EE 371 Lab 2 Report

Jiarong Qian

I. Introduction

In this lab, I went through several tasks about RAM. I first used the RAM provided in the IP catalog of Quartus and then implemented a RAM with the same function on my own. I also utilized the IP catalog to realize a two-port RAM and downloaded these designs onto the FPGA board.

II. Lab Procedures

I first started by building a 32x4 RAM with the tool from IP catalog and wrote a SystemVerilog code that uses it. I also put it into a test bench to try different combinations of inputs. Then I downloaded the RAM onto the FPGA board and used switches as binary inputs and hex LEDs to display the addresses and values.

After using the provided tool to build the RAM, I implemented another RAM with similar function by writing a SystemVerilog program. To do so, I constructed an array of 4 bit logic with the size of 32. At each clock cycle, I can access an address and do the read and write. The module worked similarly to the previous built-in one. However, there is a slight difference that in my version of RAM the written data does not update immediately, instead, the new value would appear at the next clock cycle, which isn't surprising since non-blocking assignment only updates value after the clock cycle finishes.

After implementing the one-port RAM, I started working on the two-port version. I used the IP catalog to create the RAM and set up a counter to go through all addresses to do the reading automatically at about 1 address/s. In this way I can fully utilize the two ports to read and write at the same time. I also downloaded the RAM onto the FPGA board. The design is almost the same as the previous ones except that I used extra hex LEDs to display the read address and the read value.

III. Results

I simulated the major modules used in the lab and they worked as expected. In fact, many modules from this lab are used across different tasks so I skipped the trouble of simulating the same stuff repeatedly. After loading the designs to the FPGA board, they all worked as expected.

For task 4, I also used SignalTap II Logic Analyzer to probe the switch value, read address, write address, input data and output data. The result from the analyzer also showed that the system was working properly.

IV. Problem Faced & Feedback

In this lab, one problem I ran into was about using key 0 as clock and reset. Because key 0 is active low, I needed to put a ~ sign in front of it to get the module work correctly. I also used a clock divider to get a clock running at a low rate so that the read address for the two-port RAM can be recognized.

I also had trouble using the SignalTap II Logic Analyzer. The tutorial on the course website told me to add write address, input

and output data, which caused the failure of downloading the design onto the board. Instead of following the tutorial, I ended up importing the addresses and data from the RAM module to the analyzer without declaring them as output ports.

These two issues took me a really long time to finish the lab. I think I spent probably 12 hours on it.

Appendix

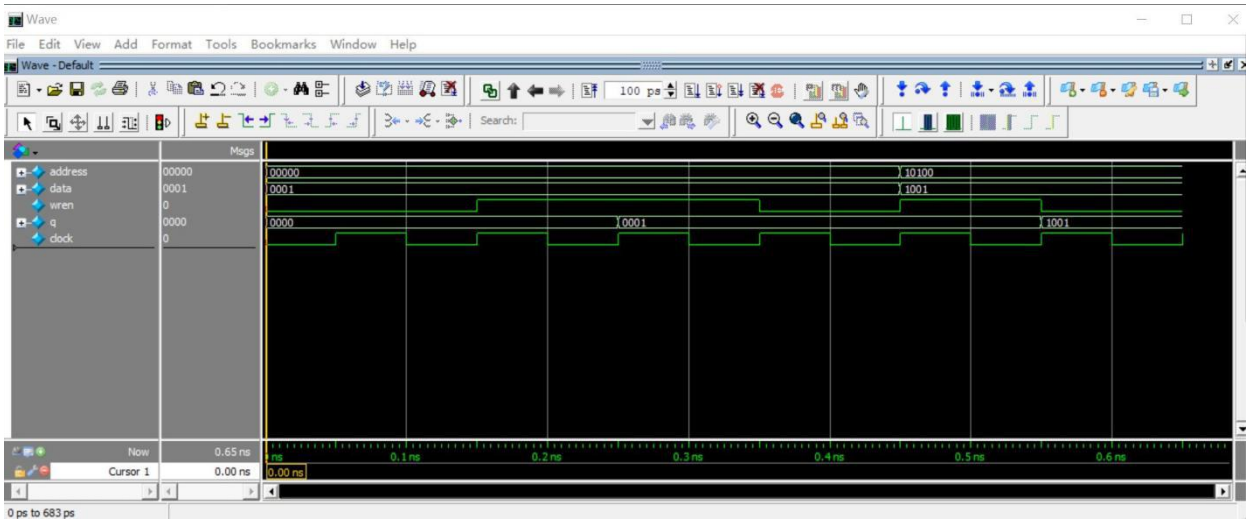

Fig 1. Task 1 Simulation

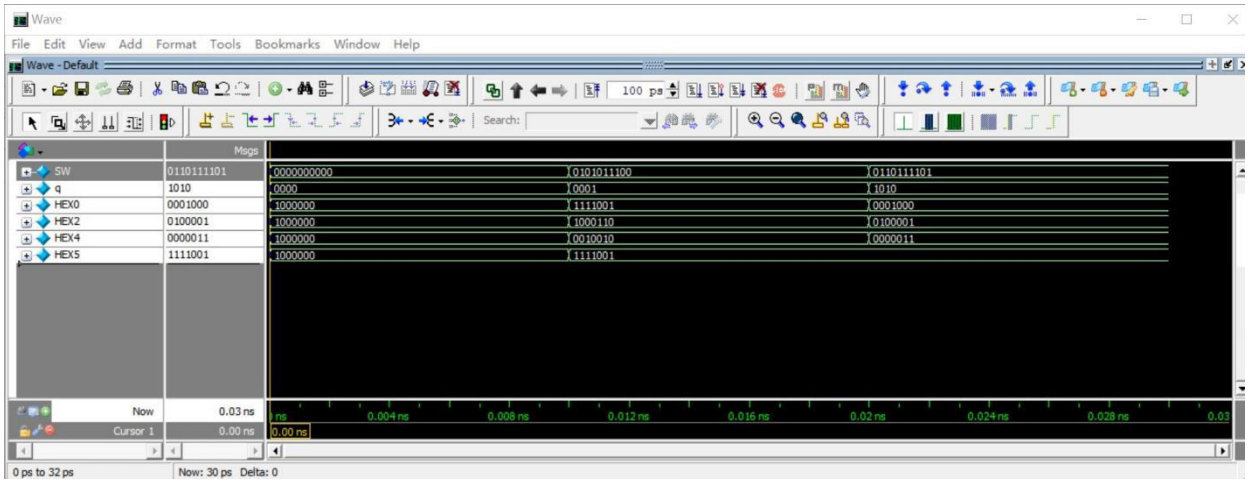

Fig 2. Display (Task 2) Simulation

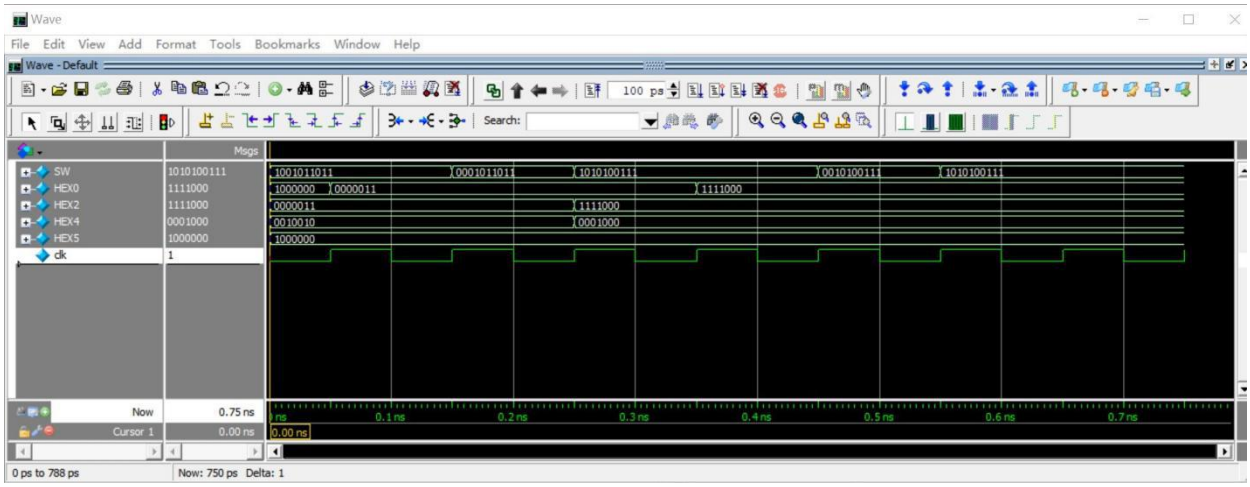

Fig 3. Task 2 Simulation

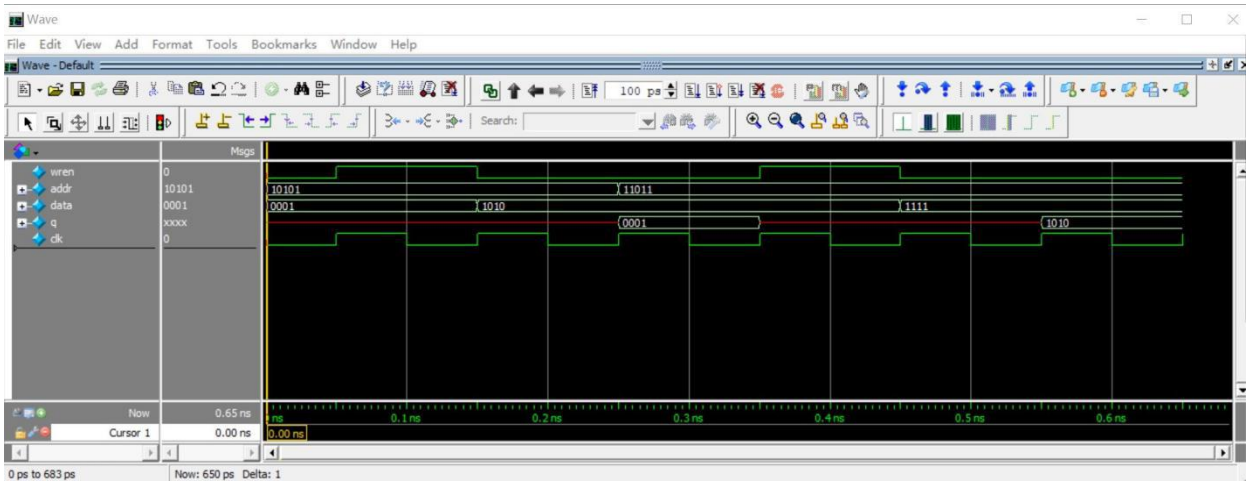

Fig 4. 32x4 RAM Simulation

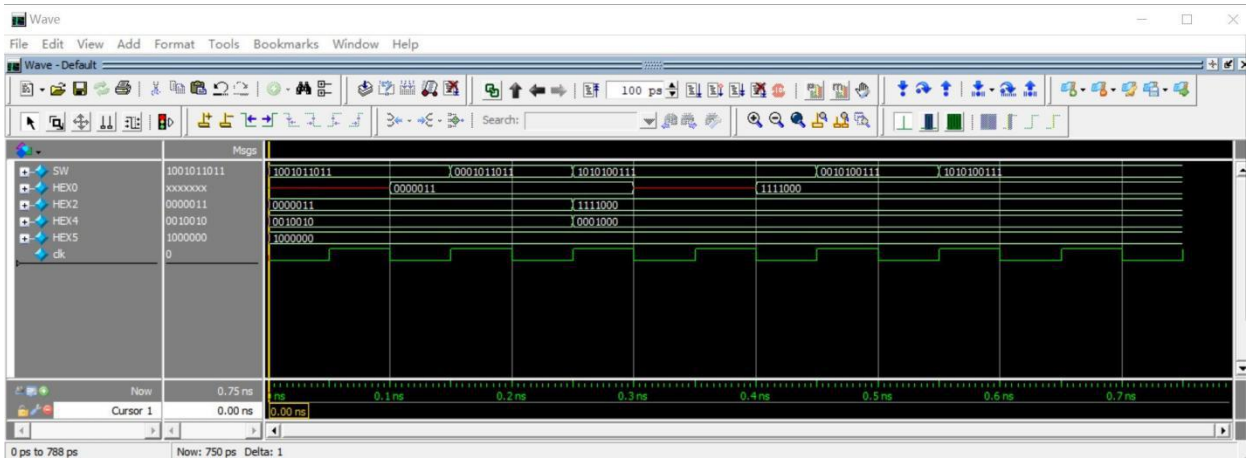

Fig. 5 Task 3 Simulation

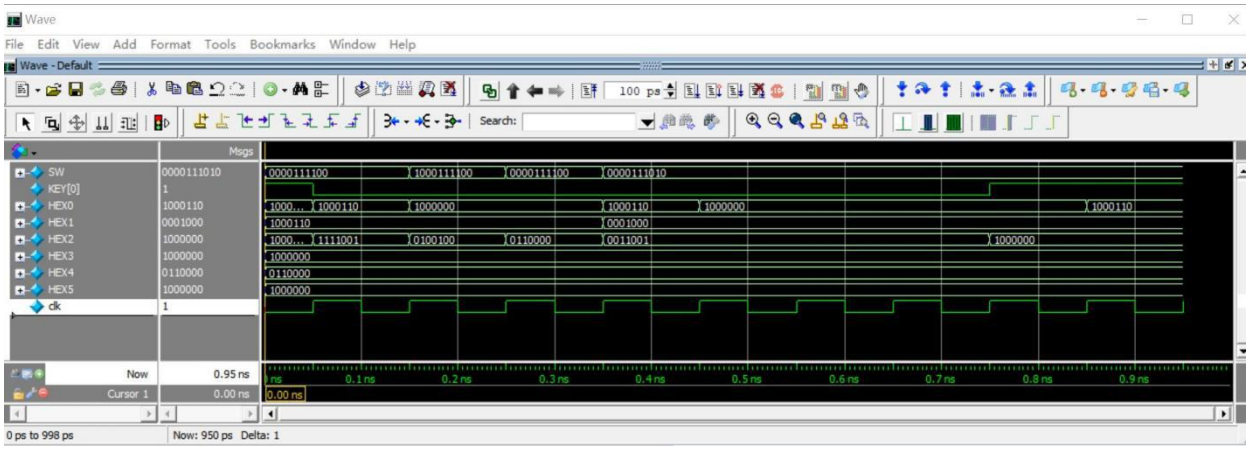

Fig 6. Task 4 Simulation