

David Ginsberg
EE134 Project
Simple ALU Layout with the Cadence Suite

Professor: Dr. Roger Lake
TA: Junjie Yang

I have designed a simple Arithmetic Logic Unit (ALU) that performs 4 operations. It can add, subtract, check if two numbers are equal, or check if one number is greater than the other number. It has two 4-bit data inputs, one 2 bit select line, and a 5-bit (4-bits and a carry/borrow) output line.

First I designed each of the basic logic components necessary to build this ALU. These can be seen in figures 1-6. The order they appear is inverter, AND, OR, transmission gate, XOR, and multiplexer. For many of these components I followed the VLSI design style given by R. Jacob Baker, Harry W. Li, and David E. Boyce in their book CMOS Circuit Design, Layout and Simulation. The inverter, AND, OR and multiplexer all follow their standard design. I used the transmission gate implementation of the XOR and multiplexer. Each of these components includes a schematic, simulation and a layout. I used the transistors from the NCSU digital library in order to design and simulate the schematics.

Once I had completed the basic logic gates, I could create the more advanced logic structures. I used these basic logic structures to build a 2-bit adder and a 2-bit subtractor. The schematics and layouts for these components can be seen in figures 7 and 8.

Once all the ALU components had been designed they just had to be put together and simulated. The ALU schematic can be seen in figure 9a. I performed a simulation on this schematic using Verilog XL. The simulation input file can be seen in Table 1, and the simulation waveforms can be seen in figure 9c. The layout for ALU can be seen in figure 9c and 9d.

After all this was completed, the ALU was small enough that there would be extra space left on the die. In the extra space I connected 800 inverters in series to create a ring oscillator. This can be seen in figure 10a and a more detailed view can be seen in figure 10b. All the inverters are minimum size. The frequency of oscillation should be about 4.07Mhz.

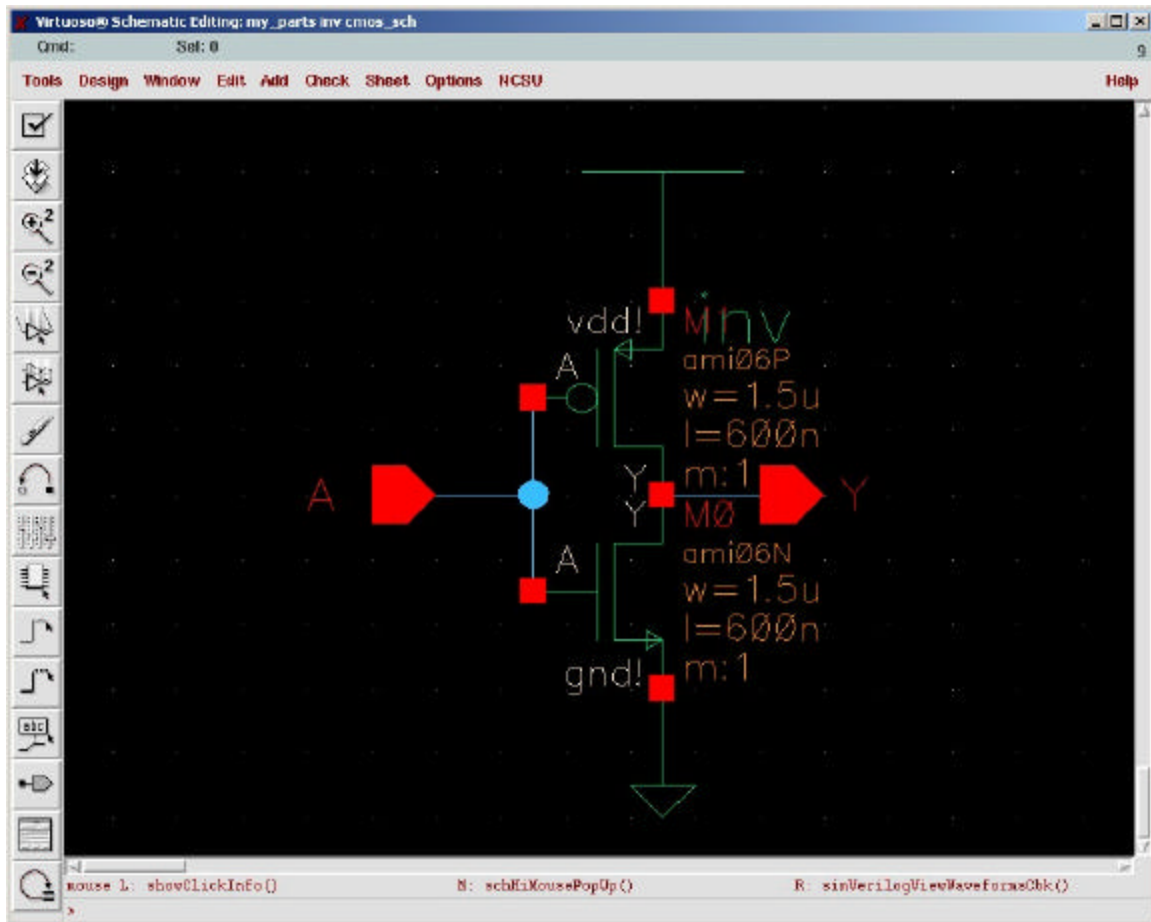


Figure 1a. Inverter Schematic

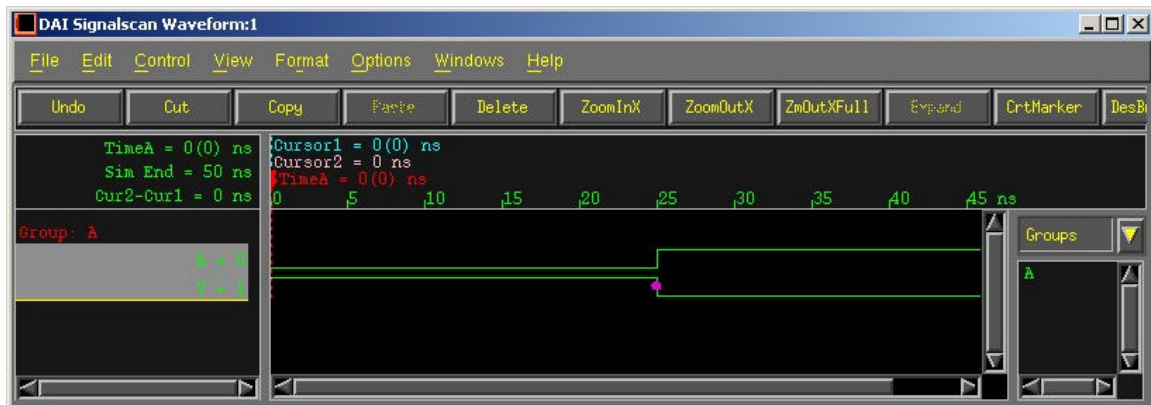


Figure 1b. Inverter Simulation

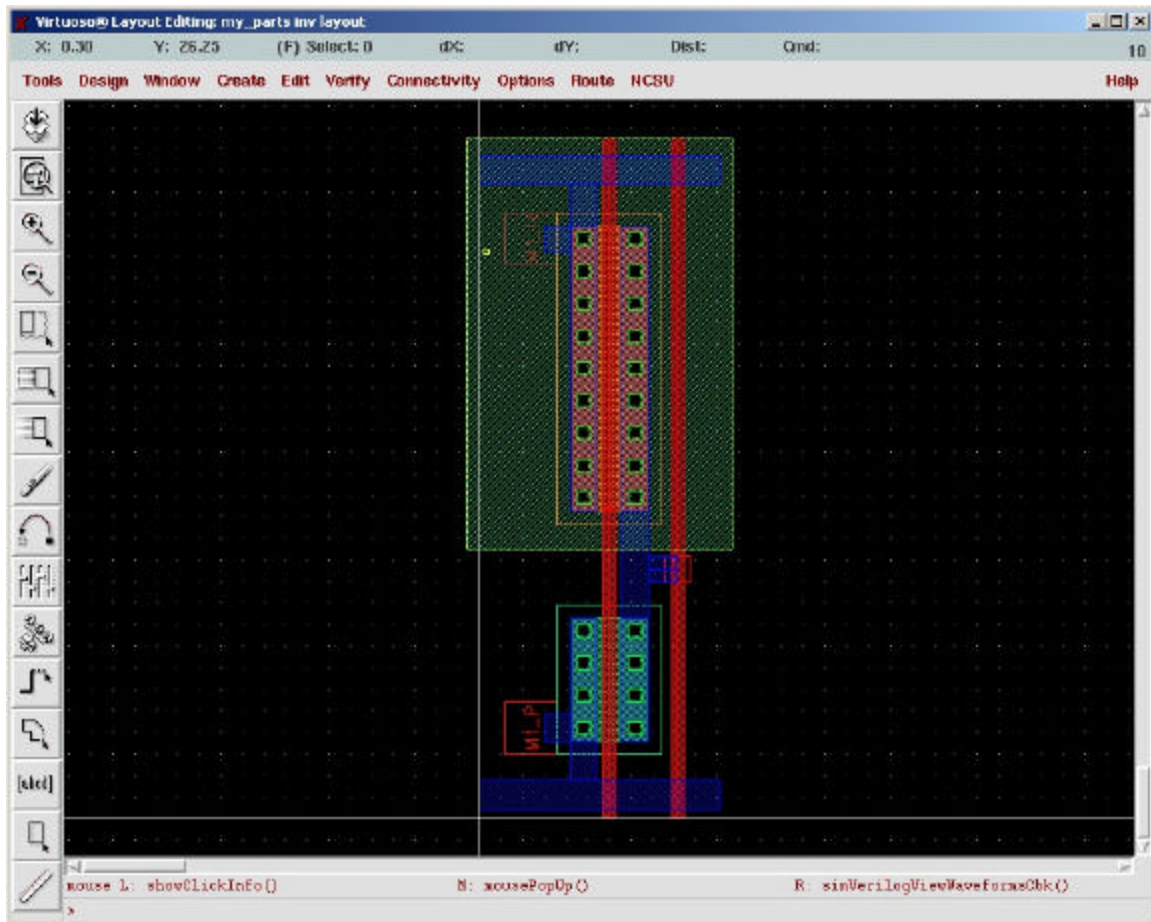


Figure 1c. Inverter Layout

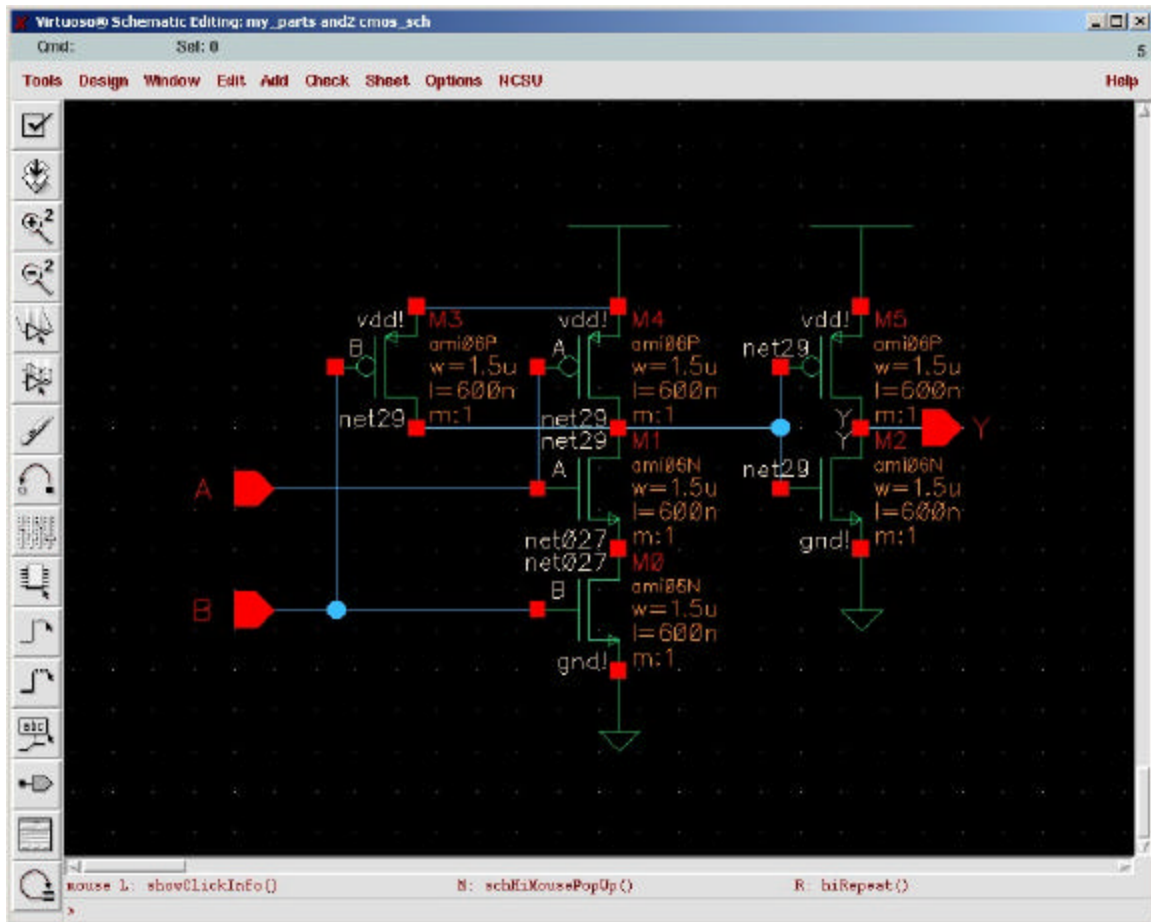


Figure 2a. AND schematic.

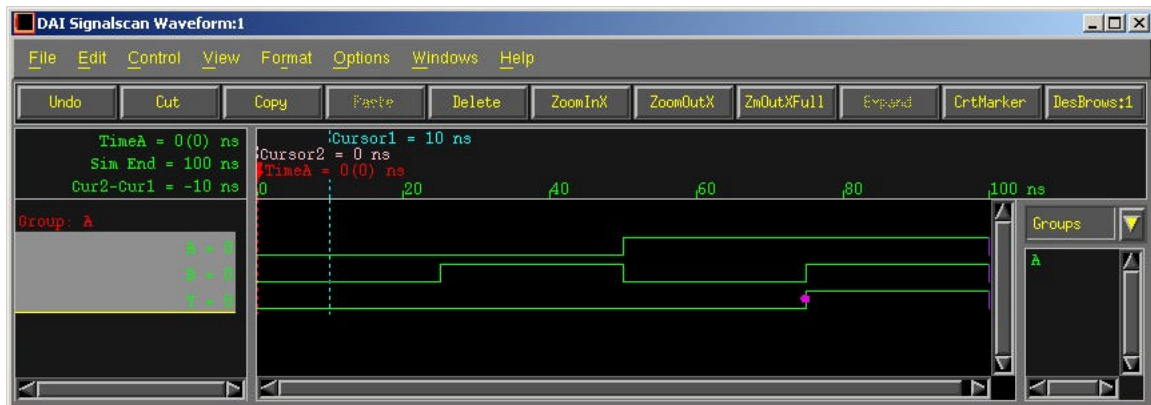


Figure 2b. AND simulation.

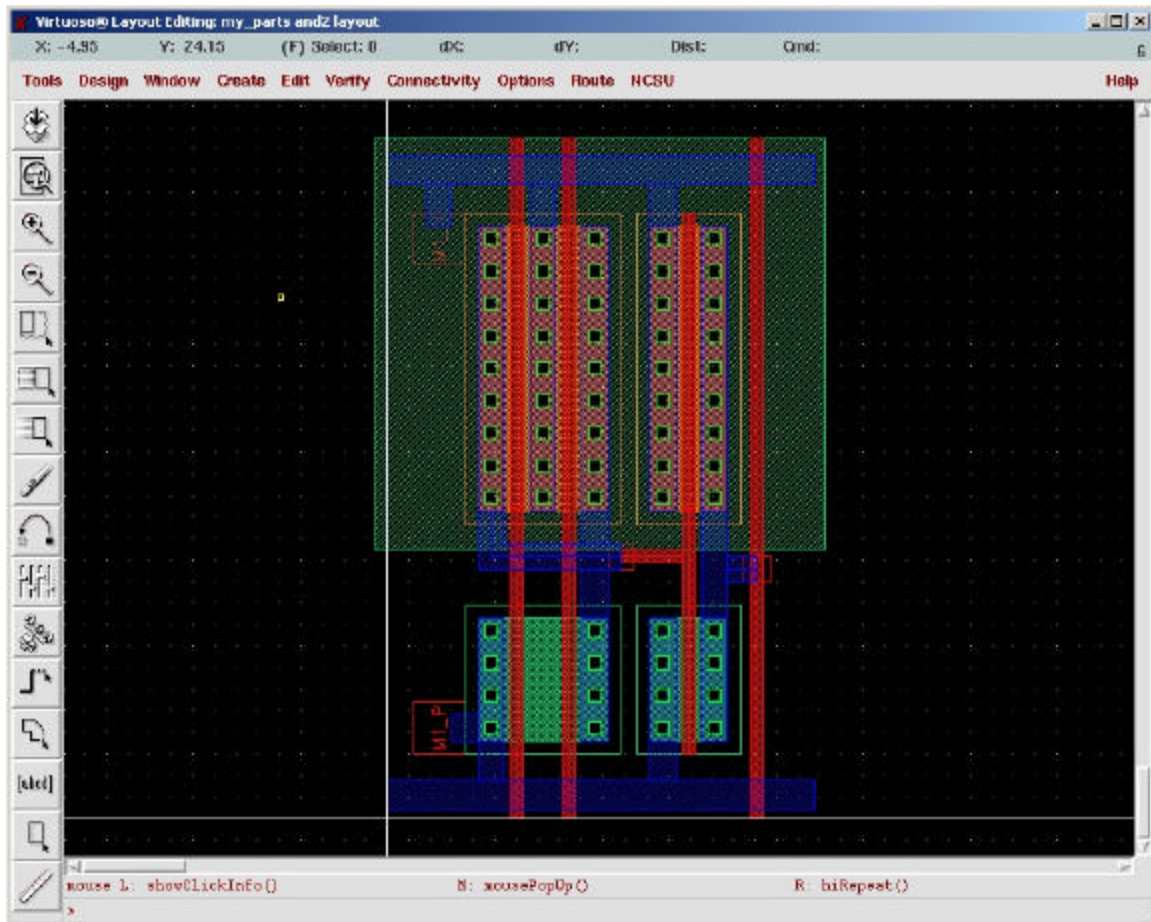


Figure 2c. AND layout.

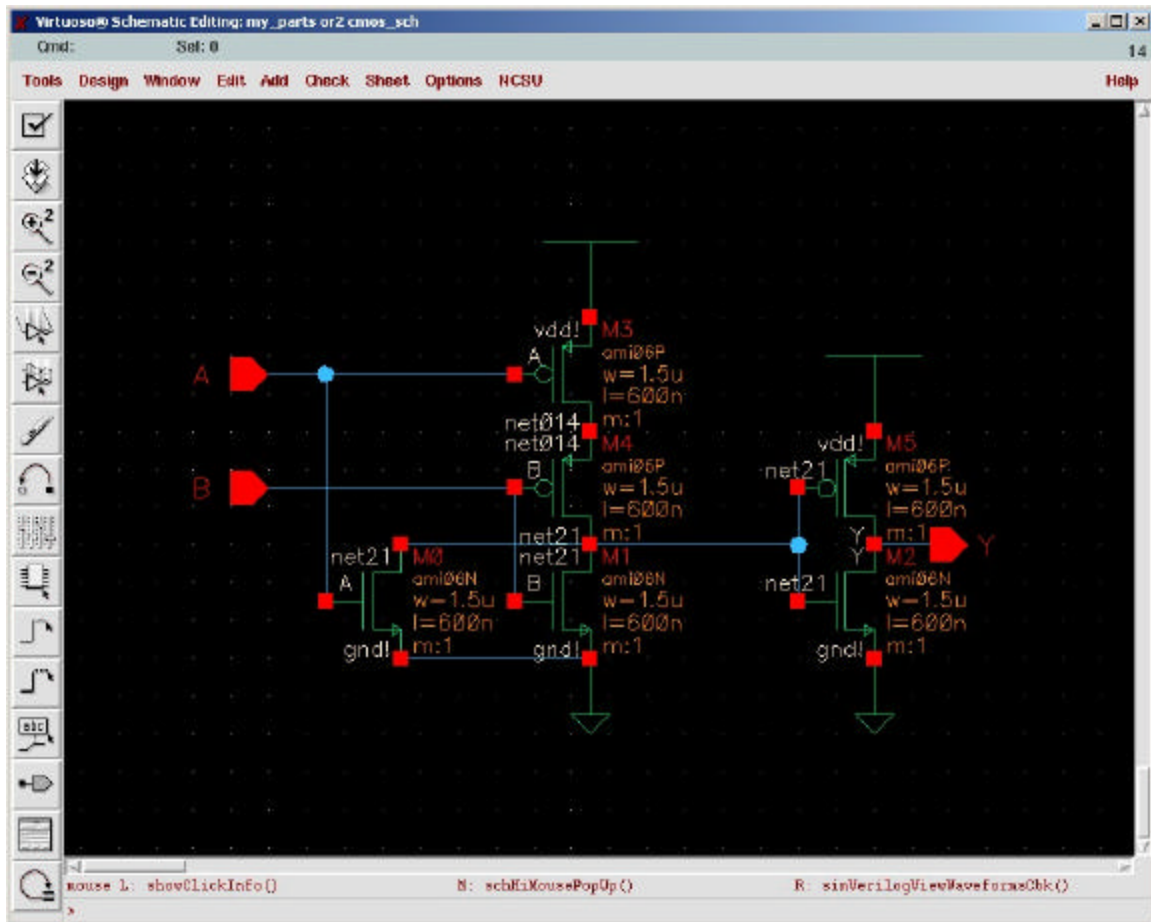


Figure 3a. OR schematic.

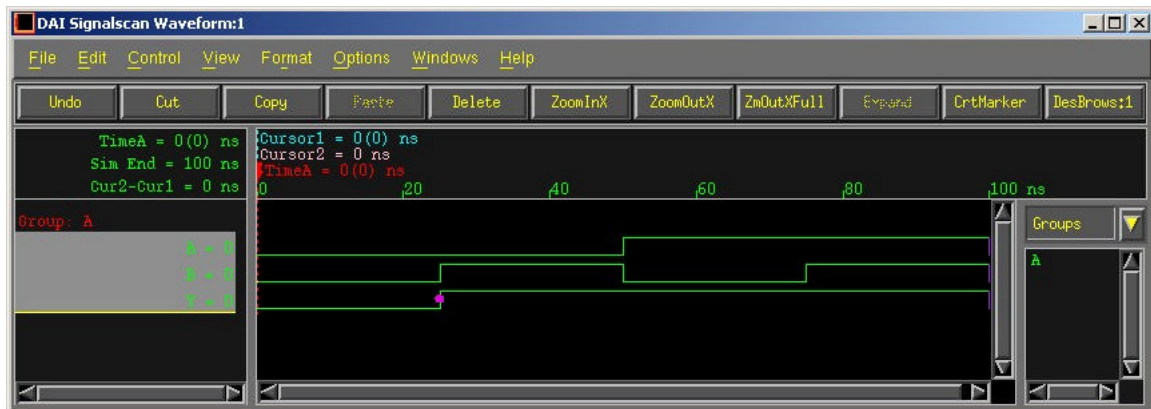


Figure 3b. OR simulation.

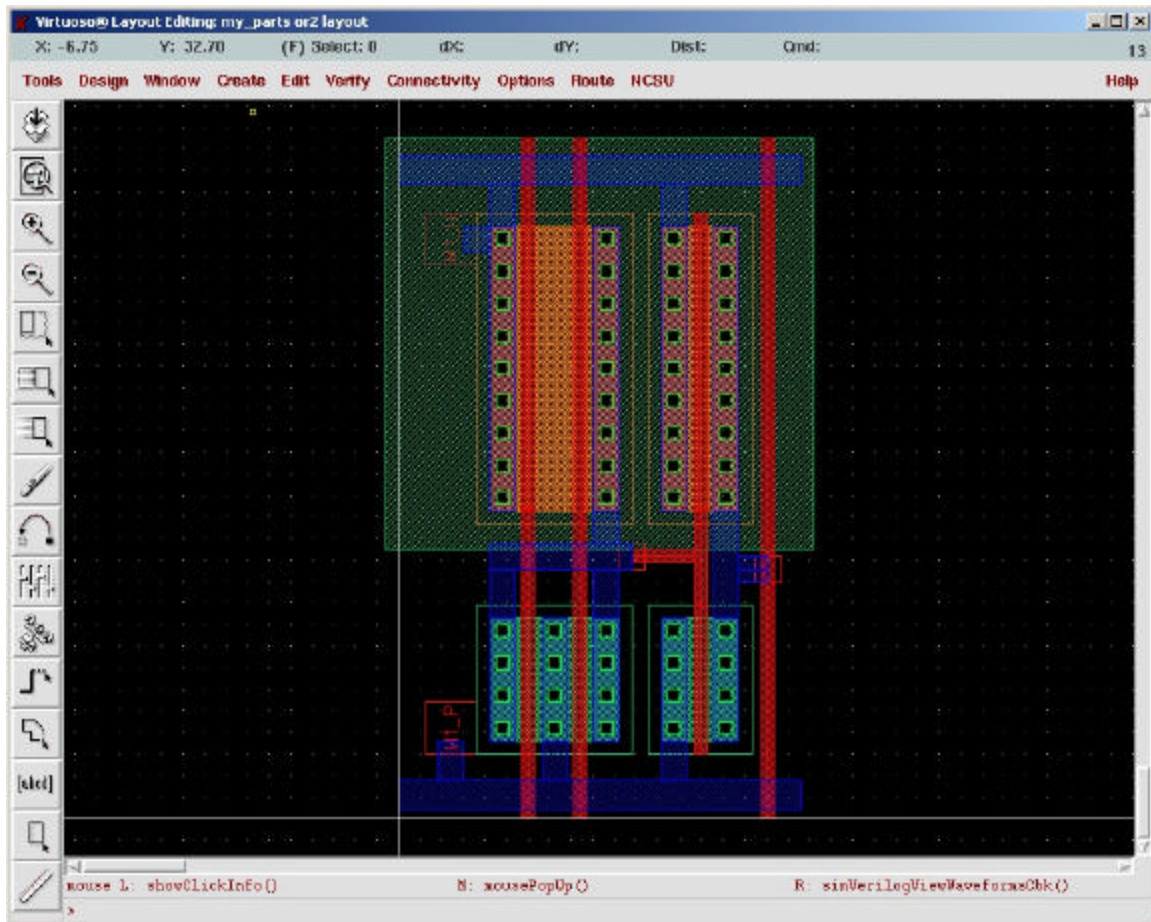


Figure 3c. OR layout.

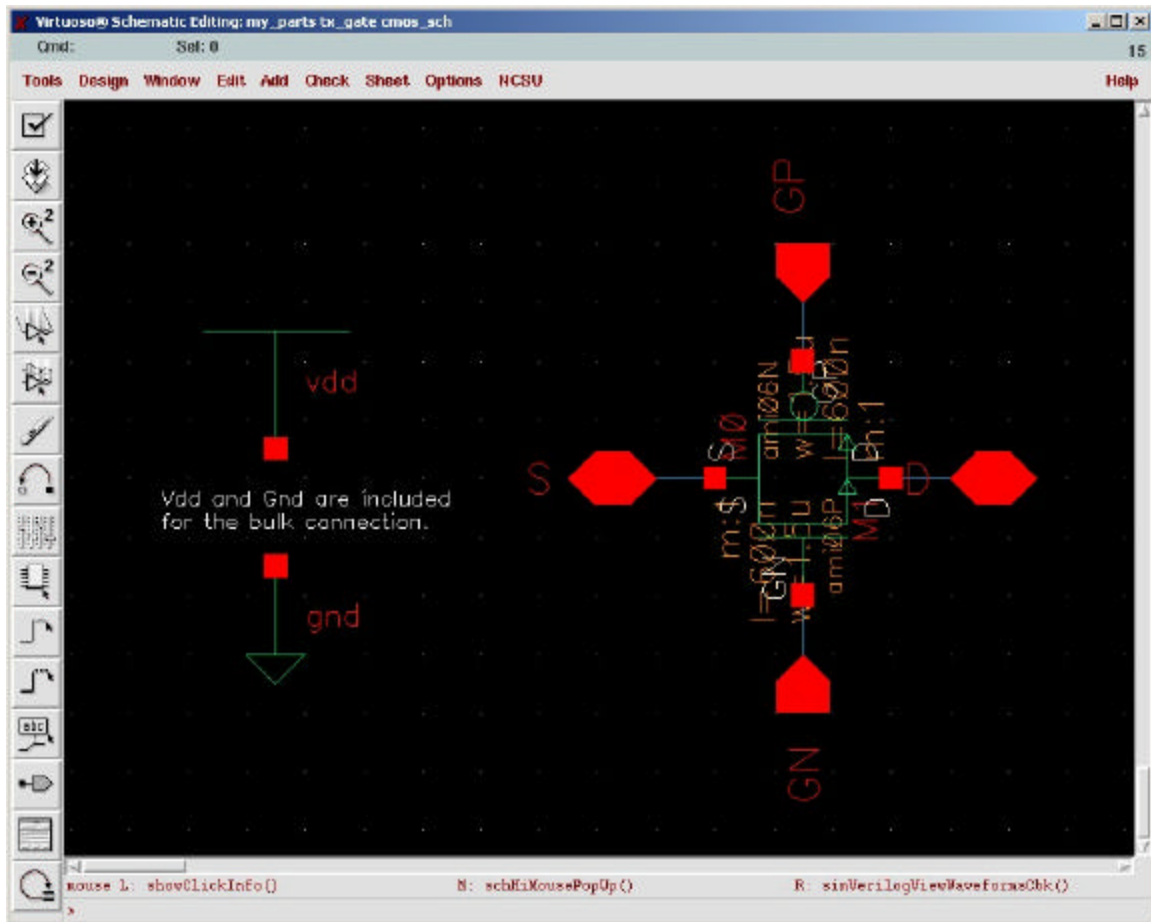


Figure 4a. Transmission Gate schematic.

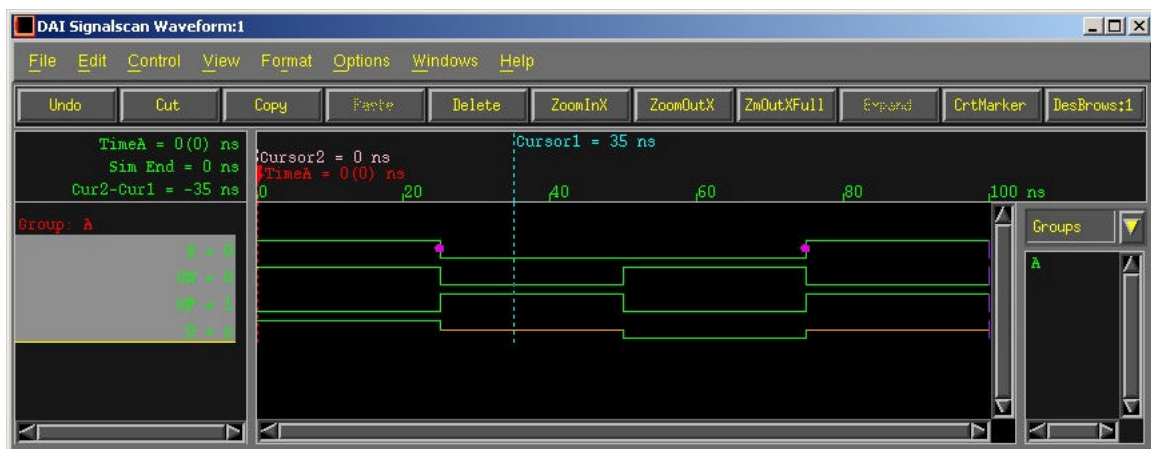


Figure 4b. Transmission Gate simulation.

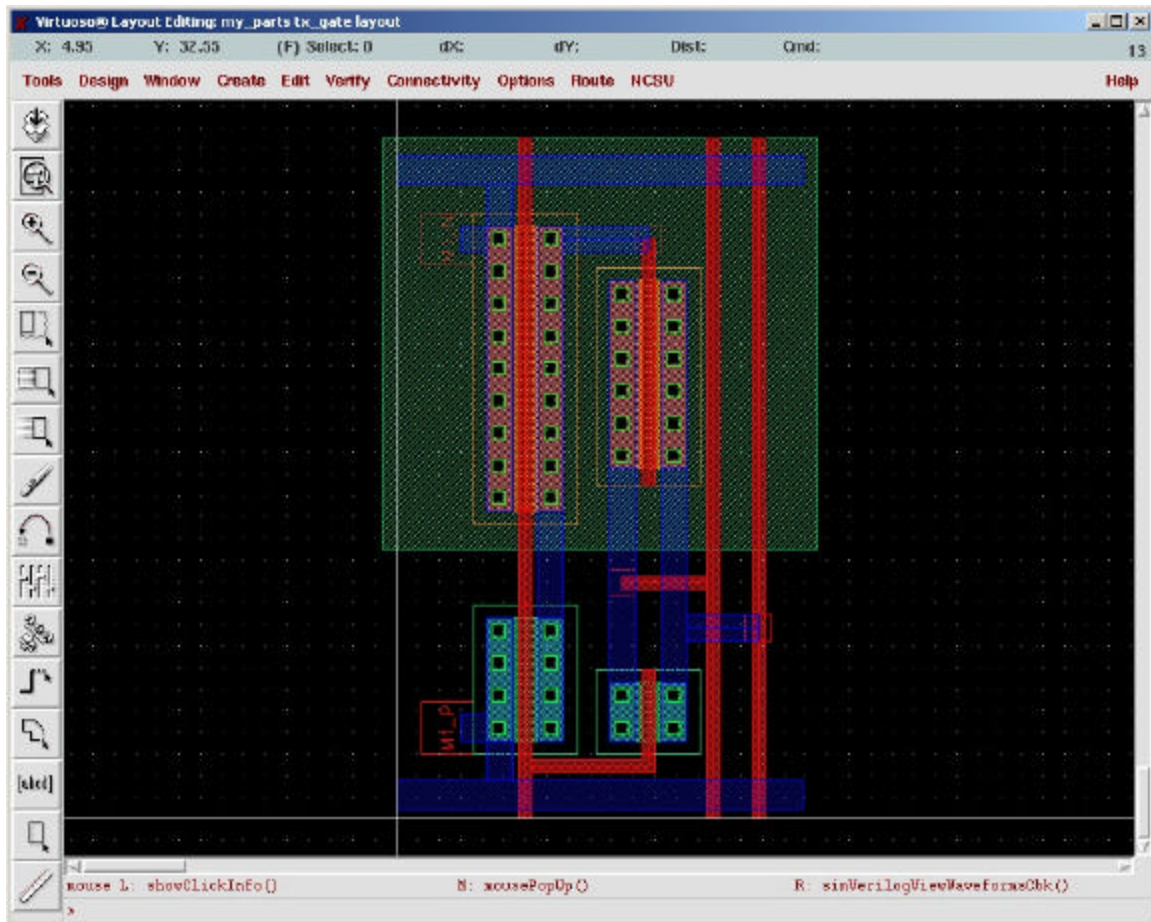


Figure 4c. Transmission Gate layout.

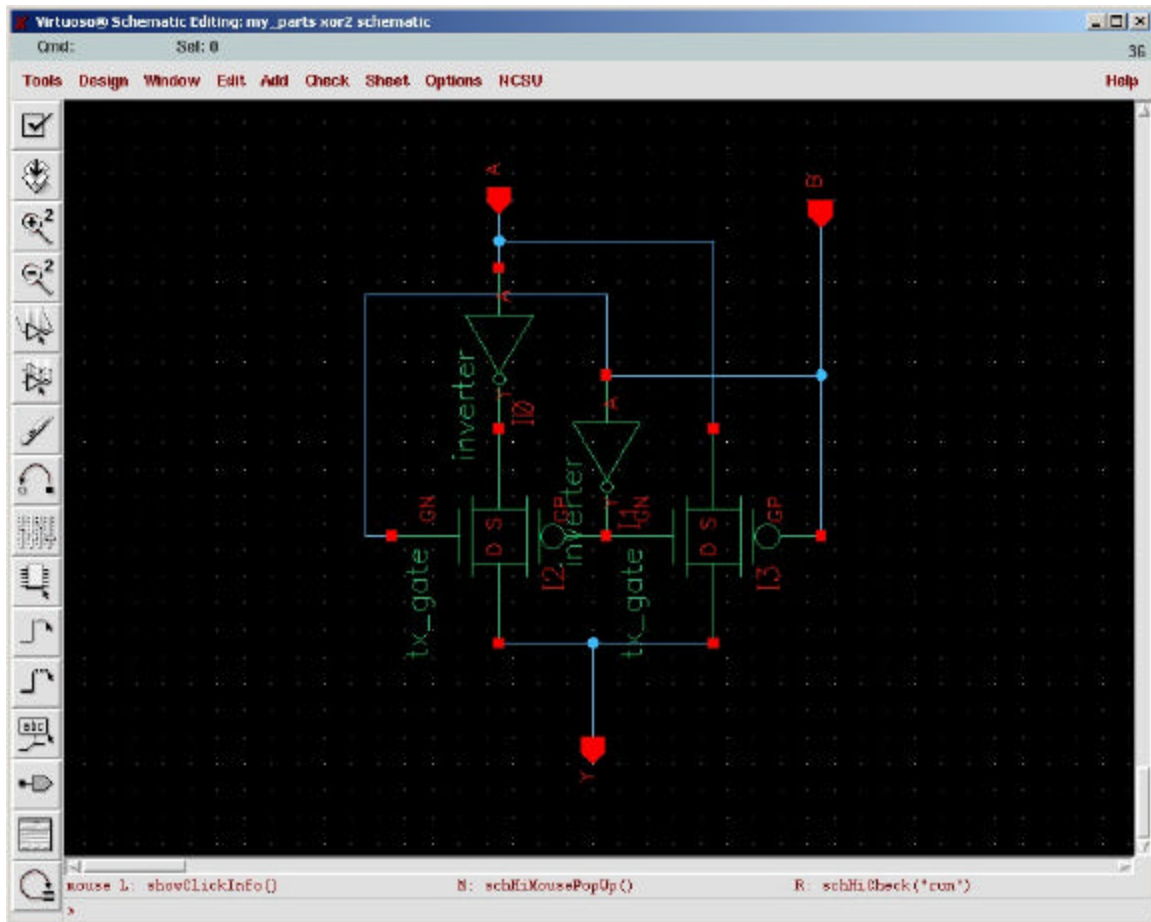


Figure 5a. XOR schematic.

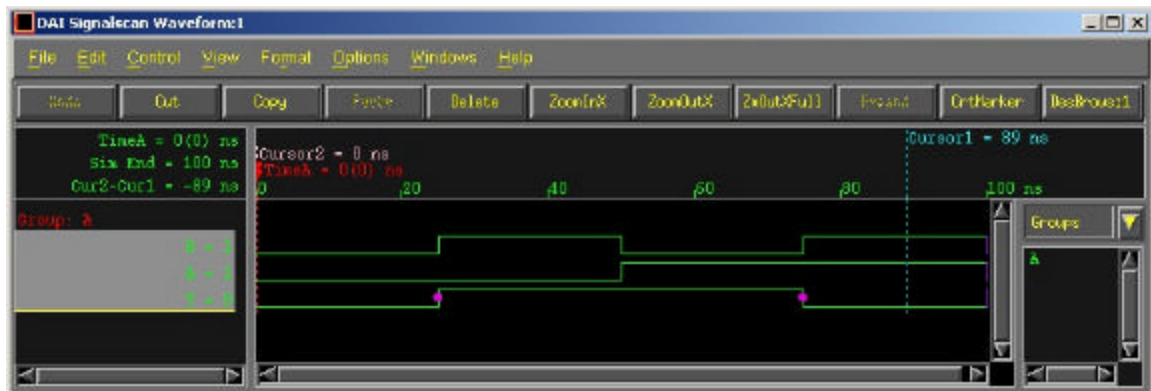


Figure 5b. XOR simulation.

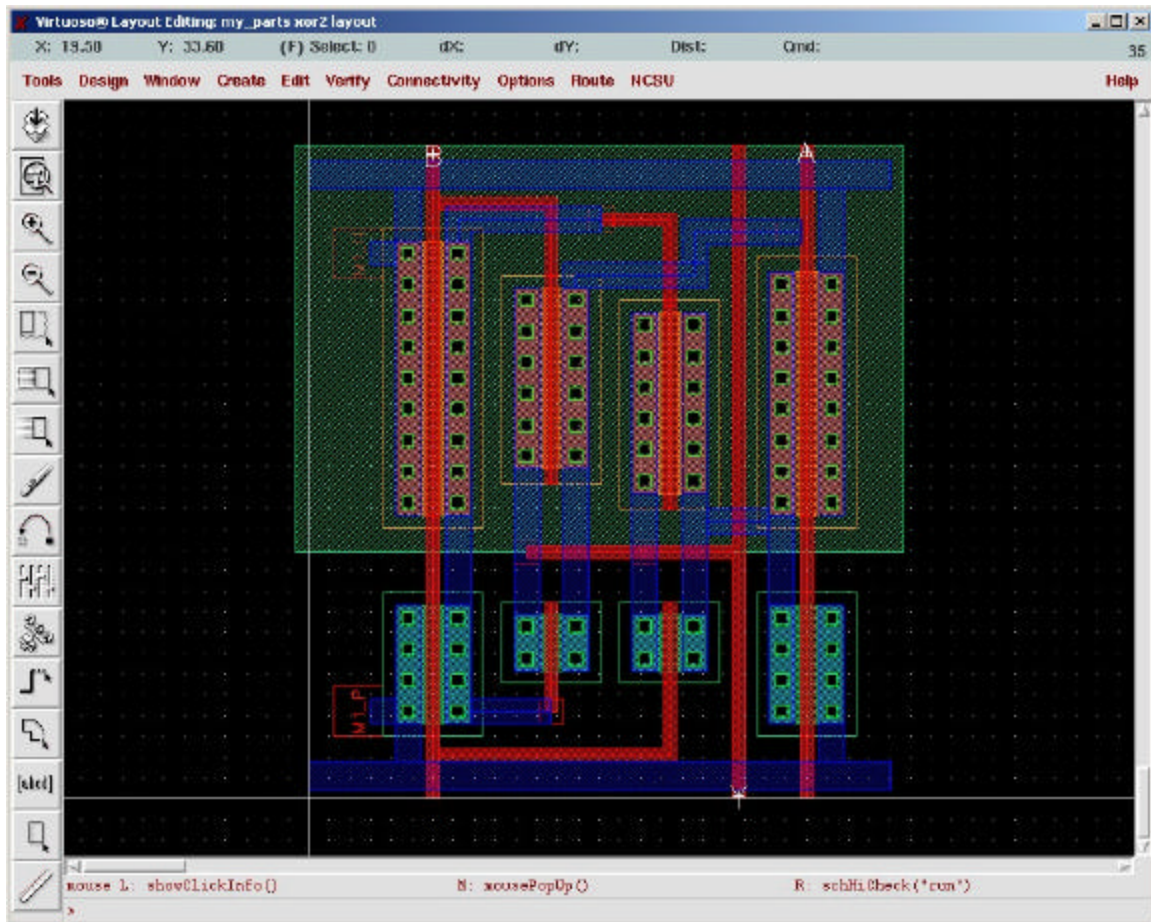


Figure 5c. XOR layout.

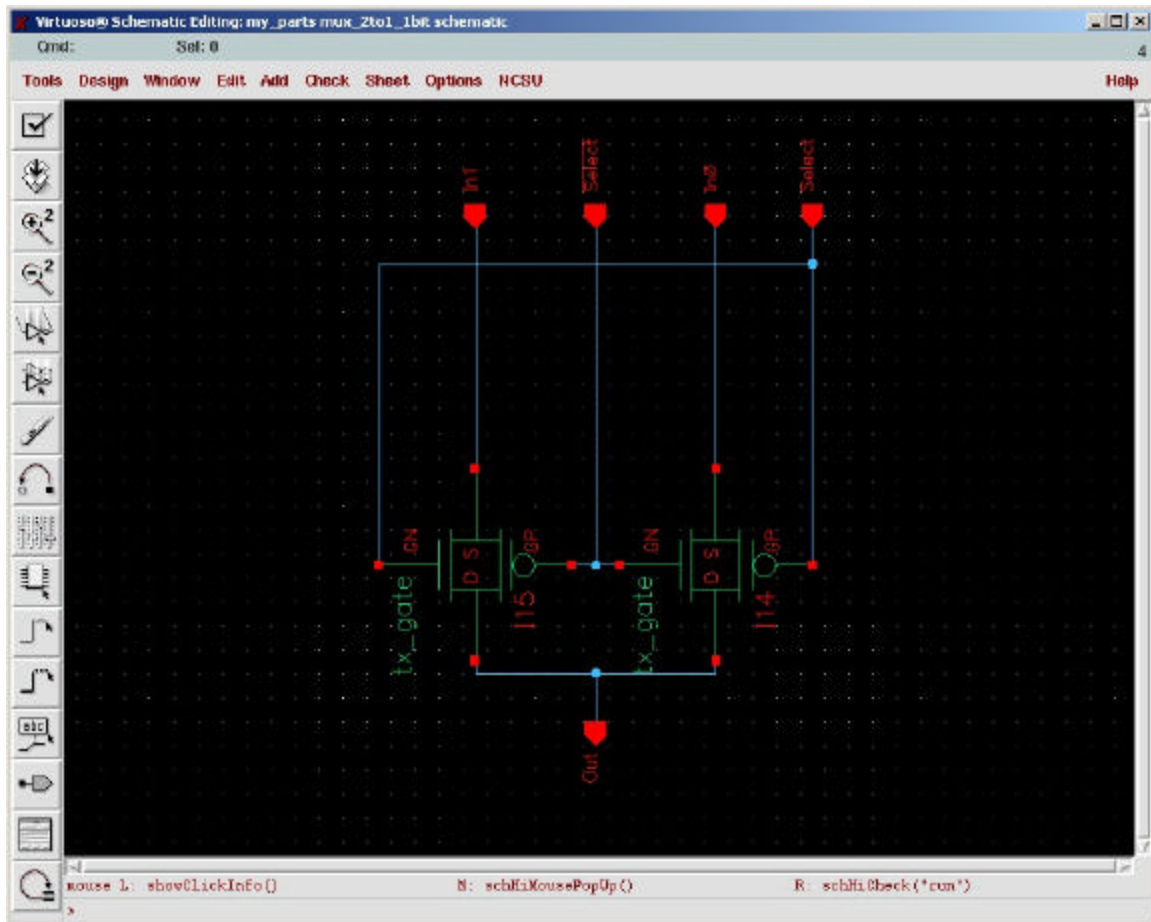


Figure 6a. Multiplexer schematic.

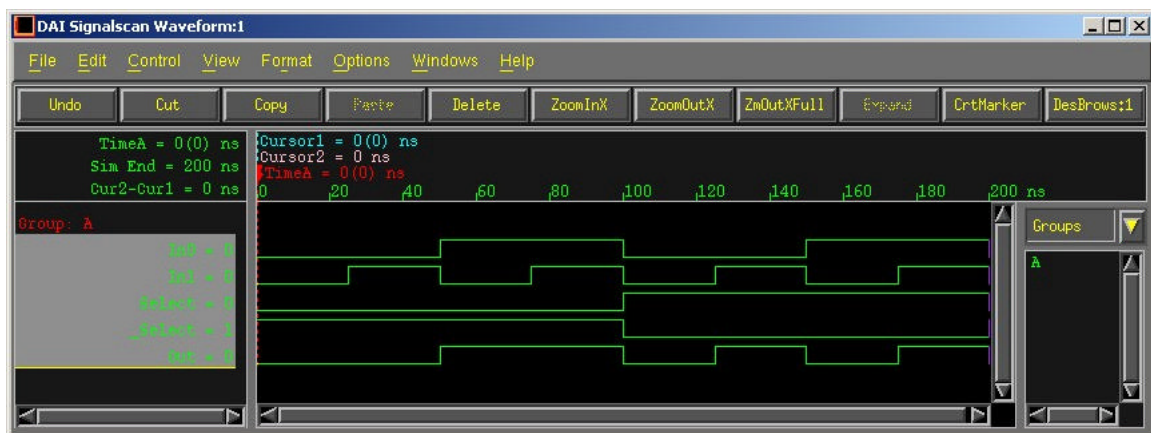


Figure 6b. Multiplexer simulation.

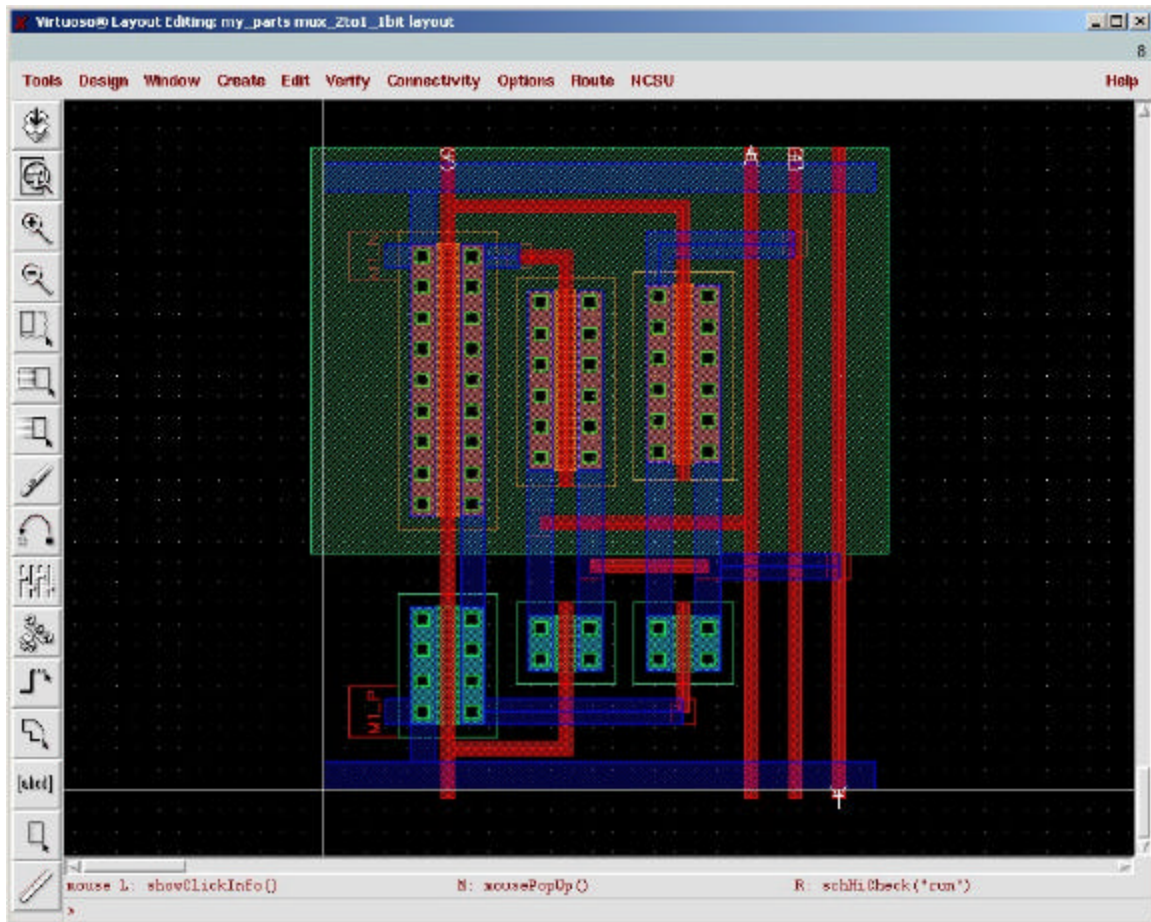


Figure 6c. Multiplexer layout.

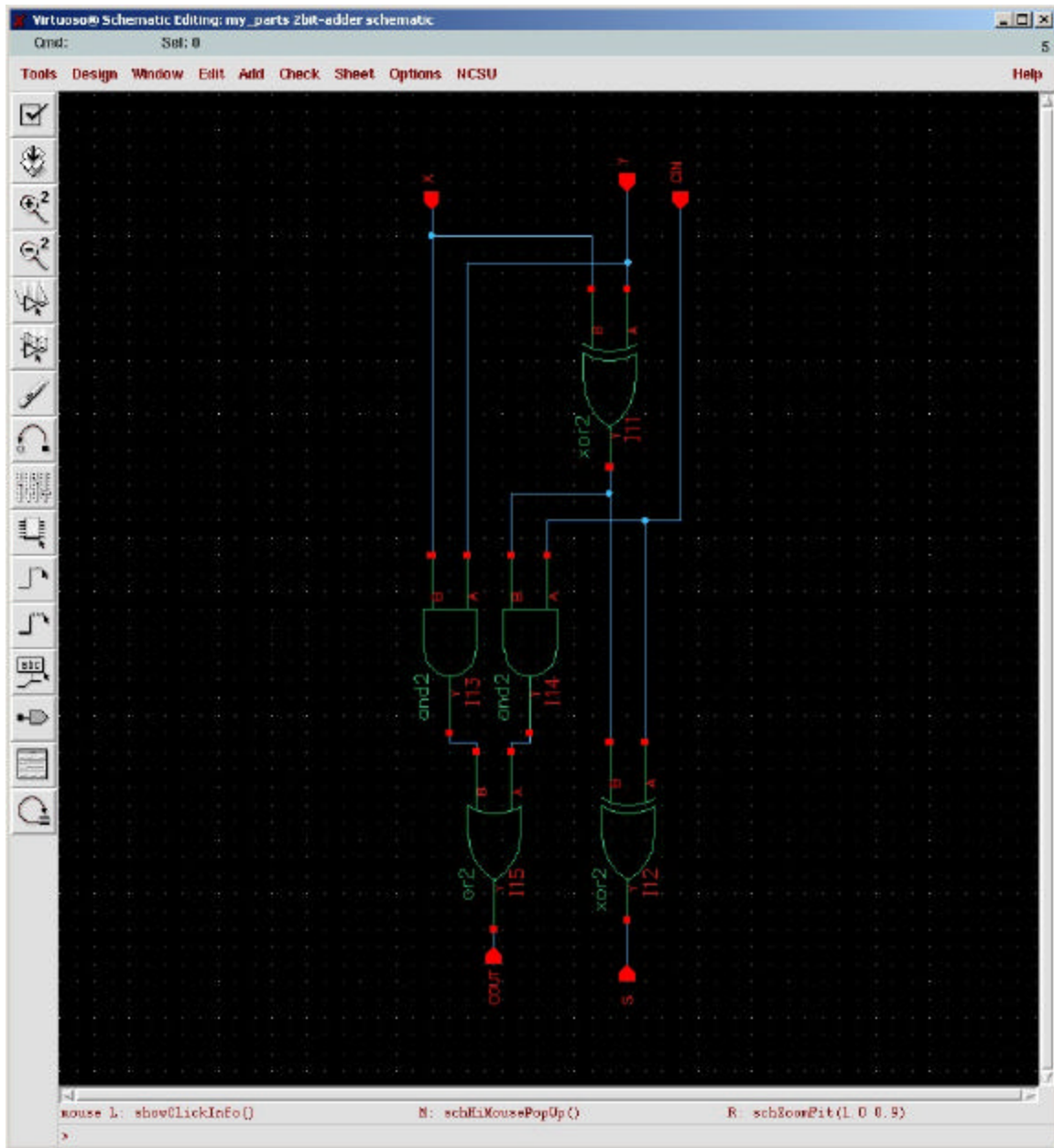


Figure 7a. Adder schematic.

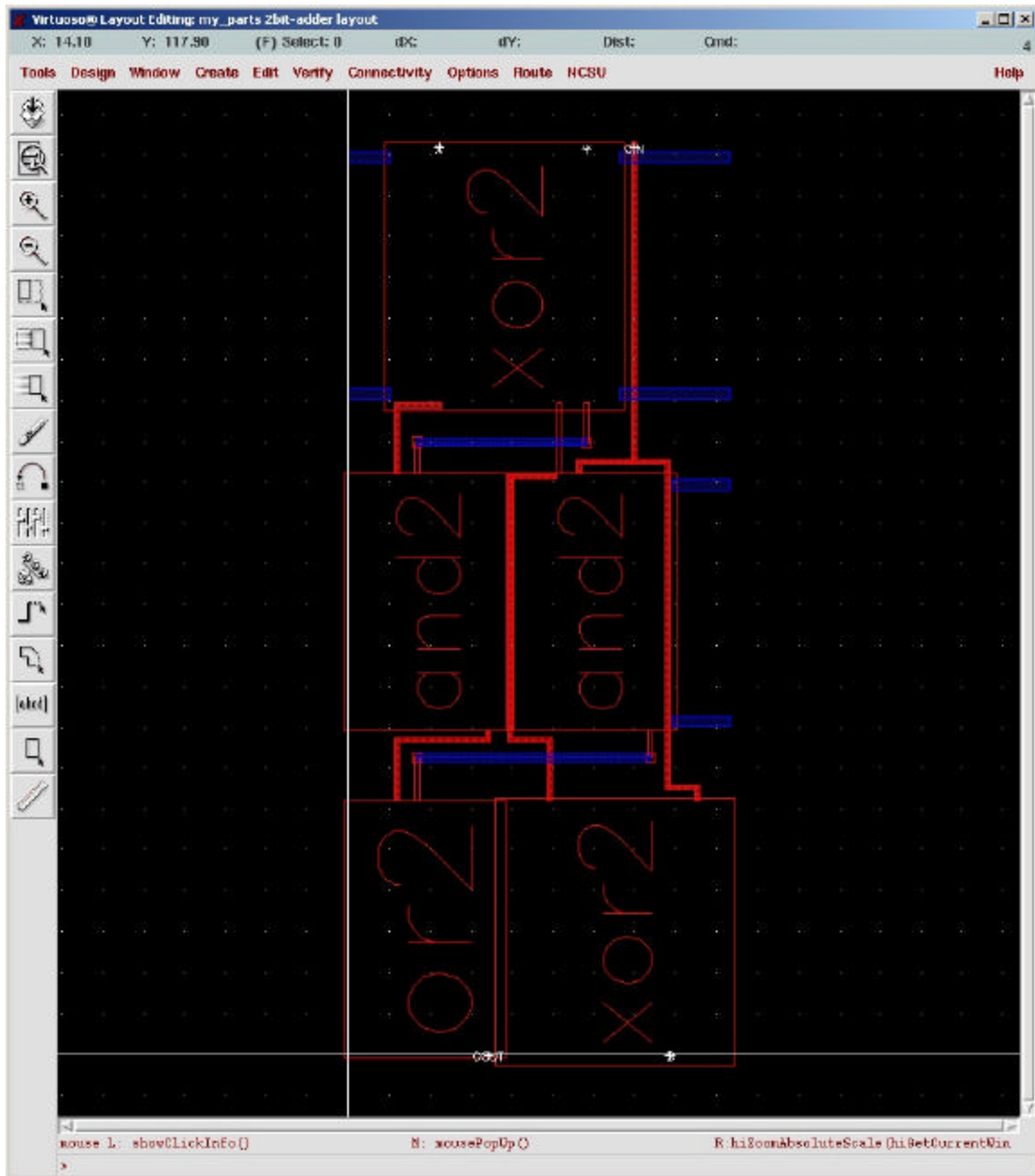


Figure 7b. Adder layout.

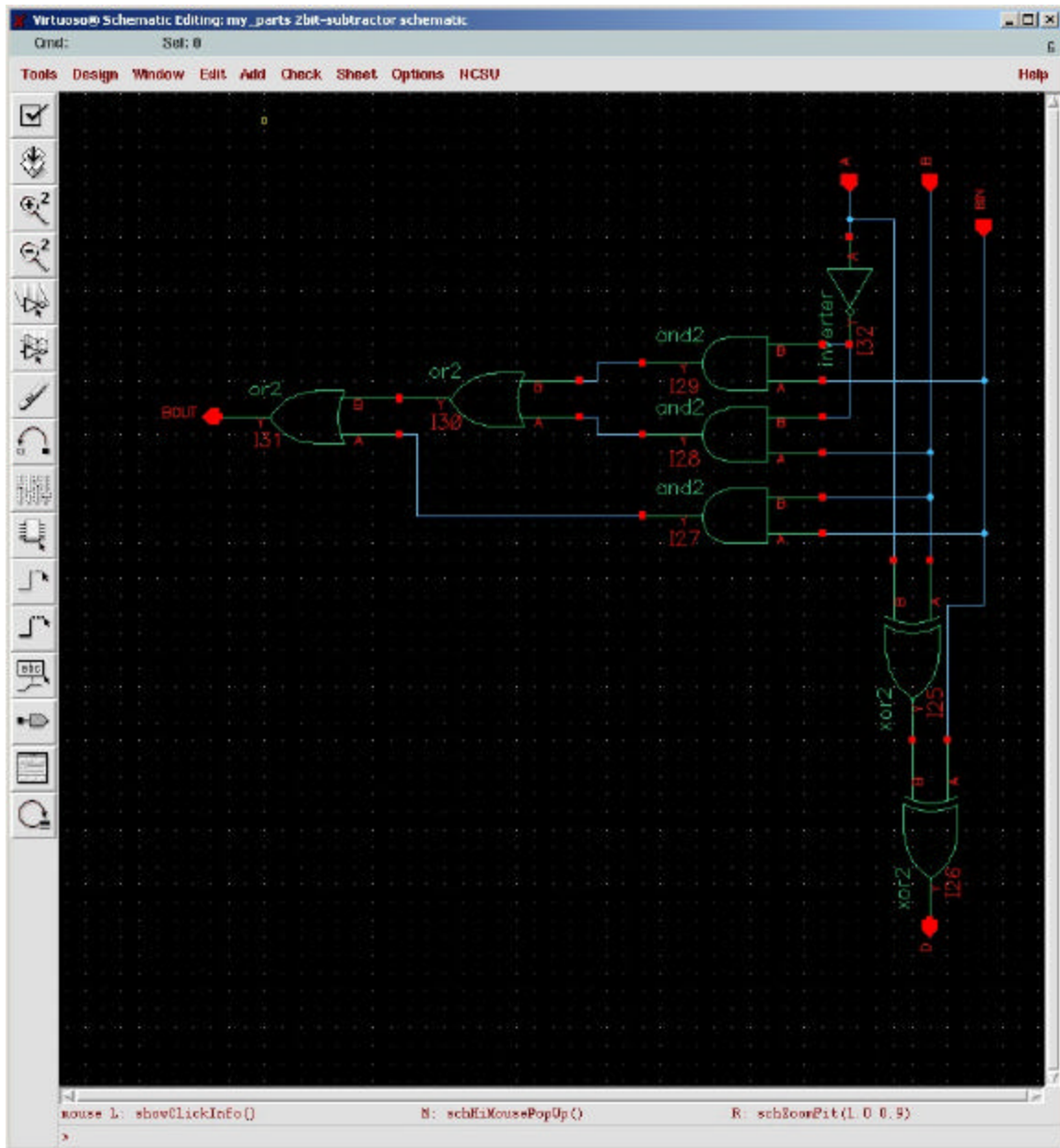


Figure 8a. Subtractor schematic.

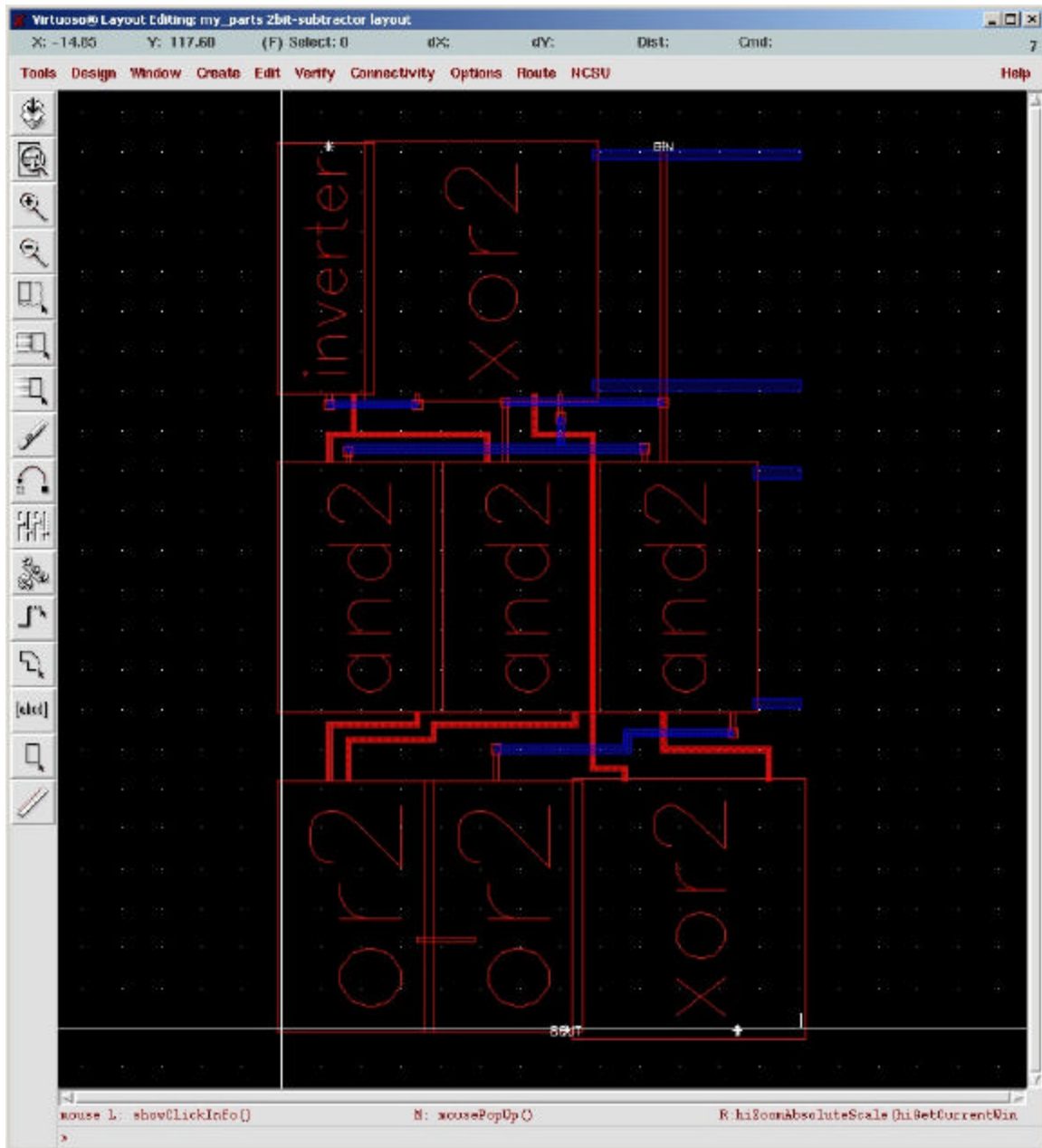


Figure 8b. Subtractor layout.



Figure 9a. ALU schematic.


```

X2 = 1'b1;          Y3 = 1'b0;
X3 = 1'b1;          #25 $finish;
Y0 = 1'b1;          end
Y1 = 1'b1;
Y2 = 1'b0;

```

Table 1. Verilog XL stimulus file.

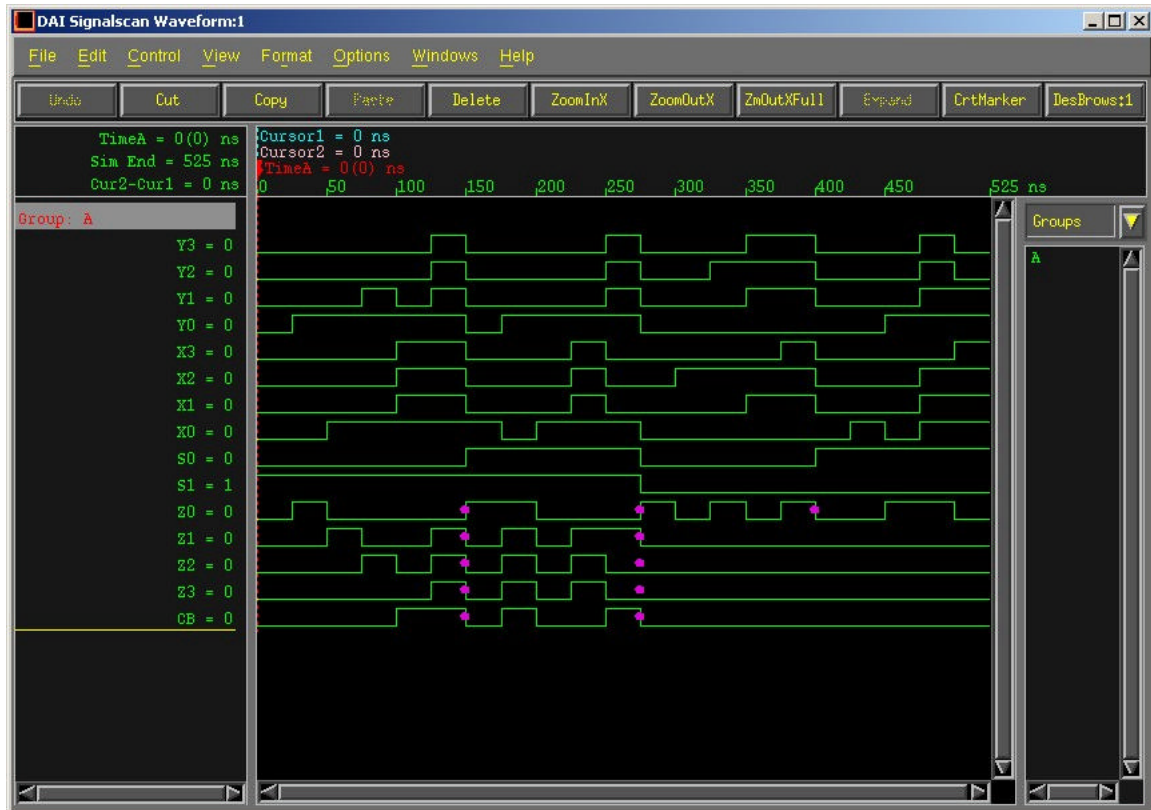


Figure 9b. ALU simulation.

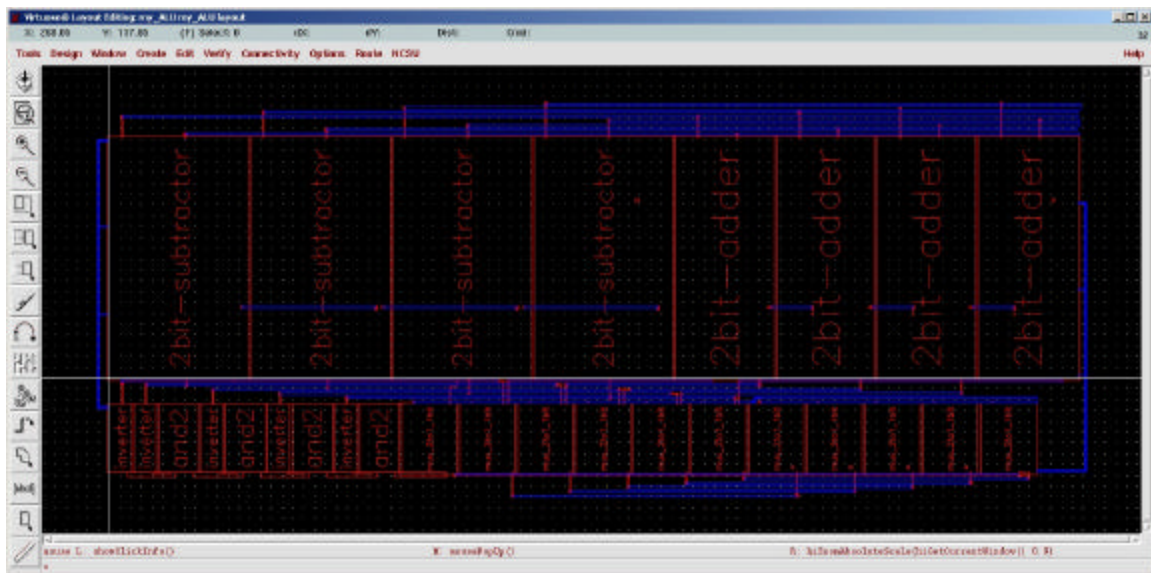


Figure 9c. ALU layout - simple.

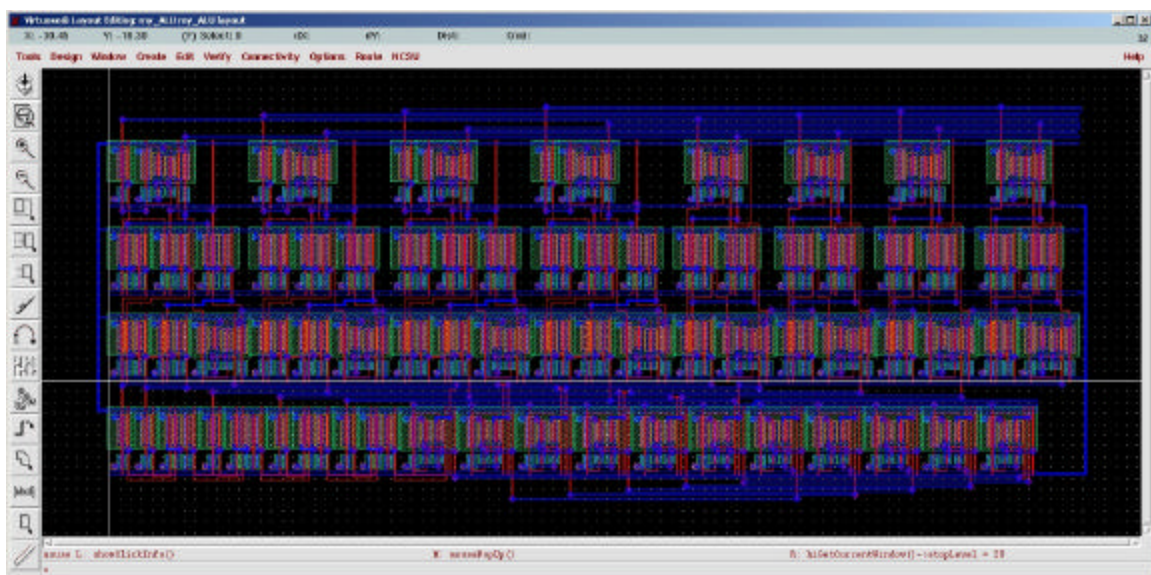


Figure 9d. ALU layout – detailed (flattened).

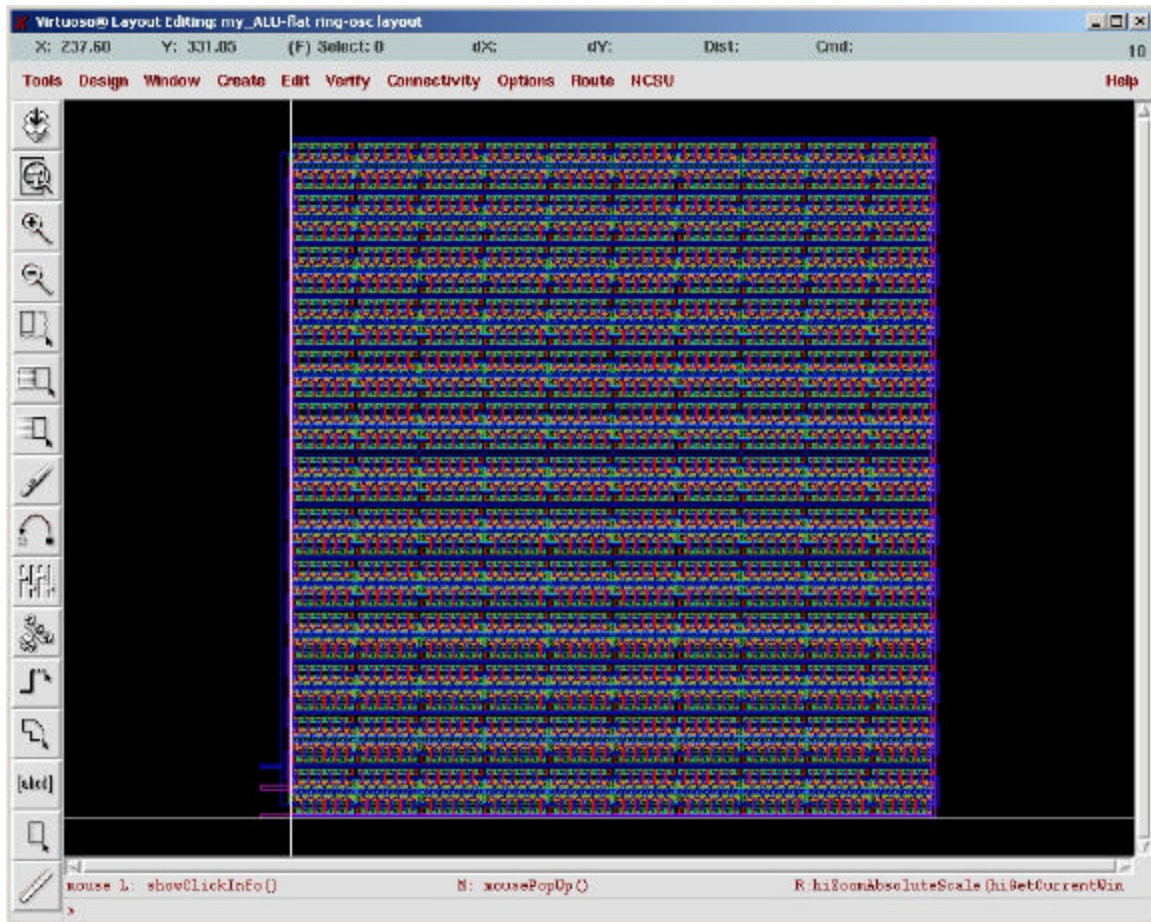


Figure 10a. Ring oscillator layout.

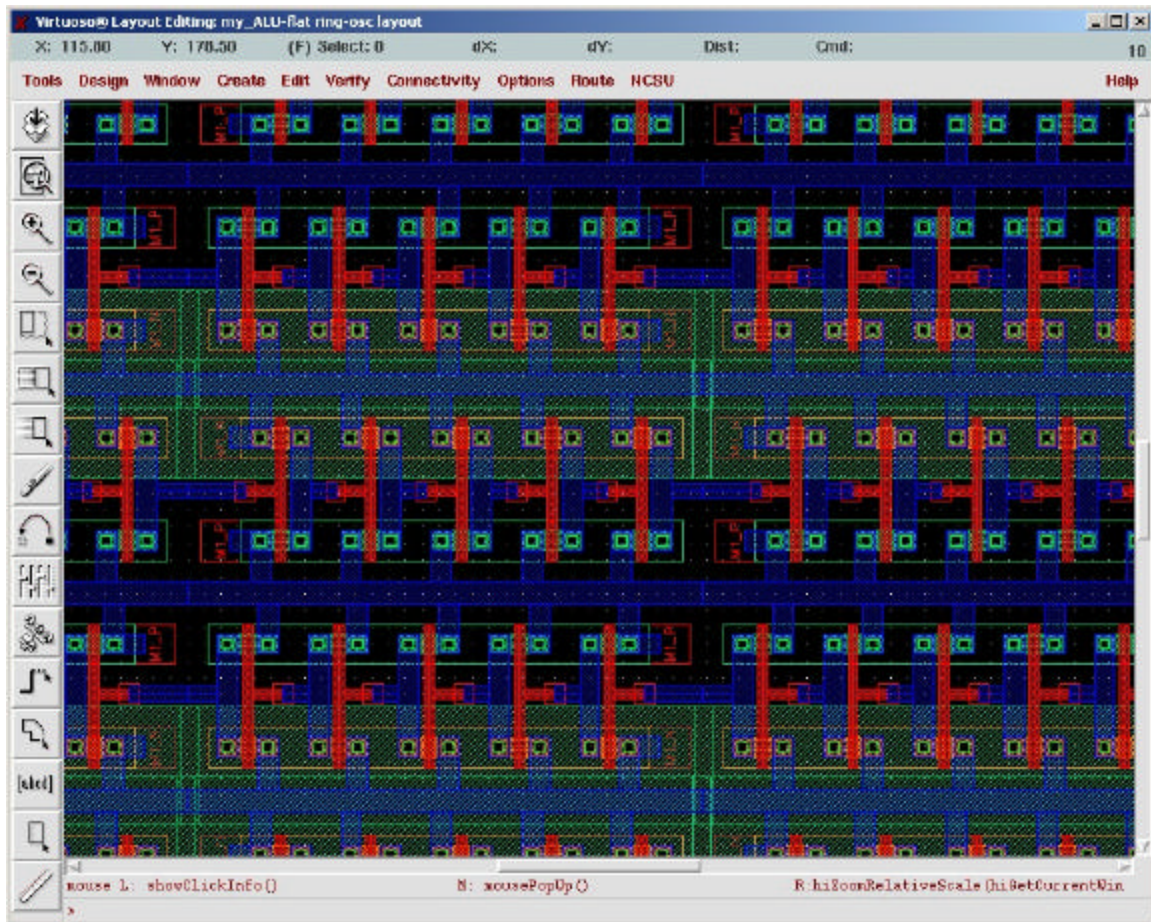


Figure 10a. Ring oscillator layout detail.