EE 4325.001

Vending Machine VLSI Design

Jayanth Domalapalli JXD180023
Maxwell Lim MRL180002
Nahiyan Muzaffar NKM190002

Introduction

In this report, we will discuss the results of our final project. For this project, we decided to create a vending machine. The vending machine allowed for a simple design that was straightforward for us to follow. Additionally, a vending machine would be easily scalable, which made it less difficult to meet the minimum 300 cell requirement. We used Synopsys and Cadence programs in order to implement our project. The goal of this project was to give us a look at the industry software used in VLSI applications, as well as being able to implement some of the things that we learned during class.

Function of Our Design

Our design is to function as a vending machine. This vending machine can take in money as an input and output one of the three objects that the user chooses or returns the users money if they either need change for the item they've purchased or have changed their mind on purchasing the item. The vending machine has three states: idle, ready and dispense. The finite state machine, or FSM, starts in the idle state and during the idle state, if money is inserted, the FSM proceeds to the ready state, if not the FSM stays in the idle state. During the ready state, if the money inserted is greater than or equal to $4, and the user doesn't request for their money to be returned, the FSM proceeds to the dispense state, if the user request for their money to be returned, after returning the money the FSM goes back to the ready state. During the dispense state, the dispense output is set high and if there is no more money left in the vending machine, the FSM goes back to the idle state, if there is money left the FSM goes back to the ready state.

Tradeoffs

The tradeoffs that we made were about the size of the layouts of each cell. They are not minimized to make them all a consistent size as the DFF which required a larger area than the other cells.

IO

Since our FSM needs to handle vending three separate items, it needs to have three separate sets of inputs and outputs. Each set of IO has a one bit input coinInserted which is activated when the user inputs money into the vending machine, a four bit input register for taking in the money, a one bit input returnMoney which the user can activate if they would like their money to be returned, an four bit output register for returning money, and a four bit output register for dispensing items.

Testing

The way that we tested each individual cell was to simulate their outputs using HSPICE and to look at the outputs with waveview and compare them to the truth table of the gate in question. The way that we tested the correctness of the final layout was to take the netlist Verilog file and fun it with the same testbench as the original Verilog code and to compare the outputs to see if they were identical.

Waveforms

As seen below in Figures 1 and 2, the output waveforms of our original Verilog code match with the waveforms from the simulation of the actual layout.

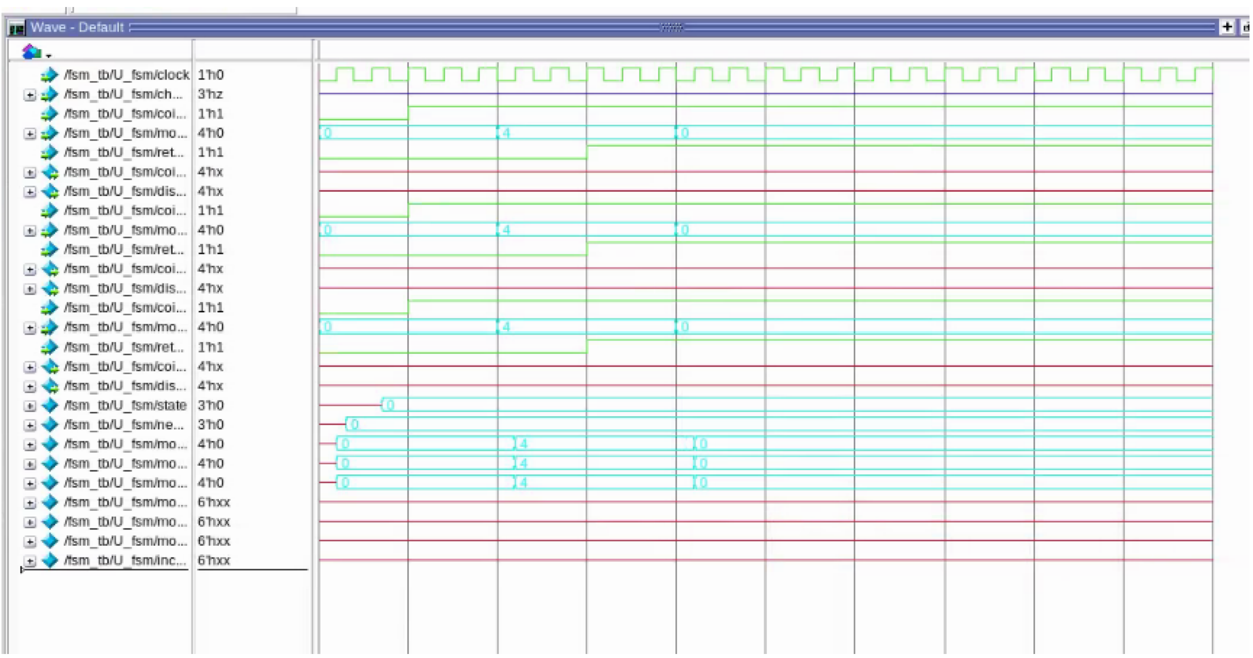

Figure 1. Waveforms produced from the original Verilog code.

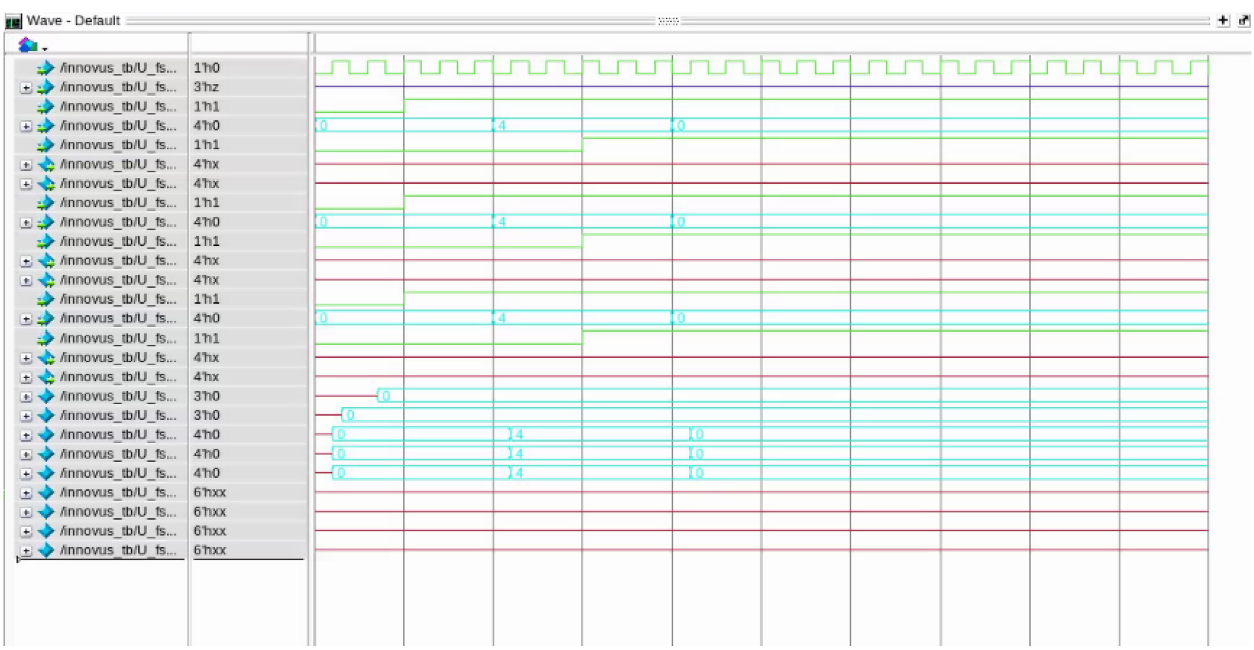

Figure 2. Waveforms produced from the netlist.

Layout

Seen below in Figure 3 is the complete layout of our design, with a height of 156.16 and a width of 136.873.

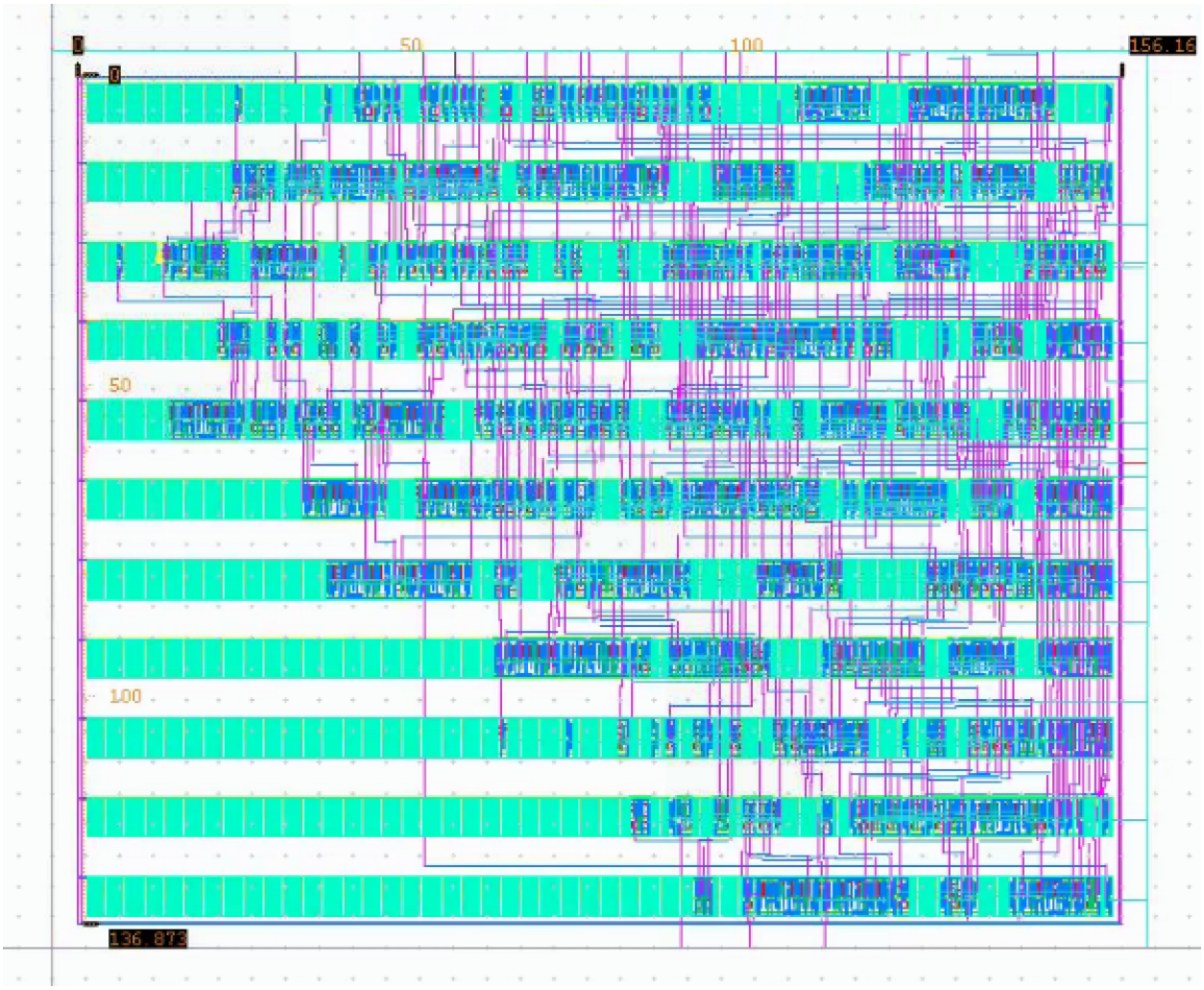

Figure 3. Image of layout in Virtuoso.

DRC and LVS

Below, Figure 4 shows the DRC report of our design passing with no errors and Figure 5 shows the LVS report of our design passing with no errors.

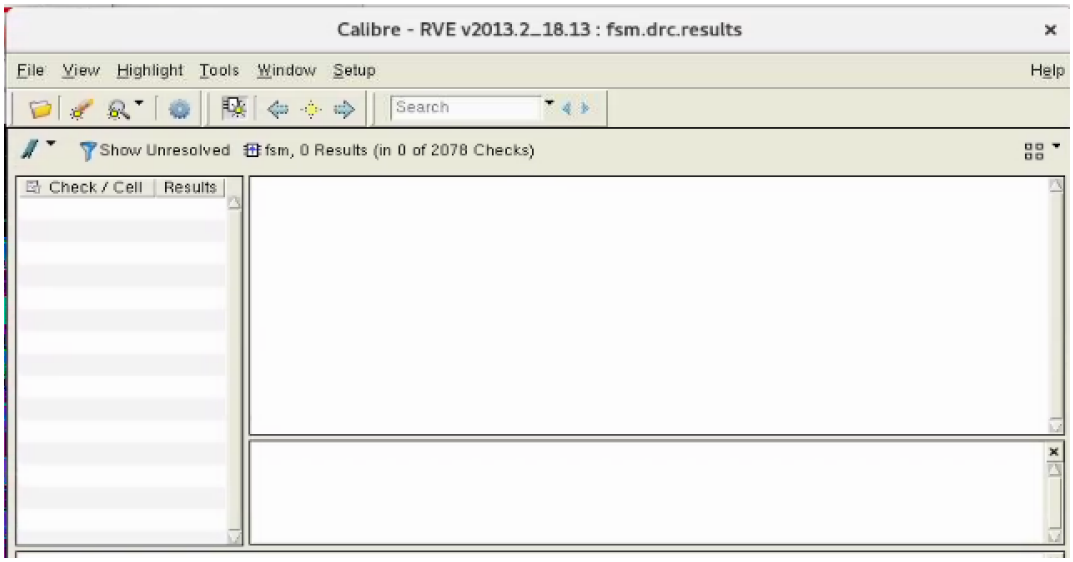

Figure 4. Image from Virtuoso showing the DRC report.

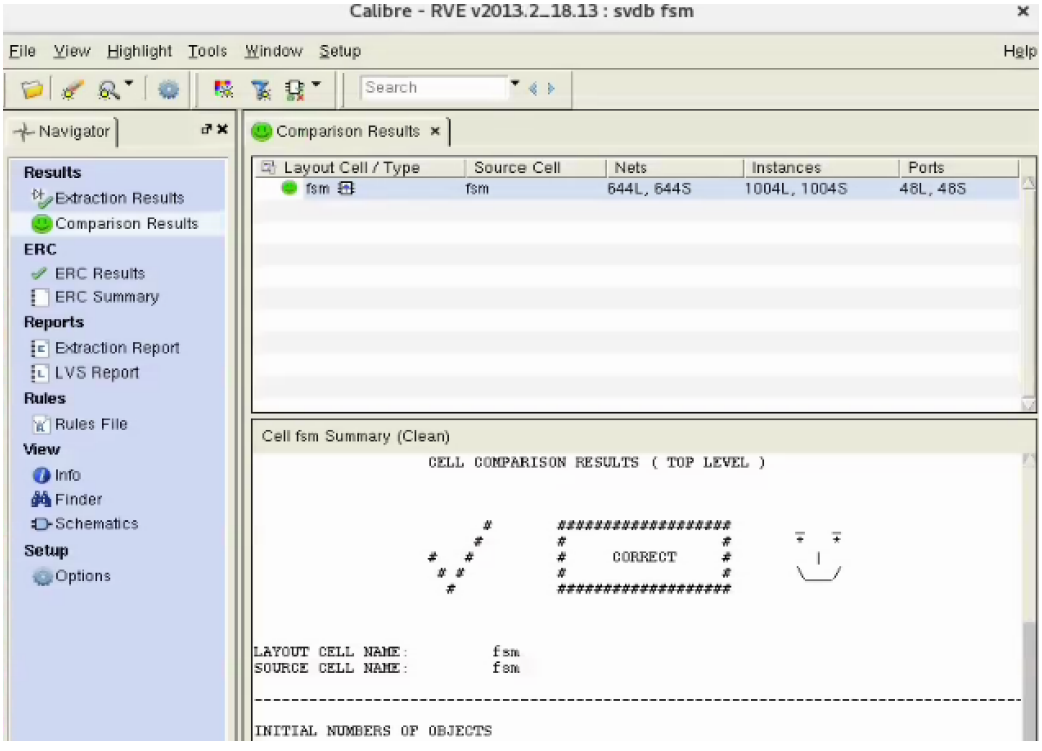

Figure 5. Image from Virtuoso showing the LVS report.

PrimeTime

Utilizing PrimeTime, we were able to find the clock period being 0.79, as well as the total power consumption being 1.208e-4, seen below in Figures 6 and 7, respectively.

```
---------------------------------------------------------------------------
clock clk (rise edge)                          0.08       0.00       0.00
clock network delay (ideal)                               0.00       0.00
state_reg[1]/CLK (DFF)                         0.08       0.00       0.00 r
state_reg[1]/Q (DFF)                  0.02     0.05       0.16       0.16 f
I_11/IN (INV)                                  0.05       0.00       0.16 f
I_11/OUT (INV)                        0.01     0.04       0.05       0.21 r
C10/B (NAND2)                                  0.04       0.00       0.21 r
C10/OUT (NAND2)                       0.00     0.03       0.04       0.25 f
U181/IN (INV)                                  0.03       0.00       0.25 f
U181/OUT (INV)                        0.00     0.02       0.02       0.27 r
U180/B (NAND2)                                 0.02       0.00       0.27 r
U180/OUT (NAND2)                      0.00     0.04       0.03       0.30 f
U92/IN (INV)                                   0.04       0.00       0.30 f
U92/OUT (INV)                         0.01     0.03       0.03       0.33 r
U69/B (NAND2)                                  0.03       0.00       0.33 r
U69/OUT (NAND2)                       0.03     0.12       0.11       0.44 f
U21/C (OAI12)                                  0.12       0.00       0.44 f
U21/OUT (OAI12)                       0.03     0.12       0.10       0.54 r
U19/B (NAND2)                                  0.12       0.00       0.54 r
U19/OUT (NAND2)                       0.00     0.04       0.06       0.60 f
U18/IN (INV)                                   0.04       0.00       0.60 f
U18/OUT (INV)                         0.02     0.06       0.06       0.66 r
U17/B (NAND2)                                  0.06       0.00       0.66 r
U17/OUT (NAND2)                       0.00     0.03       0.04       0.70 f
U16/C (OAI12)                                  0.03       0.00       0.70 f
U16/OUT (OAI12)                       0.00     0.05       0.03       0.73 r
moneyInserted_reg[0]/D (DFF)                   0.05       0.00       0.73 r
data arrival time                                                    0.73

clock clk (rise edge)                          0.00       0.79       0.79
clock network delay (ideal)                               0.00       0.79
clock reconvergence pessimism                             0.00       0.79
moneyInserted_reg[0]/CLK (DFF)                                       0.79 r
library setup time                                       -0.05       0.74
data required time                                                   0.74
---------------------------------------------------------------------------
data required time                                                   0.74
data arrival time                                                   -0.73
---------------------------------------------------------------------------
slack (MET)                                                          0.01
```

Figure 6. Image of report from PrimeTime showing clock values.

```
  Attributes
  ----------
      i  -  Including register clock pin internal power
      u  -  User defined power group

                          Internal  Switching  Leakage    Total
Power Group               Power     Power      Power      Power   (     %)  Attrs
--------------------------------------------------------------------------------
clock_network                0.0000    0.0000    0.0000    0.0000 ( 0.00%)  i
register                  2.063e-05 8.791e-06 8.528e-09 2.943e-05 (24.37%)
combinational             4.980e-05 4.152e-05 7.300e-09 9.133e-05 (75.63%)
sequential                   0.0000    0.0000    0.0000    0.0000 ( 0.00%)
memory                       0.0000    0.0000    0.0000    0.0000 ( 0.00%)
io_pad                       0.0000    0.0000    0.0000    0.0000 ( 0.00%)
black_box                    0.0000    0.0000    0.0000    0.0000 ( 0.00%)

  Net Switching Power  = 5.031e-05   (41.67%)
  Cell Internal Power  = 7.043e-05   (58.32%)
  Cell Leakage Power   = 1.583e-08   ( 0.01%)
                         ---------
Total Power            = 1.208e-04  (100.00%)
```

Figure 7. Image of report from PrimeTime showing power consumption.

Conclusion

We were able to successfully implement our vending machine finite state machine design. Throughout this project, we coded our FSM in Behavioral Verilog, used Synopsys Design Vision to generate a mapped netlist, (gate-level Verilog), used Virtuoso to create a design library, simulated our cells with HSPICE, generated the library using PrimeLib, generated abstract views of our cells using Abstract, used Innovus to place and route our design, and used PrimeTime to run static timing analysis. This project gave us a hands-on experience with industry level software used in VLSI applications and allowed us to take what we learned in class and be able to use it directly for this project.