Project 5 - Finite State Machine Layout

FSM Layout and Design Problems

Arthur Hsueh
21582168
UBC - ELEC 402

Contents

List of Figures

List of Tables

1 Finite State Machine Layout

1.1 Review of FSM Design - Assignment 1 & 2

The finite state machine (FSM) design was the one used in Assignments 1 and 2. The detailed design of the FSM is too long for the purpose of this report, so this section will reiterate the General Description of the design and functionality of the FSM from Assignment 1.

"My finite state machine (FSM) is a simplified application of a credit card payment controller. It accepts payment in the form of VISA, MasterCard or AMEX, and includes basic operations that one would see when paying with a credit card.

The state machine moves through the states of card association selection, confirmation of payment, pin processing and processing the actual payment. It also includes state transitions when confirmation of payment, pin processing and actual payment processing fails. As a more unique feature, I added in a fail counter for the pin processing, allowing the state machine to abort processing if the user enters their pin incorrectly too many times.

The FSM assumes a more modular design, and functions as a controller to the whole system. The unique inputs of the FSM are bits that trigger transitions to new states, and the outputs of the FSM are bits that function as enables to other modules."

For detailed input/outputs of the FSM as well as the testbench design used to simulate the FSM, please refer to my Assignment 1.

Because we are using the _map.v file generated by the synthesis tool of Cadence, the design name now includes an appended _map on each corresponding file.

1.2 Virtuoso Layout

Below is the image of the transistor layout of the FSM. The placement and routing of the design was first performed in the Encouter software, and then imported into Virtuoso.

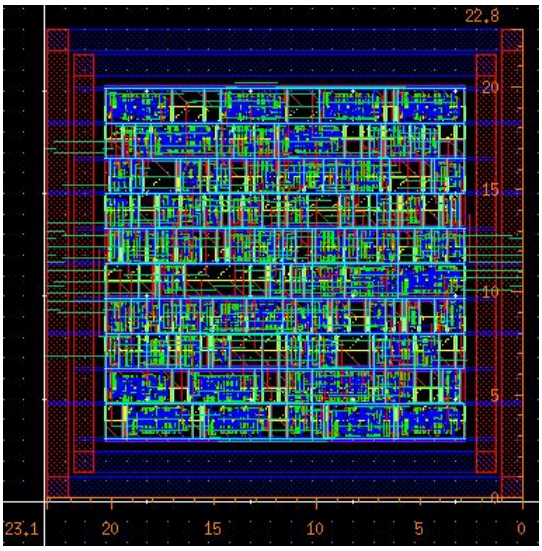

Figure 1: credit_card_payment_FSM_map layout

Rulers have been added to show the dimensions of the layout. The table below shows the dimensions and the resulting area.

Width	Length	Area
21.1 μm	22.8 μm	281.08 μm^2

Table 1: Dimensions of the FSM Circuit

1.3 Verification of Layout

To verify the functionality of the FSM circuit, I generated a new .sdf file from Encounter and used it to simulate the testbench used in Assignment 1. A Modelsim project was created and the testbench file, Assignment 2 _map file and the gpdk045 basic cell file was added. The testbench instantiates the _map module and runs with the .sdf file created in Encounter. The waveform is plotted and the output of the testbench will indicate any errors. Below is the waveform generated from the simulation.

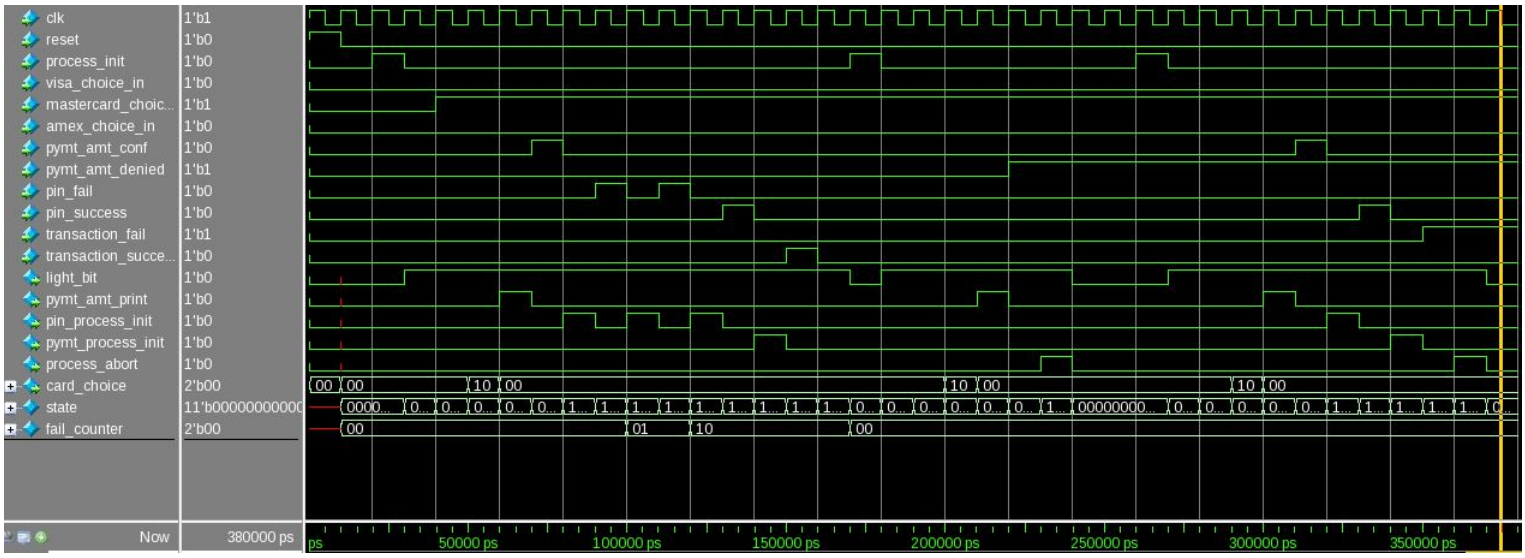

Figure 2: credit_card_payment_FSM_map layout

Compared against the waveform generated in Assignment 1, it is exactly the same. This is supported by the fact that all assertions made in the testbench were evaluated to be true, and no errors were reported.

1.4 Testing verification

1.4.1 Verilog Netlist

The verilog netlist is the new _map file generated from Encounter, and not from the original synthesis. The code is too long to include in the report while maintianing its original formatting integrity, so it is attached with the report. The file is credit_card_payment_fsm_map.v.

1.4.2 .sdf File

The .sdf file is generated by Encounter as well. Again, the code is too long to include in the report while maintianing its original formatting integrity, so it is attached with the report. The file is credit_card_payment_fsm.sdf.

1.4.3 Geometry Check

The geometry check is performed to check against all design rules. Below is both the terminal output from the test, as well as the contents of the outputted .rpt file. The .rpt file is attached with this report, named credit_card_payment_fsm_map.geom.rpt.

```
encounter 1>  *** Starting Verify Geometry (MEM: 657.5) ***

  VERIFY GEOMETRY ..... Starting Verification
  VERIFY GEOMETRY ..... Initializing
  VERIFY GEOMETRY ..... Deleting Existing Violations
  VERIFY GEOMETRY ..... Creating Sub-Areas
                  ..... bin size: 960
  VERIFY GEOMETRY ..... SubArea : 1 of 1
  VERIFY GEOMETRY ..... Cells          :  0 Viols.
  VERIFY GEOMETRY ..... SameNet        :  0 Viols.
  VERIFY GEOMETRY ..... Wiring         :  0 Viols.
  VERIFY GEOMETRY ..... Antenna        :  0 Viols.
  VERIFY GEOMETRY ..... Sub-Area : 1 complete 0 Viols. 0 Wrngs.
VG: elapsed time: 0.00
Begin Summary ...
  Cells       : 0
  SameNet     : 0
  Wiring      : 0
  Antenna     : 0
  Short       : 0
  Overlap     : 0
End Summary

  Verification Complete : 0 Viols.  0 Wrngs.

**********End: VERIFY GEOMETRY**********
 *** verify geometry (CPU: 0:00:00.2  MEM: 171.3M)
```

Figure 3: Terminal output for the Geometry Check

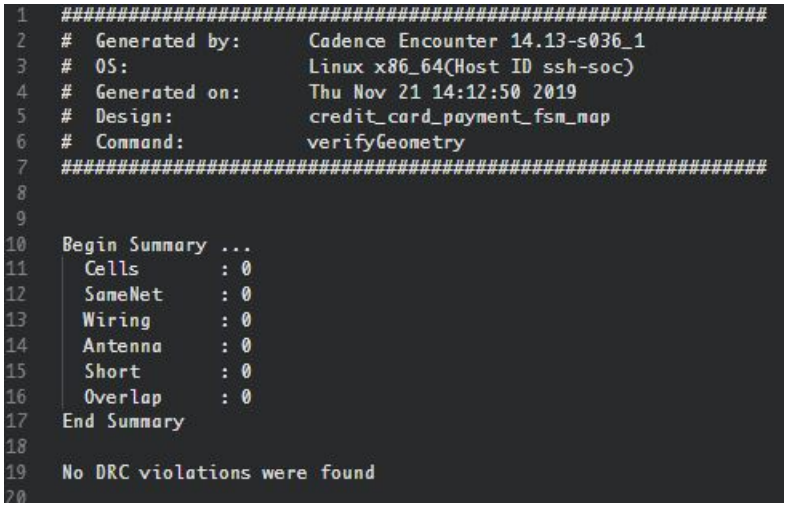

Figure 4: .rpt Contents for the Geometry Check

1.4.4 DRC Check

The DRC check is performed to check against all design rules like the geometry check. Below is both the terminal output from the test, as well as the contents of the outputted .rpt file. The .rpt file is attached with this report, named credit_card_payment_fsm_map.drc.rpt.

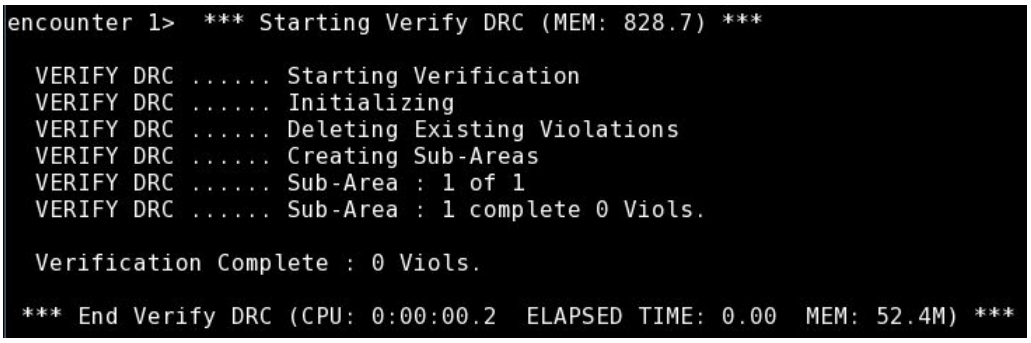

Figure 5: Terminal output for the DRC Check

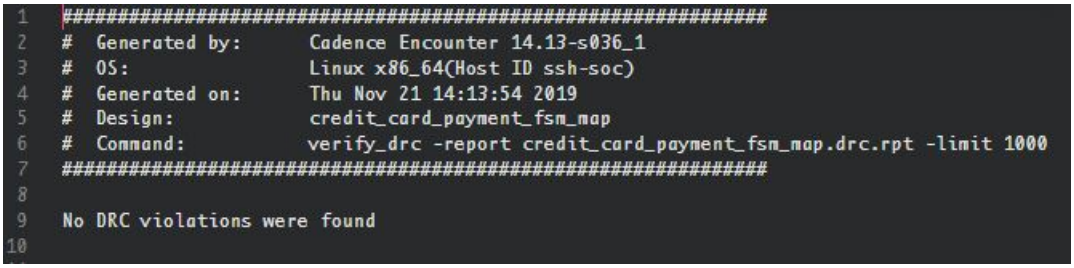

Figure 6: .rpt Contents for the DRC Check

1.4.5 Connectivity Check

The connectivity check is performed to check for any missing connections. Below is both the terminal output from the test, as well as the contents of the outputted .rpt file. The .rpt file is attached with this report, named credit_card_payment_fsm_map.conn.rpt.

```
encounter 1> VERIFY_CONNECTIVITY use new engine.

******** Start: VERIFY CONNECTIVITY ********
Start Time: Thu Nov 21 14:14:33 2019

Design Name: credit_card_payment_fsm_map
Database Units: 1000
Design Boundary: (0.0000, 0.0000) (23.4000, 23.1400)
Error Limit = 1000; Warning Limit = 50
Check all nets

Begin Summary
  Found no problems or warnings.
End Summary

End Time: Thu Nov 21 14:14:33 2019
Time Elapsed: 0:00:00.0

******** End: VERIFY CONNECTIVITY ********
  Verification Complete : 0 Viols.  0 Wrngs.
  (CPU Time: 0:00:00.0  MEM: 0.000M)
```

Figure 7: Terminal output for the Connectivity Check

```
1  ############################################################
2  #  Generated by:      Cadence Encounter 14.13-s036_1
3  #  OS:                Linux x86_64(Host ID ssh-soc)
4  #  Generated on:      Thu Nov 21 14:14:33 2019
5  #  Design:            credit_card_payment_fsm_map
6  #  Command:           verifyConnectivity -type all -error 1000 -warning 50
7  ############################################################
8  Verify Connectivity Report is created on Thu Nov 21 14:14:33 2019
9
10
11
12
13 Begin Summary
14     Found no problems or warnings.
15 End Summary
16
```

Figure 8: .rpt Contents for the Connectivity Check

2 Domino Logic Problem

2.1 Determining the logic function of OUT

The circuit shown in the logic applies the properties of Domino Logic with the PMOS and NMOS with input 'phi' begint the input clock. OUT is not defined in the circuit, but it is taken as the output of the output inverter.

The logic of the circuit can be derived by looking at the pull down network NMOS with inputs A, B, C, and D. The parallel C and D NMOS represents an OR, and the series A, B and C and D, NMOS repesents AND. The important note is that the pull down NMOS network represents the *dual* of the origin logic, and the application of the inverter ensures that the ouptut corresponds to the original logic. So, the resulting logic fucntion is

$$OUT = AB(C + D)$$

2.2 Voltage Reduction under worst case conditions

The first step is to determine the input combination with the worst charge sharing. This combination is

$$ABCD = 1100$$

The next step is the calculate the capacitances as the respective nodes. The image below labels the nodes XYZ that will be used for calculation

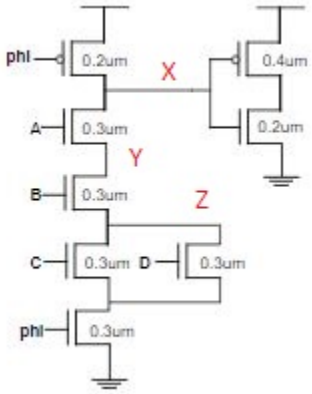

Figure 9: Problem 2 circuit with labelled nodes

For calculations the values C_{eff} and C_g will be used at the declared values of

$$C_{eff} = 1\,\text{fF}\mu\text{m}^{-1} \quad and \quad C_g = 2\,\text{fF}\mu\text{m}^{-1}$$

For C_X

Capacitance at node X is the sum of the effective capacitances of the phi PMOS and A NMOS as well as the gate capacitance at the input of the inverter.

$$C_X = C_{eff}W_{phi} + C_{eff}W_A + C_g(W_p + W_n)$$

$$C_X = 1.7\,\text{fF}$$

For C_Y

Capacitance at node X is the self capacitance of the B NMOS.

$$C_X = C_{eff} W_B$$

$$C_X = 0.3\,\text{fF}$$

For C_Z

Capacitance at node X is the sum of the self capacitances of the C and D NMOS.

$$C_X = C_{eff}(W_C + W_D)$$

$$C_X = 0.6\,\text{fF}$$

For V*

V* is the resulting voltage across all the nodes with the shared charge. Here the capacitances of B, C and D are all charged at once when A and B are input HIGH. So the charge sharing equation will take into account more than two capcitances; the original charge at node X is equal to the charge at all three nodes once the capacitances are charge. As done in class, the initial voltage is taken to be 1.2V.

$$V^*(C_X + C_Y + C_Z) = V_{initial} C_X$$

$$V^* = \frac{V_{initial} C_X}{C_X + C_Y + C_Z}$$

$$V^* = \frac{1.7 * 1.2}{1.7 + 0.3 + 0.6}$$

$$V^* = 0.7846\,\text{V}$$

This value is not reasonable, becuase node Z must have a voltage of at most $V_{DD} - 2V_T = 0.4V$. Thus to calculate the accurate voltage at node X we must find the remaining charge at node X after the charge as been shared.

$$Q_{Xremaining} = Q_{Xoriginal} - (Q_Y + Q_Z)$$

$$C_X V_{Xfinal} = C_X V_{Xoriginal} - (C_Y(V_{DD} - V_T) + C_X(V_{DD} - 2V_T))$$

$$1.7 * V_{Xfinal} = 1.7 * 1.2 - (0.3 * 0.8 + 0.6 * 0.4)$$

$$V_{Xfinal} = 0.9176\,\text{V}$$

Thus the reduction in voltage at the input of the inverter under the worst case charge sharing condition is

$$V_{reduction} = V_{DD} - V_{Xfinal}$$

$$V_{reduction} = 0.2823\,\text{V}$$

3 Transmission Gate Problem

Below is a notated image of Figure 4 with nodes labelled to assist with value identification.

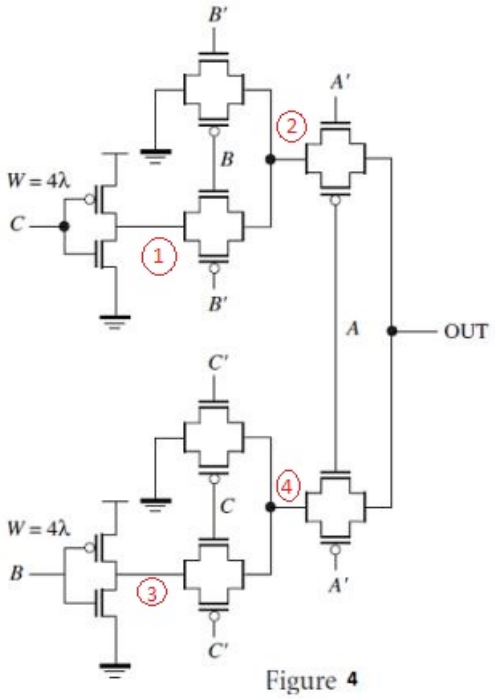

Figure 10: Figure 4 Circuit with annotated nodes

3.1 Capacitance Calculations

There are 5 nodes in the circuit 1,2,3,4 and OUT. We note that by the nature of the circuit the capacitances at node 3 and 4 are equal to 1 and 2 repsectively.

For C_1

The capacitance contributions come from the self capacitance of the PMOS and NMOS of the inverter, the self capacitance of the transmission gate and the gate capcitance of the transmission gate.

$$C_1 = C_{eff-inv-out} + C_{g-TG} + C_{eff-TG}$$
$$C_1 = C_{eff}(W + 2W) + C_g W + 2C_{eff} W$$
$$C_1 = C_3 = 5C_{eff} W + C_g W$$

For C_1

The capcitance contributions come from 3 transmission gates, both the self and gate capacitance.

$$C_2 = 3(C_{g-TG} + C_{eff-TG})$$
$$C_2 = 3(C_g W + 2C_{eff} W)$$

$$C_2 = C_4 = 6C_{eff}W + 3C_g$$

For C_{out}

We are assuming that we are not driving any circuit. Thus the capacitance contributions come from 2 transmission gates.

$$C_{OUT} = 2(C_{g-TG} + C_{eff-TG})$$
$$C_{OUT} = 2(C_gW + 2C_{eff}W)$$
$$C_{OUT} = 4C_{eff}W + 2C_g$$

For delay calculations we can take the effective resistance of each inverter and gate to be the same as the effective resistance of an NMOS transistor.

$$R = R_{inv} = R_{TG} = R_{nmos} * \frac{L}{W}$$

Delay calculations apply Elmore delay methods.

3.2 Minimum Delay

The minimum delay is given by the path from the ground at the input to the most top transmission gate (input 0 of the B-select mux) to OUT. The calculation is as follows.

$$\tau_{min} = C_2 * R + C_{OUT} * 2R$$

$$\tau_{min} = R(6C_{eff}W + 3C_gW) + 2R(4C_{eff}W + 2C_gW)$$
$$\tau_{min} = RW(14C_{eff} + 17C_g)$$

3.3 Maximum Delay

The maximum delay is given by the pathing of Input C to OUT, or Input B to OUT. The calculation is as follows

$$\tau_{max} = \tau_1 + \tau_2 + \tau_{OUT}$$
$$\tau_{max} = C_1 * R + C_2 * 2R + C_{OUT} * 3R$$
$$\tau_{max} = R(5C_{eff}W + C_gW) + 2R(6C_{eff}W + 3C_gW) + 3R(4C_{eff}W + 2C_gW)$$
$$\tau_{max} = RW(29C_{eff} + 13C_g)$$

3.4 Delay Comparison

There is no defined technology in the problem, but if we assume a $0.1\,\mu$m technology and the following values

$$C_{eff} = 1\,\text{fF}\mu\text{m}^{-1} \quad and \quad C_g = 2\,\text{fF}\mu\text{m}^{-1} \quad and \quad R = 12.5\,\text{k}\Omega$$

Solving for both maximum and minimum delay yields

$$\tau_{min} = 60\,\text{ns} \quad and \quad \tau_{max} = 68.75\,\text{ns}$$

4 SRAM Cell Problem

Not Solved for Submission

5 FPGA implementation with LUTs

Not Solved for Submission