The goal of the assignment is to develop an understanding of the different sequential circuit design constructs in Verilog, and to synthesis digital circuits using the Altera DE1/DE2 FPGA board.

***Homework 5: PART 1***

To test the Static RAM on the **Altera DE2,** the following PIN I/O assignments were made:



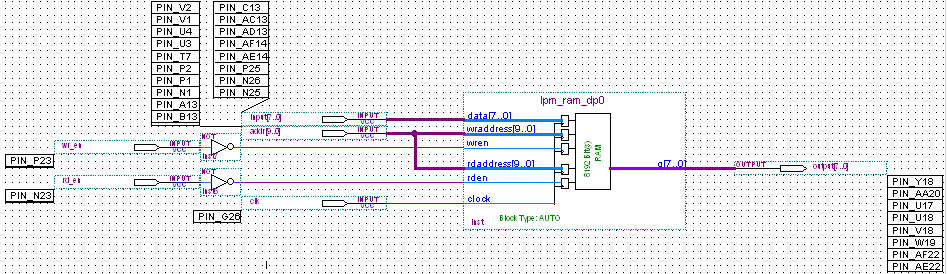
**The project files (DE2 and TestBench) can be found in the Pt1 folder of the ZIP file.**

In order to fit the address and input lines fully into the DE2, the following is my memory model:

* **8192 bit** Static RAM **= 1024** **bit** address [9..0], with **8 bit** words [7..0]
* **addr** and **input** map directly to the **18** switches on the DE2
* *StaticRam.mif* was used to initialize the memory
  + Repeated incremental values of 0 – 255 in order to populate .mif file

Here is a screenshot of the Block Diagram/Schematic file:

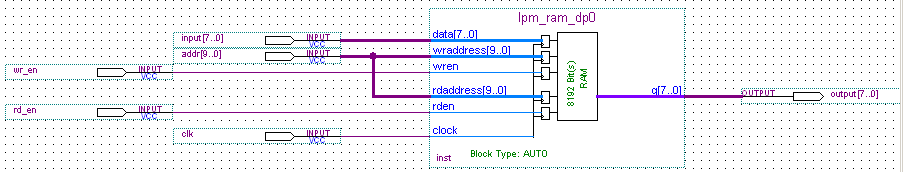
**Block Diagram/Schematic for StaticRAM**



One thing to notice is that both read and write enable signals (rd\_en & wr\_en) have a NOT gate before the input into the lpm\_ram. This was done so that Pushbutton[1] & [0] work as active HIGH. All pins were assigned correctly, and this file works exactly as intended on the DE2.

Here is a screenshot of the Block Diagram/Schematic file for the Testbench, without the NOT gates:

**Block Diagram/Schematic for StaticRAM\_Tester**



I decided to develop my TestBench in Quartus II using a Vector Waveform to prove the functionality of the sequential circuit. For reference, the *.mif file* used initialize 3 addresses in the memory to the following three values:

Memory @ (D) 1023, (B) 1111111111 = (D) 255, (B) 11111111

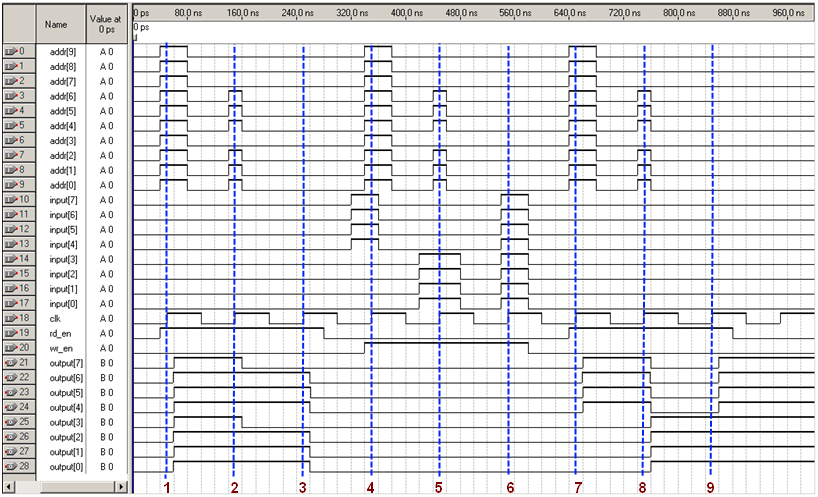
Memory @ (D) 119, (B) 0001110111 = (D) 119, (B) 01110111

Memory @ (D) 0, (B) 0000000000 = (D) 0, (B) 00000000

My test bench instantiates a clock and utilized the first 9 positive edges to perform operations:

1. PosEdges 1 – 3: memory is read when ***rd\_en*** =1; shown on ***output*** bus at the specified ***addr***
2. PosEdges 3 – 6: memory is written when ***wr\_en*** =1 at the specified ***addr***; ***output*** has no values
3. PosEdges 7 – 9: *new* memory is read when ***rd\_en*** =1; shown on ***output*** bus at the specified ***addr***

**Waveform Output for StaticRAM\_Tester**



Here is the analysis of the waveform output:

**READ CYCLE #1**

* PosEdge #1 **CORRECT**
  + addr = (D) 1023, (B) 1111111111
  + Read operation performed, Output = (D) 255, (B) 11111111
* PosEdge #2 **CORRECT**
  + addr = (D) 119, (B) 0001110111
  + Read operation performed, Output = (D) 119, (B) 01110111
* PosEdge #3 **CORRECT**
  + addr = (D) 0, (B) 0000000000
  + Read operation performed, Output = (D) 0, (B) 00000000

**WRITE CYCLE**

* PosEdge #4 **CORRECT**
  + addr = (D) 1023, (B) 1111111111
  + Write operation performed, addr = input = (D) 240, (B) 11110000
* PosEdge #5 **CORRECT**
  + addr = (D) 119, (B) 0001110111
  + Write operation performed, addr = input = (D) 15, (B) 00001111
* PosEdge #6 **CORRECT**
  + addr = (D) 0, (B) 0000000000
  + Write operation performed, addr = input = (D) 255, (B) 11111111

**READ CYCLE #2**

* PosEdge #7 **CORRECT**
  + addr = (D) 1023, (B) 1111111111
  + Read operation performed, Output = (D) 240, (B) 11110000
* PosEdge #8 **CORRECT**
  + addr = (D) 119, (B) 0001110111
  + Read operation performed, Output = (D) 15, (B) 00001111
* PosEdge #9 **CORRECT**
  + addr = (D) 0, (B) 0000000000
  + Read operation performed, Output = (D) 255, (B) 11111111

Based on the outputs displayed, the memory is initialized correctly, as the expected values are seen at the specified addresses for Read Cycle #1. Read Cycle #2 displays the correct functionality of the write function of the sequential logic, as values are overwritten correctly in the Write Cycle.

This concludes the analysis for Homework 5, Part 1.

***Homework 5: PART 2***

The Verilog code for this section can be found in the Pt2 folder of the ZIP file submitted:

*minMemValue.v*

*minMemValueTester.v*

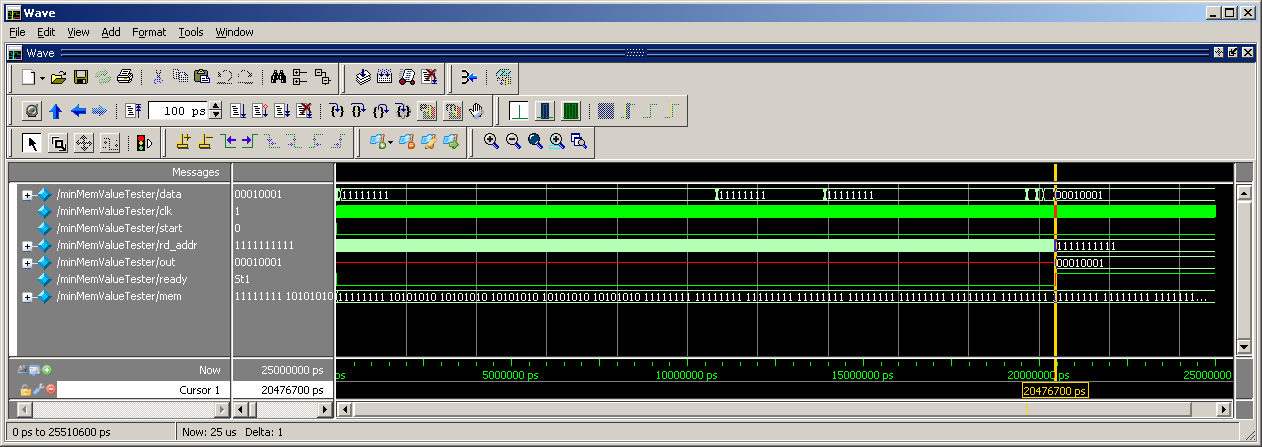
An input file, *mem.dat*, is used to initialize an instantiated memory like the one in Pt1:

*reg [7:0] mem [0:1023];*

The smallest value contained in the 1024 values of *mem.dat* is **(H) 11**, or **(B) 00010001**.

In running the minMemValueTester module in ModelSim, the following waveform was generated:

**Simulation Waveform Output of minMemValueTester**



A clock is instantiated at the following frequency:

*initial repeat (2500) #10 clk <= ~clk;*

As seen in the *minMemValue.v* code, a new value of the memory is read in at every positive edge of the clock. After the value is scanned, As soon as the last *rd\_addr* becomes *10’b1111111111*, the minimum value is output from the circuit. The output above displays out = 00010001, which is correct.

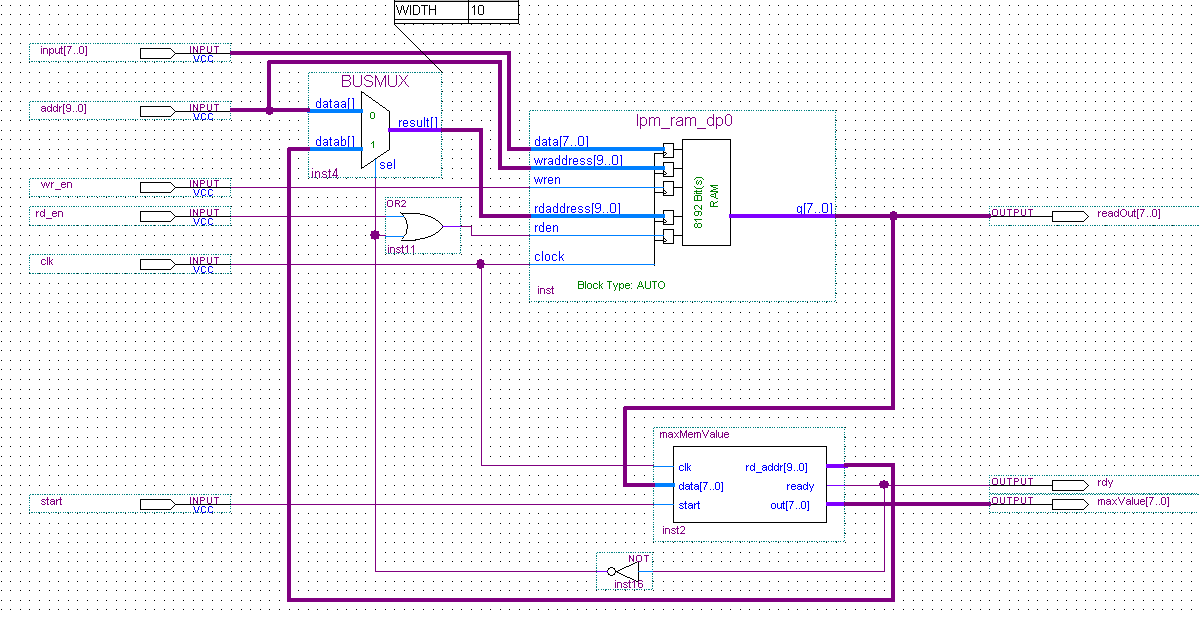
This concludes the analysis for Homework 5, Part 2.

***Homework 5: PART 3***

A testbench was first built in order to test the functionality of this circuit implementation.

Here is a screenshot of the Block Diagram/Schematic file:

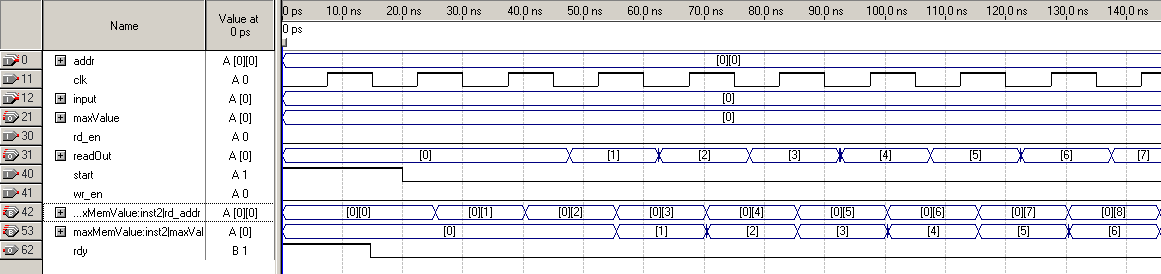
**Block Diagram/Schematic for MaxValStaticRAMTester**



In generating my simulation waveform, it was apparent that I was getting timing errors when having the period of the simulated clock set too fast. With this being the case, I chose to look at internal signals from the different functional blocks in the .bdf to ensure the code was working correctly.

After generating a simulated waveform, the following output is produced:

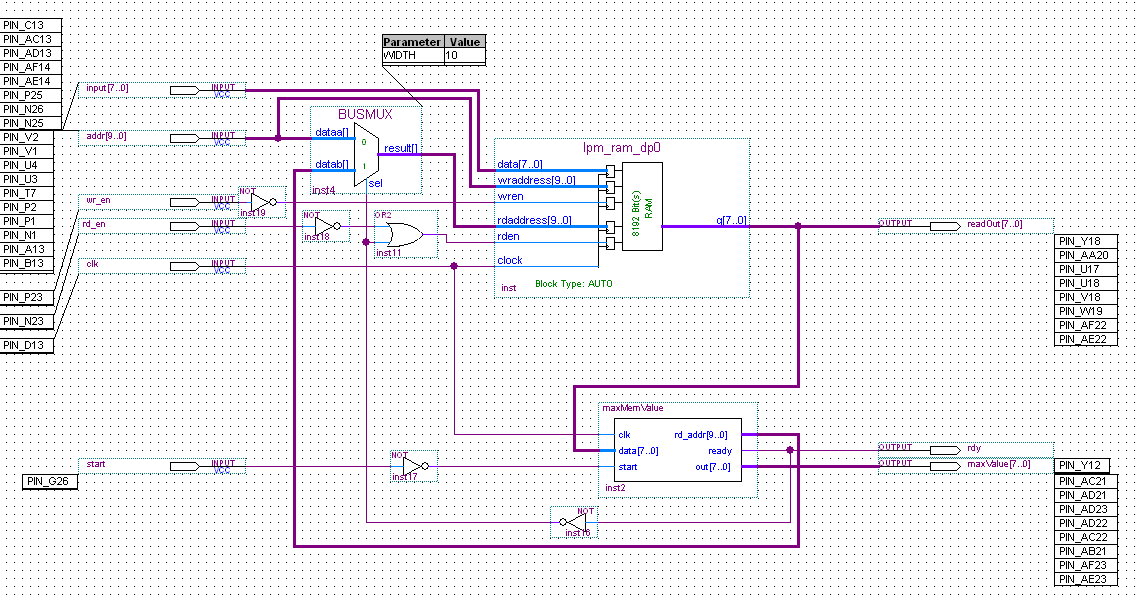
**Simulation Waveform Output of MaxValueStaticRAMTester**



As seen above, the counting of the circuit (*rd\_addr*) begins incrementing on the clock pulse after *start* is asserted to 1. With each positive edge of the clock, *readOut*, the output of the static RAM, is displaying values correctly. These are then fed into the *maxMemValue* logic block created and is processed.

Upon creation of the Testbench, the circuit was modified slightly to include all FPGA pins, with inverters:

**Block Diagram/Schematic for MaxValStaticRAMTester**



To test the MaxValueStaticRAM on the **Altera DE2,** the following PIN I/O assignments were made:



In loading the project onto the Altera DE2 board, the simulation works as expected.

* Both the read/write functionally of the RAM itself is maintained, through use of the multiplexer selection criteria.
* An internal clock is being used, so that the user does not need to worry about manually clocking the sequential circuit.
* The highest value in the *.mif* file used to load the static RAM is 255 (8’b11111111):
  + When the *start* pushbutton[0] is hit, the ready indicator flashes quickly, as expected based on the clock frequency, to indicate that the search is complete.
  + Upon completion, 8 red LEDs are lit up, to indicate the highest value of 255.
  + Upon completion, 8 green LEDs are lit up, to indicate the last value ready in the RAM.
* In order to **fully ensure** the fact that the maximum value was being read correctly, and that the green and red LEDs were not linked:
  + After the first iteration on *start*, *write* the value 8’b00001111 into the RAM at the last address in the RAM, 10’b1111111111.
  + Now hit *start* again 🡪 output is now:
    - 8 red LEDs, to signify maximum value of 255 in RAM.
    - 4 green LEDs, in configuration 8b’00001111, to indicate the last value read.

After running the circuit through a variety of tests, it was determined that it works, fully, as intended.

This concludes the analysis for Homework 5, Part 3.