Jennifer Wei
Computer Architecture HW b010
25 September 2015

Decoder

```
# Loading work.testDecoder
# Loading work.structuralDecoder
# En A0 A1| O0 O1 O2 O3 | Expected Output
# 0  0  0 |  0  0  0  0 | All false
# 0  1  0 |  0  0  0  0 | All false
# 0  0  1 |  0  0  0  0 | All false
# 0  1  1 |  0  0  0  0 | All false
# 1  0  0 |  1  0  0  0 | O0 only
# 1  1  0 |  0  1  0  0 | O1 only
# 1  0  1 |  0  0  1  0 | O2 only
# 1  1  1 |  0  0  0  1 | O3 only
```

Above is the table generated by testing my device against the test bench. This is correct as it checks all possible combinations. The outputs (*o0,o1,o2,o3*) align with the expected outputs.

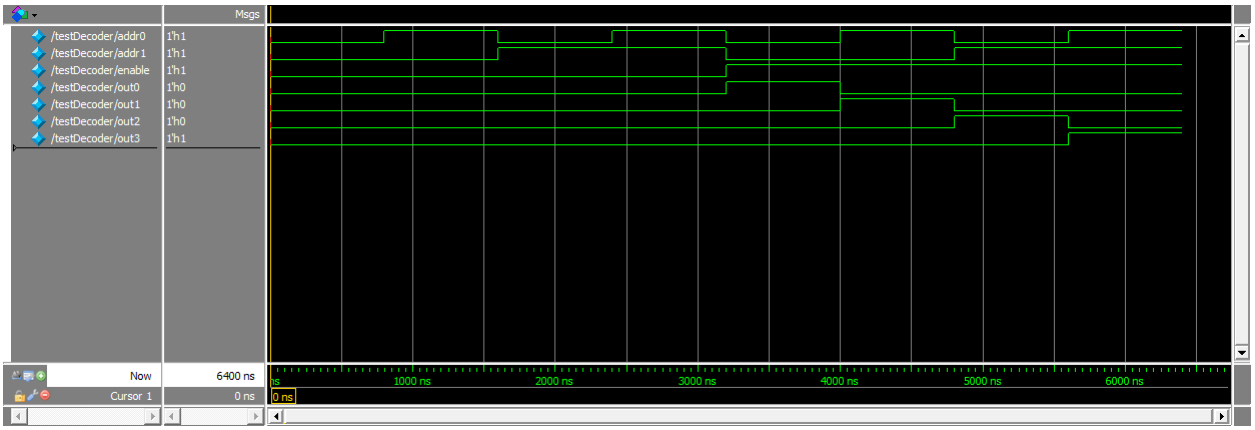


Figure 1a. Behavioral Decoder

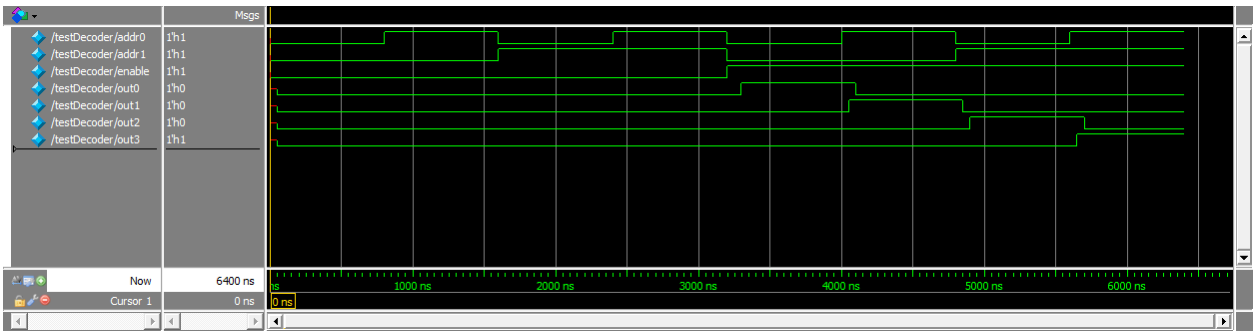


Figure 1b. Structural Decoder

Since Verilog statements run instantaneously, delays are added to see the effects of the gates. Above are the waves of the behavioral and structural devices. Overall, the waves produced look similar, but when looked at more closely, the delays in the structural decoder are visible around the 4000ms,

5000ms, and 5500ms marks. When *addr0* and *addr1* change while *enable* is off, all the outputs are off. Once *enable* is toggled on and after the 50ms delay, *out0* toggles on. When *addr0* toggles on (with *addr1* still off, *enable* still on, and a slight delay again), *out0* toggles off, and *out1* toggles on. This sequential off/on toggling continues through *out3*.

Full Adder

```
# Loading work.testFullAdder
# Loading work.structuralFullAdder
# A B C_in|   S  C_out | Expected Output
# 0  0  0 |   0    0    | Both False
# 0  0  1 |   1    0    | Sum Only
# 0  1  0 |   1    0    | Sum Only
# 0  1  1 |   0    1    | Carryout Only
# 1  0  0 |   1    0    | Sum Only
# 1  0  1 |   0    1    | Carryout Only
# 1  1  0 |   0    1    | Carryout Only
# 1  1  1 |   1    1    | Both True
```

The table above is correct as it checks all possible combinations. The outputs (sum (*s*) and carryout (*C_out*)) align with the expected outputs.

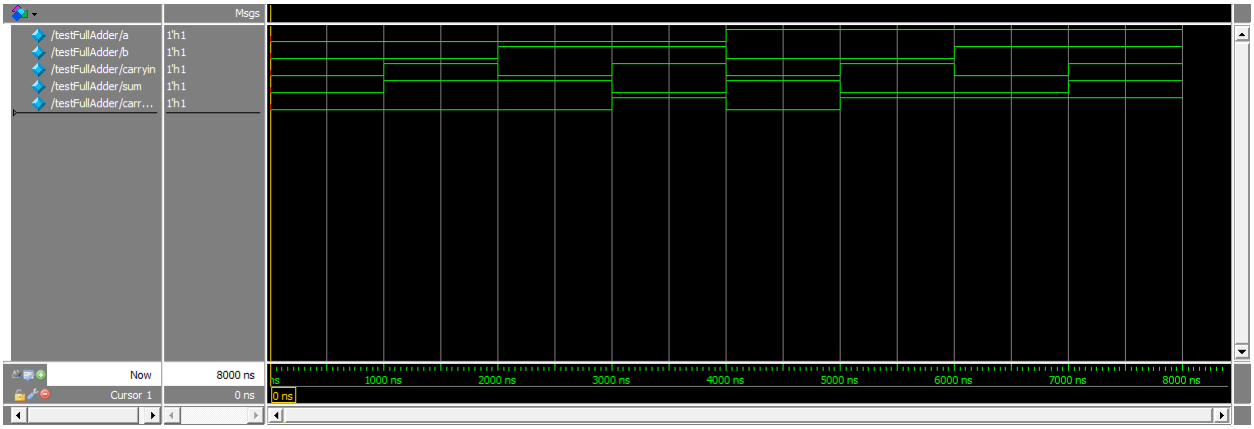


Figure 2a. Behavioral Adder

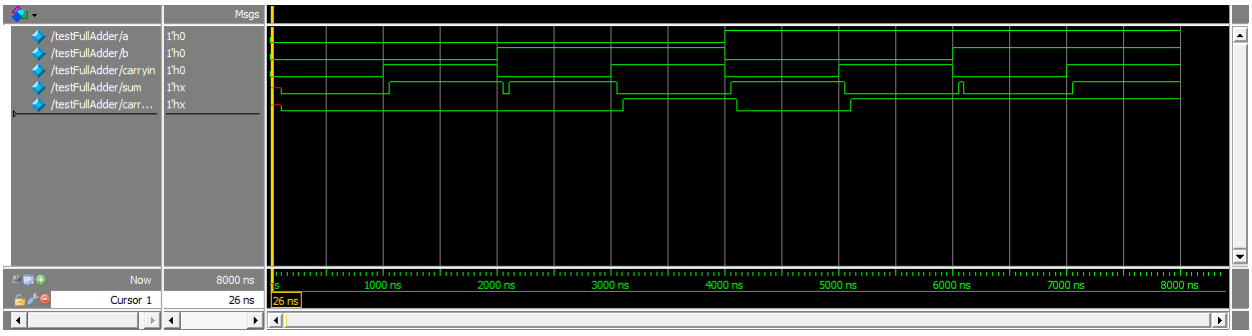


Figure 2b. Structural Adder

Like the decoder, the behavioral and structural adders both have similar wave forms, but again, the implemented delays affect the structural device, causing offsets and sudden drops (i.e. dip around

2000ms, 3000ms, 4000ms, slightly after 5000ms mark, a quick bump around 6000ms, and an offset again at 7000ms). *Sum* and *carryout* both have a delayed start as they are the results of several gate calculations. When *b* toggles on, *carryin* is on and *a* remains off, *sum* toggles off (due to the XOR) but quickly toggles back on when *carryin* is toggled off (with some delay, causing the small dips). As for *carryout*, it starts toggled off but toggles on when *carryin* is toggled on, *b* is on, and *a* is off.

Multiplexer

```
# Loading work.testMultiplexer
# Loading work.structuralMultiplexer
# S1 S0  | I0 I1 I2 I3 | Out | Expected Output
#  0  0  |  0  0  0  0 |  0  | 0
#  0  0  |  1  0  0  0 |  1  | 1
#  0  1  |  0  0  0  0 |  0  | 0
#  0  1  |  0  1  0  0 |  1  | 1
#  1  0  |  0  0  0  0 |  0  | 0
#  1  0  |  0  0  1  0 |  1  | 1
#  1  1  |  0  0  0  0 |  0  | 0
#  1  1  |  0  0  0  1 |  1  | 1
```

For the multiplexer, there are actually 64 different combinations. However, only eight are in this table. That is enough because the two addresses (S1,S0) point to specific inputs, so checking the other inputs is unnecessary. As shown above, the outputs align with the expected outputs.

```
# Loading work.testMultiplexer
# Loading work.structuralMultiplexer
# S1 S0  | I0 I1 I2 I3 | Out | Expected Output
#  0  0  |  0  x  x  x |  0  | 0
#  0  0  |  1  x  x  x |  1  | 1
#  0  1  |  x  0  x  x |  0  | 0
#  0  1  |  x  1  x  x |  1  | 1
#  1  0  |  x  x  0  x |  0  | 0
#  1  0  |  x  x  1  x |  1  | 1
#  1  1  |  x  x  x  0 |  0  | 0
#  1  1  |  x  x  0  1 |  1  | 1
```

Above is a modified version of the code using ‘X’s to prove that the other inputs do not affect the output (i.e. all the other inputs do not necessarily have to be set to 0 to get the expected output).

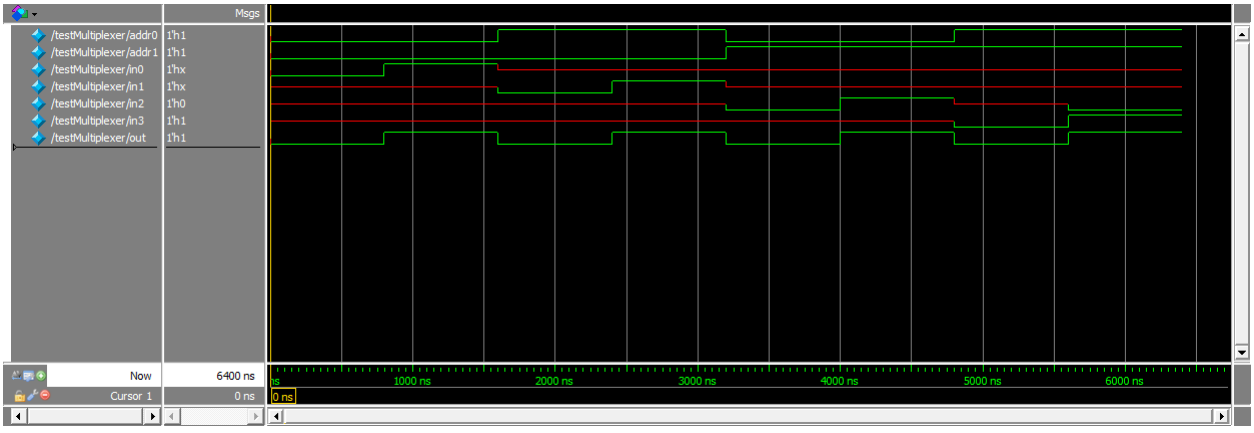


Figure 3a. Behavioral Multiplexer

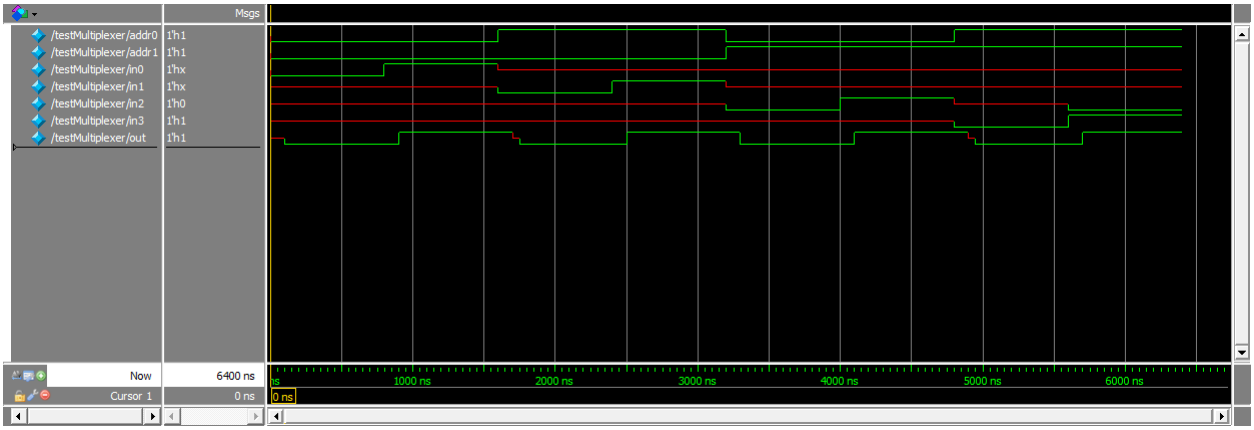


Figure 3b. Structural Multiplexer

Again, the waves for the two multiplexers (like the adders and the decoders) are similar, with the difference being that the structural waveform clearly shows a delay between the output and the input (i.e. near 1000ms, 1500ms, 2500ms, and so on). Around 1750ms and 5000ms, there are a couple "glitches" where the drop of the square wave has a small step embedded in it. This is because the circuit is still trying to finish computations at those points.

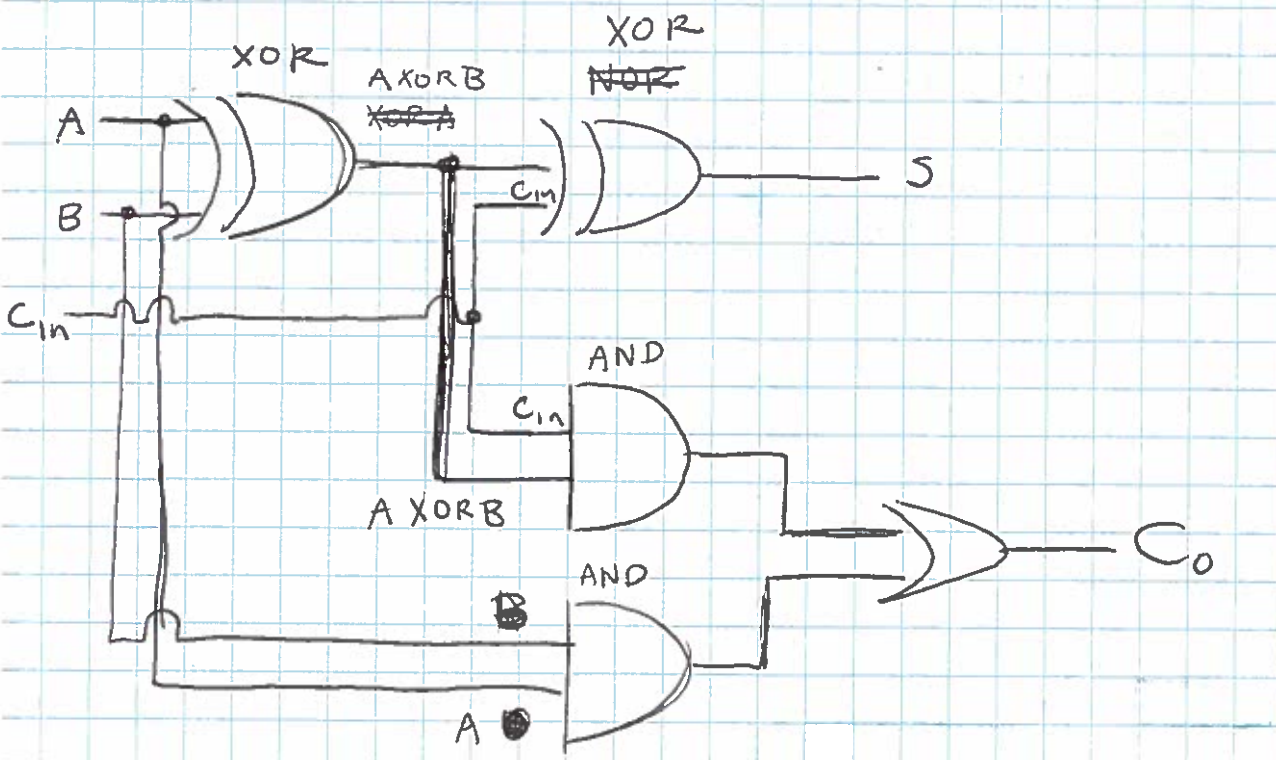
Comments/Acknowledgements

On the scans attached are sketches and tables for the three devices. I received some help from NINJAs when figuring out how to draw/setup the logic for the full adder, adding X's to the multiplexer table to make it better and account for more cases, and understanding the waveforms.

Note that variable names changed between the sketches and the Verilog, but the overall structure should be the same.

1 BIT FULL ADDER

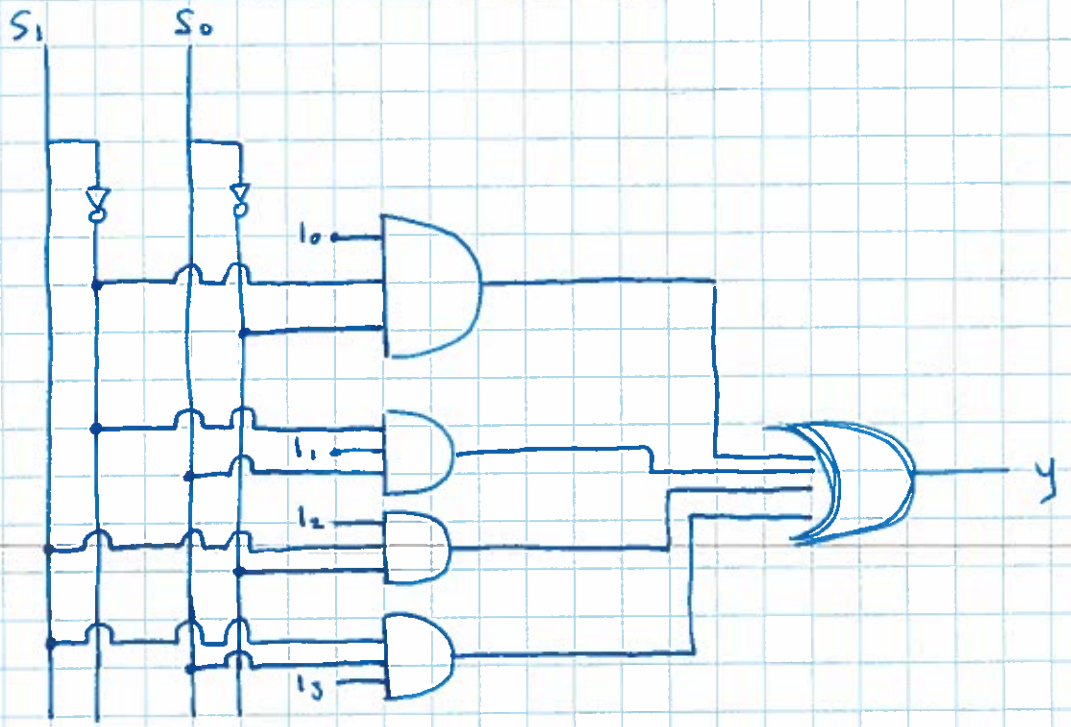
~~CON~~:



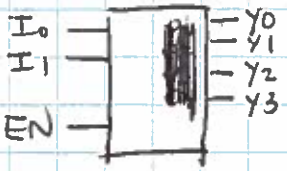
A	B	C_{in}	S	C_o
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

4 : 1 FOUR INPUT MULTIPLEXOR

S_1	S_0	I_3	I_2	I_1	I_0	y
0	0	X	X	X	0	0
0	0	X	X	X	1	1
0	1	X	X	0	X	0
0	1	X	X	1	X	1
1	0	X	0	X	X	0
1	0	X	1	X	X	1
1	1	0	X	X	X	0
1	1	1	X	X	X	1

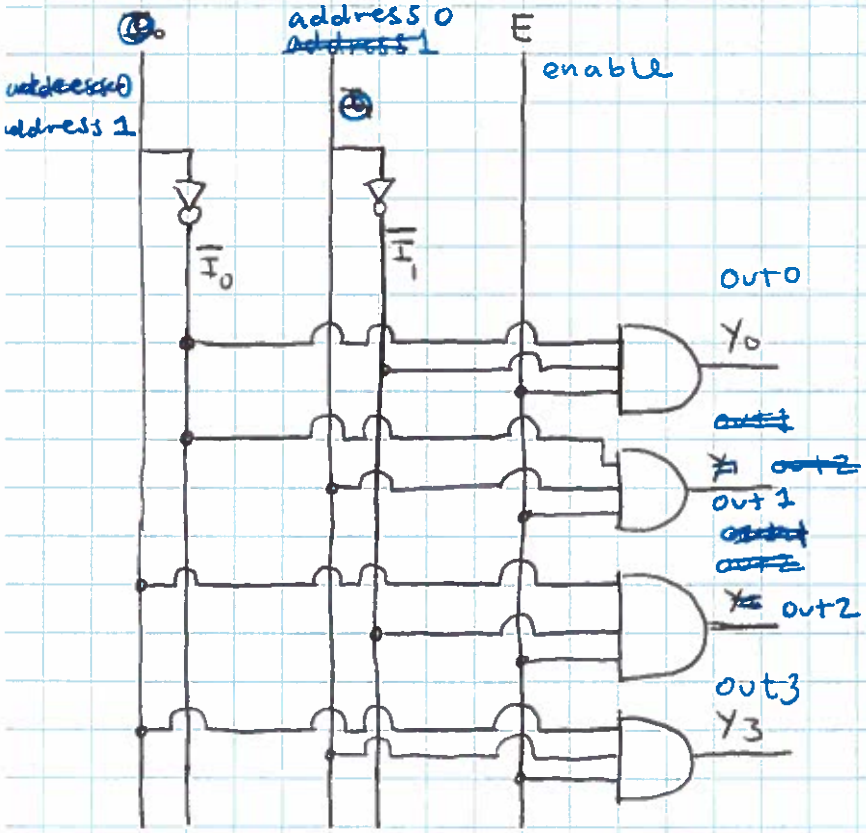


2 BIT DECODER W/ ENABLE (2+1 IN, 4 OUT)



I_1 I_0

~~I_0~~ ~~I_1~~	E	Y_0	Y_1	Y_2	Y_3
0 0	0	0	0	0	0
0 0	1	1	0	0	0
0 1	0	0	0	0	0
0 1	1	0	1	0	0
1 0	0	0	0	0	0
1 0	1	0	0	1	0
1 1	0	0	0	0	0
1 1	1	0	0	0	1



* EXCUSE ALL THE SCRIBBLES I ATTEMPTED TO KEEP NAMING CONVENTION CONSISTENT BETWEEN THIS + VERILOG