Experiment 5
Multiplier circuit
ECE 385

Jeffrey Huang JHUANG61, Jordan Tan JTAN41

February 20, 2017

Date Performed: February 20, 2017

1 Introduction

In this lab we created a simple 8 bit processor that is able to multiply two numbers and store it in a register. We wrote this processor in system verilog and ran it on the altera board. To do this we used a given multiplication algorithm that is capable of multiplying twos complements.

2 Pre-Lab Question

Function	X	A	B	M
ClearA-LoadB	0	00000000	00000111	1
ADD	1	11000101	00000111	1
SHIFT	1	11100010	10000011	1
ADD	1	10100111	10000011	1
SHIFT	1	11010011	11000001	1
ADD	1	10011000	11000001	1
SHIFT	1	11001100	01100000	1
SHIFT	1	11100110	00110000	0
SHIFT	1	11110011	00011000	0
SHIFT	1	11111001	10001100	0
SHIFT	1	11111100	11000110	0
SHIFT	1	11111110	01100011	1

Figure 1. 7 * -59.

3 Written description and diagrams of multiplier circuit

3.1 Summary of operation

The processor is controlled with two buttons, the ClearA_LoadB switch, and the run switch. The ClearA_LoadB switch will take the value set on the switches and put it into register B. If you wanted to perform a multiplication operation, you first put a value in register B, then you load a value on the switches and press the run button. The result of the multiplication will show up on the hex display.

The circuit performs the operation using the algorithm detailed below. The final value is stored in a register and shown on the hex displays when the operation is done

```
MULTIPLICATION(multiplicand, multiplier):
    result=0
    for i in range(0..7):
        if multiplicand and 0x1 is 0:
            result = result + multiplier
            result = result ≪ 1
        multiplicand = multiplicand ≫ 1
    if multiplicand is 0x1:
        result = result - multiplier
    return result
```

3.2 Written Description of .sv Modules

Module:	addernsubtractor9bit.sv
Inputs:	[7:0] A, [7:0] S, Sub
Outputs:	[8:0]result
Description:	Adder and subtraction unit
Purpose:	Adds the input values or subtracts them if it's the 8th bit. Outputs the result

Module:	control.sv
Inputs:	Clk, Reset, C_L, Run, M_bit
Outputs:	cl_out, shift, add, sub
Description:	Control unit for the processor
Purpose:	Controls the other units in the processor. Necessary to ensure the right outcome

Module:	dflipflop.sv
Inputs:	D, Clk, Reset, Load
Outputs:	Q
Description:	D-FlipFlop
Purpose:	Used to holds state information

Module:	fulladder.sv
Inputs:	x, y, z
Outputs:	s, c
Description:	Adds two numbers
Purpose:	To add numbers

Module:	HexDriver.sv
Inputs:	[3:0] In0
Outputs:	[6:0] Out0
Description:	Takes an input and outputs the hex value to be used on the hex displays
Purpose:	To display data correctly on the hex displays

Module:	multiplier8bit.sv
Inputs:	[7:0] S, Clk, Reset, Run, ClearA_LoadB
Outputs:	[6:0] AhexU, [6:0] AhexL, [6:0] BhexU, [6:0] BhexL, [7:0] Aval, [7:0]Bval, X
Description:	Top level for multiplier
Purpose:	The top level of the multiplier, instantiates all of the modules

Module:	shiftreg8bit.sv
Inputs:	Clk, Reset, Load, ShiftInBit, ShiftEn, [7:0] DataIn
Outputs:	[7:0] DataOut, LSB
Description:	8 bit Shift register
Purpose:	Used to hold the multiplicand and shifts it according to the algorithm.

3.3 Finite State Machine

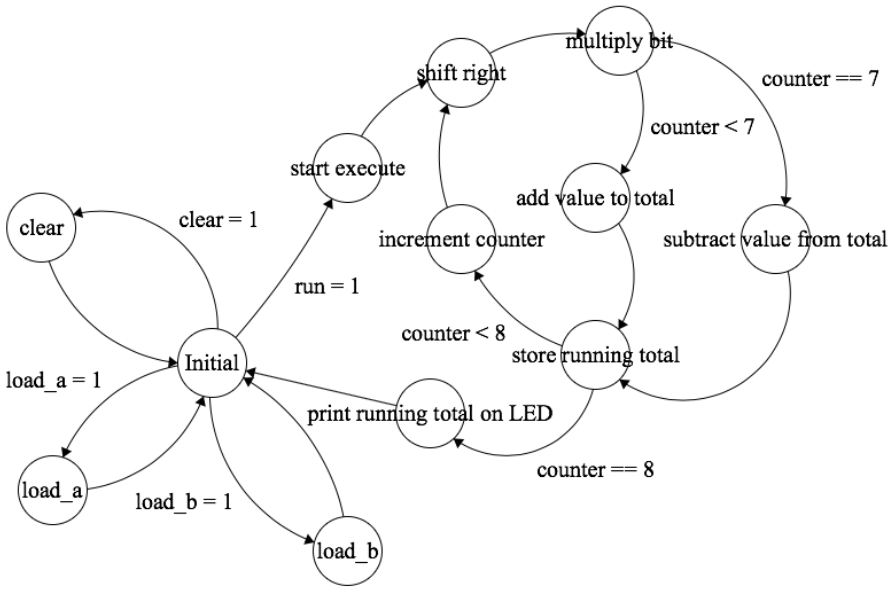

Figure 2. State machine of circuit.

4 Bugs

We ran into a couple bugs debugging this circuit. We only accounted and tested for the positive case. Because of this, we didn't have the correct value when we tested in lab, this caused us to only get a 1/5 on this lab. In future labs we will test more thoroughly before getting to lab.

5 Conclusion

We didn't completely finish this lab due to the bugs mentioned above. Other than that, the instructions to this lab were straightforward and easy to follow.