**Lab 6**

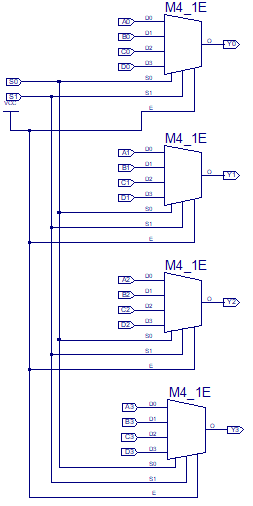
**Objective:**

To practice using decoders and multiplexers to implement functions as well as making more complex multiplexers from simpler ones.

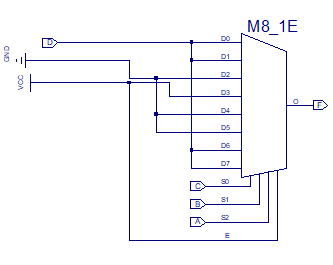
**Design:**



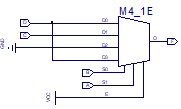




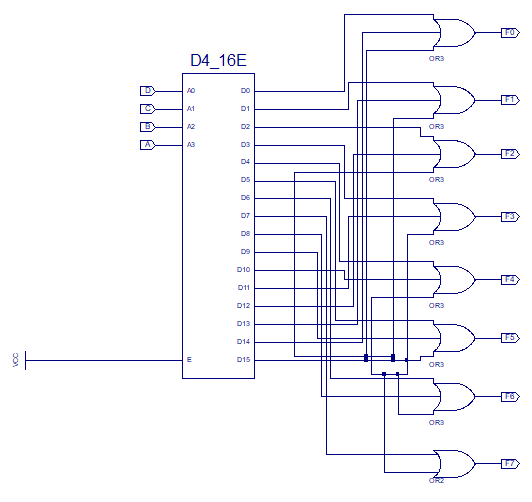
Quad 4:1 MUX Circuit



8:1 MUX Circuit



4:1 MUX Circuit



4:8 Multiplexer Circuit

**Procedure:**

1. Design a Quad 4:1 MUX using 4:1 MUXs
   * Test with the provided data by using a .do file
2. Use the following MUXs to implement the function [F(A,B,C,D) = ∑(1,3,6,7,13,15)]
   * 8:1 MUX
     1. Connect the MUX inputs per prelab design
     2. Simulate the design for all inputs by the clock method (16ns for MSB)
     3. Test the design on the board
   * 4:1 MUX
     1. Construct the MUX per prelab design
     2. Simulate the design for all inputs by the clock method (16ns for MSB)
     3. Test the design on the board (Hint: rename the .ucf file from the 8:1 MUX)
3. Implement the series of functions using a 4:19 decoder

F0 ( A, B, C, D ) = Σ m( 0 , 14, 15 )

F1 ( A, B, C, D ) = Σ m( 1, 13 , 15 )

F2 ( A, B, C, D ) = Σ m( 2 , 12 , 15 )

F3 ( A, B, C, D ) = Σ m( 3, 11 , 15 )

F4 ( A, B, C, D ) = Σ m( 4 , 10 , 15 )

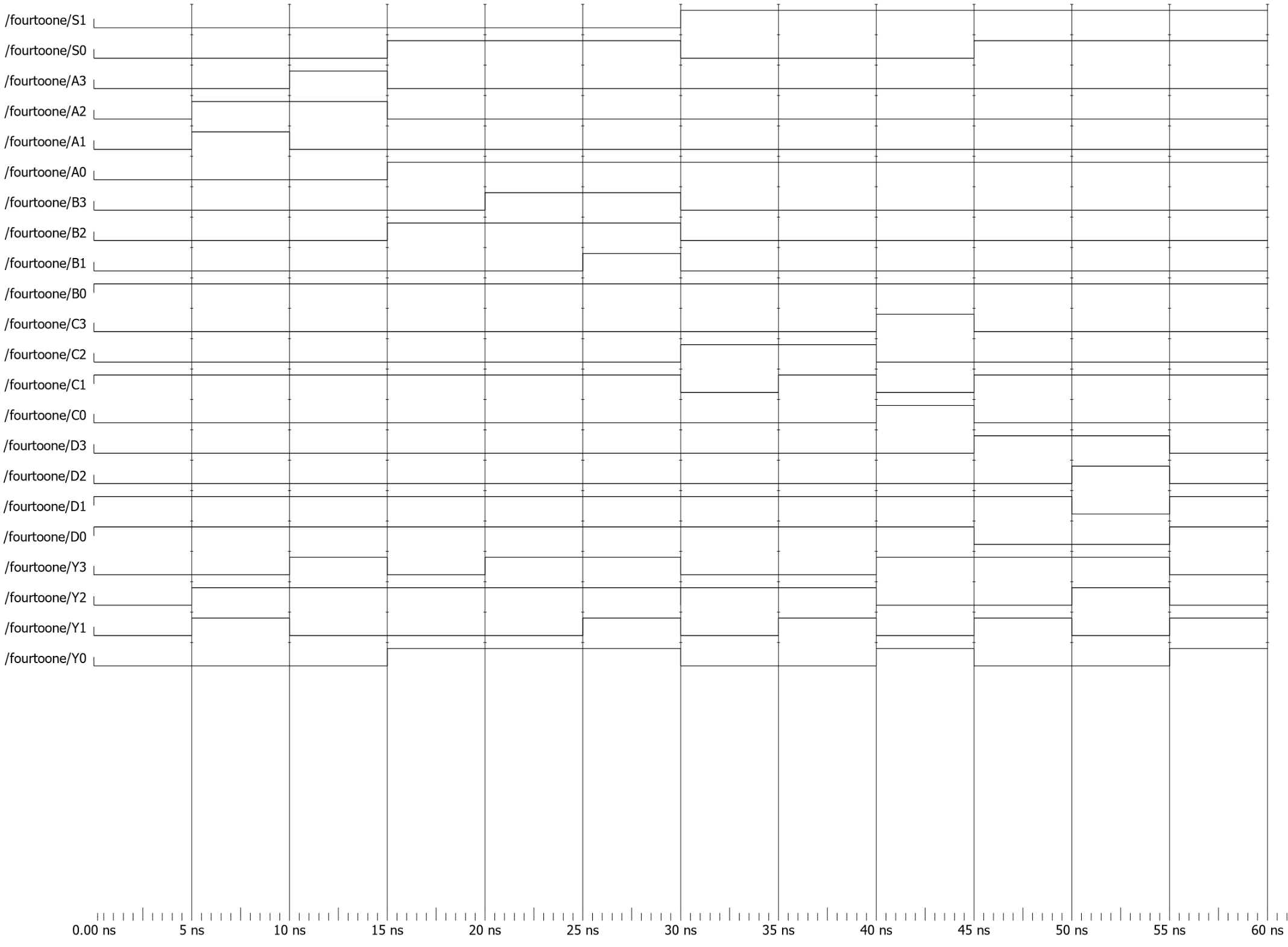
F5 ( A, B, C, D ) = Σ m( 5 , 9 , 15 )

F6 ( A, B, C, D ) = Σ m( 6 , 8 , 15 )

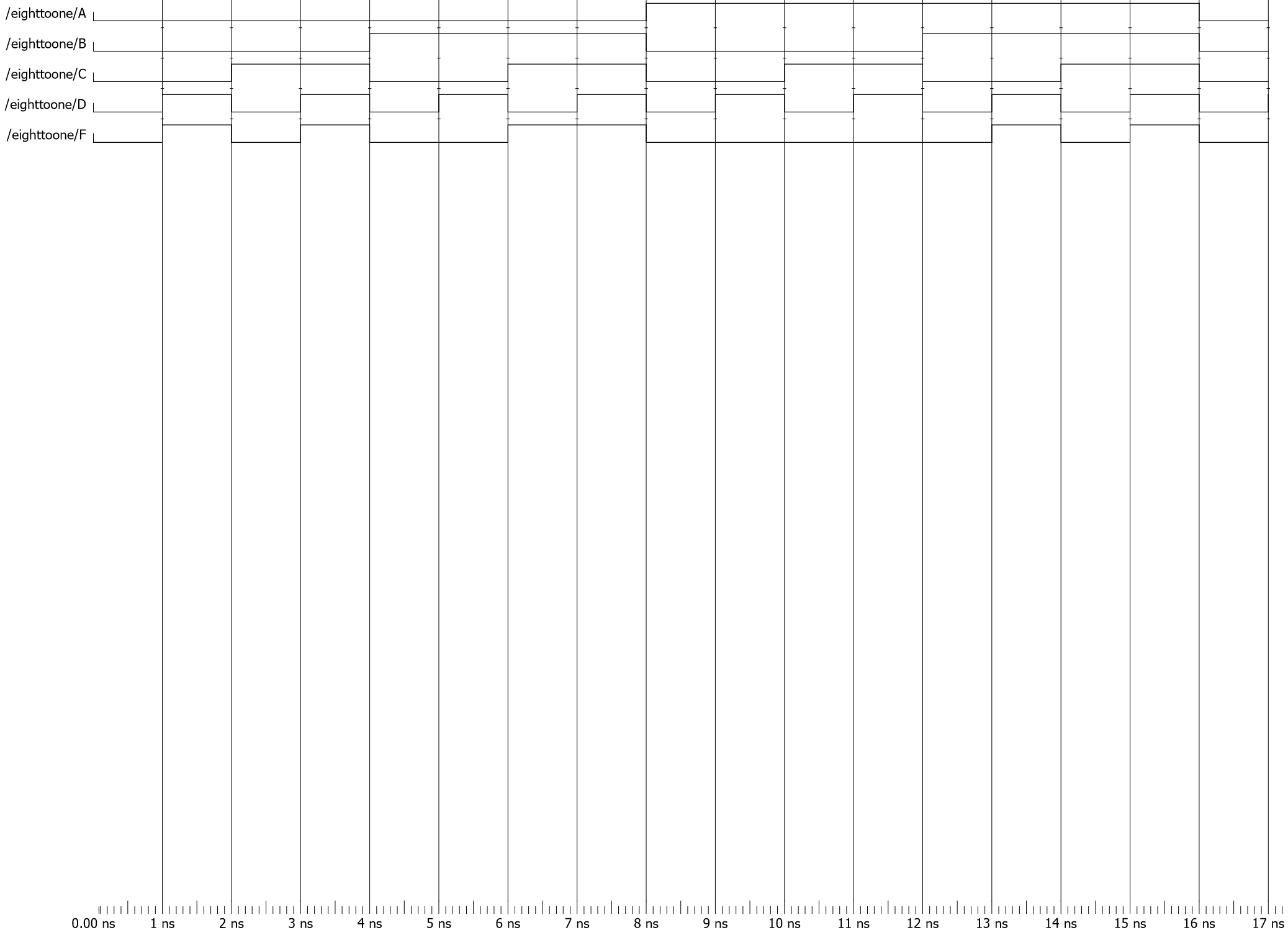
F7 ( A, B, C, D ) = Σ m( 7 , 15 )

* Construct the circuit per prelab design
* Simulate the design for all combinations using the clock method (16ns for MSB)
* Download and test on the board
  + Connect F0 to the right-most LED and F7 to the left-most
  + LED should travel right to left and back again
  + All LEDs should be on when all inputs are 1

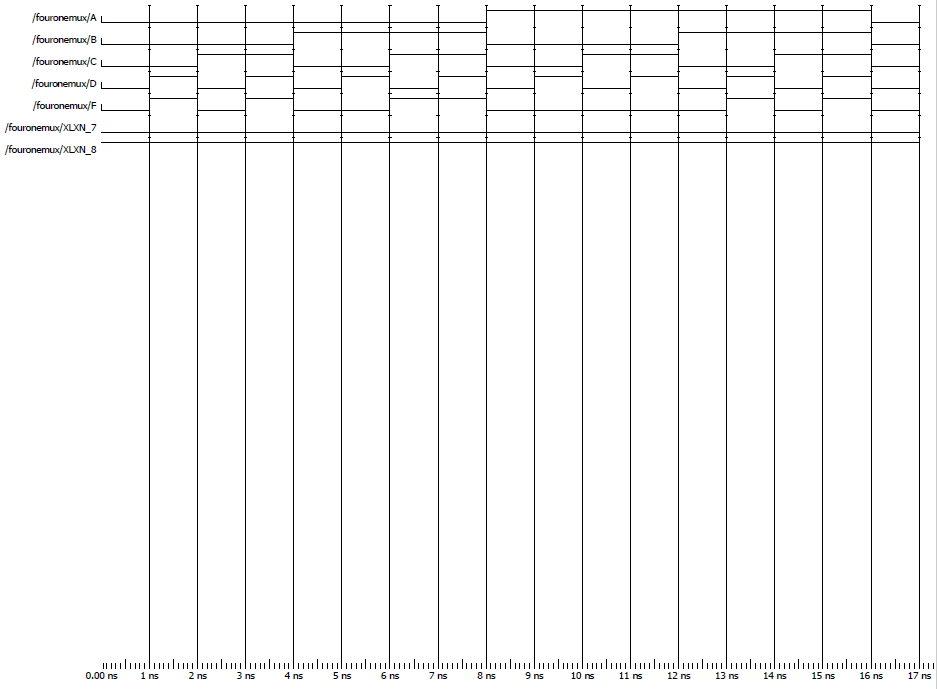
**Data:**

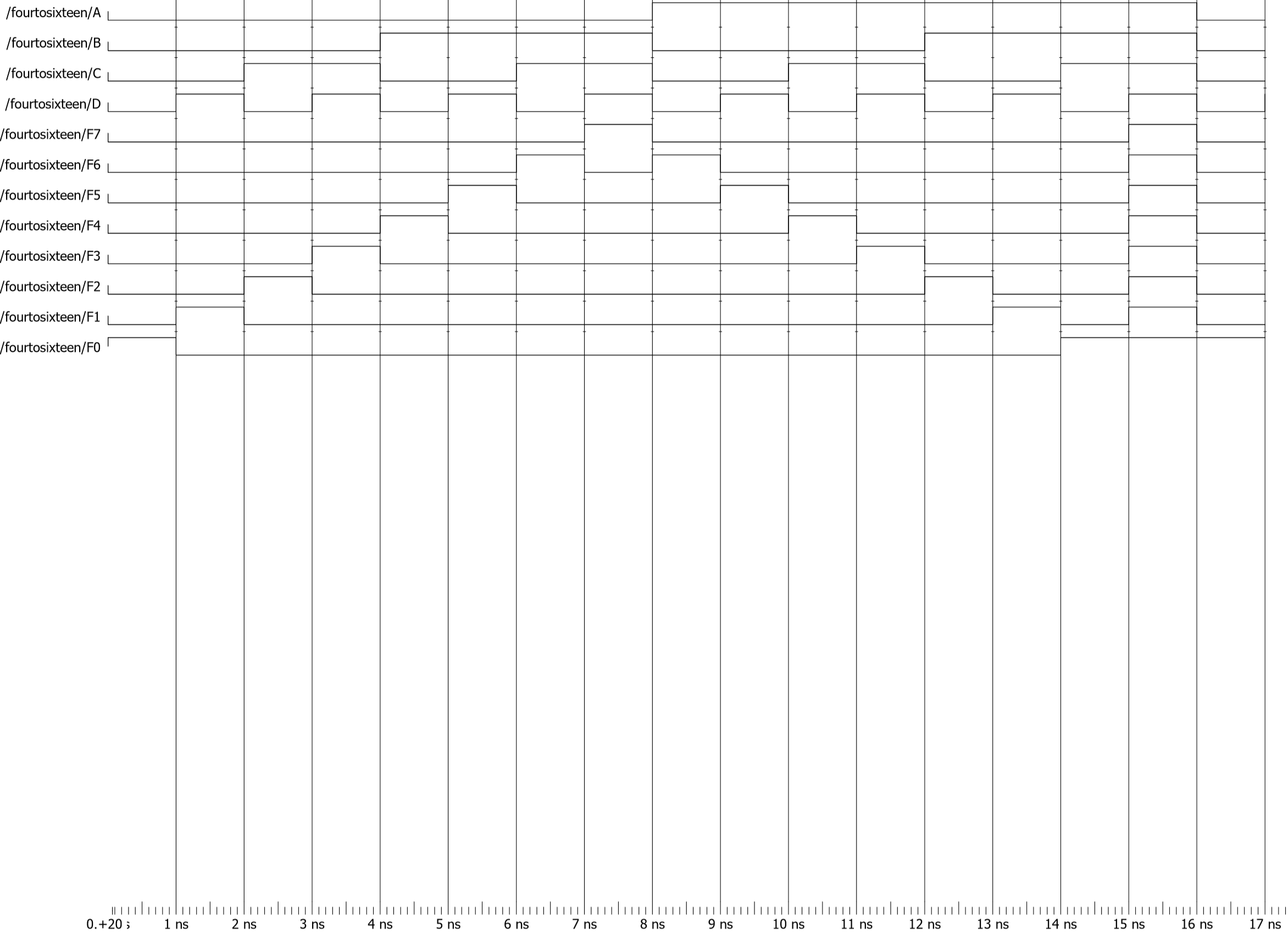


Quad 4:1 MUX Circuit Simulation



8:1 MUX Circuit Simulation



4:1 MUX Circuit Simulation

4:16 Decoder Circuit Simulation

**Data Analysis:**

The designs worked as implemented.

For implementing functions with a using MUXs (part 2), using the 4:1 MUX would seemingly be more efficient as it would require fewer gates to function as compared to the 8:1 MUX. Doing a Post-Route simulation (including time delays) would have made it clearer if that would be the case.

For part three, I noticed that it is much easier to parse if a circuit is working if the output has some easily discernable pattern, though that isn’t always practical to implement. Since the LEDs lit in an easily predictable matter (right to left, left to right, then all on), it was the quickest I have tested and verified that my circuit worked appropriately.

**Conclusion:**

One of the goals of this experiment was to reinforce the ongoing idea from the beginning of the semester of the benefits of modularity; of using simpler pieces that we have designed before to make a more complex circuit. Designing the Quad 4:1 MUX was where this was most prominent and, as in previous weeks, it greatly simplified the process of designing something to give the same output (outputting the indexed values from four separate 4-bit inputs).

The other goal of implementing functions using multiplexers and decoders was a surprisingly simple process. By having a design that you know the output of, or what input will pass through to the output, it allows one to add a little complexity to one side of a circuit to get a desired result rather than design an entirely new circuit to get the same results. At that, it also allows one to simplify some inputs to get the same result, such as some of the 8:1 MUX implementation inputs simplifying to 1s and 0s instead of a changing input.