Arithmetic and Logic Unit (ALU)

Keonwoong Min
San Jose State University
keonwoong.min@sjsu.edu

Abstract: This report show how to implement arithmetic & logic unit (ALU) module HDL using the "ModelSim", a multi-language HDL simulation tool. This project includes following things.

1) Instructions to install a digital simulation tool, "ModelSim", and its setup.
2) Implementation of Arithmetic & Logic Unit (ALU) module using Verilog HDL
3) Implementation of testing of implemented ALU using HDL.
4) Simulation and observation of output waveforms of ALU using ALU test bench.

General Information

Table 1.1: List of Simulation Tool, ModelSim

Name of Tool	ModelSim
Founded by	Mentor Graphics
Objective	Test and Simulation

1. STEPS ON HOW TO INSTALL THE SIMULATION TOOL

There are three major simulators such as Enterprise Simulator, Mentor ModelSim, and Synopsys VCS. However only Mentor Graphics provides a free simulation tool, ModelSim, for students.

Following steps shows how to install the student edition simulator, "ModelSim".

1) Open this link https://www.mentor.com/company/higher_ed/modelsim-student-edition
2) Click on "Download Student Edition" button to download it
3) Open the file, 'modelsim-pe_student_edition', and complete the installation.
4) Fill out the form that will come out after the installation.
5) Get your license and a file named 'student license.dat' through email from ModelSim.
6) Save that file, 'student license.dat' and move into the file that contains 'win32pe_edu'.

2. STEPS OF SIMULATION PROJECT CREATION

This section contains the instructions of how to simulate the project for ALU and test it with test becnch.

Following steps shows how to simulate the ALU with Verilog Code in ModelSim

1) Download the prj_01.zip file, containing 'alu.v', 'prj_definition.v', and 'prj_01_tb.v', on Canvas and extract it all into a folder that you want.

Table 2.1: List of Files of 'prj_01_zip'

Name of File	**Used for**
'alu.v'	ALU module code
'prj_01_tb.v'	Test bench
'prj_definition'	Definitions, creating ALU

2) Open the ModelSim simulator and Close the 'IMPORTANT information' window. Then click project in figure 2.1

- To create project, go to "File" and "New" and click the "Project" as figure 2.1
- "Create Project" window will show up after click "Project" in figure 2.2.
- Give a project name such as "CS147_Project1" in figure 2.2 and push the OK button.

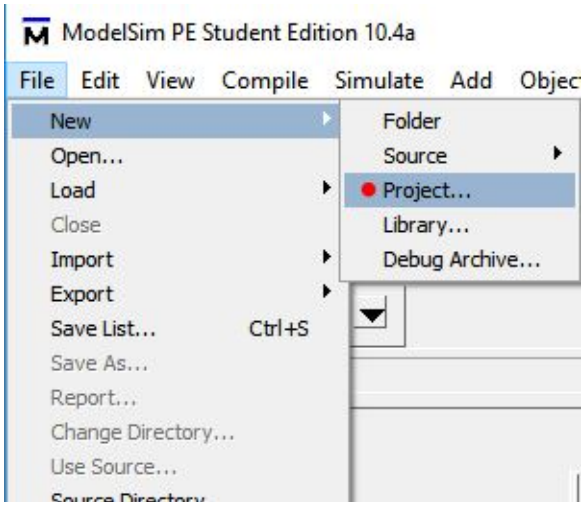

Figure 2.1 Click Project

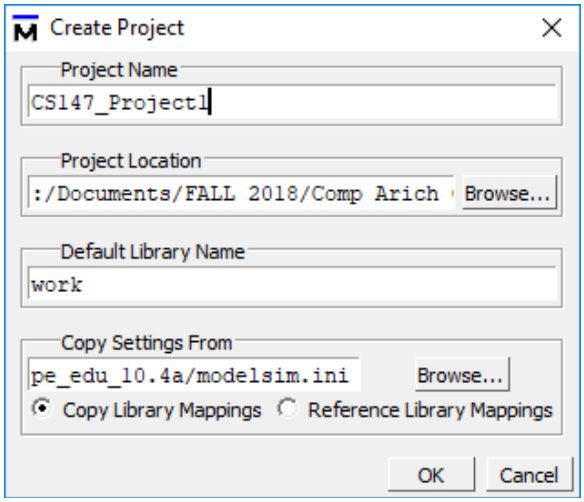

Figure 2.2 Create Project

3) After the second step, click “ADD Existing File” as Figure 2.3.
- Then click “Browse” in Figure 2.4
- After clicking “Browse”, select every Verilog files in figure 2.5 and push “Open” button.

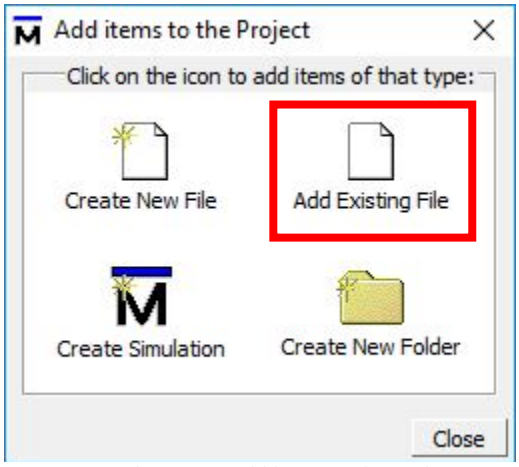

Figure 2.3 Add items to the Project

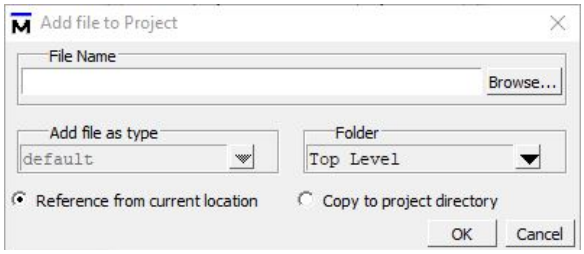

Figure 2.4 add file to Project

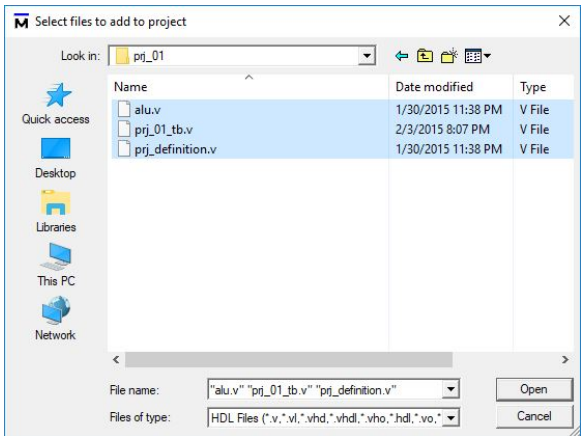

Figure 2.5 Select files to add to project

4) Select all files on “Project” tab and click “Compile all” and green check point mark will appear as in Figure 2.6

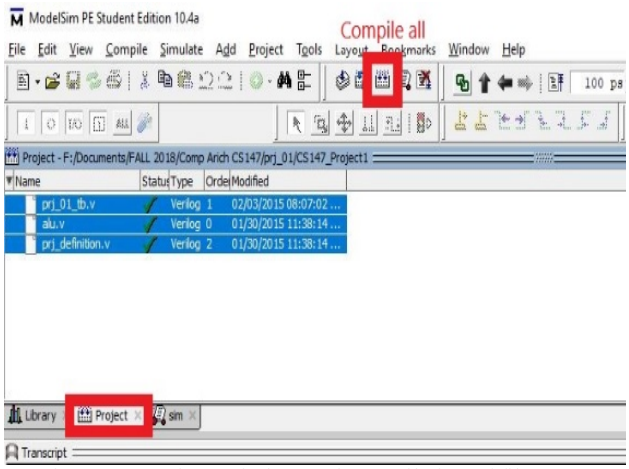

Figure 2.6 Project Window and Compilation of files

5) Go to Library tab and open work folder and double click the 'proj_01tb.v' as in figure

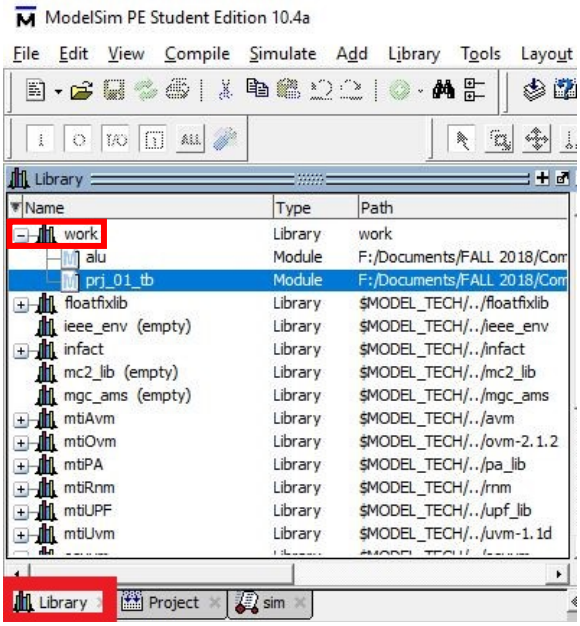

Figure 2.7 Test Bench code run

6) "Sim - Default" window will appear as in Figure 2.8 and double click the 'prj_01_tb'.

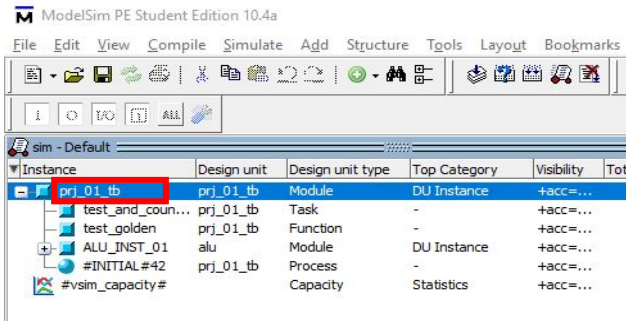

Figure 2.8 Sim Window

7) "Object" window will appear as in Figure 2.9

- Select all files on "Objects" window then hit the right click and click "Add Wave" as in Figure 2.9

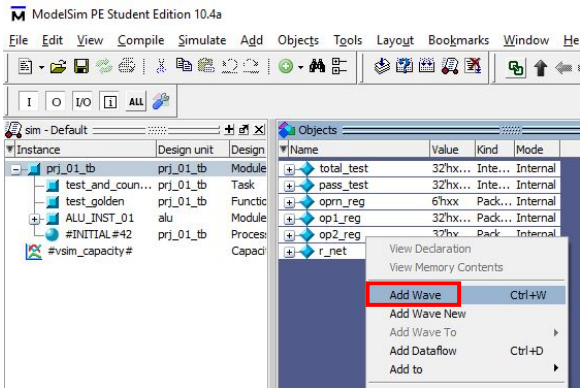

Figure 2.9 Object Window

8) Run and Change the maps values on "Wave – Default" window from hex to unsigned.

- Click "Run All' as in Figure 2.10
- Select every file on "Wave – Default" window and right click it, then go to "Radix" and click "Unsigned" as in Figure 2.10, since the value is in Hex in default

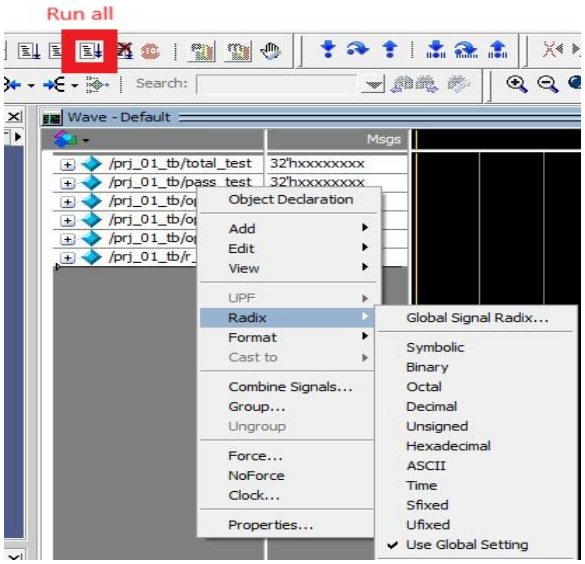

Figure 2.10 Waveform Window

3. REQUIREMENTS OF ALU

ALU It is a fundamental unit block of central processing unit (CPU), graphics processing units (GPUs) and some of them have multiple ALUs. It is a digital electronic circuit that generally supports many basic arithmetic and bitwise logic functions on a computer. The input data is called operands that is operated with operations in ALU.

Since an ALU supports basic arithmetic and logic functions, it takes any mathematic and logical programs with two operand operations such as in Figure 3.1. It also has a special shape to represent it as shown in Figure 3.1.

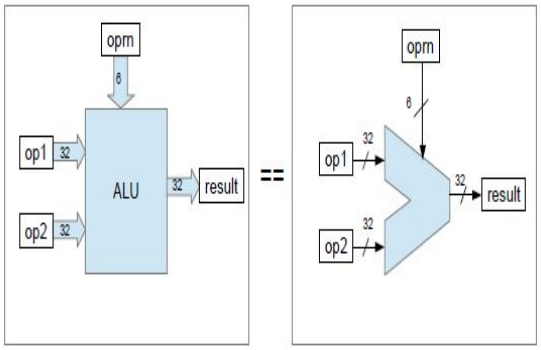

Figure 3.1 Diagram of ALU

4. DESIGN AND IMPLEMENTATION OF ALU

Electronic systems like ALU are modeled by a hardware description language such as Verilog.

Design of ALU

- ALU needs three input ports for operand 1, operand 2, and operation and one output ports for the result.

Instruction Set of ALU, Arithmetic operations and Bitwise logical operations:

- These are the instruction set of ALU of this project as shown in Figure 4.1.

Name	Mnemonic	Format	Operation
Addition	add	R	R[rd] = R[rs] + R[rt]
Subtraction	sub	R	R[rd] = R[rs] - R[rt]
Multiplication	mul	R	R[rd] = R[rs] * R[rt]
Logical AND	and	R	R[rd] = R[rs] & R[rt]
Logical OR	or	R	R[rd] = R[rs] | R[rt]
Logical NOR	nor	R	R[rd] = ~(R[rs] | R[rt])
Set less than	slt	R	R[rd] = (R[rs] < R[rt])?1:0
Shift left logical	sll	R	R[rd] = R[rs] << shamt
Shift right logical	srl	R	R[rd] = R[rs] >> shamt
Jump Register	jr	R	PC = R[rs]

Figure 4.1 Instruction Set

- Electrical conductors in an ALU that send digital signals between ALU and external circuitry. External circuits send signals to the ALU inputs and the ALU makes and gets signals to external circuits in return when an ALU is operating. For example, a CPU sends operands to the ALU's input from the register or sources to begin an addition operation, at the same time control unit simultaneously gives an opcode to ALU's opcode input. Plus, the CPU send the result of ALU's output to a register which is a destination that gets the sum.

5. TEST STRATEGY AND TEST IMPLEMENTATION

To Test the implementation code of ALU, test bench is used on ModelSim simulator. Following depicts the result of the test on ALU design.

OVERALL OUTPUT WAVEFORM

The overall output is shown in Figure 5.1 in unsigned decimal

Figure 5.1 Subtraction Waveform

A. Addition

```
#5      op1 = 15;
        op2 = 3;
        oprn = h01;
```

First arithmetic operation, Addition, is operated after 5ns from the program running since it is implemented on the test bench as shown in Figure 5.2.

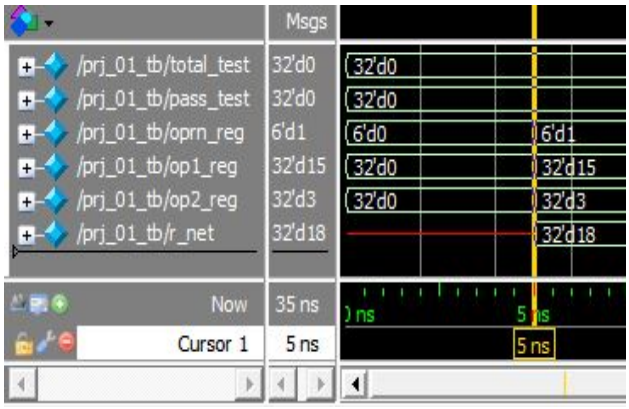

Figure 5.2 Addition Waveform

B. Subtraction

```
#5      op1 = 15;
        op2 = 5;
        oprn = h02;
```

Subtraction which is the value of 02h is operated at 15ns as implemented in test bench as shown in Figure 5.3

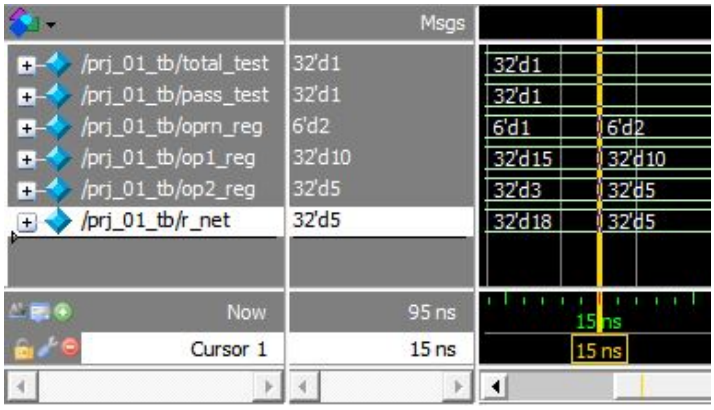

Figure 5.3 Subtraction Waveform

C. Multiplication

```
#5      op1 = 10;
        op2 = 2;
        oprn = h02;
```

Multiplication which is the value of 03h is operated at 25ns as implemented in test bench as shown in Figure 5.4

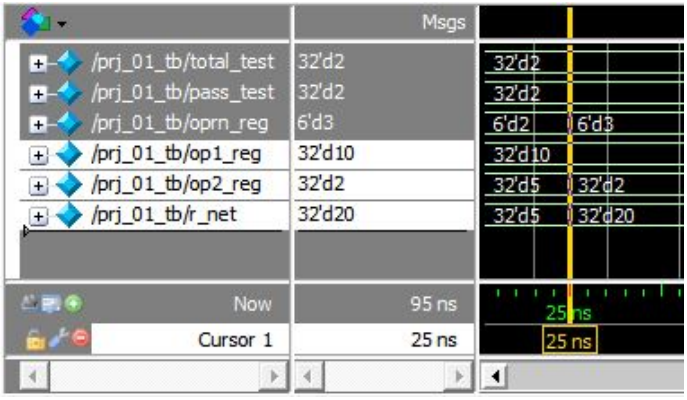

Figure 5.4 Multiplication Waveform

D. Bitwise AND

```
#5      op1 = 5 (0101);
        op2 = 1 (0001);
        oprn = h04;
```

Bitwise AND which is the value of 04h is operated at 35ns as implemented in the test bench as shown in Figure 5.5

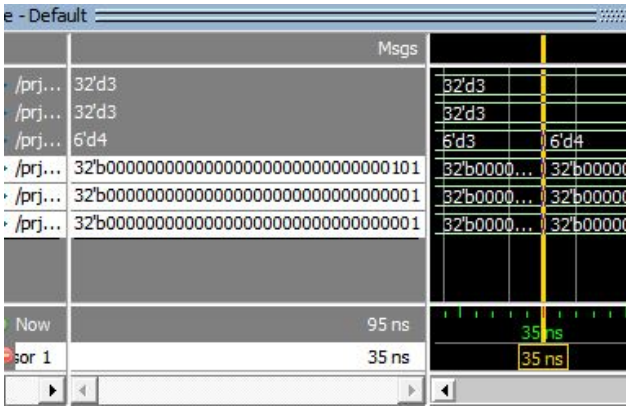

Figure 5.5 Bitwise AND Waveform

E. Bitwise OR

#5 op1 = 8 (1000);
op2 = 2 (0010);
oprn = h05;

Bitwise OR is operated at 45ns which is the value of 05h as shown if Figure 5.6.

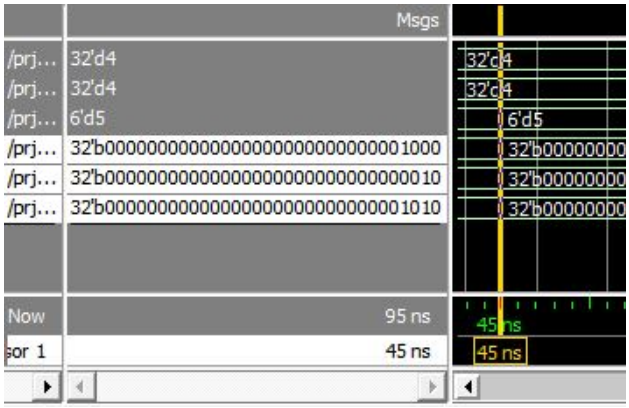

Figure 5.6 Bitwise OR Waveform

F. Bitwise NOR

#5 op1 = 10 (1010);
op2 = 3 (0011);
oprn = h06;

Bitwise NOR which is the value of 06h is operated at 55ns as implemented in test bench as shown in Figure 5.7

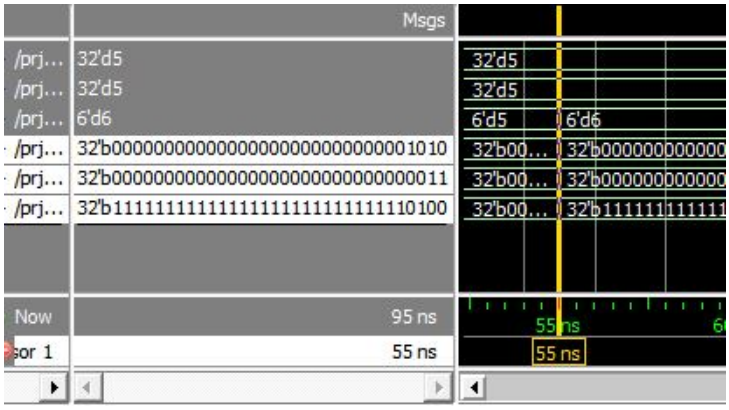

Figure 5.7 Bitwise NOR Waveform

G. Set Less Than

#5 op1 = 15;
op2 = 5;
oprn = h07;

Set less than which is the value of 07h is operated at 65ns as implemented in test bench as shown in Figure 5.8

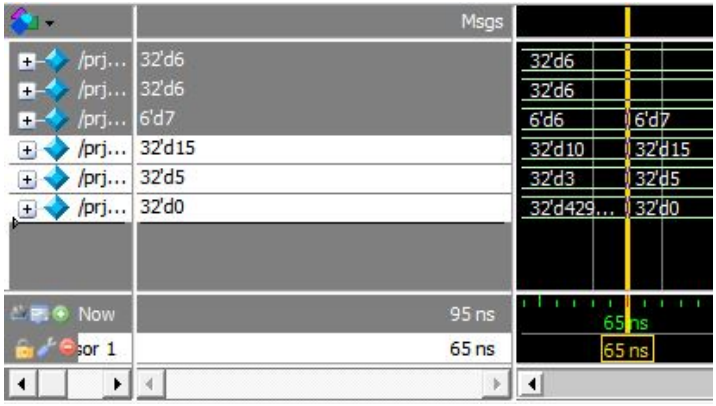

Figure 5.8 SetLessThan Waveform

H. Shift Left

#5 op1 = 2 (0010);
op2 = 4 (0100);
oprn = h08;

Shift Left which is the value of 08h is operated at 75ns as implemented in test bench as shown in Figure 5.9

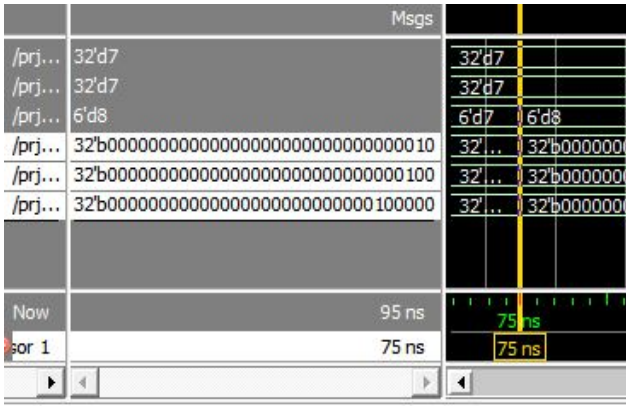

Figure 5.9 Shift Left Waveform

I. Shift Right

#5 op1 = 5 (0101);
op2 = 8 (1000);
oprn = h09;

Shift Right which is the value of 09h is operated at 85ns as implemented in test bench as shown in Figure 5.10

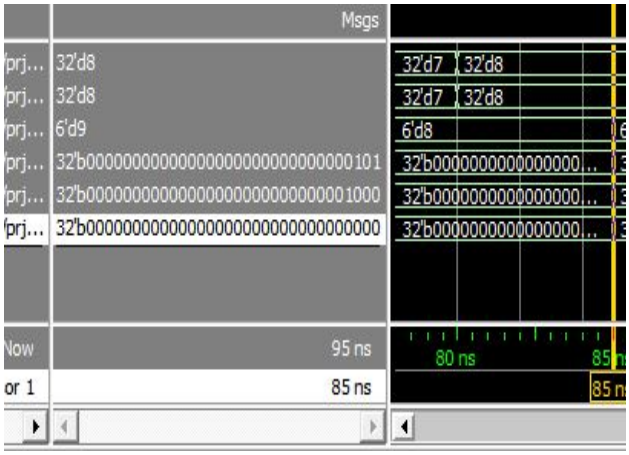

Figure 5.10 Shift Right Waveform

6. CONCLUSION

It had me a hard time to follow the instructions using a new tool, ModelSim, that I had never used and seen, so I had to spend tremendous of time and effort to deal with all these instructions. However, this project became a useful lesson eventually. I learned how to install the simulator, ModelSim and how to use the tool. Moreover, it helped me to understand and get more ideas of ALU in detail such as its functions, design, implementation and even how to implement the test bench based on Verilog.

7. REFERENCES

1. Computer Organization and Design (5th Edition) by David A. Patterson and John L. Hennessy, 2014

2. CS147 Lecture 01 by Kaushik Patra, 2014