled
Reset Configuration Word (RCW):
00000000: 16070019 18101916 00000000 00000000
00000010: 04023030 00558c00 ec020000 f5000000
00000020: 00600000 ee0000ee 00000000 000307fc
00000030: 00000000 00000000 00000000 00000028
Board: T4240QDS, Sys ID: 0x1e, Sys Ver: 0x24, vBank: 0
FPGA: v3 (T4240QDS_2012_1113_1114), build 438 on Tue Nov 13 17:14:23 2012
SERDES Reference Clocks: SERDES1=125MHz SERDES2=125MHz SERDES3=100MHz SERDES4=100MHz
I2C: ready
SPI: ready
DRAM: Initializing...using SPD
Detected UDIMM 9JSF25672AZ-2G1K1
Detected UDIMM 9JSF25672AZ-2G1K1
Detected UDIMM 9JSF25672AZ-2G1K1
4 GiB left unmapped
    
```


Procedure

```

DDR: 6 GiB (DDR3, 64-bit, CL=7, ECC on)
  DDR Controller Interleaving Mode: 3-way 4KB
Flash: 128 MiB
L2: 2048 KB enabled
enable l2 for cluster 1 fec60000
enable l2 for cluster 2 fec90000
Corenet Platform Cache: 1536 KB enabled
Using SERDES1 Protocol: 1 (0x1)
Using SERDES2 Protocol: 1 (0x1)
Using SERDES3 Protocol: 6 (0x6)
Using SERDES4 Protocol: 6 (0x6)
SRIO1: enabled
SRIO2: enabled
NAND: 512 MiB
MMC: FSL_SDHC: 0
EEPROM: NXID v1
PCIE1: Endpoint, no link, regs @ 0xfe240000
PCIE1: Bus 00 - 00
PCIE2: disabled
PCIE3: Endpoint, no link, regs @ 0xfe260000
PCIE3: Bus 01 - 01
PCIE4: disabled
In: serial
Out: serial
Err: serial
Net: Fman1: Uploading microcode version 106.4.8
PHY reset timed out
PHY reset timed out
Fman2: Uploading microcode version 106.4.8
FM1@DTSEC5, FM1@TGEC1, FM1@TGEC2, FM2@DTSEC5
Hit any key to stop autoboot: 0

```

2. Update the RCW to the binary supplied in the software package:
 - a. First, erase the current RCW
 1. Type **Protect off all**
 2. Type **Erase e8000000 +118** This erases current RCW
 3. Type **md e8000000** All f's should be content of this area of flash
 - b. Type the following at the U-boot prompt => **loady 1000000**
 - c. Select : **File-> Transfer-> YMODEM -> Send -> SRIO1& SRIO2_61614.bin** from the TeraTerm menu (See figure below).

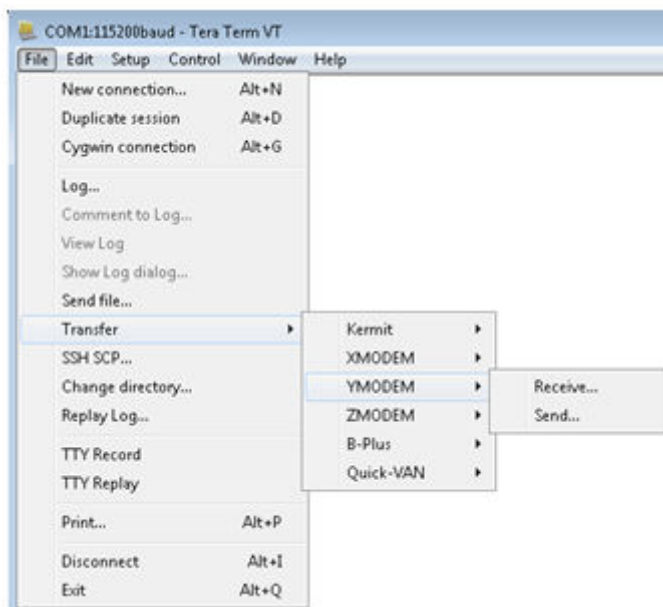


Figure 4. TeraTerm screen shot

- The U-boot screen should display:

```
=> loady 1000000
## Ready for binary (ymodem) download to 0x01000000 at 115200 bps...
CCCxyzModem - CRC mode, 0(SOH)/2(STX)/0(CAN) packets, 8 retries
## Total Size      = 0x00000060 = 96 Bytes
```

d. Display the contents of the DDR memory to make sure your RCW loaded correctly. **Type: md 1000000**, it should display the following:

```
=> md 1000000
01000000: aa55aa55 010e0100 16070019 18101916      .U.U.....
01000010: 00000000 00000000 04023030 00558c00      .....00.U..
01000020: ec020000 f5000000 00600000 ee0000ee      .....`.....
01000030: 00000000 000307fc 00000000 00000000      .....
01000040: 00000000 00000028 095fc030 00008148      .....(._.0...H
01000050: 095fd030 00808148 08138040 81ec7c04      .._0...H...@...|.
01000060: deadbeef deadbeef deadbeef deadbeef      .....
```

e. Copy the RCW located at 0x1000000 to the required U-boot location e8000000

1. Type **cp.b 1000000 e8000000 118**
2. Type **md e8000000**, this displays the RCW at location e8000000
3. Reset the board and you will see the new RCW as in 1 .

3. Repeat steps 2 and 3 for Board B.

3.3 Part 3: Reconfigure the push button switches

1. Set the switches SW1 through SW9 on both boards as shown in the following figure:

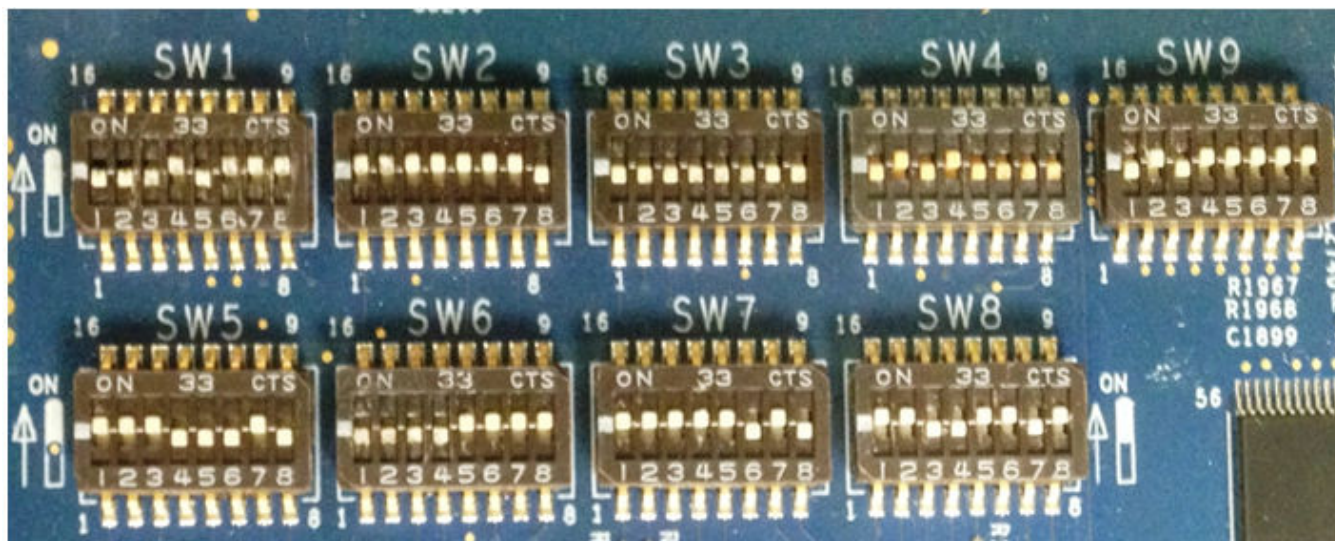


Figure 5. T4240QDS switch settings

2. Power cycle both boards at the same time.
 - Powering boards at the same time helps prevent initiation link errors during SRIO software training.
 - Immediately catch the U-boot prompt by pressing the space bar on your keyboard.
3. Compare your U-boot screen to the U-boot log in Part 2: Set-up the software, step 2. Both Board A and Board B should have the same U-boot screen configurations. Key items to compare are:
 - RCW
 - Platform Speed
 - SRIO1 enabled
 - SerDes3 Reference Clock Speed
4. Check and clear any initiation link errors by checking the Port 1 Error and Status Command and Status Register (SRIO_P1ESCSR) at offset C_0158 in the following steps:
 - a. Type **md fe0c0158**.

- If the value of this register is 00000002, then link is up and PORT OK bit is set with no errors. Proceed to [Part 4: Run the test](#).
 - If not, continue with the next step.
- b. Reset both boards at the same time and re-check register.
- If “fe0c0158” register value is anything other than 00000002, then error bits will need to be cleared.
- For example, if the value of the register is 00020202:
- Type **mm fe0c0158**
 - Type the value **00020202**, then press enter
 - Type **x**, then press enter
 - Type **md fe0c0158**
- The value should now be 00000002.
- c. The equipment is ready to run the test.

3.4 Part 4: Run the test

Steps 1–3 load and run the script that sets the inbound and outbound SRIO windows.

1. On Board A, Type **loady 1000000** at the U-boot prompt
2. Navigate to **File-> Transfer-> YMODEM -> Send -> srio1_windows_script.img** from the Teraterm menu
3. Type **source 1000000**
 - The U-boot screen displays the following output:

```
=> loady 0x1000000
## Ready for binary (ymodem) download to 0x01000000 at 115200 bps...
CCxyzModem - CRC mode, 0(SOH)/2(STX)/0(CAN) packets, 8 retries
## Total Size      = 0x000002db = 731 Bytes
=> source 0x1000000
## Executing script at 01000000
===== Test SRIO1 Link =====
fe0c015b: 02      .
Total of 1 byte(s) were the same
SRIO1 LINK UP
===== SRIO1LB Test =====
===== Setting up Outbound and Inbound Windows =====
=>
```

4. Repeat steps 1–3 for Board B

Steps 5–7 load and run the script that carries out the DMA Transfer from Board A DDR memory to Board B’s DDR Memory via SRIO.

5. On Board A, Type **loady 1000000** at the U-boot prompt
6. Navigate to **File-> Transfer-> YMODEM -> Send -> dma_SRIO1.img**
7. Type **source 1000000**,
 - The U-boot screen displays the following output:

```
=> loady 0x1000000
## Ready for binary (ymodem) download to 0x01000000 at 115200 bps...
CCxyzModem - CRC mode, 0(SOH)/3(STX)/0(CAN) packets, 4 retries
## Total Size      = 0x00000631 = 1585 Bytes
=>source 0x1000000
## Executing script at 01000000
===== DMA Desc 1 =====
===== DMA Desc 2 =====
===== Set DMA MR[CS] to zero, set DMA CLNDAR =====
===== Set MR[CS] bit, starts the transfer =====
==== Read the block of source data from DDR address 0x10000 =====
00100000: babeface babeface babeface babeface      .....
00100010: babeface babeface babeface babeface      .....
00100020: babeface babeface babeface babeface      .....
00100030: babeface babeface babeface babeface      .....
Read back the same block of data from SRIO1 for DMA data transfer comparison
```



```

a1000000: babeface babeface babeface babeface .....
a1000010: babeface babeface babeface babeface .....
a1000020: babeface babeface babeface babeface .....
a1000030: babeface babeface babeface babeface .....
Read the second block of source data from DDR address 0x200000
00200000: facebabe facebabe facebabe facebabe .....
00200010: facebabe facebabe facebabe facebabe .....
00200020: facebabe facebabe facebabe facebabe .....
00200030: facebabe facebabe facebabe facebabe .....
Read back the same block of data from SRIO1 for DMA data transfer comparison
a1002000: facebabe facebabe facebabe facebabe .....
a1002010: facebabe facebabe facebabe facebabe .....
a1002020: facebabe facebabe facebabe facebabe .....
a1002030: facebabe facebabe facebabe facebabe .....
Total of 256 word(s) were the same
===== SRI01 DMA data transfer 1 PASSED! =====
Total of 256 word(s) were the same
===== SRI01 DMA data transfer 1 PASSED! =====

```

The DMA is setup in basic chaining mode, and consists of two descriptors. It transfers two blocks of data located in DDR memory at locations 00100000 (populated with the value babeface), and 00200000 (populated with the value facebabe). The DMA transfers the data via the SRIO port 1. The address a1000000 (effective address) is accessing the Board B's DDR memory via SRIO outbound window. Check Board B for example, by typing **md 01000000**. The DMA test can also go the other direction—from Board B to Board A. To run the test on Board B, setup the SRIO windows (per steps 1–3 and then complete steps 5–7 for Board B instead of Board A. Text versions of the scripts are available in [Appendix A](#).

Appendix A Text versions of the U-boot scripts

A.1 Text versions of the U-boot scripts

sriol_windows_script.img

```

echo ===== Test SRI01 Link =====

mw.b 700000 02                #Saving value of 0x02 to DDR Memory at location 0x700000
md.b fe0c015b 1              #Reading bit 30, checking for PO (PORT OK)
cmp.b 700000 fe0c015b 1;     #if 0x02 = bit 30, SRI0 link up , otherwise test fails.
if test $? -eq 0; then echo SRI01 LINK UP;
else echo SRI01 LINK DOWN, TEST FAILED;
exit
fi;

echo ===== SRI01LB Test =====
echo ===== Setting up Outbound and Inbound Windows =====
mw.l fe0d0c20 3fc10000 #target ID for a large transport system
mw.l fe0d0c24 0
mw.l fe0d0c28 00c20000 #SRIO port 1 U-boot physical address 0xC20000000
mw.l fe0d0c2c 0
mw.l fe0d0c30 80045019 # (8)Port1 Outbnd Trans enbld, (4)NREAD (5)NWRITE (19)wndw size
mw.l fe0d0dc8 0001000 # Port1 Trans addr 0x1000000 for SRI01
mw.l fe0d0dc0 0
mw.l fe0d0dd0 80f55019 # (8)Port 1 Inbound Trans Addr enabled, (f) local
                        address space, (5) read w/snoop, (5) write with snoop (19)
                        windows size

```

dma_SRI01.img

```

echo ===== DMA Desc 1 =====
mw.l 100000 0 1000          #Init to zero to a block of mem in DDR at address 100000
mw.l 100000 babeface 1000   #Write "babeface" to addr 100000 (Descriptor 1 src addr)
mw.l 200000 facebabe 1000   #Write "facebabe" to addr 200000 (Descriptor 1 src add)

```

```

mw.l 1000000 00050000      #(SATRn), set to 0101, Read, snoop local processor
mw.l 1000004 100000      #(SARn), set to address that is init above 0x100000
mw.l 1000008 0005000C     #(DATRn), set to 0101, wrt snoop, C is phys ext addr bit.
mw.l 100000c 21000000     #(DARn), set to init physical addr for SRI01 21000000
mw.l 1000010 00000000     #ENLNDARn,not using ext addr,not last link descriptor
mw.l 1000014 01000020     #NLNDARn),addr of the next descriptor(2) in mem
mw.l 1000018 00001000     #Byte count (BCRn), bytes for DMA to transfer
mw.l 100001c 00000000     #Reserved
echo ===== DMA Desc 2 =====
mw.l 1000020 00050000     #(SATRn), set to 0101, Read, snoop local processor
mw.l 1000024 200000      #(SARn), set to address that is initialized above 0x200000
mw.l 1000028 0005000C     #(DATRn), set to 0101, wrt snoop, C is physical ext addr bit
mw.l 100002c 21002000     #(DARn), set to init physical address for SRI01 0x2100000
mw.l 1000030 00000001     #ENLNDARn,not using ext addr,set to 1 for last link descriptor
mw.l 1000034 00000001     #NLNDARn), EOLND bit 31 set to 1 as this is last descriptor
mw.l 1000038 00001000     #Byte count (BCRn), bytes for DMA to transfer
mw.l 100003c 00000000     #Reserved

echo ===== Set DMA MR[CS] to zero, set DMA CLNDAR =====
mw.l fe10010c 01000000
mw.l fe100100 0f0003C0

echo ===== Set MR[CS] bit, starts the transfer =====
mw.l fe100100 0f0003C1

echo ===== Read the block of source data from DDR address 0x10000 =====
md.l 100000 10
echo Read back the same block of data from SRI01 for DMA data transfer comparison
md.l a1000000 10
echo Read the second block of source date from DDR address 0x200000
md.l 200000 10
echo Read back the same block of data from SRI01 for DMA data transfer comparison
md.l a1002000 10

cmp.l 100000 a1000000 100;
if test $? -eq 0;
then echo ===== SRI01 DMA data transfer 1 PASSED! =====;
else echo ===== ERROR: SRI01 DMA descriptor 1 data transfer FAILED! =====;
setenv ERRCODE 1
exit;
fi;
cmp.l 200000 a1002000 100;
if test $? -eq 0;
then echo ===== SRI01 DMA data transfer 2 PASSED! =====;
else echo ===== ERROR: SRI01 DMA descriptor 2 data transfer FAILED! =====;
setenv ERRCODE 1
exit;
fi;

```

Appendix B Revision history

B.1 Revision history

This table summarizes revisions to this document.

Table B-1. Revision history

Revision	Date	Description
0	07/2014	Initial public release.

How to Reach Us:

Home Page:

freescale.com

Web Support:

freescale.com/support

Information in this document is provided solely to enable system and software implementers to use Freescale products. There are no express or implied copyright licenses granted hereunder to design or fabricate any integrated circuits based on the information in this document. Freescale reserves the right to make changes without further notice to any products herein.

Freescale makes no warranty, representation, or guarantee regarding the suitability of its products for any particular purpose, nor does Freescale assume any liability arising out of the application or use of any product or circuit, and specifically disclaims any and all liability, including without limitation consequential or incidental damages. "Typical" parameters that may be provided in Freescale data sheets and/or specifications can and do vary in different applications, and actual performance may vary over time. All operating parameters, including "typicals," must be validated for each customer application by customer's technical experts. Freescale does not convey any license under its patent rights nor the rights of others. Freescale sells products pursuant to standard terms and conditions of sale, which can be found at the following address: freescale.com/SalesTermsandConditions.

Freescale, the Freescale logo, AltiVec, CodeWarrior, Energy Efficient Solutions logo, and QorIQ are trademarks of Freescale Semiconductor, Inc., Reg. U.S. Pat. & Tm. Off. CoreNet is a trademark of Freescale Semiconductor, Inc. All other product or service names are the property of their respective owners. The Power Architecture and Power.org word marks and the Power and Power.org logos and related marks are trademarks and service marks licensed by Power.org.

© 2014 Freescale Semiconductor, Inc.

