Quinn Roemer

Engineering – 303

Lab 4

2/22/2017

Introduction/Description

The purpose of this lab was to design several circuits that would be incorporated in a hierarchical design at the end of the lab. I was supposed to learn how to create a circuit that would translate a binary input into a hex display output, a circuit that would add a couple bits, and a circuit capable of adding 8 bits. In the end I was supposed to learn how to incorporate these three previous circuits into one design.

Design

In the first part of the lab I was instructed to design a circuit that would translate a four bit binary input into a seven bit output that could be hooked up to a hex display. This display would correctly display a 0 for a binary zero of 0000 a 1 for a binary one 0001 and so on. In this part of the lab I was told it was not necessary to simplify the equations using Kmaps. However, I decided to create the simplified equations for extra practice. These equations that I created were not used in the final design.

Here is the truth table that I filled out in the process of creating the circuit.

	Inputs				Outputs						
Hex	B3	B2	B1	B0	S6	S5	S4	S3	S2	S1	S0
0	0	0	0	0	1	0	0	0	0	0	0
1	0	0	0	1	1	1	1	1	0	0	1
2	0	0	1	0	0	1	0	0	1	0	0
3	0	0	1	1	0	1	1	0	0	0	0
4	0	1	0	0	0	0	1	1	0	0	1
5	0	1	0	1	0	0	1	0	0	1	0
6	0	1	1	0	0	0	0	0	0	1	0
7	0	1	1	1	1	1	1	1	0	0	0
8	1	0	0	0	0	0	0	0	0	0	0
9	1	0	0	1	0	0	1	1	0	0	0
A	1	0	1	0	0	0	0	1	0	0	0
b	1	0	1	1	0	0	0	0	0	1	1
C	1	1	0	0	1	0	0	0	1	1	0
d	1	1	0	1	0	1	0	0	0	0	1
E	1	1	1	0	0	0	0	0	1	1	0
F	1	1	1	1	0	0	0	1	1	1	0

Here is the Minterm chart that I used to create the Kmaps and the equations used in the circuit.

Minterms						
mS6	mS5	mS4	mS3	mS2	mS1	mS0
B3'B2'B1'B0'						
B3'B2'B1'B0	B3'B2'B1'B0	B3'B2'B1'B0	B3'B2'B1'B0			B3'B2'B1'B0
	B3'B2'B1B0'			B3'B2'B1B0'		
	B3'B2'B1B0	B3'B2'B1B0				
		B3'B2B1'B0'	B3'B2B1'B0'			B3'B2B1'B0'
		B3'B2B1'B0			B3'B2B1'B0	
					B3'B2B1B0'	
B3'B2B1B0	B3'B2B1B0	B3'B2B1B0	B3'B2B1B0			
		B3B2'B1'B0	B3B2'B1'B0			
			B3B2'B1B0'			
					B3B2'B1B0	B3B2'B1B0
B3B2B1'B0'				B3B2B1'B0'	B3B2B1'B0'	
	B3B2B1'B0					B3B2B1'B0
				B3B2B1B0'	B3B2B1B0'	
			B3B2B1B0	B3B2B1B0	B3B2B1B0	

Here are the Kmaps that I designed to figure out the simplified algebraic expressions for the circuit.

S6 Kmap	B1B0	B1'B0	B1'B0'	B1B0'
B3B2			1	
B3'B2	1			
B3'B2'		1	1	
B3B2'				
S6=B3'B2'B1'+B3'B2B1B0+B3'B2B1'B0'				

S5 Kmap	B1B0	B1'B0	B1'B0'	B1B0'
B3B2		1		
B3'B2	1			
B3'B2'	1	1		1
B3B2'				
S5=B3'B1B0+B3'B2'B0+B3B2B1'B0+B3'B2'B1B0'				

S4 Kmap	B1B0	B1'B0	B1'B0'	B1B0'
B3B2				
B3'B2		1		1
B3'B2'	1	1		
B3B2'	1	1		
S4=B2'B0+B3'B1'B0+B3'B2B1B0'				

S3 Kmap	B1B0	B1'B0	B1'B0'	B1B0'
B3B2	1			
B3'B2	1		1	
B3'B2'		1		
B3B2'		1		1
S3=B2B1B0+B2'B1'B0+B3'B2B1'B0+B3B2'B1B0'				

S2 Kmap	B1B0	B1'B0	B1'B0'	B1B0'
B3B2	1		1	1
B3'B2				
B3'B2'				1
B3B2'				
S2=B3B2B0'+B3B2B1+B3'B2'B1B0'				

S1 Kmap	B1B0	B1'B0	B1'B0'	B1B0'
B3B2	1			1
B3'B2		1		1
B3'B2'				
B3B2'	1		1	
S1=B3B1B0+B3'B2B1'B0+B3B2'B1'B0'+B2B1B0'				

S0 Kmap	B1B0	B1'B0	B1'B0'	B1B0'
B3B2				
B3'B2			1	
B3'B2'		1		
B3B2'	1	1		
S6=B3B2'B0+B2'B1'B0+B3'B2B1'B0'				

Here is the finished Verilog circuit design.

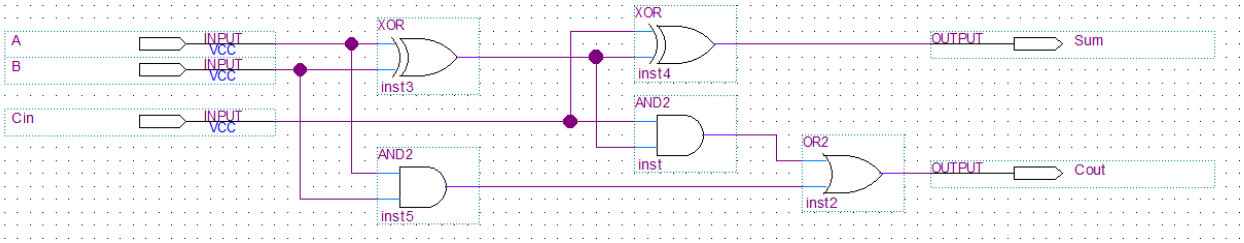
```
1   //A 4 bit binary to seven-segment decoder.
2   module BinHex (B, S);
3
4   input [3 : 0] B;        //Declaring a set of four input wires.
5   output [6 : 0] S;       // Declaring a set of seven ouput wires.
6
7   //Assigning all of the necessary values to the outputs.
8
9   assign S[0] = ~B[3] & ~B[2] & ~B[1] &  B[0]
10              | ~B[3] &  B[2] & ~B[1] & ~B[0]
11              |  B[3] & ~B[2] &  B[1] &  B[0]
12              |  B[3] &  B[2] & ~B[1] &  B[0];
13
14  assign S[1] = ~B[3] &  B[2] & ~B[1] &  B[0]
15              | ~B[3] &  B[2] &  B[1] & ~B[0]
16              |  B[3] & ~B[2] &  B[1] &  B[0]
17              |  B[3] &  B[2] & ~B[1] & ~B[0]
18              |  B[3] &  B[2] &  B[1] & ~B[0]
19              |  B[3] &  B[2] &  B[1] &  B[0];
20
21  assign S[2] = ~B[3] & ~B[2] &  B[1] & ~B[0]
22              |  B[3] &  B[2] & ~B[1] & ~B[0]
23              |  B[3] &  B[2] &  B[1] & ~B[0]
24              |  B[3] &  B[2] &  B[1] &  B[0];
25
26  assign S[3] = ~B[3] & ~B[2] & ~B[1] &  B[0]
27              | ~B[3] &  B[2] & ~B[1] & ~B[0]
28              | ~B[3] &  B[2] &  B[1] &  B[0]
29              |  B[3] & ~B[2] & ~B[1] &  B[0]
30              |  B[3] & ~B[2] &  B[1] & ~B[0]
31              |  B[3] &  B[2] &  B[1] &  B[0];
32
33  assign S[4] = ~B[3] & ~B[2] & ~B[1] &  B[0]
34              | ~B[3] & ~B[2] &  B[1] &  B[0]
35              | ~B[3] &  B[2] & ~B[1] & ~B[0]
36              | ~B[3] &  B[2] & ~B[1] &  B[0]
37              | ~B[3] &  B[2] &  B[1] &  B[0]
38              |  B[3] & ~B[2] & ~B[1] &  B[0];
39
40  assign S[5] = ~B[3] & ~B[2] & ~B[1] &  B[0]
41              | ~B[3] & ~B[2] &  B[1] & ~B[0]
42              | ~B[3] & ~B[2] &  B[1] &  B[0]
43              | ~B[3] &  B[2] &  B[1] &  B[0]
44              |  B[3] &  B[2] & ~B[1] &  B[0];
45
46  assign S[6] = ~B[3] & ~B[2] & ~B[1] & ~B[0]
47              | ~B[3] & ~B[2] & ~B[1] &  B[0]
48              | ~B[3] &  B[2] &  B[1] &  B[0]
49              |  B[3] &  B[2] & ~B[1] & ~B[0];
50
51  endmodule
```

In the next part of the lab I was supposed to design a circuit called a full adder. This circuit was supposed to have two main inputs with an additional input for the carry-in. In addition, it was supposed to have one main output with a carry-out. I was told to implement the design as a Block-Diagram.

Here is the truth table that I was given in the lap to design the circuit.

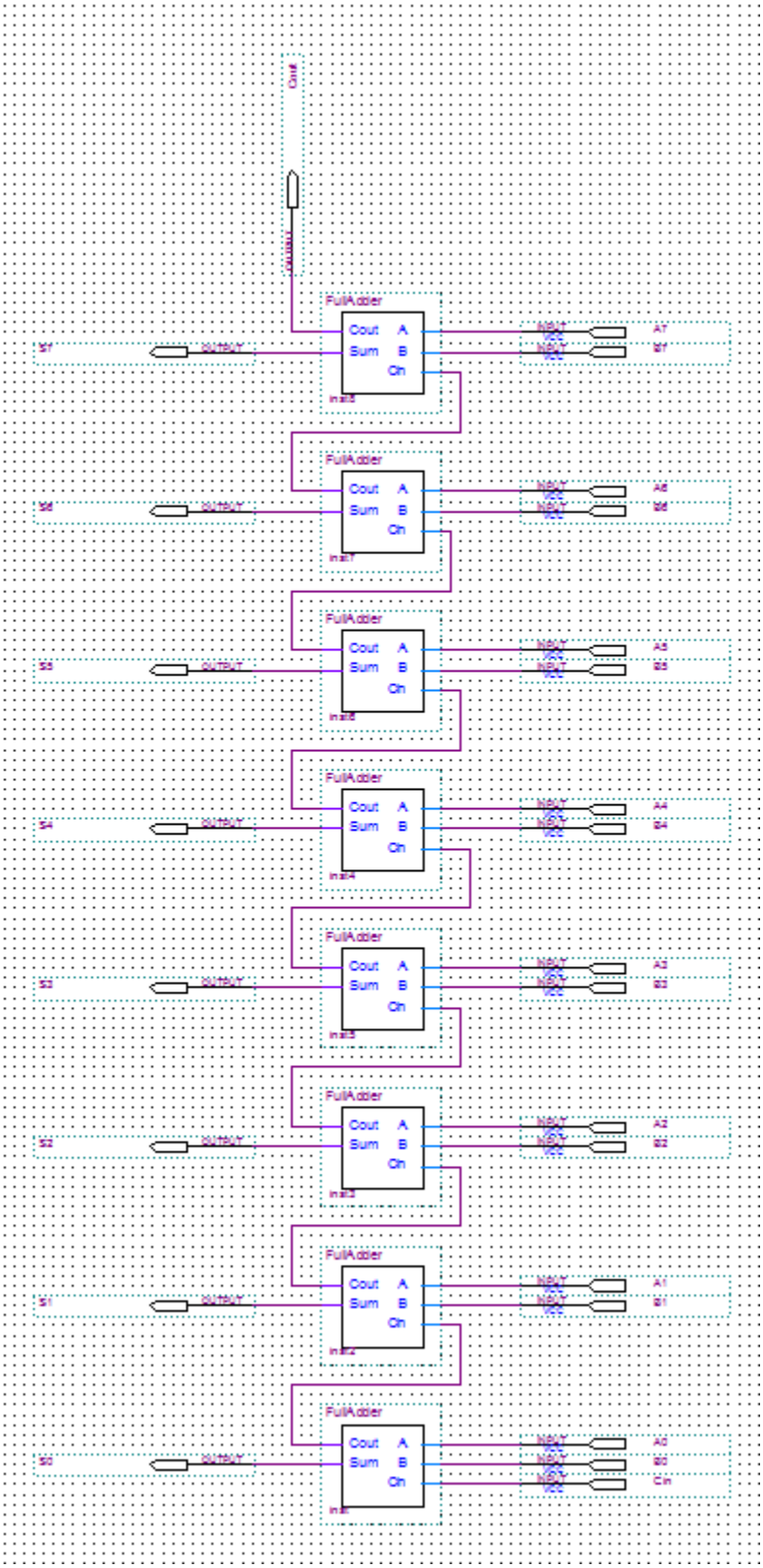
Inputs			Outputs	
A	B	Carry-In	Carry-Out	Sum
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1

Here is the Block-Diagram that I used for the circuit.

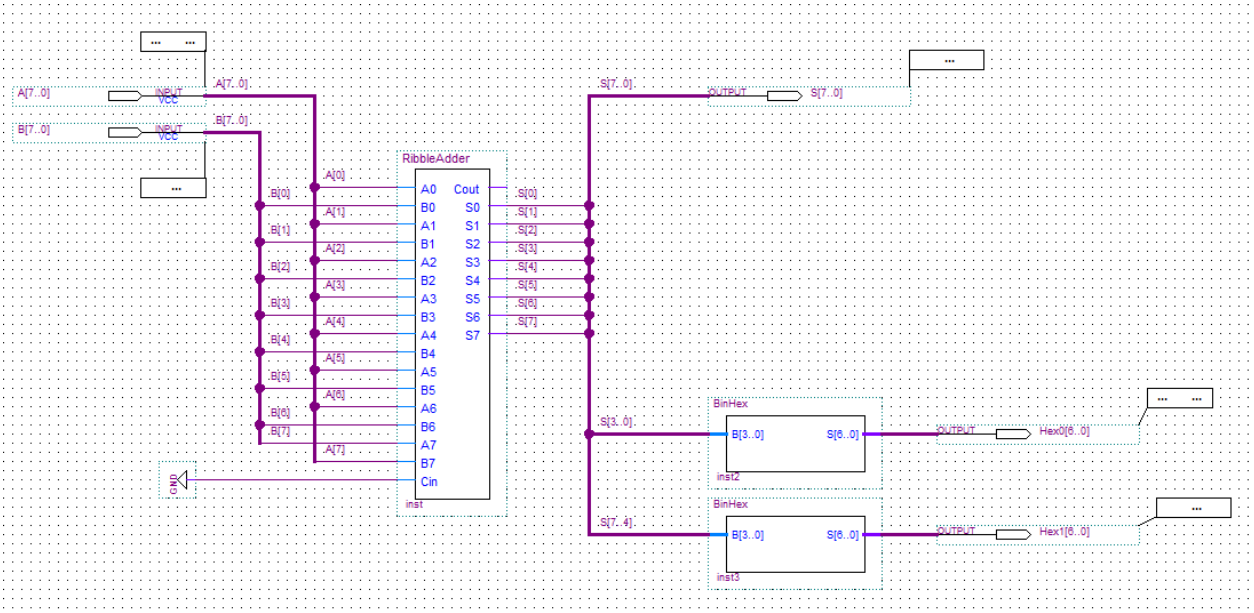


In the next part of the lab I was told to create a circuit called a ripple adder that would incorporate the previous circuit designed in the lab. I was not given a truth table for this circuit nor was I instructed to create one. I was only told to create the Block-Diagram design by following the picture given to me in the lab handbook.

Here is the Block-Diagram for the ripple adder circuit.

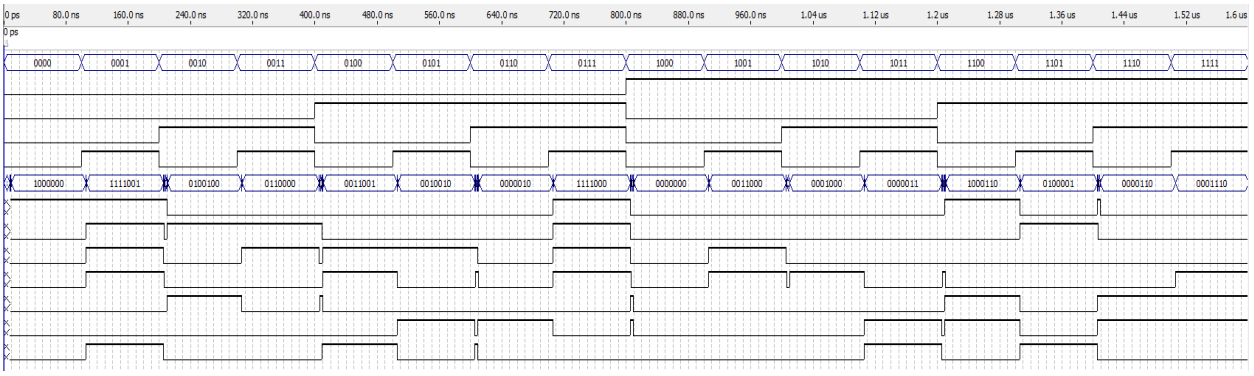


In the last part of the lab I was told to build a circuit that would incorporate all the previous designs into one circuit. This circuit was supposed to be capable of adding four bit numbers and then displaying the result in hexadecimal on a hex display. I was not instructed to build a truth table for this circuit.

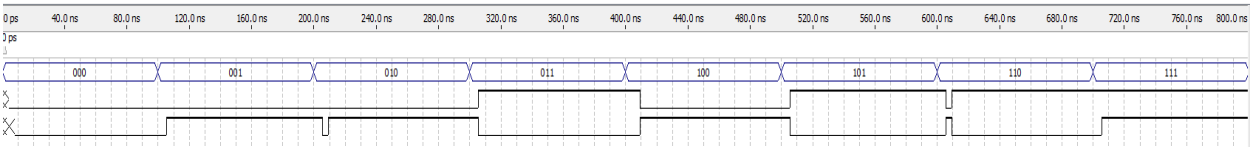


Testing

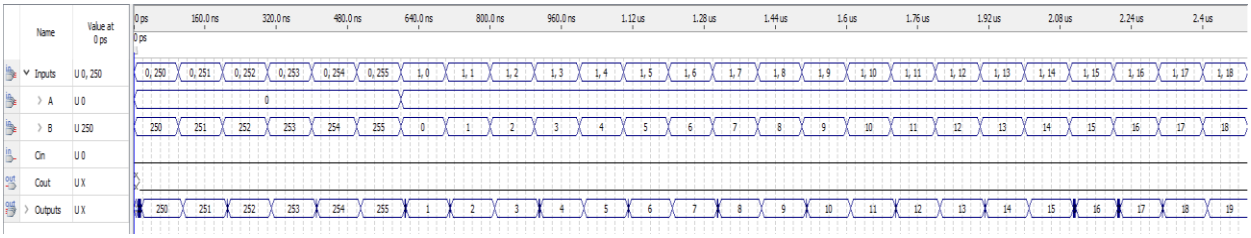
When testing the binary to hex circuit, I encountered no problems and it performed as expected for every single input combination.



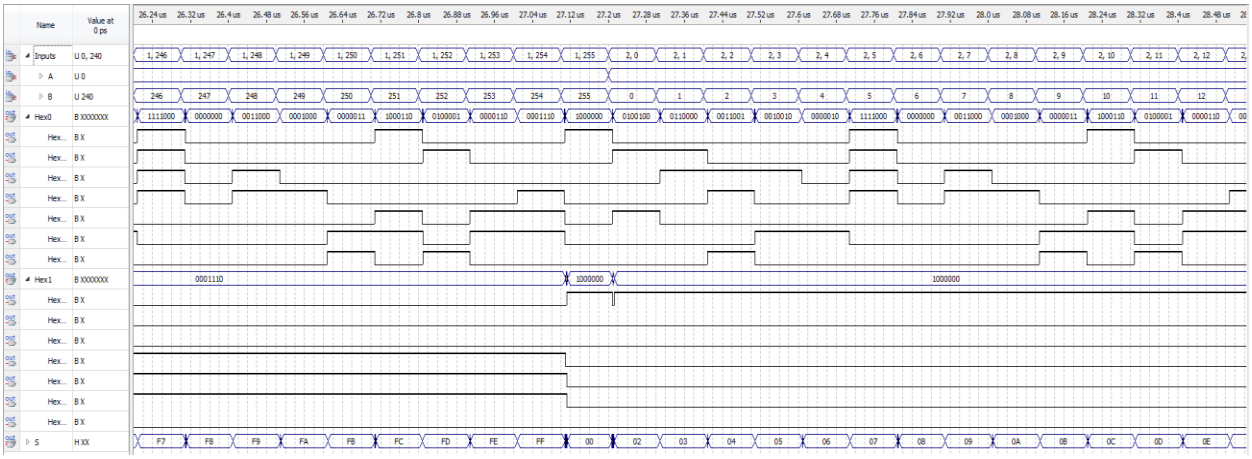
When testing the full adder circuit, I encountered no problems and it performed as expected for every single input combination.



When testing the ripple adder circuit, I encountered a small problem in creating the waveform. It took some trial and error to figure out how to correctly setup the waveform by grouping the inputs and outputs in order from greatest to least. Also, I was unfamiliar with using a radix different then binary. However, despite these issues I still managed to get the circuit to test correctly.



When testing the hierarchal circuit at the end of the lab, I amazingly encountered no problems. Also, it performed as expected for every single input combination.



Conclusion

In this lab I learned how to create more complex hierarchical designs in Quartus. I learned that a complex circuit can easily be simplified by putting a box around it. In other words to create a complex circuit you simply work on one element of it at a time. This lab enabled me to get more practice incorporating these elements in circuits. If I was to do this lab again I would use my simplified algebraic circuit expressions to program the first circuit. In doing this it would increase the efficiency of the circuit and simplify its design. Overall the exercises in this lab proved to be enjoyable and was great fun to implement.