

Introduction to Zynq Hardware

Lab 6

Improving Data flow between PL and PS utilizing PS DMA



November 2013
Version 03

Lab 6 Overview

In the last lab, we expanded our block design by extending our memory space with a PL-based Block RAM (BRAM). The BRAM can be used to buffer data going between the PS and PL. In this lab, we'll add a software application that enables the PS DMA engine to show how efficiency gains can be achieved when passing data between the PL-based BRAM and external DDR3 memory.

Lab 6 Objectives

When you have completed Lab 6, you will know how to do the following:

- Create a new C Software Application and import C source code
- Use PS DMA and GIC controllers
- Perform DMA operations using the PS DMA

Experiment 1: Test PS DMA Controller

This experiment shows how the PS DMA Controller can improve data transactions between the PS and PL.

Experiment 1 General Instruction:

Create new C application, import C-code (dma_test) to test DMA transaction.

Experiment 1 Step-by-Step Instructions:

1. <Optional> If you did not complete Lab 5 or wish to start with a clean copy, delete the ZynqDesign and SDK_Workspace folders in the ZynqHW/2013_3 folder. Then unzip **Solutions\ZynqHW_Lab5_Solution.zip** to the 2013_3 folder. If you have 7-zip installed, you can do this by right-clicking and dragging **ZynqHW_Lab5_Solution.zip** to the 2013_3 folder. Select **7-Zip → Extract Here**. (If the BSP does not build, try cleaning the SDK workspace (Project → Clean.)
2. In Vivado, **Open Implemented Design**. Click **Reload** if necessary.

3. **Export design** for SDK (File → Export → Export Hardware for SDK) and select a new software workspace, name it **SDK_BRAM**. Make sure you export the bitstream as well as we have IP in the PL now.

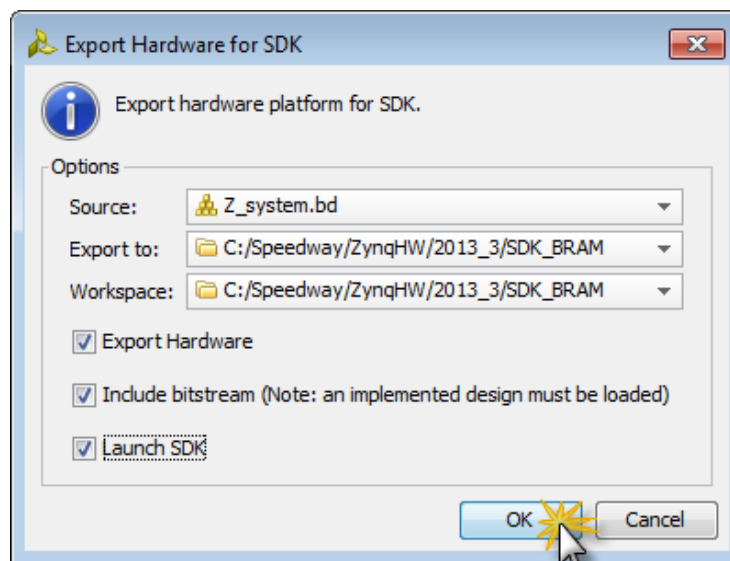
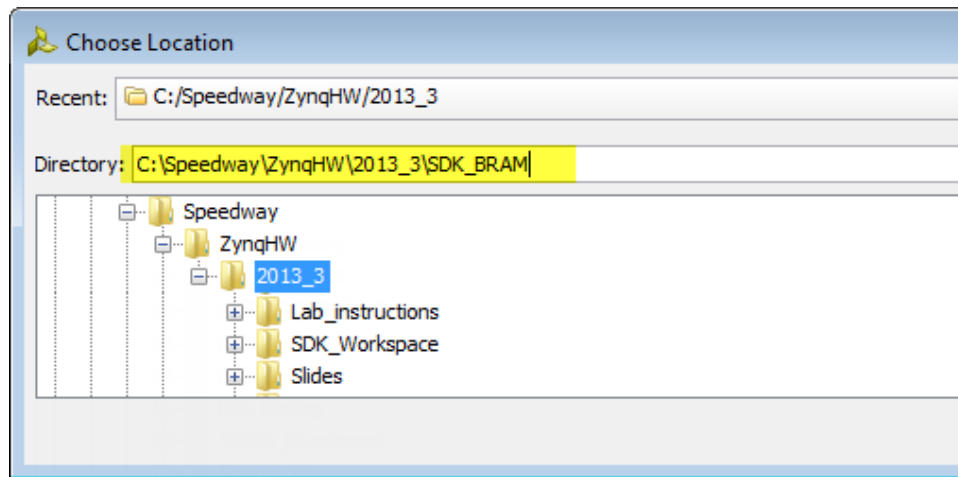


Figure 1 - New SDK Workspace

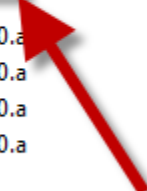
We've created a new workspace and the reason why is because it is good practice to do this every time we create a new hardware platform. Creating new workspaces allow software engineers the ability to revert back to the previous platform in the event the new hardware platform has issues.

Creating a new workspace means our existing projects will not appear in this workspace. Also with a new workspace we must create a new BSP. If you wish to bring your existing projects into this workspace, you can import those using *File → Import → General → Existing Projects into Workspace*.

4. Verify your Hardware Platform changes were accepted by SDK. View the **system.xml** file. If not open already, expand *HW_platform_0*, open **system.xml**. Verify the IP versions were loaded for the BRAM, if not, close SDK and export hardware platform from Vivado again.

IP blocks present in the design

axi_bram_ctrl_0	axi_bram_ctrl	3.0
axi_interconnect_0_s00_couplers_auto_pc	axi_protocol_converter	2.1
axi_interconnect_0_s00_couplers_auto_us	axi_dwidth_converter	2.1
blk_mem_gen_0	blk_mem_gen	8.0
ps7_uart_1	ps7_uart	1.00.a
ps7_qspi_0	ps7_qspi	1.00.a
ps7_qspi_linear_0	ps7_qspi_linear	1.00.a
ps7_axi_interconnect_0	ps7_axi_interconnect	1.00.a
ps7_cortexa9_0	ps7_cortexa9	5.2
ps7_cortexa9_1	ps7_cortexa9	5.2
ps7_dds_0	ps7_dds	1.00.a
ps7_ethernet_0	ps7_ethernet	1.00.a



IP Versions

Figure 2 - Hardware Platform – IP Version for BRAM

5. To test our BRAM, software code is provided; however a C application project needs to be created first. Create a new application project (**File → New → Application Project**)
6. Enter **BRAM_DMA_Test** for the *project name* and select **Create New BRAM_DMA_Test_bsp**. Click **Next >**.
7. Select **Empty Application**, click **Finish**.

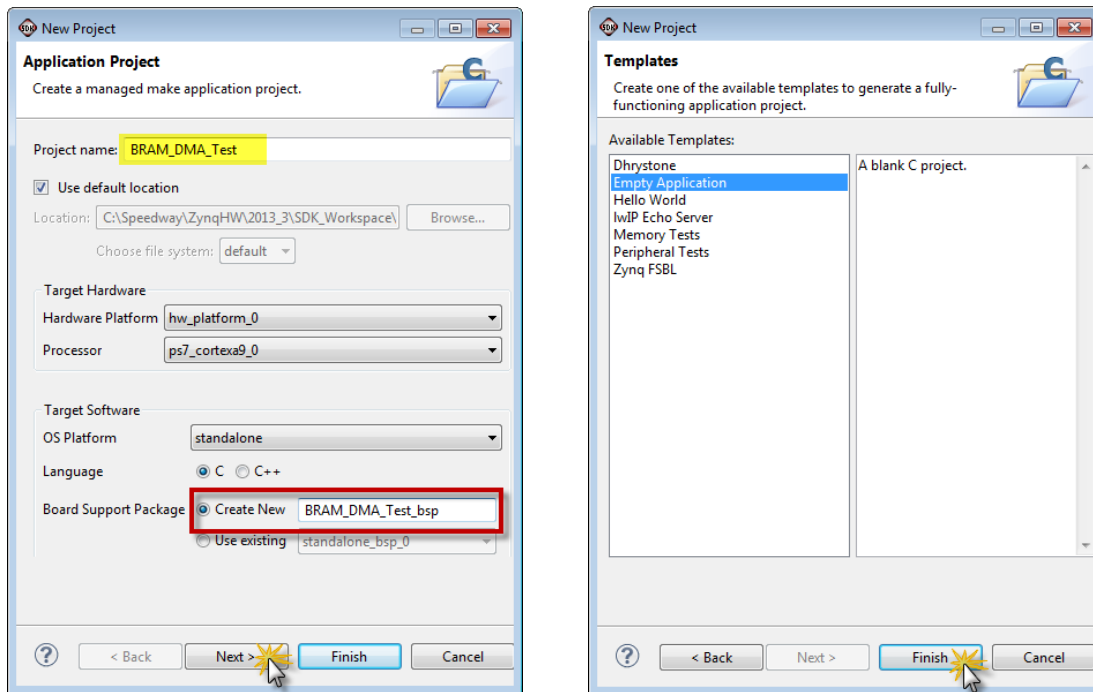


Figure 3 - Create New Application

It may take a minute to build the new BSP.



Figure 4 - Building new Workspace

8. In the Project Explorer Window, expand **BRAM_DMA_TEST**, right-click on **src** and choose **Import**.

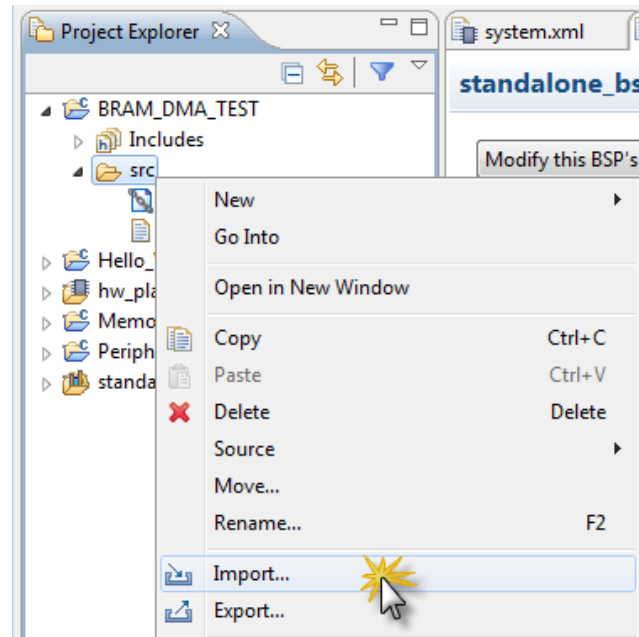


Figure 5 - Import code source

9. In Import window, expand **General**, select **File System**, then click **Next >**.

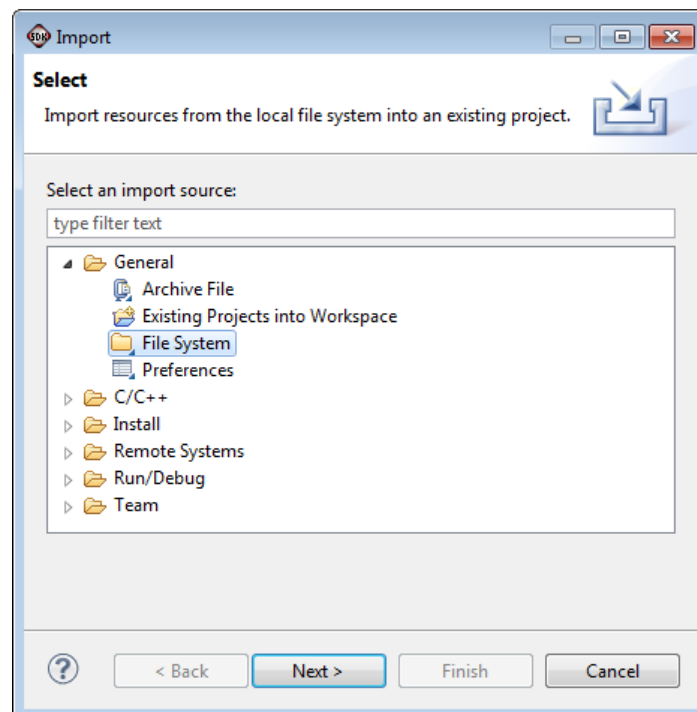


Figure 6 - Import File System

- Click the Browse button and navigate to the support documents directory, **C:\Speedway\ZynqHW\2013_3\Support_documents**. Select the checkbox next to **dma_test.c**, then click **Finish**.

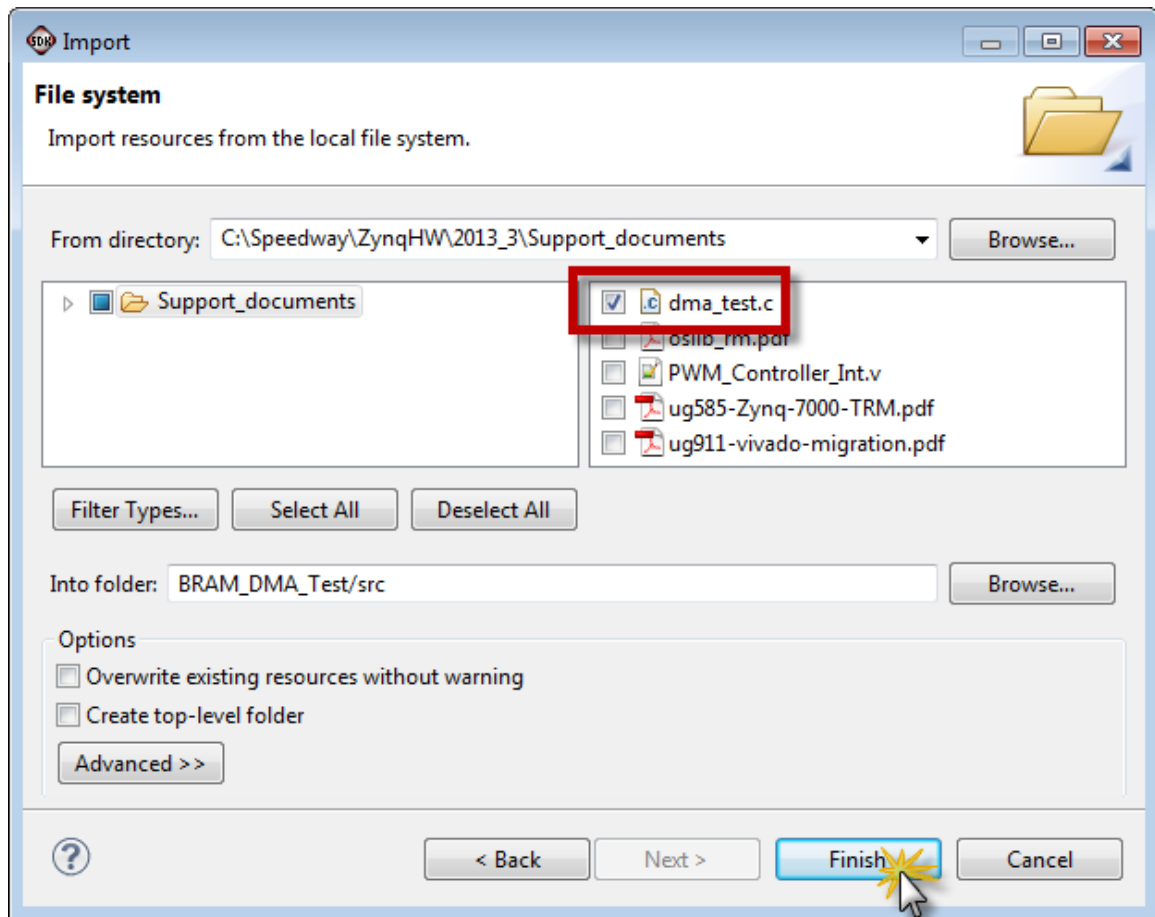


Figure 7 - Import C Source

Immediately the application will start building, when complete you should see the result in the Console. (The warnings can be safely ignored.)

```
arm-xilinx-eabi-gcc -Wl,-T -Wl,../src/lscript.ld -L../BRAM_DMA_Test_bsp/ps7_cortexa9_0/lib -o  
"BRAM_DMA_Test.elf" ./src/dma_test.o -Wl,--start-group,-lxil,-lgcc,-lc,--end-group  
'Finished building target: BRAM_DMA_Test.elf'  
,,  
'Invoking: ARM Print Size'  
arm-xilinx-eabi-size BRAM_DMA_Test.elf |tee "BRAM_DMA_Test.elf.size"  
text    data    bss     dec     hex filename  
41224   1912   32356   75492   126e4 BRAM_DMA_Test.elf  
'Finished building: BRAM_DMA_Test.elf.size'  
,,  
  
19:36:44 Build Finished (took 1s.570ms)
```

Figure 8 - Application Build Results

11. Now that we are utilizing the PL, it must be programmed. Click the **Program FPGA** button on the top shortcut bar:

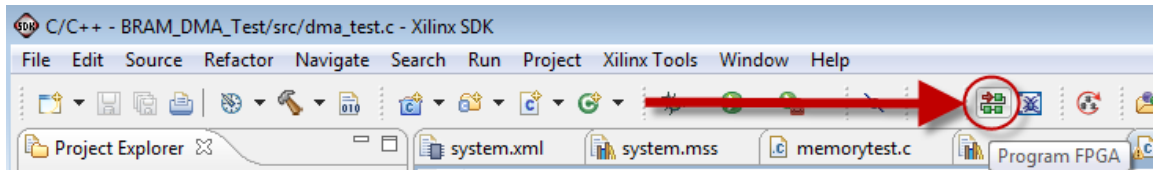


Figure 9 - Program FPGA

12. The default program options are OK. Click **Program**. Note: we are using the hardware platform found in the new workspace.

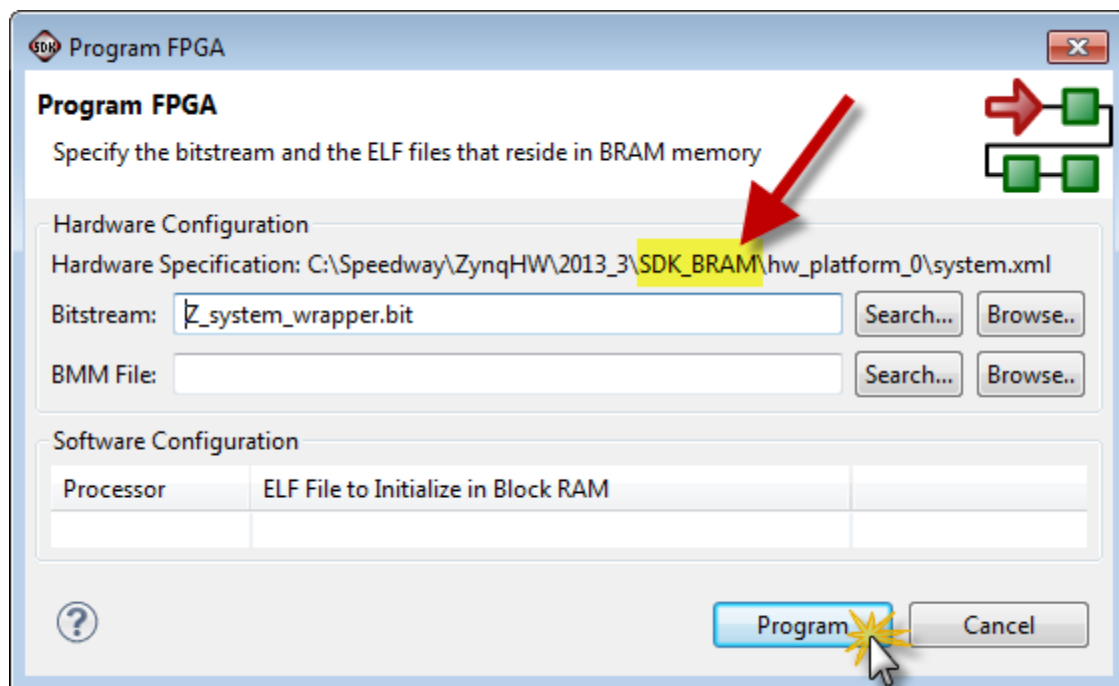
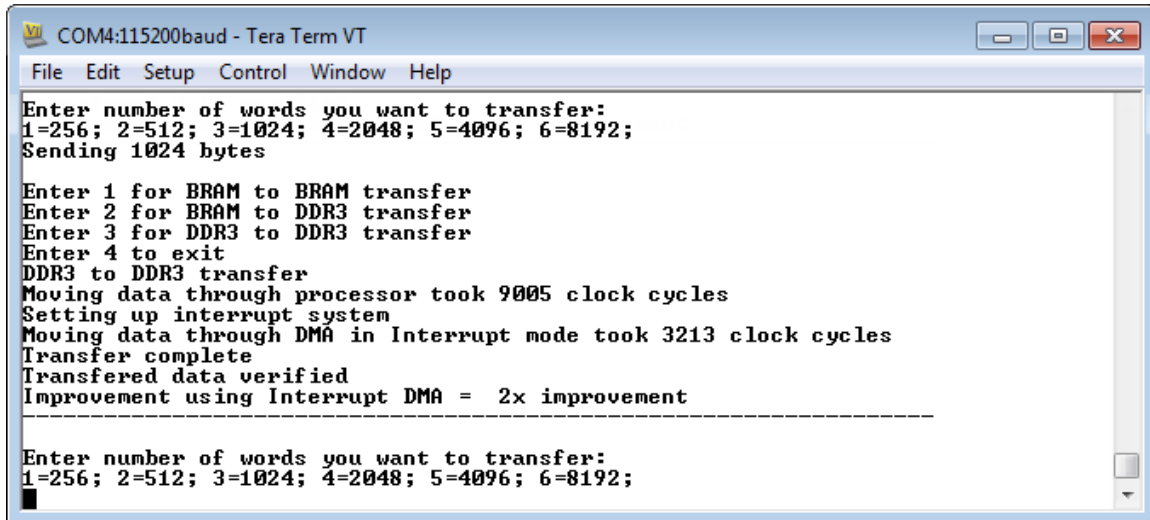


Figure 10 - Program FPGA Options

13. Open Terminal, such as **Tera Term**, and set the **COM port** to active COM setting for your board and set the **Baud Rate at 115,200**.
14. Right-click on **BRAM_DMA_Test** and select **Run As → Launch on Hardware (GDB)**.

15. Once the code is downloaded to the board, TeraTerm should show the burst size prompt. Explore different byte sizes and transfer modes to compare the improvement using the DMA.



```
COM4:115200baud - Tera Term VT
File Edit Setup Control Window Help
Enter number of words you want to transfer:
1=256; 2=512; 3=1024; 4=2048; 5=4096; 6=8192;
Sending 1024 bytes

Enter 1 for BRAM to BRAM transfer
Enter 2 for BRAM to DDR3 transfer
Enter 3 for DDR3 to DDR3 transfer
Enter 4 to exit
DDR3 to DDR3 transfer
Moving data through processor took 9005 clock cycles
Setting up interrupt system
Moving data through DMA in Interrupt mode took 3213 clock cycles
Transfer complete
Transferred data verified
Improvement using Interrupt DMA = 2x improvement
-----
Enter number of words you want to transfer:
1=256; 2=512; 3=1024; 4=2048; 5=4096; 6=8192;
```

Figure 11 - BRAM_DMA_TEST running on Terminal

This lab led you through exercising block RAM connected to a processor system so that you can see how to perform DMA transfers within a memory or between two different kinds of memory. You verified the functionality by using the provided application. You observed that DMA is considerably improved for block RAM compared to DDR because the transactions to block RAM are across the AXI interconnect and block RAM is operating at a slower speed.

16. When done with the questions below, **Close SDK**.

Questions:

Answer the following questions:

- *Was a new BSP required for this application?*

- *What is the performance increase when transferring 8192 bytes from BRAM to DDR3? Try again from DDR3 to DDR3? BRAM to BRAM?*

- *What is the performance increase when transferring 256 bytes from BRAM to DDR3? Try again from DDR3 to DDR3? BRAM to BRAM?*

Exploring Further

If you have more time and would like to investigate more...

- Explore the provided C source code. Examine the PS DMA Configuration.

This concludes Lab 6.

Revision History

Date	Version	Revision
6 Nov 13	02	Initial Draft
19 Nov 13	03	Pilot updates

Resources

www.microzed.org

www.zedboard.org

www.xilinx.com/zyng

www.xilinx.com/sdk

www.xilinx.com/vivado

Answers

Experiment 1

- *Was a new BSP required for this application?*

No. We could have used the BSP created for Hello_World_BRAM in the past lab. That BSP included drivers for the PL BRAM. However, it's not necessarily bad practice to generate a new BSP for each new HW platform.

- *What is the performance increase when transferring 8192 bytes from BRAM to DDR3? Try again from DDR3 to DDR3? BRAM to BRAM?*

11x, 3x, 20x (Results may vary)

- *What is the performance increase when transferring 256 bytes from BRAM to DDR3? Try again from DDR3 to DDR3? BRAM to BRAM?*

10x, 1x, 18x (Results may vary)