Cory Snooks

Aaron Pedersen

EE 465 Lab 8

10/28/2015

System Area Optimization

# Introduction

This lab demonstrated some of the tradeoffs between the size and speed of a circuit. We started with the circuit shown in Figure 1 and removed the multiplier on the top left. In order to keep the same functionality, D flip flops were added as shown in Figure 3. Removing the multiplier saves physical space on the board layout, but slows the circuit down since it now takes two clock cycles to process the four inputs through the single multiplier instead of one clock cycle with the two multipliers. The area is analyzed in comparison to the pipelined circuit from lab 7 in the results.

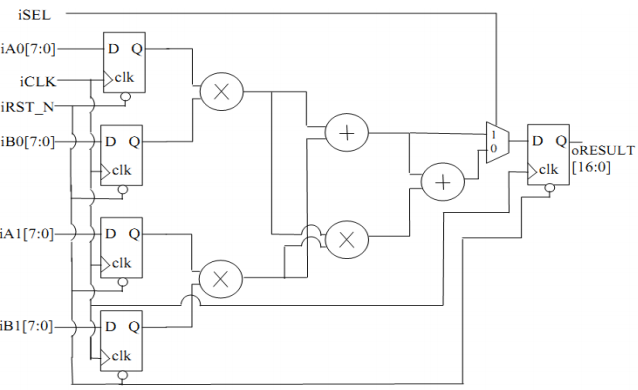


Figure : The original circuit is shown above.

# Lab Tasks

1. In Fig. 2, one of the registers can actually be eliminated. Please figure out which register is not necessary and modify the circuit diagram to eliminate it. You may just draw the modified circuit by hand but draw it clearly.

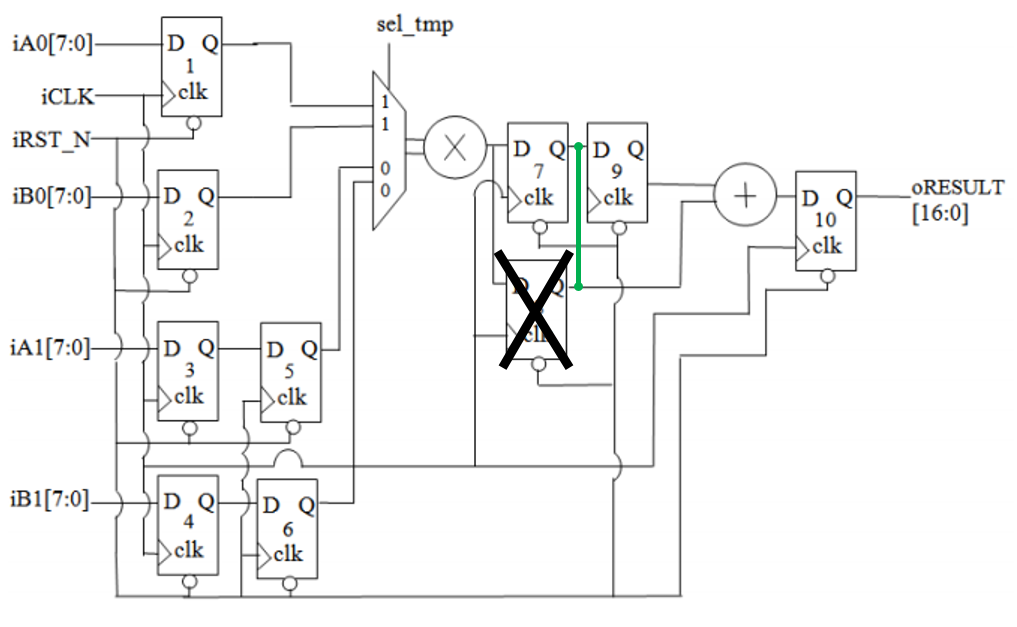


Figure : A sample circuit is shown above. There is an X over DFF 8 because that flip flop is not required for the circuit to operate correctly as long as the output from DFF 7 goes directly to the adder and to the input of DFF 9 as shown with the green line.

1. Please draw the modified circuit diagram of our design in Fig. 3 to save the multiplier at the lower left corner. You may just draw it by hand but draw them clearly.

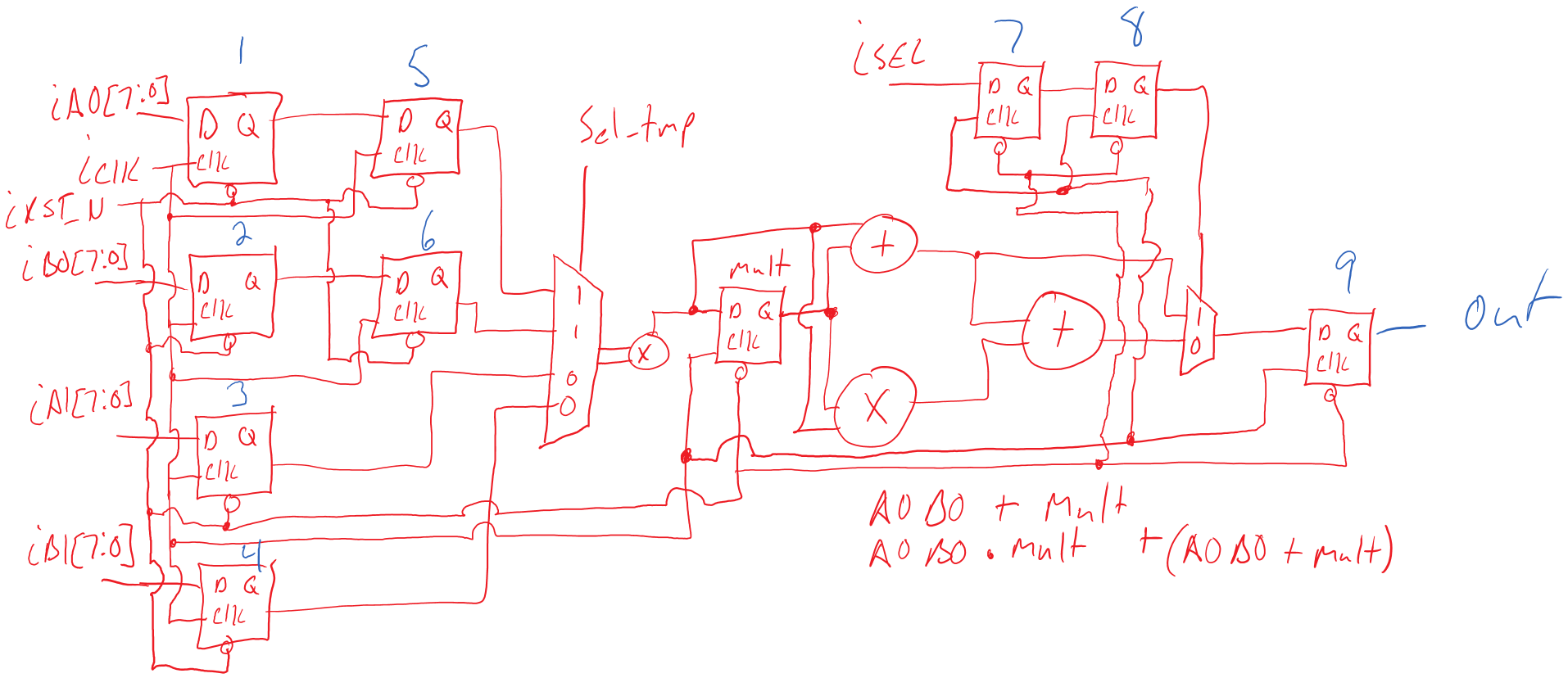


Figure : The schematic above shows a circuit with the same functionality as the circuit shown in Figure 2. This circuit will be smaller, but also a little slower due to the removal of the DFF to pipeline the logic flow.

1. Modify your Verilog code of Lab 4 according to the circuit diagram in Task 2 above. Verify its function in ModelSim for at least 5 sets of inputs. If you find that your circuit has bugs, please correct both your circuit diagram and your Verilog code. You should notice that the circuit will not function correctly if a new set of inputs is provided in every clock cycle. How many clock cycles does the circuit need to wait before it can accept another set of inputs? Please do simulation in ModelSim such that sets of inputs are supplied as frequent as possible so as to maximize the throughput of the circuit.

New inputs can be put into the circuit every two clock cycles due to the addition of the flip flops 5 and 6 shown in Figure 3. The Verilog code is shown below and was updated from the code used in lab 4 to include flip flops 5, 6, 8, and Mult as shown in Figure 3. The Verilog test bench results are shown below in Figure 4.

# Verilog Code

module ALT\_MULTADD\_NEW(iCLK, iRST\_N, iSEL, iA0, iA1, iB0, iB1, oR);

input iCLK, iRST\_N, iSEL, iA0, iA1, iB0, iB1;

output oR;

wire iCLK, iRST\_N, iSEL;

wire [7:0] iA0, iA1, iB0, iB1;

reg SEL1, SEL2, sel\_tmp;

reg [7:0] A0, A1, B0, B1, A01, B01, AxB;

reg [16:0] oR;

reg [1:0] count;

always @ (posedge iCLK)begin

A0 <= iA0;

B0 <= iB0;

A1 <= iA1;

B1 <= iB1;

A01 <= A0;

B01 <= B0;

SEL1 <= iSEL;

SEL2 <= SEL1;

if(iRST\_N)begin

if(count)begin

sel\_tmp <= 0;

end

else begin

sel\_tmp <= 1;

count <= count + 1;

end

if(~sel\_tmp)begin

AxB <= A01 \* B01;

end

else begin

AxB <= A1 \* B1;

end

if(SEL2) begin

oR <= AxB + A1 \* B1;

end

else begin

oR <= AxB + A1 \* B1 + AxB \* A1 \* B1;

end

end

else begin

sel\_tmp <= 0;

oR <= 0;

count <= 0;

end

end

always @(iA0 or iA1 or iB0 or iB1)begin

count <= 0;

end

endmodule

# Verilog Testbench

`timescale 10ns/1ns

module ALT\_MULTADD\_TB ();

reg iCLK\_t, iRST\_N\_t, iSEL\_t;

reg [7:0] iA0\_t, iA1\_t, iB0\_t, iB1\_t;

wire [16:0] oRESULT\_t;

ALT\_MULTADD\_re X(iCLK\_t, iRST\_N\_t, iSEL\_t, iA0\_t, iA1\_t, iB0\_t, iB1\_t, oRESULT\_t);

initial $display ("Test control");

initial $display ("%10s %10s %10s %10s %10s %10s %10s %10s",

"iCLK\_t", "iRST\_N\_t", "iSEL\_t", "iA0\_t", "iA1\_t",

"iB0\_t", "iB1\_t", "oRESULT\_t");

initial $monitor ("%10b %10b %10b %10d %10d %10d %10d %10d",

iCLK\_t, iRST\_N\_t, iSEL\_t, iA0\_t, iA1\_t, iB0\_t, iB1\_t, oRESULT\_t);

initial begin

iCLK\_t = 0;

iRST\_N\_t = 0;

iSEL\_t = 0;

iA0\_t = 0;

iA1\_t = 0;

iB0\_t = 0;

iB1\_t = 0;

#1

iRST\_N\_t = 1;

#1

iA0\_t = 3;

iA1\_t = 2;

iB0\_t = 3;

iB1\_t = 2;

#4

iSEL\_t = 1;

iA0\_t = 1;

iA1\_t = 1;

iB0\_t = 1;

iB1\_t = 1;

#4

iSEL\_t = 0;

iA0\_t = 2;

iA1\_t = 2;

iB0\_t = 2;

iB1\_t = 2;

#10

iSEL\_t = 1;

iA0\_t = 2;

iA1\_t = 2;

iB0\_t = 2;

iB1\_t = 2;

#10

iSEL\_t = 0;

iA0\_t = 1;

iA1\_t = 1;

iB0\_t = 1;

iB1\_t = 1;

#10

iSEL\_t = 0;

iA0\_t = 26;

iA1\_t = 1;

iB0\_t = 15;

iB1\_t = 19;

#4

$stop;

end

always #1 iCLK\_t = ~iCLK\_t;

endmodule

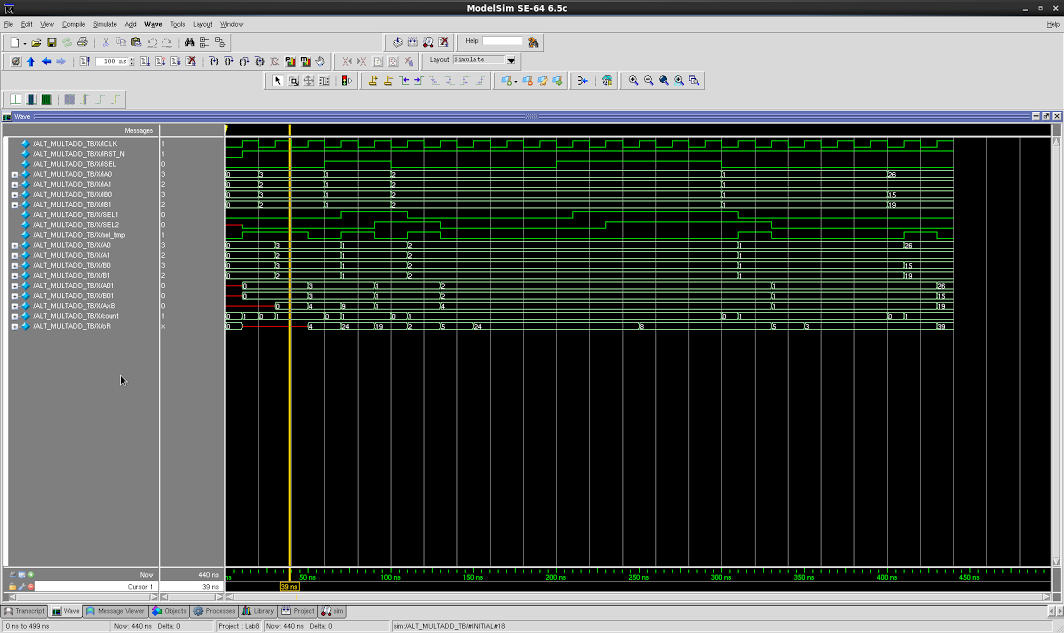


Figure 4: There Verilog output for the circuit shown in Figure 3 is shown above.

1. Go through the flow of synthesis, and compare the area result based on the same constraints with your pipelined design in Lab 7. Please compare both the versions without retiming and with retiming. Please note that the area saving also depends on how many new registers are introduced. Please also note that the throughput of our design may decrease. Basically, we trade it for area.

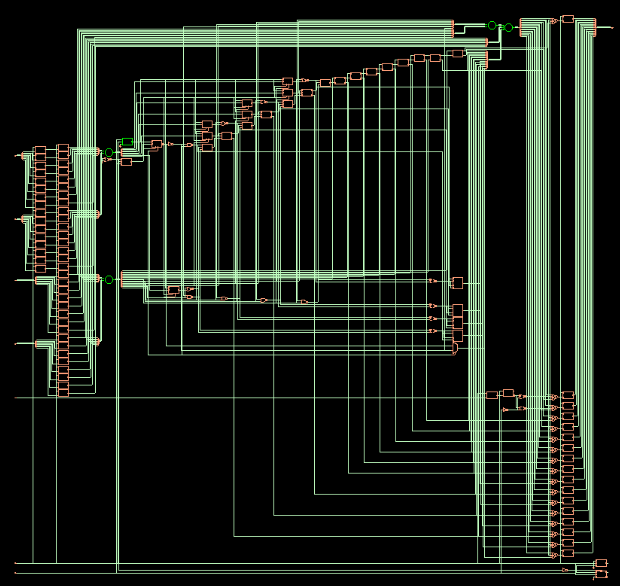


Figure : Final circuit synthesis is shown above with RETIME disabled.

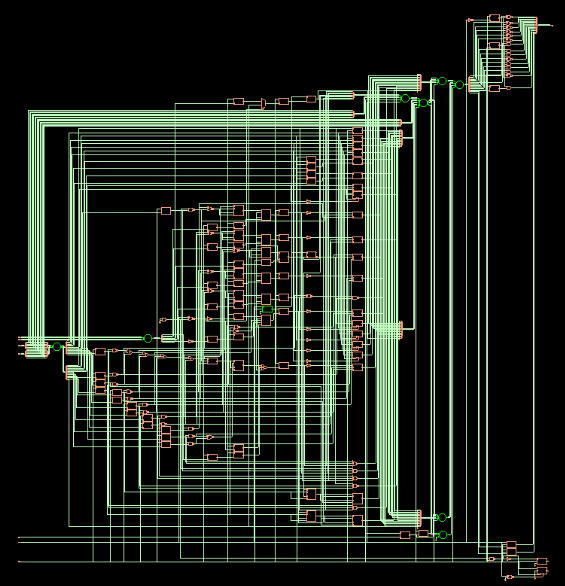


Figure : Final circuit synthesis with RETIME enabled.

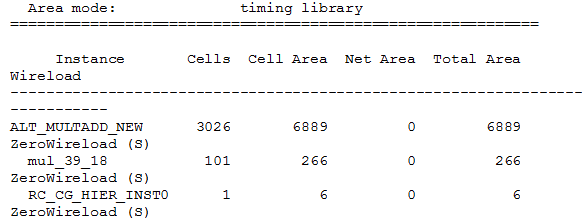


Figure 8: The lab 8 circuit area with RETIME disabled is shown above.

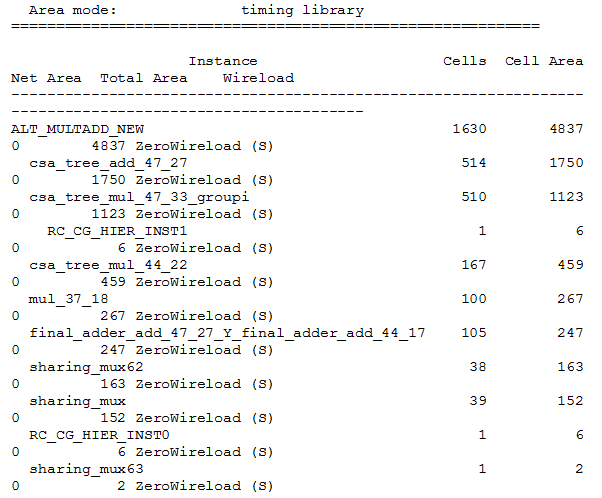


Figure : The lab 8 circuit area with RETIME enabled is shown above.

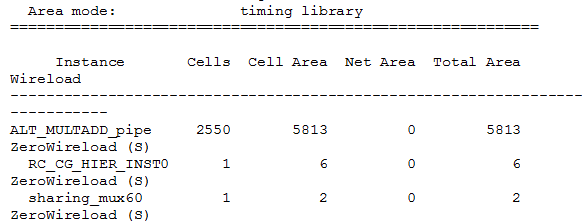


Figure : The area for the pipelined circuit from lab 7 is shown above with RETIME disabled.

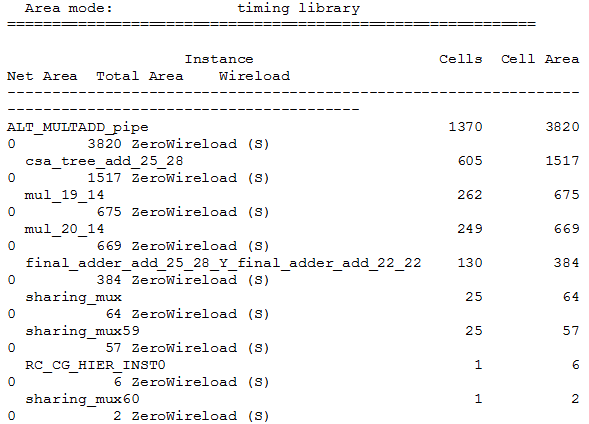


Figure : The area for the pipelined circuit from lab 7 is shown above with RETIME enabled.

The areas for the new circuits are larger than the areas for the lab 7 circuits when strictly comparing the RETIME enabled versions and the RETIME disabled versions. That means that we have not gained anything by removing the multiplier. In fact the area is larger and the circuit is slower. The area gained by removing a single multiplier was not enough to offset the area of the four additional D flip flops.

# Conclusion

This lab was supposed to demonstrate that the multiplier operations for the four inputs could be completed by a single multiplier while saving physical area. According the area reports generated however, we see that is not that case. The functionality of the two circuits shown in Figure 1 and Figure 3 operate the same, but the pipelined version of Figure 1 is still smaller. We have learned in previous labs that the synthesized circuit depends heavily on the coding style of the Verilog. It is possible that the area for the new circuit could be reduced by changing the way the Verilog code is written.