EE 577B Project Phase2 Part1
Spring 2014 Nazarian

Score:___/100

Student ID: _____ Name: _____
Assigned: Friday, Mar. 7th, 2014
Due: Friday, Mar 28th at 11:59pm (View/Complete)

You have the option of working individually, or in a team of two students. However, you will be tested individually on the project. Every team member must understand every part of the project including the parts that their teammate was the primary designer.

Introduction

Phase2 of the DDR2 Controller Design Project assumes that students have successfully implemented Phase 1(DDR2 Initialization Engine). In Phase 2 of the project, the goal is to implement scalar read and write, block read and write capability and refresh logic. Students should refer to the JEDEC DDR2 SDRAM Standard (JESD79-2C) for all timings, bus interface and specifications. The controller would initialize the DDR2 model (chip) with the given parameters like CAS latency and Burst length etc. from Phase 1. Normal data transactions would start after a successful completion of the DDR2 initialization sequence.

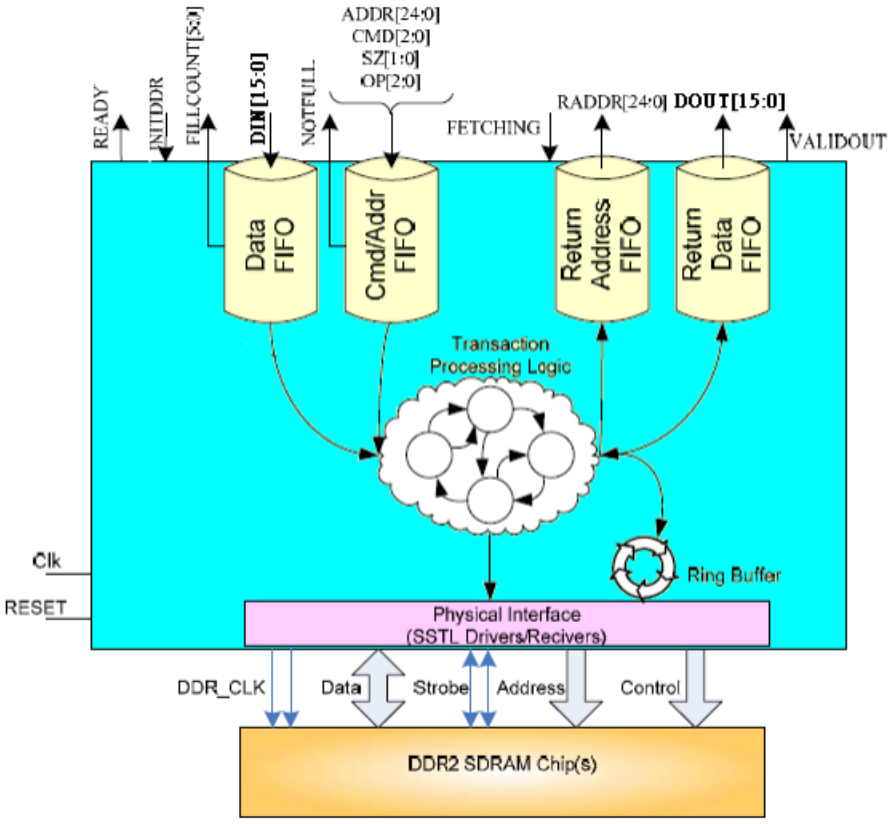

Figure 1

1, Implement the RTL for Scalar operations

I/O Name	Width	IN/OUT	Description
CMD[2:0]	3 bit	IN	000: No Operation (NOP) 001: Scalar Read (SCR) 010: Scalar Write (SCW) 011: Block Read (BLR) 100: Block Write (BLW) TBA
SZ[1:0]	2 bit	IN	Not needed for part1

i) **Scalar Read (SCR)**: a 25-bit address is given to the controller at the ADDR[24:0] port. A single 16-bit word is the read output that is read from the DRAM using the specified address. You should pick the correct data from the burst response of Denali. In the DDR2 module datasheet, you can find the timing diagrams corresponding to read-with-auto-precharge and read-without-precharge operations. The former is easier to implement while the latter one could be utilized to improve latency.

ii) **Scalar Write (SCW)**: a 25-bit address and a 16-bit data are given to the controller at ports ADDR[24:0], and DIN[15:0]. The data should be written into the DRAM using the address specified by ADDR, by setting the data mask (DM) and DRAM timing appropriately. Timing diagrams corresponding to write-with-auto-precharge and write-without-precharge operations could be found in the datasheet.

iii) **Ring Buffer**: Clock applied to the controller and to the Denali module have different frequencies. Therefore, you need a medium to synchronize the two frequency domains. This is done using a "ring buffer" which is provided by the TA. It essentially captures 8 words (=Burst Length) of the data that comes out of the DDR2 module (with help of strobe signals) and stores it in a buffer to be read by the controller. The content of memory location of buffer pointed by "readPtr" is available at its data output port, "dout". The controller needs to provide a pulse on the ring buffer's input signal "listen" before data is being read out of DDR2 module (DDR2 module sends data along with strobe when it becomes available). You are strongly recommended to try to understand its functionality by looking into its Verilog code.

Once the data (8 words) is returned from DDR2 module, and is stored in the ring buffer, the controller reads the requested word out of ring buffer and stores it in the Return Data FIFO along with the corresponding address in the Return Address FIFO (shown in Figure 1) to be read by the testbench. You are welcome to modify the Ring Buffer design if you see the need.

.

2, Synthesis of Phase 2 Part 1 and post-synthesis simulation

Synthesize your entire design using Synopsys Design Compiler. You may reuse the synthesis script files used in phase 1. Run post-synthesis simulation to verify the correct operation of your synthesized design. You should use a clock period of 380MHz for synthesis of your controller and maximum delays of 1ns at input and output ports relative to the clock edge.

What you are provided:

ddr2_ring_buffer8.v (complete)
Processing_logic.v (incomplete)
tb.v (complete RTL test bench used for phase2-part1, your design should be 100% match this testbench)
ddr2_controller.tcl (complete tcl file used in synthesis step)
ddr2_test_pattern.dat (input test pattern for phase2-part1)
ddr2_out.dump (output pattern for you to check your result)

What you need to modify:

Processing_logic.v (design the logic to implement Scalar Read/Write)
ddr2_initial_engine.v (change your previous settings, set BL=8, AL=1, CL=3)
ddr2_controller.v (change the FIFO size in order to match test bench, design the logic to implement FULL/EMPTY signal of each FIFO, design the interface of Processing_logic module)
FIFO.v (you may need to re-design your FIFO to meet the timing constraints)

Submission Guidelines

1) Name the top level of your controller as ddr2_controller.v
and your process logic module as process_logic.v

2) 3 folders:
a) RTL folder: the entire pre-synthesis simulation directory
b)SYN folder: tcl script, area report, timing report, other reports
c) POST folder: the entire post-synthesis simulation directory

3) readme.txt

In readme.txt, please state the problem you have in your design.
For example,

Name1:
E-mail1:
Name2:
E-mail2:
Scalar in RTL
Working or NOT:
Reason(s):
Scalar in Post Syn
Working or NOT:
Reason(s):

4) Compress your folders into a tar ball with the gtar command:
aludra > gtar czf First1_Last1_First2_Last2.tgz *

Please check you have include all the 3 folders and readme.txt in the tar ball

5) Submit the tar ball using the View/Complete Links (We have 2 separate ones for each part of phase 2).

6) No report is required.