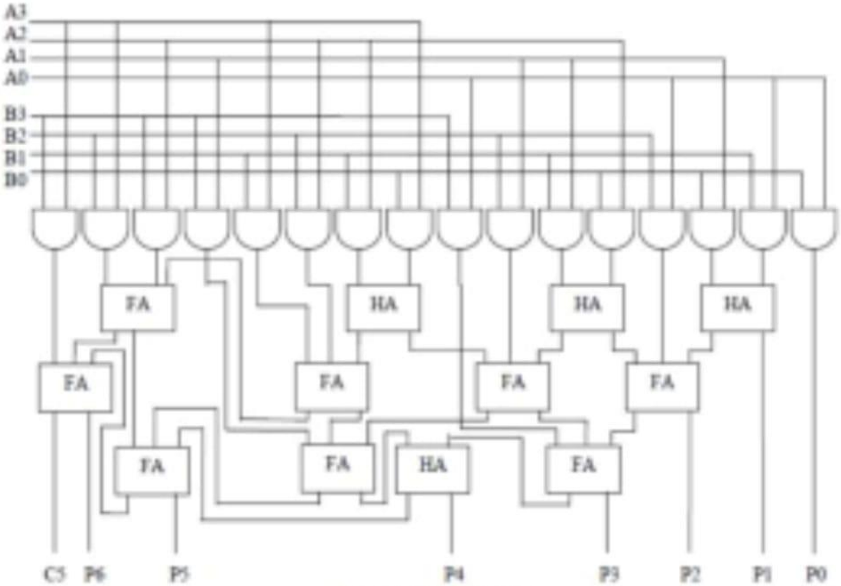
VLSI 4x4 Multiplier

2021112010

Vedansh Agrawal

Project Details:

This project involves the making and testing of a 4x4 multiplier. It will be implemented using NGSPICE for the post layout results, Magic for the pre layout results, and Verilog. We will calculate the propagation delays for every different path and report the longest delay of all of them and find the leakage currents for all kinds of inputs. We implement the following model,

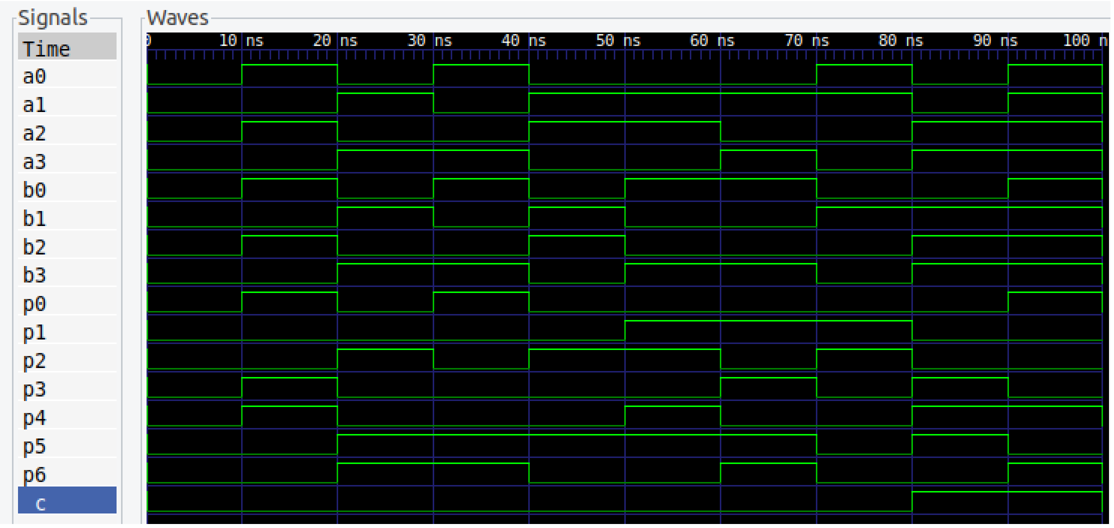


Binary 4x4 Array Multiplier

1. Verilog Coding:

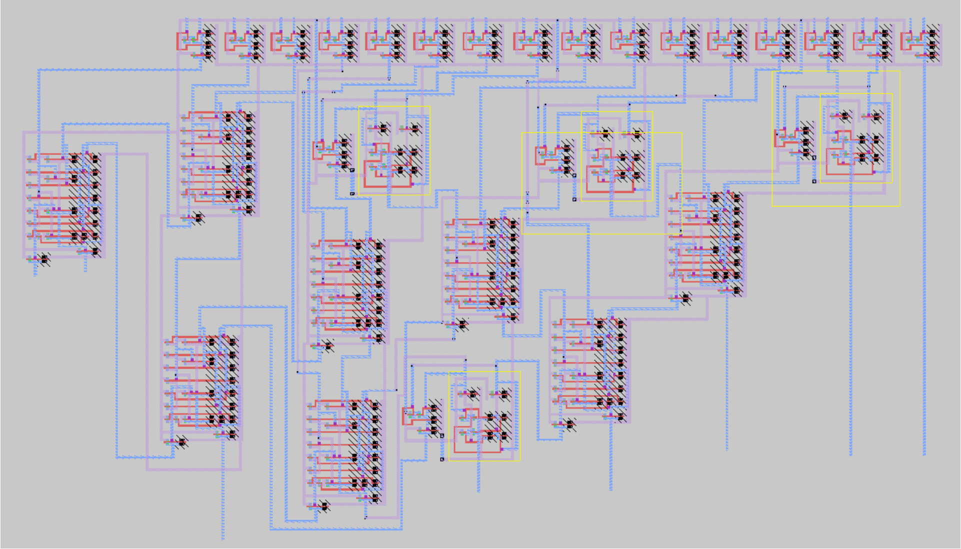
We write a normal Verilog code for the combinational circuit above by using behavioral model of coding which uses basic functions such as and() or or().

We write the testbench for the above code which inputs some sample inputs and plots the outputs. We can verify the results from the following plot of inputs and outputs,



2. Magic Layout:

Now we implement the 4x4 multiplier using magic and we obtain the following model,

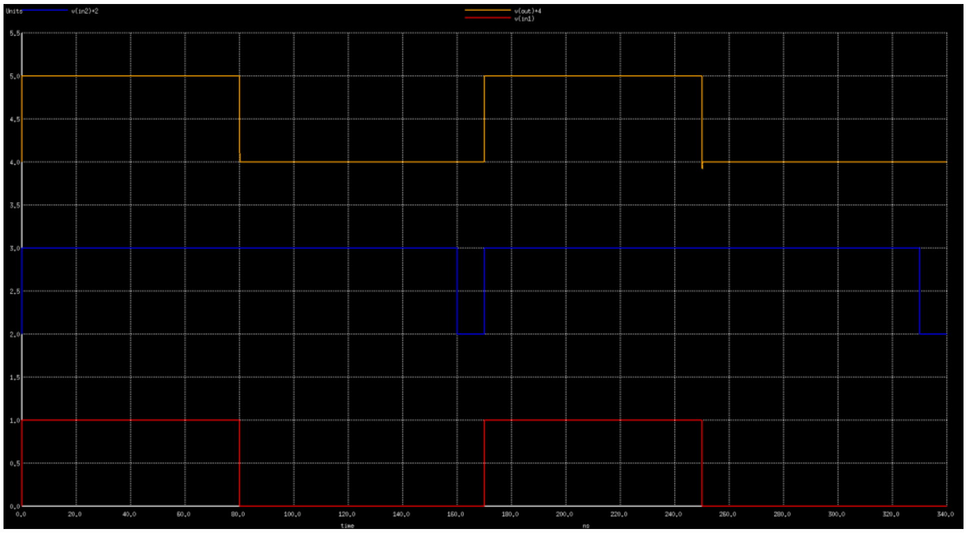


We extract the netlist of the above model and run ngspice for it so we can find the prelayout propagation delays.

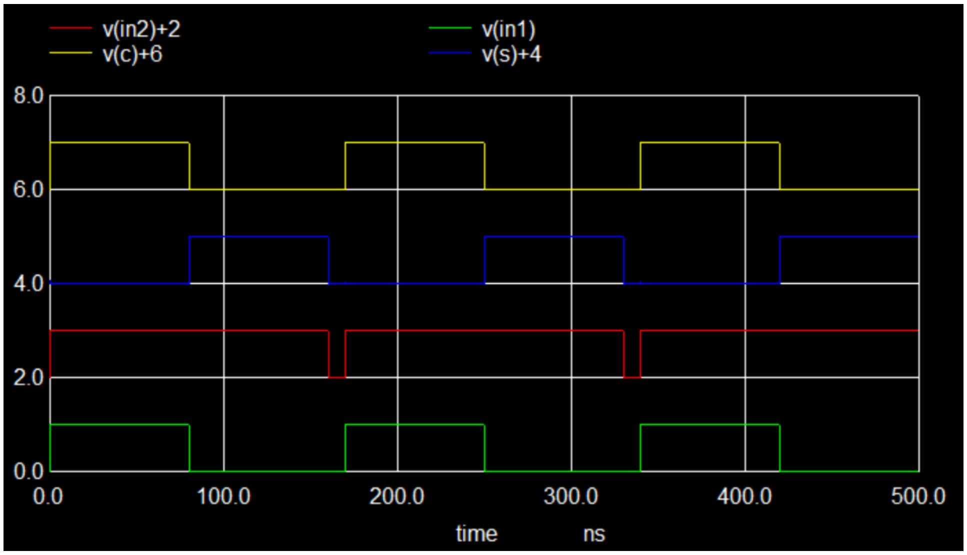
3. NGSPICE Layout:

We write the code for the above layout and obtain the plots to verify their working. Following are plots for the individual components and the final multiplier:

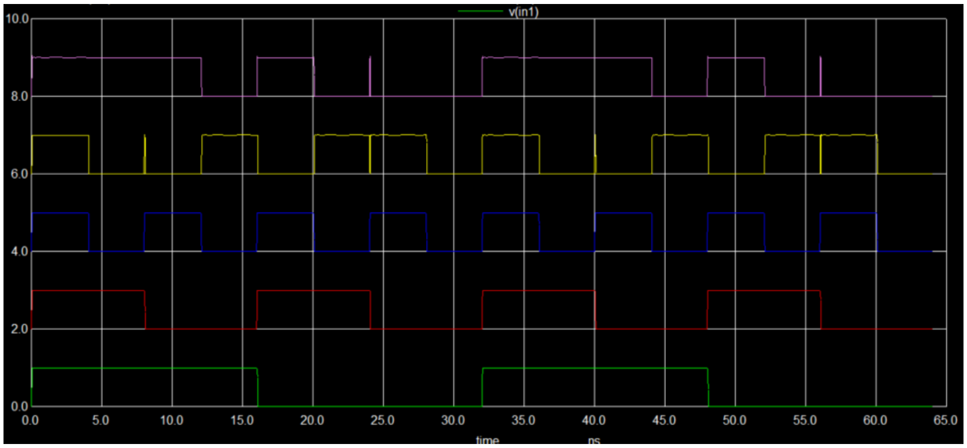
- And Gate:



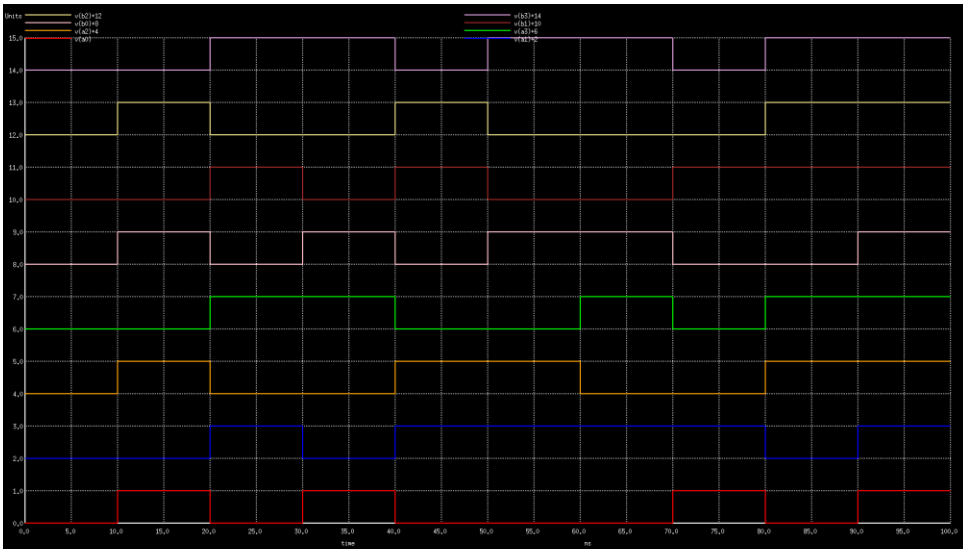
- Half Adder:

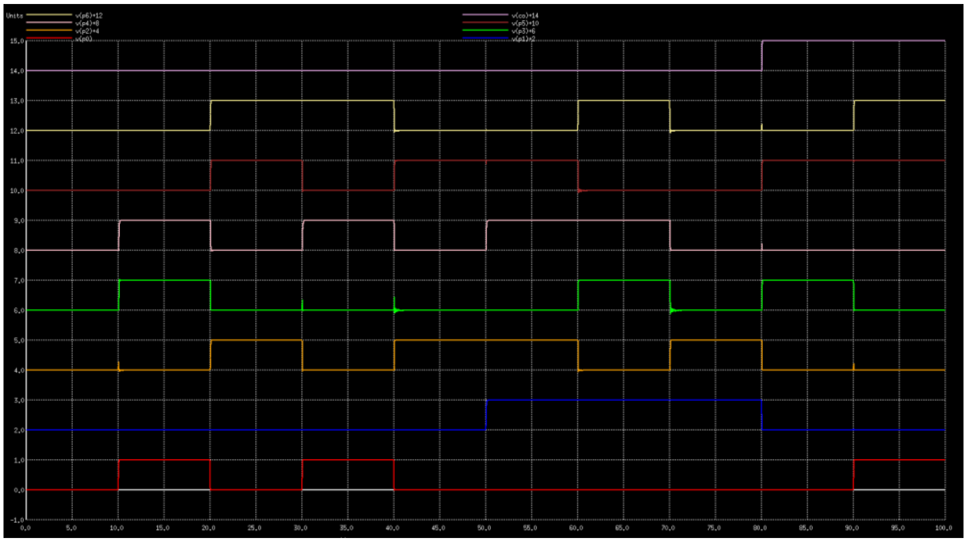


- Full Adder



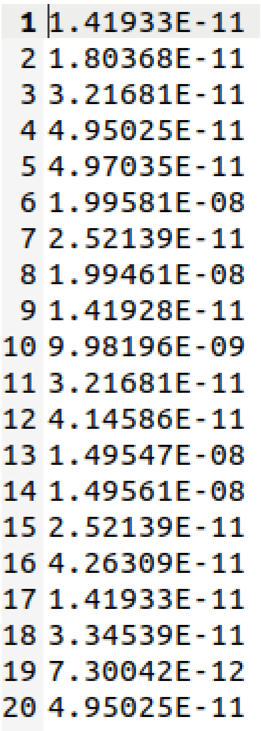
- 4x4 Multiplier:





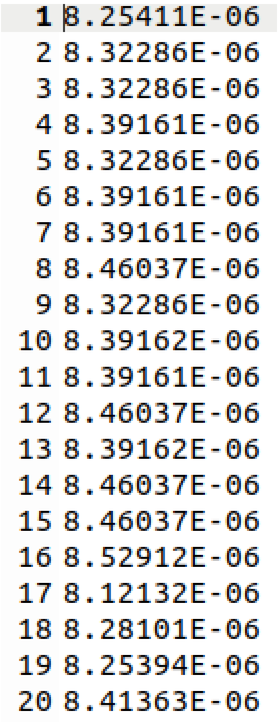
We calculate the propagation delays and output it into a text file which is included in the submission of this project. The worst-case propagation can be found between either inputs a0, a1, b0, b1 and the output p6. We report either of these delays as 2.52139E-08.

Following is some of the delay values found,



```
1.41933E-11
1.80368E-11
3.21681E-11
4.95025E-11
4.97035E-11
1.99581E-08
2.52139E-11
1.99461E-08
1.41928E-11
9.98196E-09
3.21681E-11
4.14586E-11
1.49547E-08
1.49561E-08
2.52139E-11
4.26309E-11
1.41933E-11
3.34539E-11
7.30042E-12
4.95025E-11
```

We also find the leakage currents by running a python script which will run the file 256 times for every different kind of input and output it to another text file. Following is some of the obtained values,



```
 1 8.25411E-06
 2 8.32286E-06
 3 8.32286E-06
 4 8.39161E-06
 5 8.32286E-06
 6 8.39161E-06
 7 8.39161E-06
 8 8.46037E-06
 9 8.32286E-06
10 8.39162E-06
11 8.39161E-06
12 8.46037E-06
13 8.39162E-06
14 8.46037E-06
15 8.46037E-06
16 8.52912E-06
17 8.12132E-06
18 8.28101E-06
19 8.25394E-06
20 8.41363E-06
```

When all the inputs are 0, the leakage power is more, because with low input supply voltage, the subthreshold level reduces to maintain the circuit which increases the leakage power.