Lab 2

Cameron Jennings (ID: 2029631), Donovan Clay (ID: 2276005)

October 13, 2023

CSE 371

Contents

1 Design Procedure

In this lab, we were tasked to implement computer memory or RAM. The memory module is to have 32 addresses and 3 bits per address to store data. The inputs of the RAM are a signal to determine if we are reading or writing a value, data to fill the bits in an address when the system is notified to write, and an address to determine what address we are reading or writing. Additionally, the system uses switches and HEX displays of the DE1_SoC board to process data. The switches control the address, data, and write signal being input to the system, while the HEX displays output what address(es) is being used and the number within that address.

In total, there were three different tasks to complete. Tasks 1 and 3 consisted of using modules defined in the Quartus library to run our system. Task 2 was to be implemented using SystemVerilog on our own.

1.1 Task #1

The main purpose of this task was to familiarize ourselves with the library/IP catalog of the Quartus software. We navigate to the on-chip memory and initialize the RAM: 1-PORT file with all the proper options for set-up. Utilizing this module is simply done by creating another module to provide inputs and outputs for the system. Finally, a test bench was created to ensure the module is running properly. There is no diagram or state machine for this task.

1.2 Task #2

The next task required us to build our own version of task #1 using SystemVerilog. Using the same inputs and outputs copied over from the previous tasks, additional logic is added to the module to track the state of all the registers and the address passed into the system. The state of the system is then logged using flip-flops, which is also necessary to ensure that the system is synchronous. The state of the machine is as follows...

Add state diagram with caption and description

1.3 Task #3

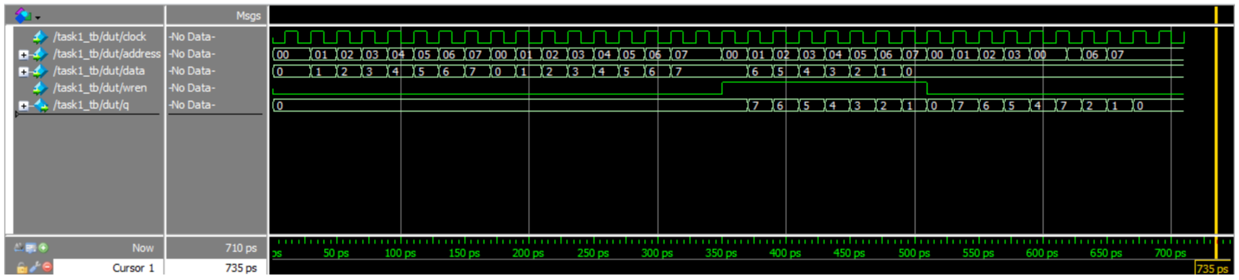
The final task combined the last two steps. This time, we were to initialize a RAM: 2-PORT file similar to task #1 and additionally create a memory initialization file (.mif) in Quartus to provide our memory with data. There are not many implementation details as we only needed another module to run our RAM system, so a DE1_SoC module, similar to the one used to run task #2, was created to provide inputs and outputs from the controller and interconnect the other modules. There is no diagram or state machine for this task. There is also a counter which cycles through the possible addresses. A clock divider module was used to slow down the clock for the counter which cycles through the addresses to read. This module also uses a D flip flop to resolve metastability issues with the keys on the DE1_SoC.

2 Results

As described before, the purpose of this lab was to model and implement RAM using the Quartus library and SystemVerilog. Considering the lab was more of a walkthrough than an assignment, there is not much to be said about the implementation of the individual systems that was not mentioned above. The DE1_SoC modules are where a majority of the implementation occurred as well as the task benching.

2.1 Task #1

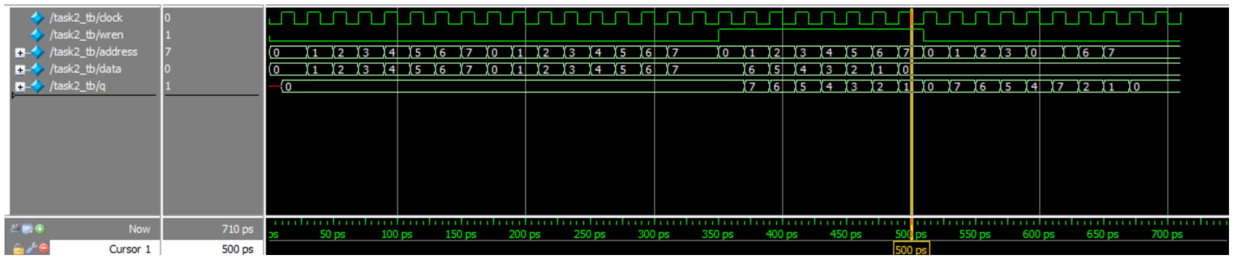
The test bench for task #1 can be seen below.



This testbench demonstrates the expected behavior of the ram module. It shows that when the ram module is first instantiated, the values at the addresses are 0. It also shows that nothing is written to the memory when `wren` is not on. It then shows data being written to the memory. Finally, it shows the data being read is the same data that was written.

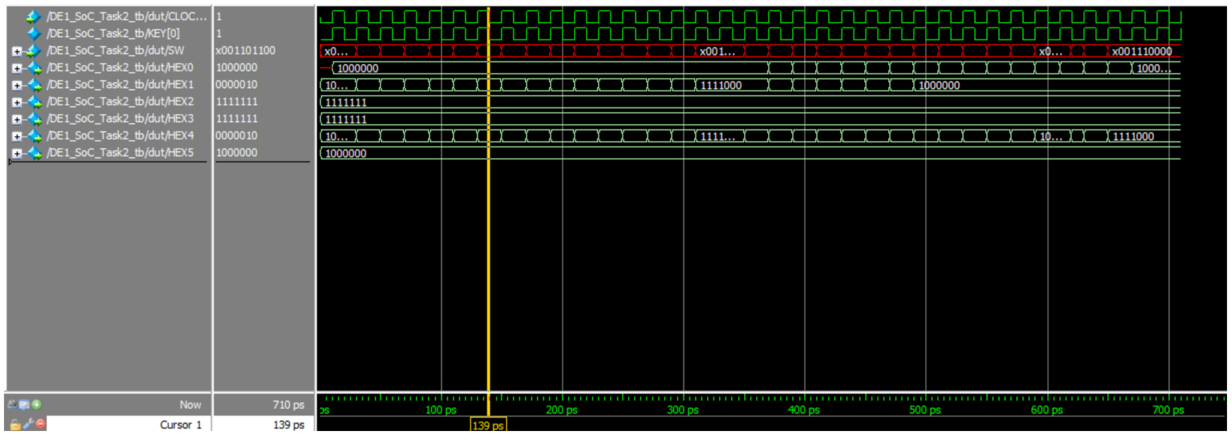
2.2 Task #2

The test bench for the task #2 module can be seen below.



2.3 DE1_SoC Task #2

The test bench for the DE1_SoC module of task #2 can be seen below.

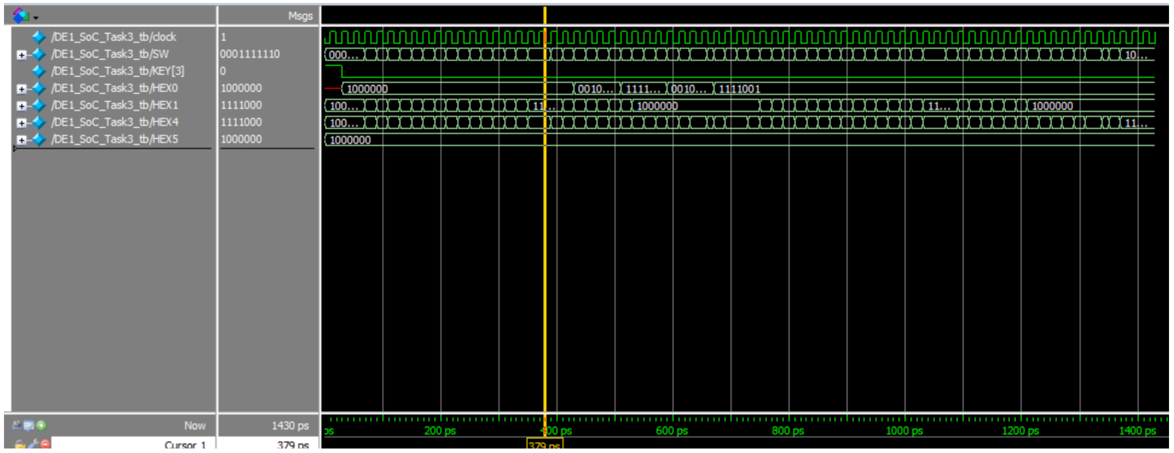


```
/DE1_SoC_Task2_tb/dut/CLOC...   1
/DE1_SoC_Task2_tb/KEY[0]        1
/DE1_SoC_Task2_tb/dut/SW        x001101100
/DE1_SoC_Task2_tb/dut/HEX0      1000000
/DE1_SoC_Task2_tb/dut/HEX1      0000010
/DE1_SoC_Task2_tb/dut/HEX2      1111111
/DE1_SoC_Task2_tb/dut/HEX3      1111111
/DE1_SoC_Task2_tb/dut/HEX4      0000010
/DE1_SoC_Task2_tb/dut/HEX5      1000000
Now        710 ps
Cursor 1   139 ps
```

The inputs for this system were SW3-SW1 to specify dataIn, SW8-SW4 to specify address, and SW0 for write. The output of the system was to the HEX displays, HEX5-HEX4 for address, HEX1 for dataIn, and HEX0 for dataOut. The hex displays were controlled using a seg7 module. This testbench demonstrates that all the modules are connected correctly. The hex displays update correctly when the switches change and when data is written or read.

2.4 DE1_SoC Task #3

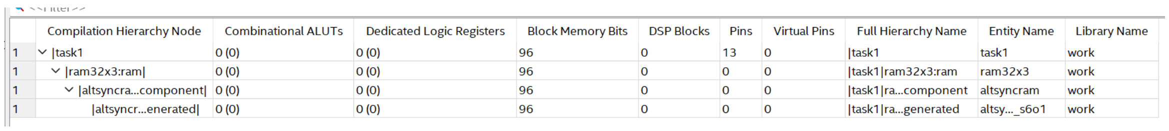
The test bench for the DE1_SoC module of task #3 can be seen below.



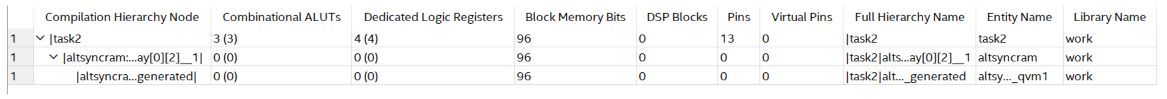
```
                                Msgs
/DE1_SoC_Task3_tb/clock         1
/DE1_SoC_Task3_tb/SW            0001111110
/DE1_SoC_Task3_tb/KEY[3]        0
/DE1_SoC_Task3_tb/HEX0          1000000
/DE1_SoC_Task3_tb/HEX1          1111000
/DE1_SoC_Task3_tb/HEX4          1111000
/DE1_SoC_Task3_tb/HEX5          1000000
Now        1430 ps
Cursor 1   379 ps
```

The inputs for this system were SW3-SW1 to specify writeData, SW8-SW4 to specify write address, and SW9 to switch from task #2 RAM to task #3 RAM or vice versa. The output of the system was to the HEX displays, HEX5-HEX4 for write address, HEX1 for writeData, HEX3-HEX2 for the read address, and HEX0 for read output. The hex displays again were controlled using a seg7 module.

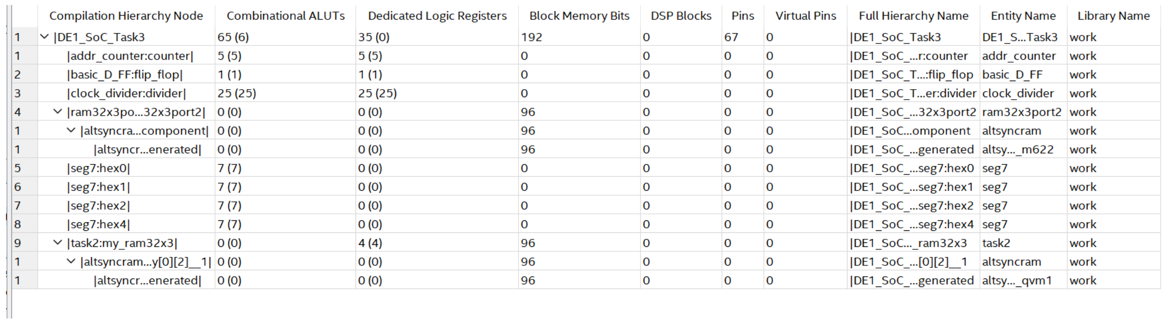
2.5 Flow Summary



	Compilation Hierarchy Node	Combinational ALUTs	Dedicated Logic Registers	Block Memory Bits	DSP Blocks	Pins	Virtual Pins	Full Hierarchy Name	Entity Name	Library Name
1	|task1	0 (0)	0 (0)	96	0	13	0	|task1	task1	work
1	|ram32x3:ram|	0 (0)	0 (0)	96	0	0	0	|task1|ram32x3:ram	ram32x3	work
1	|altsyncra...component|	0 (0)	0 (0)	96	0	0	0	|task1|ra...component	altsyncram	work
1	|altsyncr...enerated|	0 (0)	0 (0)	96	0	0	0	|task1|ra...generated	altsy..._s6o1	work



	Compilation Hierarchy Node	Combinational ALUTs	Dedicated Logic Registers	Block Memory Bits	DSP Blocks	Pins	Virtual Pins	Full Hierarchy Name	Entity Name	Library Name
1	|task2	3 (3)	4 (4)	96	0	13	0	|task2	task2	work
1	|altsyncram:...ay[0][2]__1|	0 (0)	0 (0)	96	0	0	0	|task2|alts...ay[0][2]__1	altsyncram	work
1	|altsyncra...generated|	0 (0)	0 (0)	96	0	0	0	|task2|alt..._generated	altsy..._qvm1	work



	Compilation Hierarchy Node	Combinational ALUTs	Dedicated Logic Registers	Block Memory Bits	DSP Blocks	Pins	Virtual Pins	Full Hierarchy Name	Entity Name	Library Name
1	|DE1_SoC_Task3	65 (6)	35 (0)	192	0	67	0	|DE1_SoC_Task3	DE1_S...Task3	work
1	|addr_counter:counter|	5 (5)	5 (5)	0	0	0	0	|DE1_SoC_...r:counter	addr_counter	work
2	|basic_D_FF:flip_flop|	1 (1)	1 (1)	0	0	0	0	|DE1_SoC_T...:flip_flop	basic_D_FF	work
3	|clock_divider:divider|	25 (25)	25 (25)	0	0	0	0	|DE1_SoC_T...er:divider	clock_divider	work
4	|ram32x3po...32x3port2|	0 (0)	0 (0)	96	0	0	0	|DE1_SoC_...32x3port2	ram32x3port2	work
1	|altsyncra...component|	0 (0)	0 (0)	96	0	0	0	|DE1_SoC...omponent	altsyncram	work
1	|altsyncr...enerated|	0 (0)	0 (0)	96	0	0	0	|DE1_SoC_...generated	altsy..._m622	work
5	|seg7:hex0|	7 (7)	0 (0)	0	0	0	0	|DE1_SoC_...seg7:hex0	seg7	work
6	|seg7:hex1|	7 (7)	0 (0)	0	0	0	0	|DE1_SoC_...seg7:hex1	seg7	work
7	|seg7:hex2|	7 (7)	0 (0)	0	0	0	0	|DE1_SoC_...seg7:hex2	seg7	work
8	|seg7:hex4|	7 (7)	0 (0)	0	0	0	0	|DE1_SoC_...seg7:hex4	seg7	work
9	|task2:my_ram32x3|	0 (0)	4 (4)	96	0	0	0	|DE1_SoC..._ram32x3	task2	work
1	|altsyncram...y[0][2]__1|	0 (0)	0 (0)	96	0	0	0	|DE1_SoC_...[0][2]__1	altsyncram	work
1	|altsyncr...enerated|	0 (0)	0 (0)	96	0	0	0	|DE1_SoC_...generated	altsy..._qvm1	work

3 Experience Report

This lab took approximately 8 hours, broken down as follows:

Reading: 45 minutes

Design: 30 minutes

Coding: 6 hours

Testing: 1 hours