IC Design Final Project : Coin Bank

109006723 鍾宛里

I. Table of Contents

II. Circuit Diagram

The circuit design for this homework is fbased on the block diagram image in the pdf spec. Below is the image of the block diagram.

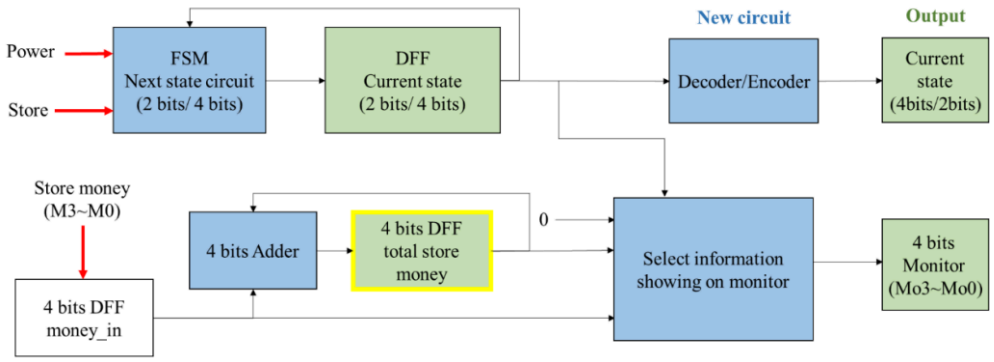
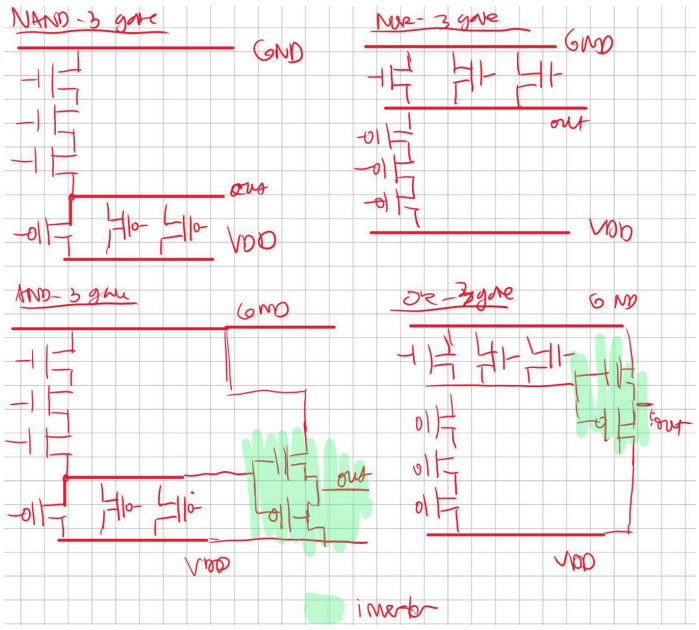
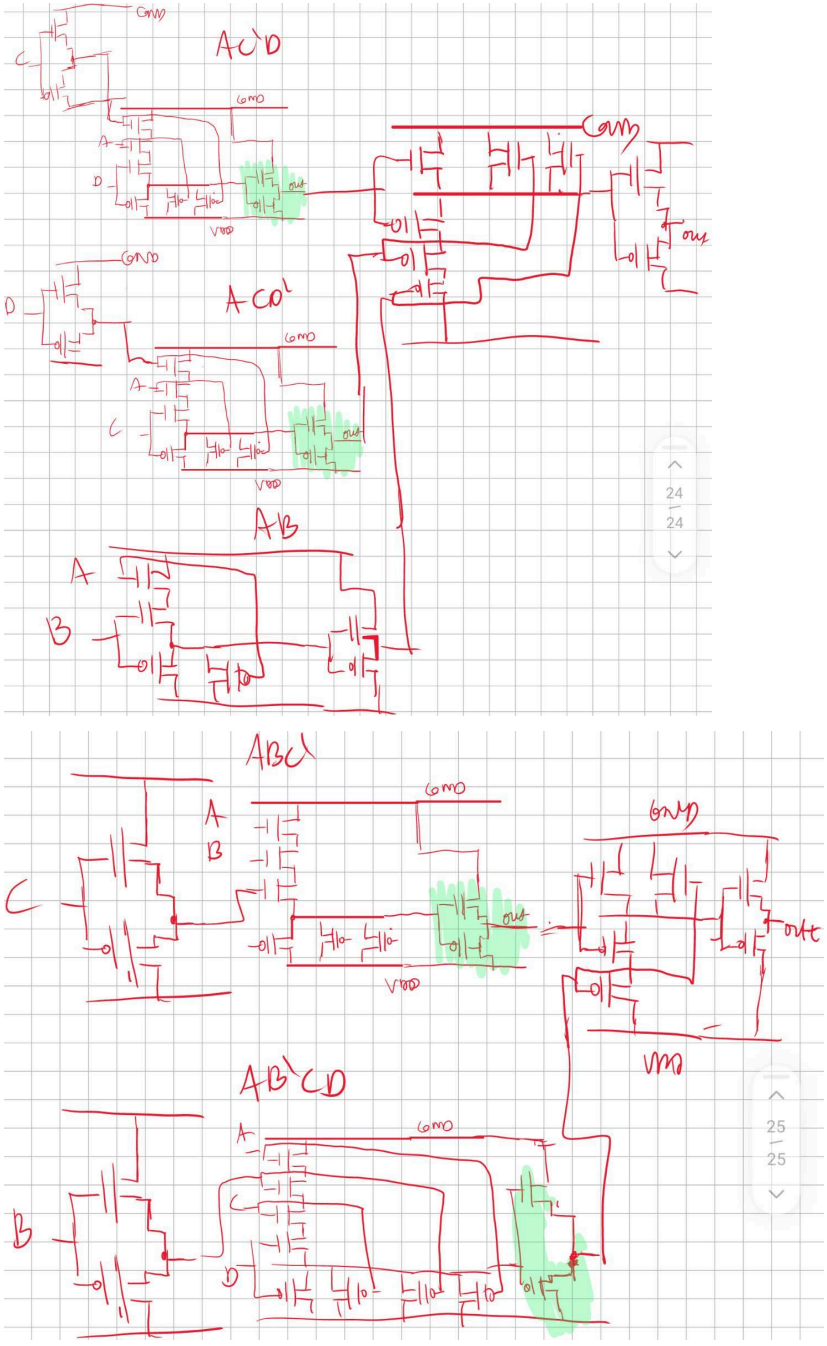


Fig. 2 Sample architecture

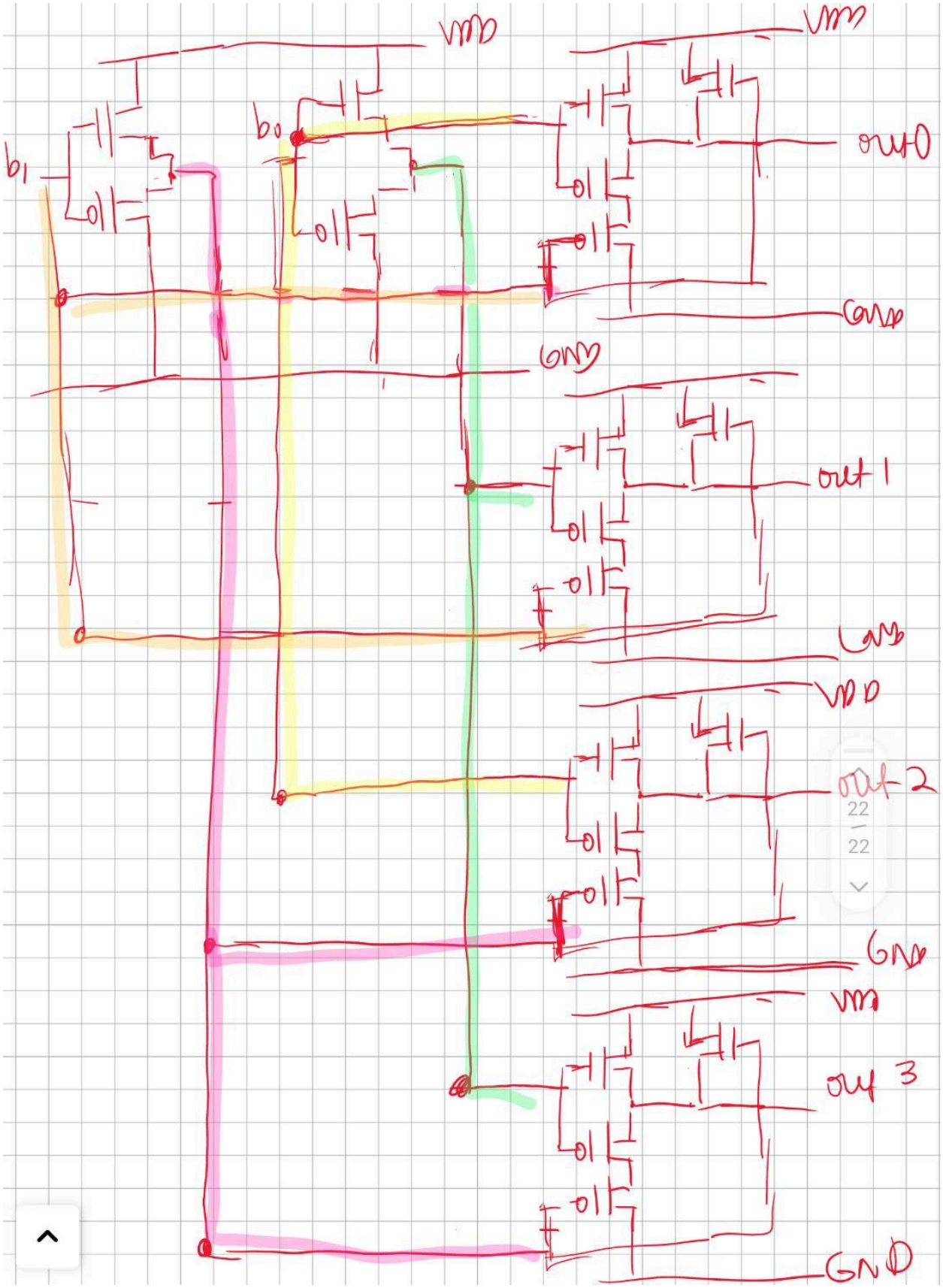
Since I only have 2 states as output of the DFF of the current state. So for the FSM, a karnaugh map was used to solve for the equation to get the next state. The function for next_state_0 is AB + AC'D + ACD' and the function for next_state_1 is ABC' + AB'CD Since during the layouting, instances are used. So, the only circuit diagram needed is AND and OR which is the combination of NAND and Inverter. Below is the circuit diagram for NAND, NOR, AND, NOT. Furthermore, as has been mentioned in the class. These circuits can be extended to 3 and 4 bits easily. For example, 3 bit NAND gates.

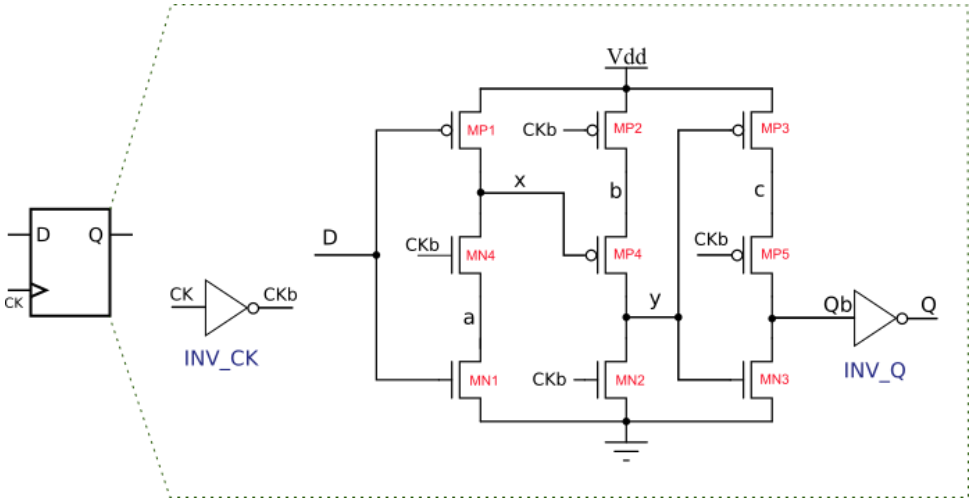


Finite state machine circuit diagram (1st is bit 0, and 2nd is bit 1)



For DFF, I thought that the DFF from homework 3 might have too large of a delay. So, after some time searching, I found a better and faster DFF which has faster switching time than the DFF from homework 3. Below are the circuits for the DFF called TSPC DFF.

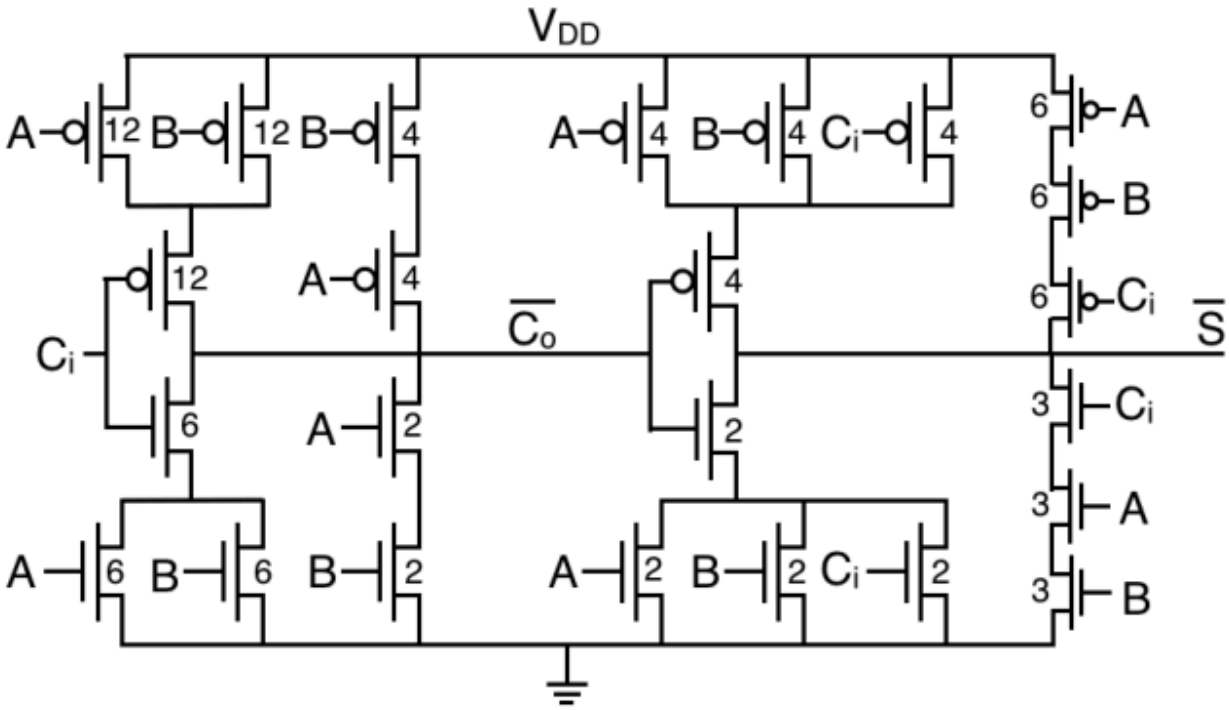

VDD
b1
b0
out0
GND
out1
out2
out3



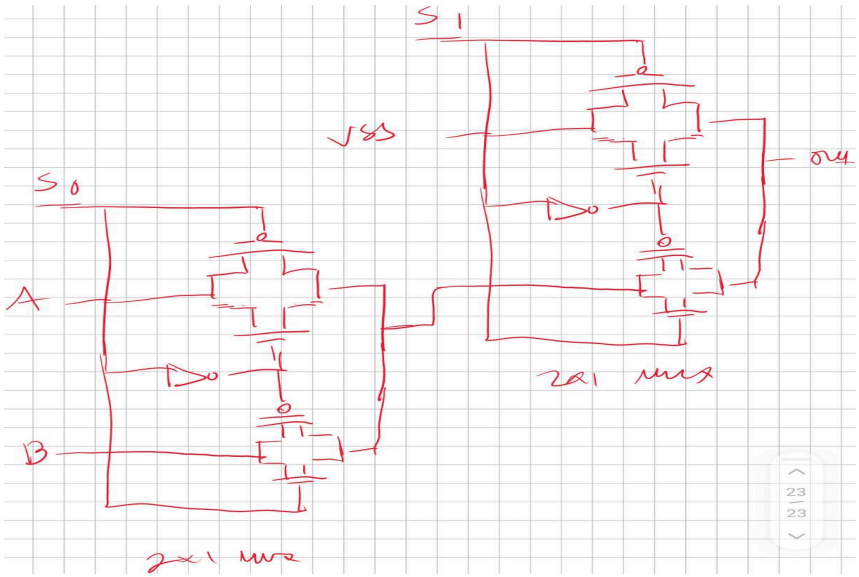
Source :
https://perso.telecom-paristech.fr/mathieu/ICS904_TP_LAYOUT/html/schematic.html
https://www.diva-portal.org/smash/get/diva2:19406/FULLTEXT01.pdf

A decoder was used to output a 4 bit current state (s0, s1, s2, s2). The decoder is made with a nor gate. Similar to the FSM, I only had to make the circuit diagram for NOR gate and INVERTER to make an OR gate. Then combine them together, Into a decoder.

For the adder, since I was also aiming to have a smaller area, so I tried looking for an adder based on circuits not based on majority gate etc. I found one after searching and it's called a mirror adder, and it seems that it is faster since it has less transistors than the adder with a majority gate. Below is the circuit diagram of the mirror adder.



To select information for the monitor, a MUX was used, to be more exact, a 4 bit 4x1 MUX. However, since input of state 00 and state 01 are the same. So, the 4x1 mux could be simplified. Also, to make a 4 bit of every 1 bit gate. 4 of them are used and each of them output corresponding bits from bit0 to bit3. MUX 2x1 circuit diagram and MUX 4x1 circuit diagram. The 2x1 mux is made from the transmission gate.

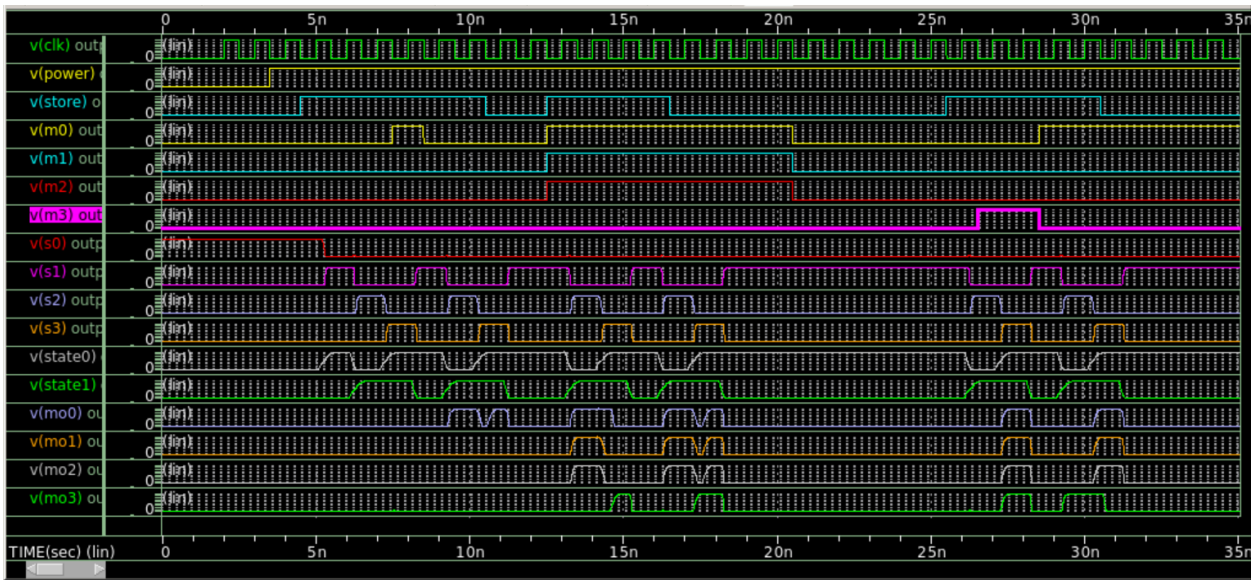


III. Pre-sim & Post-sim Waveform

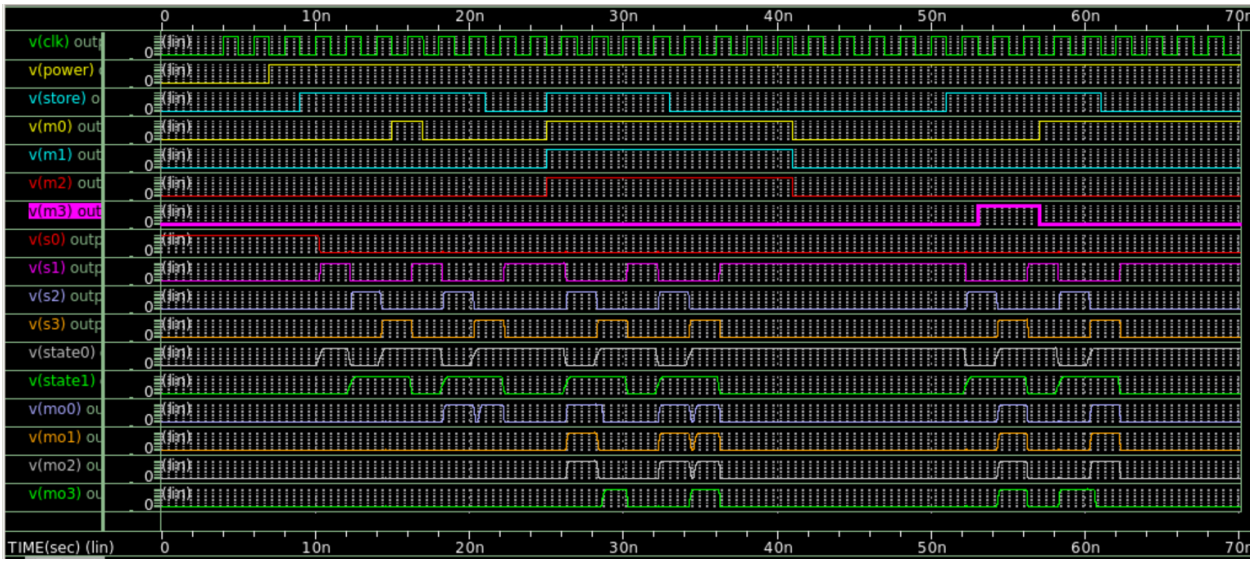
This is the pre-sim waveform screenshot with the testbench that has been made. Tested it up to 35 ns.

Pre-sim :

The first image is when the clock period is 1ns

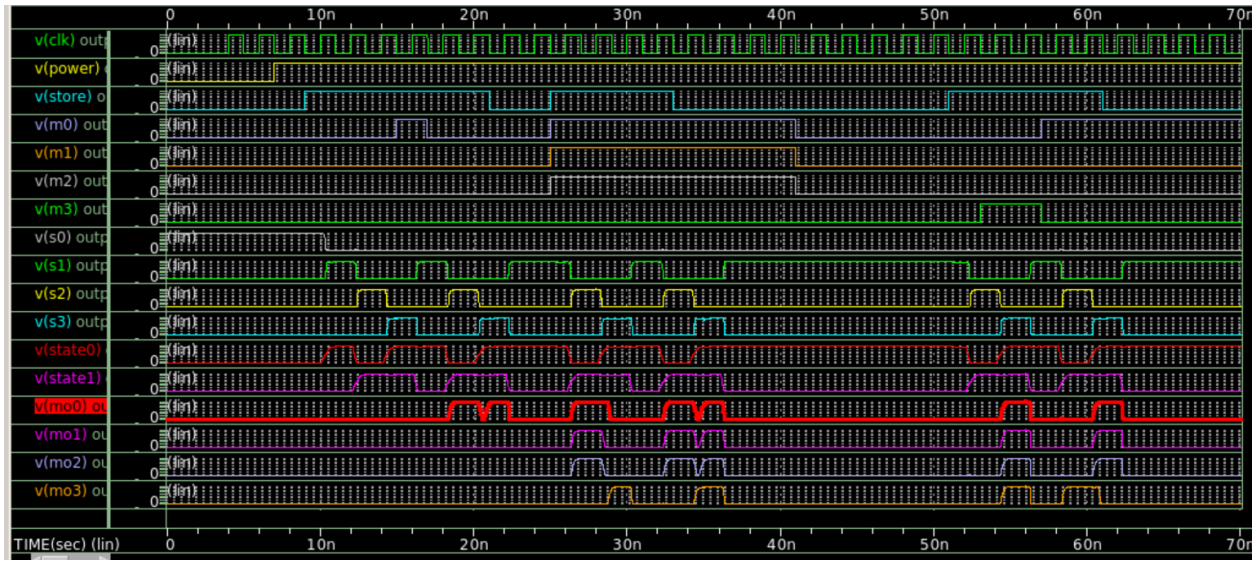


The second image is when the clock period is 2ns



Lowering down the clock period still outputs the correct waveform so i changed my clock period to 1ns

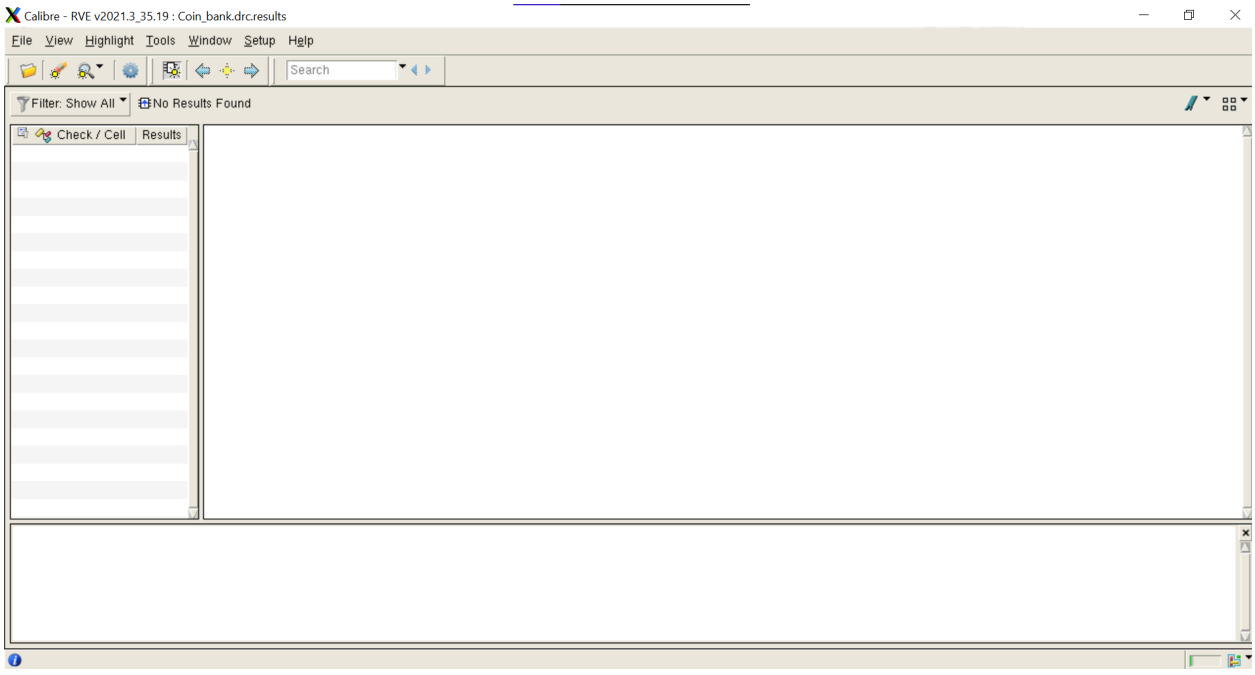
Post-sim :



Clock period : 2ns

IV. DRC Summary Report

I did a bottom up approach for the DRC. every instance starting from the bottom level was made first then checked by the DRC. This is a screenshot of passing the DRC of the top circuit.

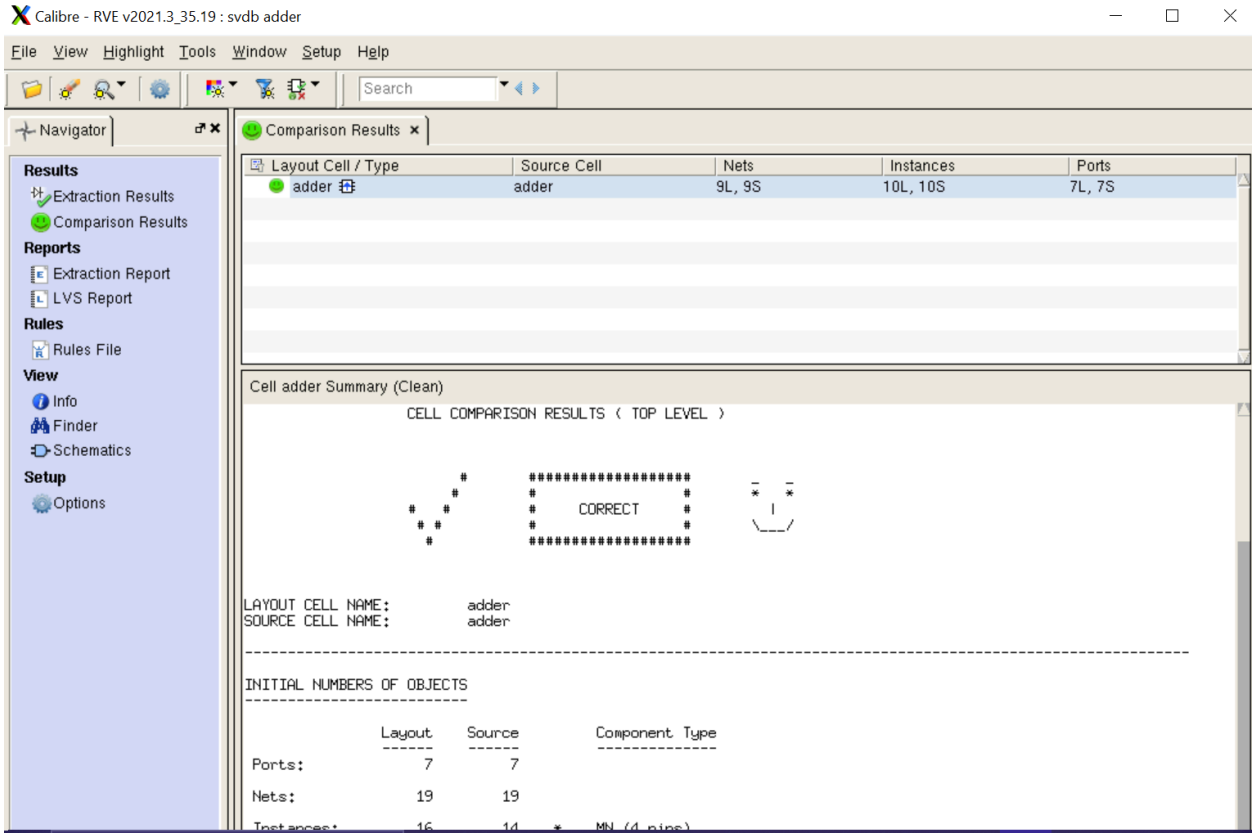


```
================================================================================
=== CALIBRE::DRC-H SUMMARY REPORT
===
Execution Date/Time:       Wed Jan 17 12:30:06 2024
Calibre Version:           v2021.3_35.19    Wed Sep 1 15:25:28 PDT 2021
Rule File Pathname:        /users/course/2023F/cs312000110001/u109006273/layout/final_project/Coin_
Rule File Title:
Layout System:             GDS
Layout Path(s):            Coin_bank.calibre.db
Layout Primary Cell:       Coin_bank
Current Directory:         /users/course/2023F/cs312000110001/u109006273/layout/final_project/Coin_
User Name:                 u109006273
Maximum Results/RuleCheck: 1000
Maximum Result Vertices:   4096
DRC Results Database:      Coin_bank.drc.results (ASCII)
Layout Depth:              ALL
Text Depth:                PRIMARY
Summary Report File:       Coin_bank.drc.summary (REPLACE)
Geometry Flagging:         ACUTE = YES  SKEW = NO  ANGLED = NO  OFFGRID = YES
                           NONSIMPLE POLYGON = NO  NONSIMPLE PATH = NO
Excluded Cells:
CheckText Mapping:         COMMENT TEXT + RULE FILE INFORMATION
```

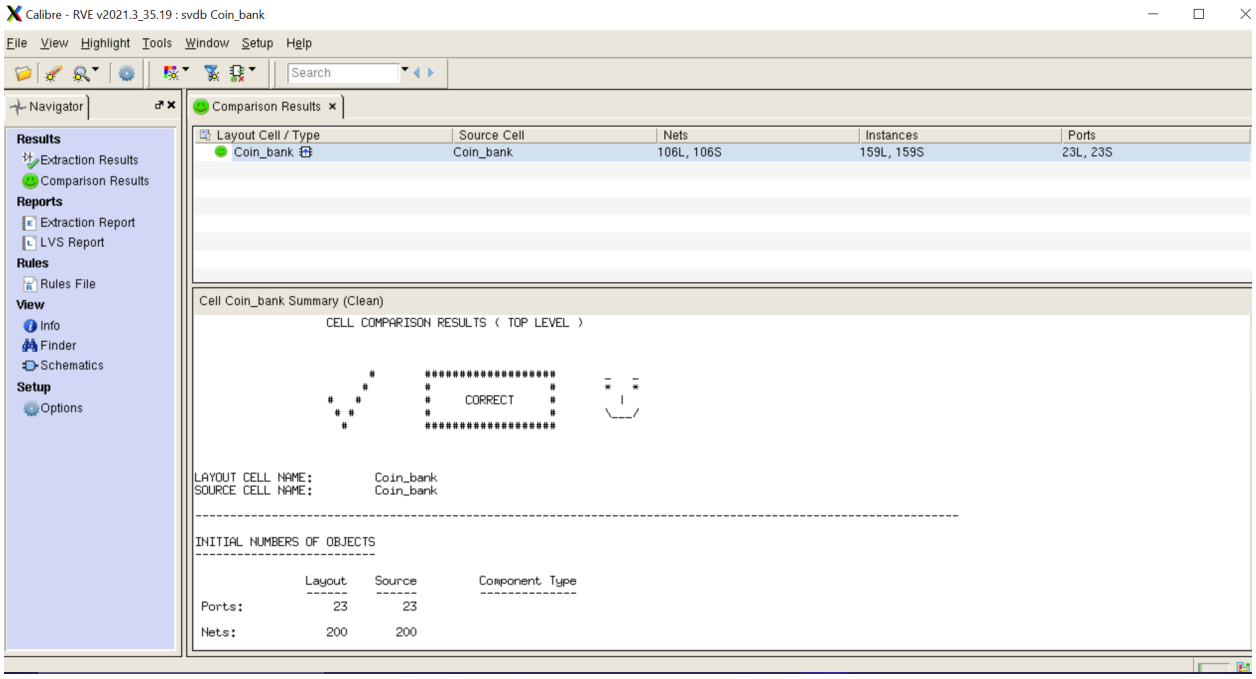
V. Message of Passing LVS

Same thing with DRC I did a bottom up approach, where I do the LVS one by one from the most bottom hierarchy to the highest.
Example.

Adder lvs



Coin bank lvs



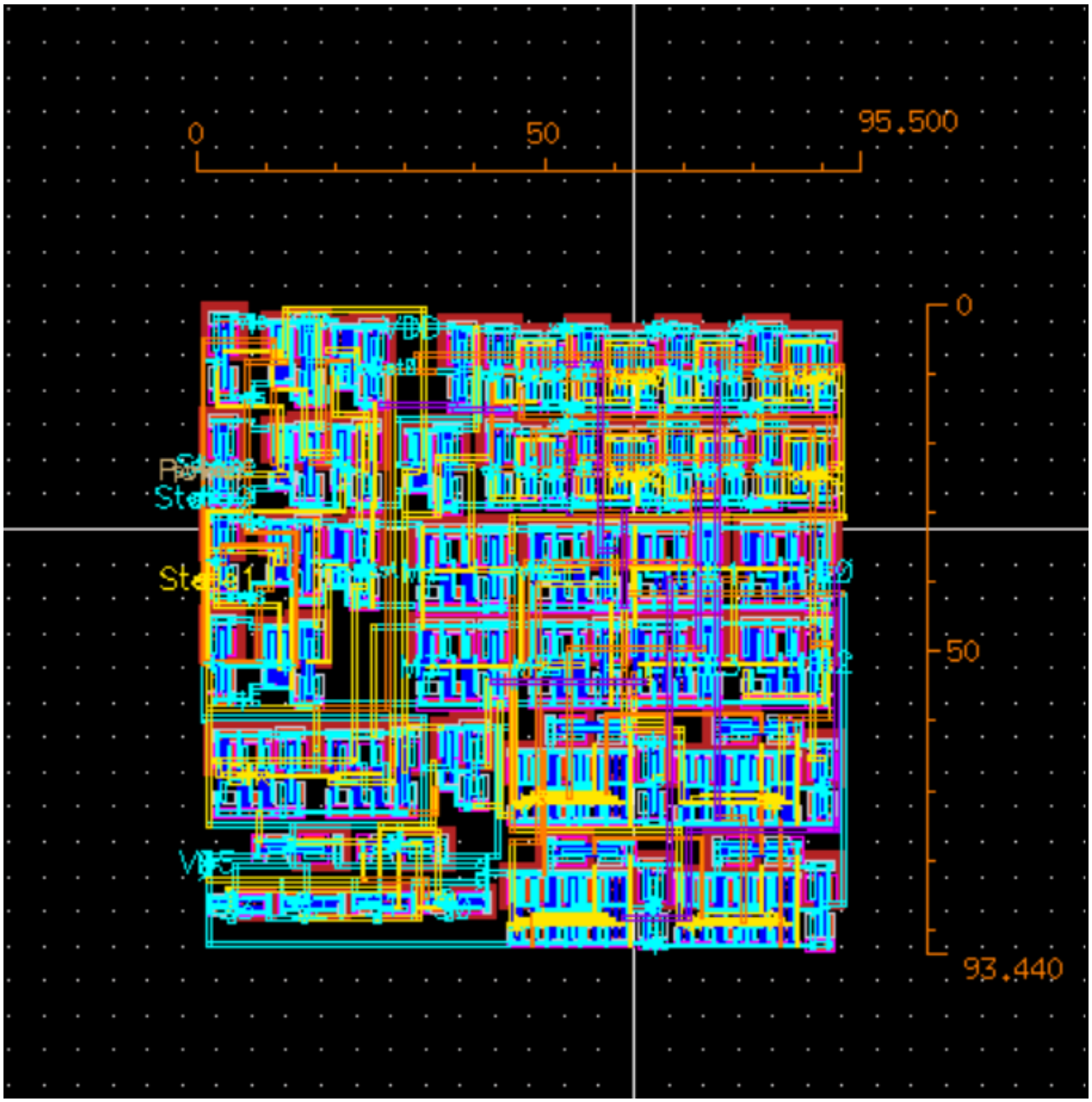
VI. Latency and Area of circuit

Latency :

Pre-sim : 1ns * 35 clock cycle = 35 ns

Post-sim: 2ns * 35 clock cycle = 70 ns

Area of circuit :

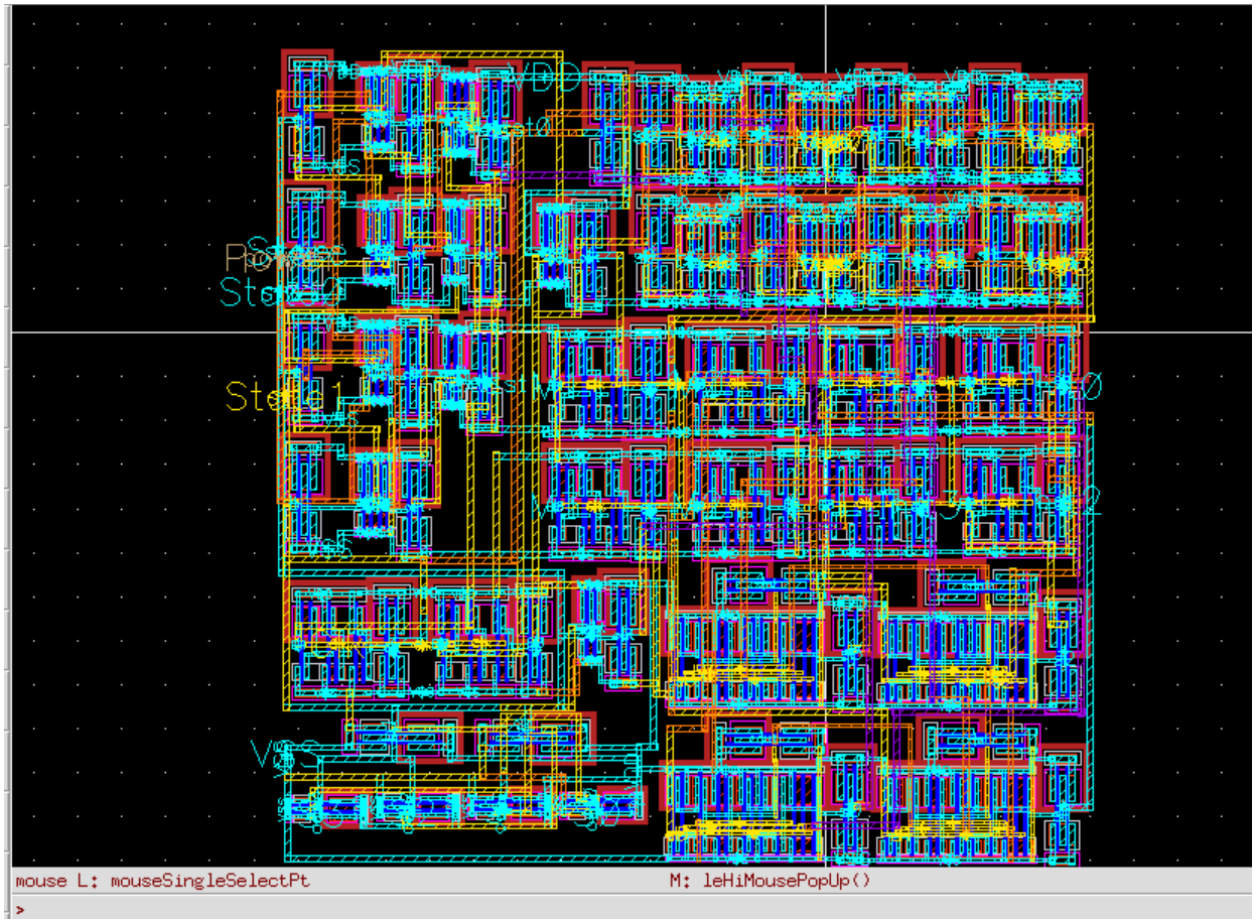


Length : 95.500um

Width : 93.440um

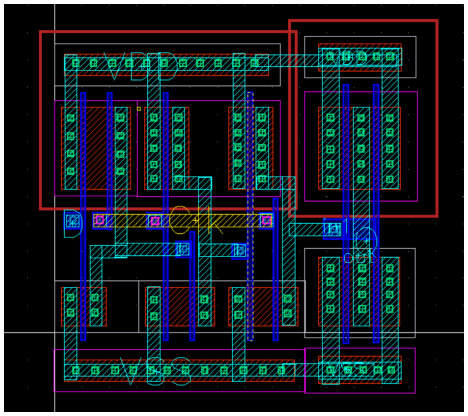
VII. Image of Layout

Coin_bank



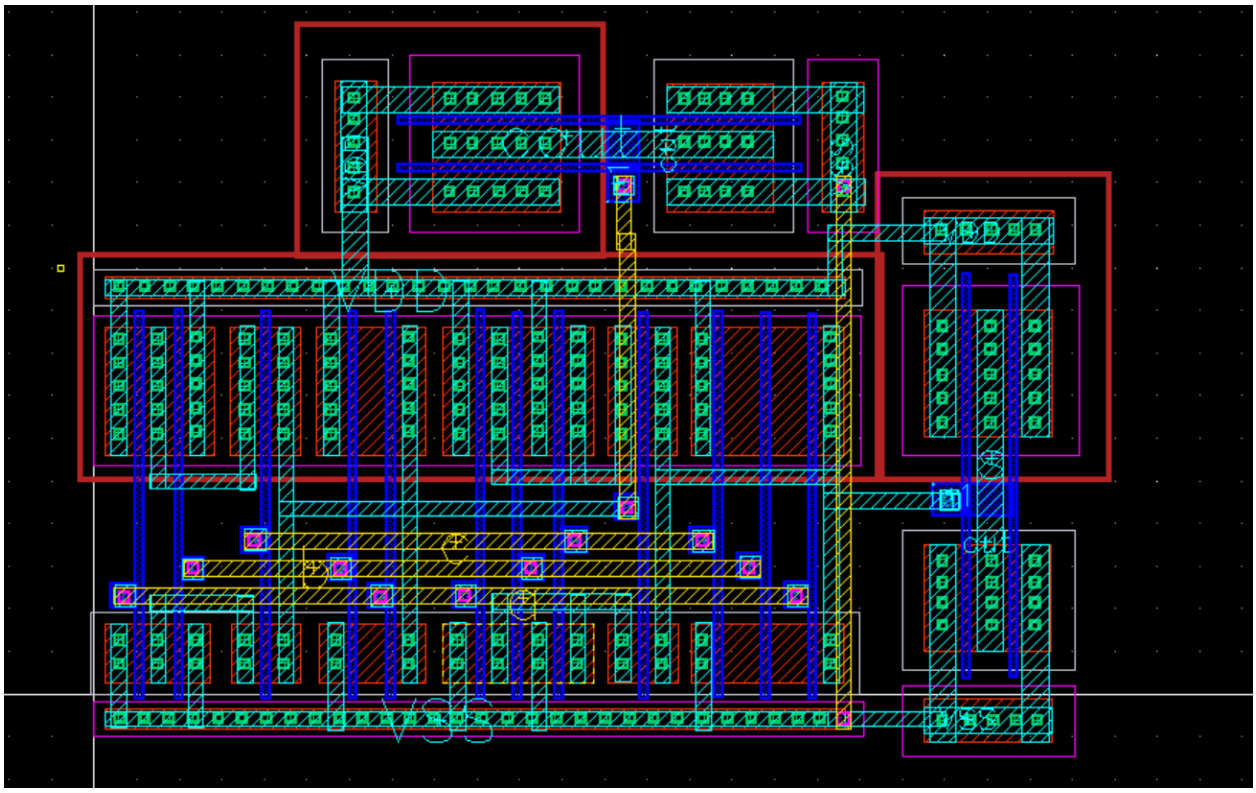
Top Module which is the Coin bank itself. Combination of 4 bit DFF, 4 bit 4 x 1 MUX, 2 bit DFF and 2 4 bit DFF. Finite State machine to calculate next state, 2x4 decoder and some logic circuit to control the DFF.

DFF



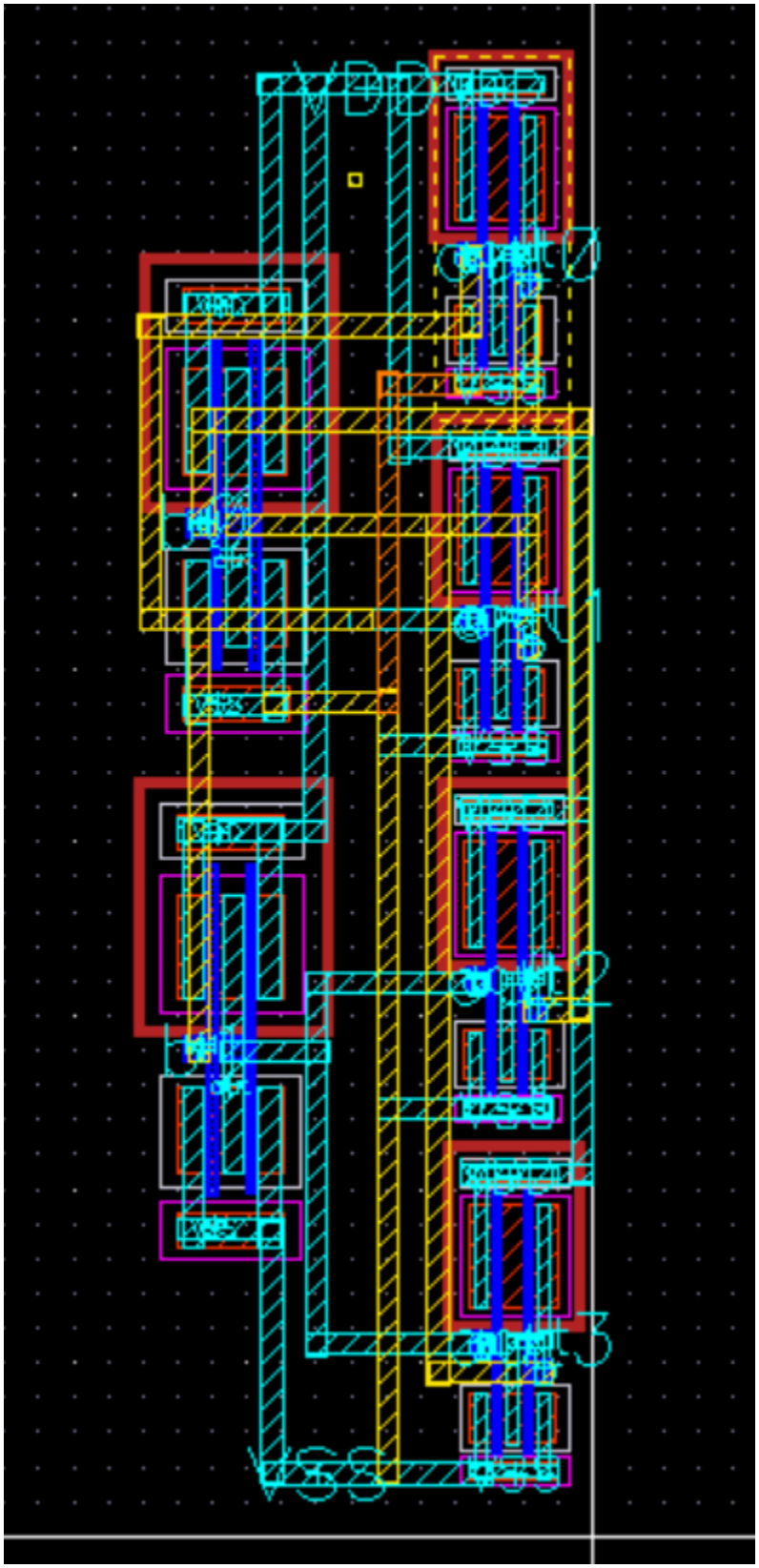
TSPC DFF, this layout is based on the drawing on circuit drawing part I.

Mirror adder



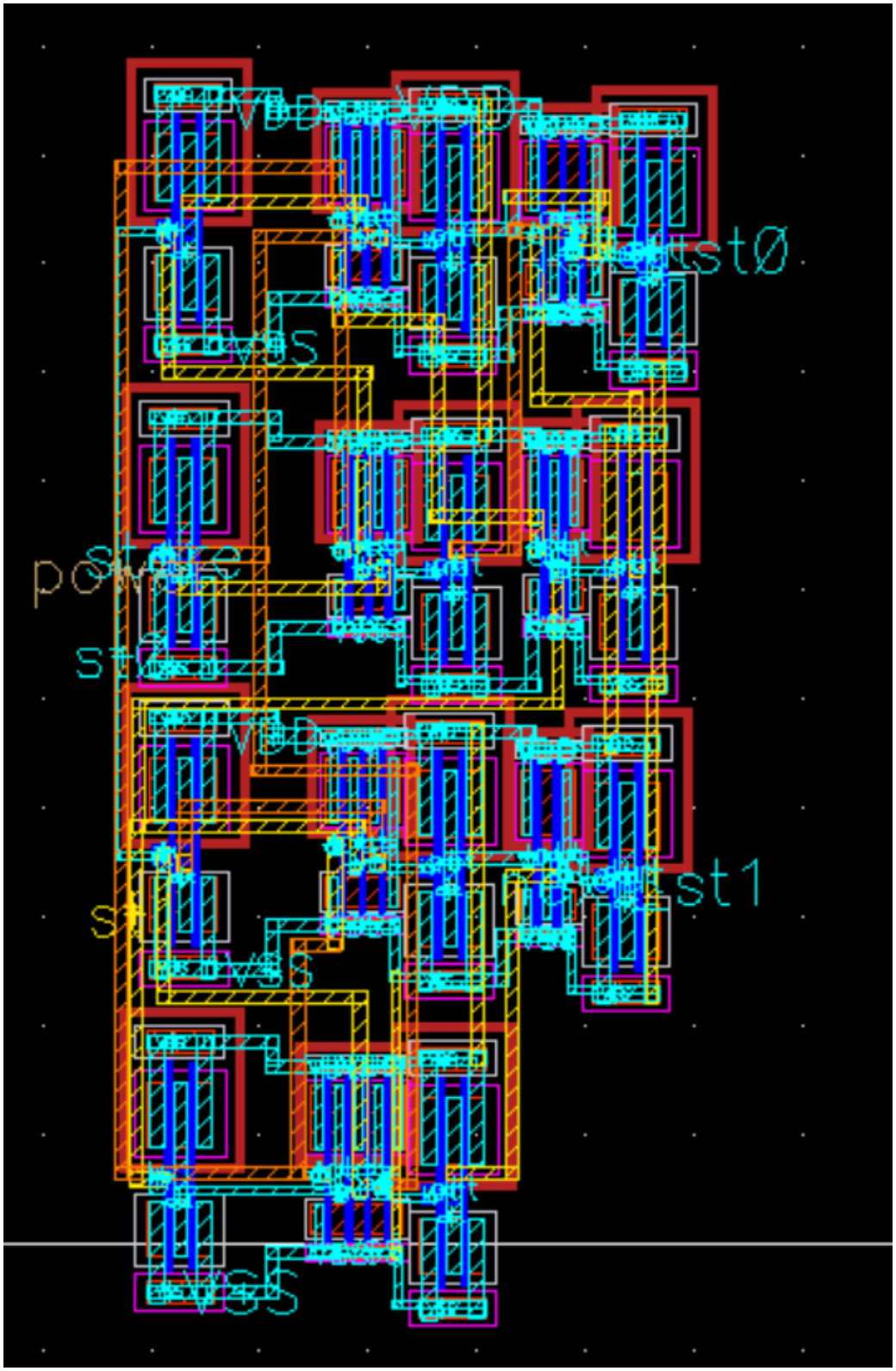
This layout is based on the circuit drawing above for mirror adder.

Decoder 2x4



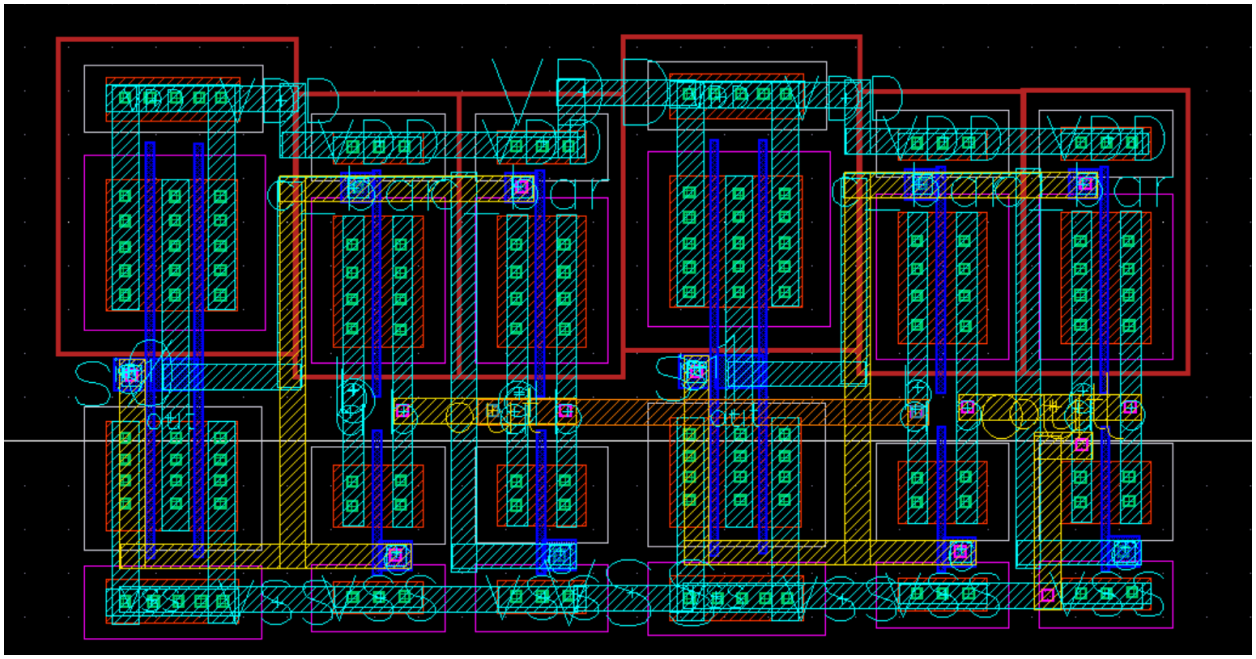
Implemented with NOR Gate

Next_state_cal (to calculate next state)



This is a combination of using OR and AND gate

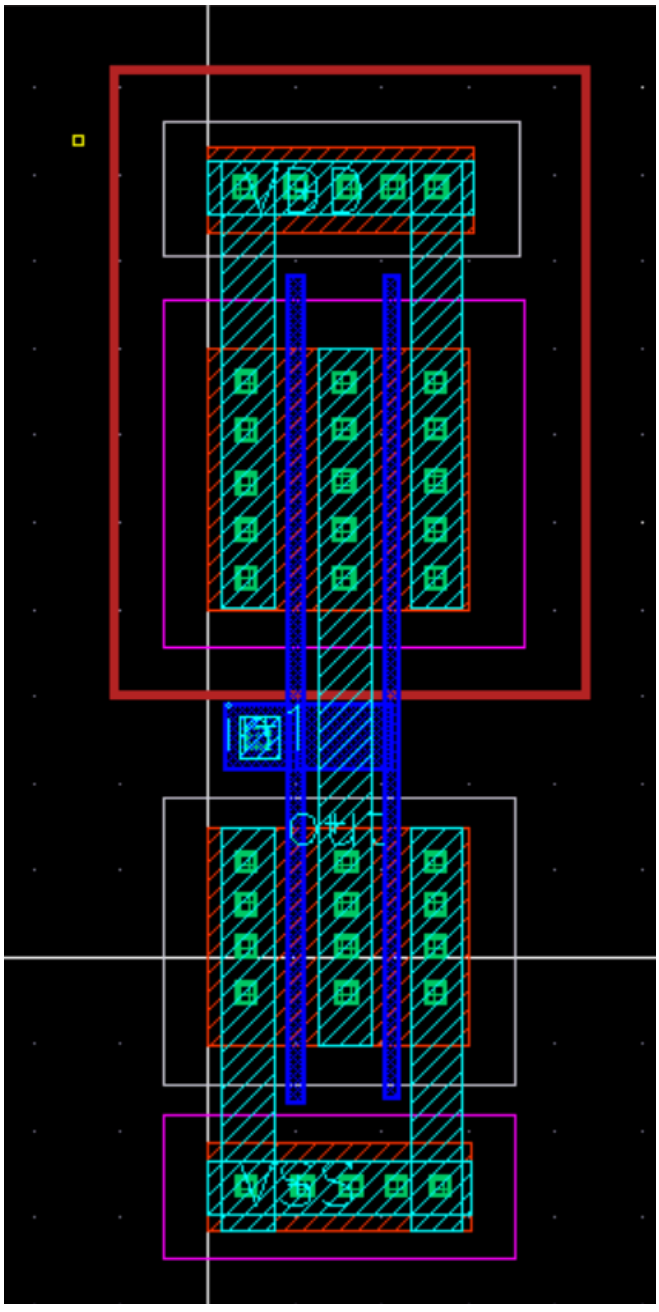
Mux_4x1



This is done with transmission gate 2x1 mux, and since the 2 of the inputs are 0 so the mux looks simpler than the normal mux.

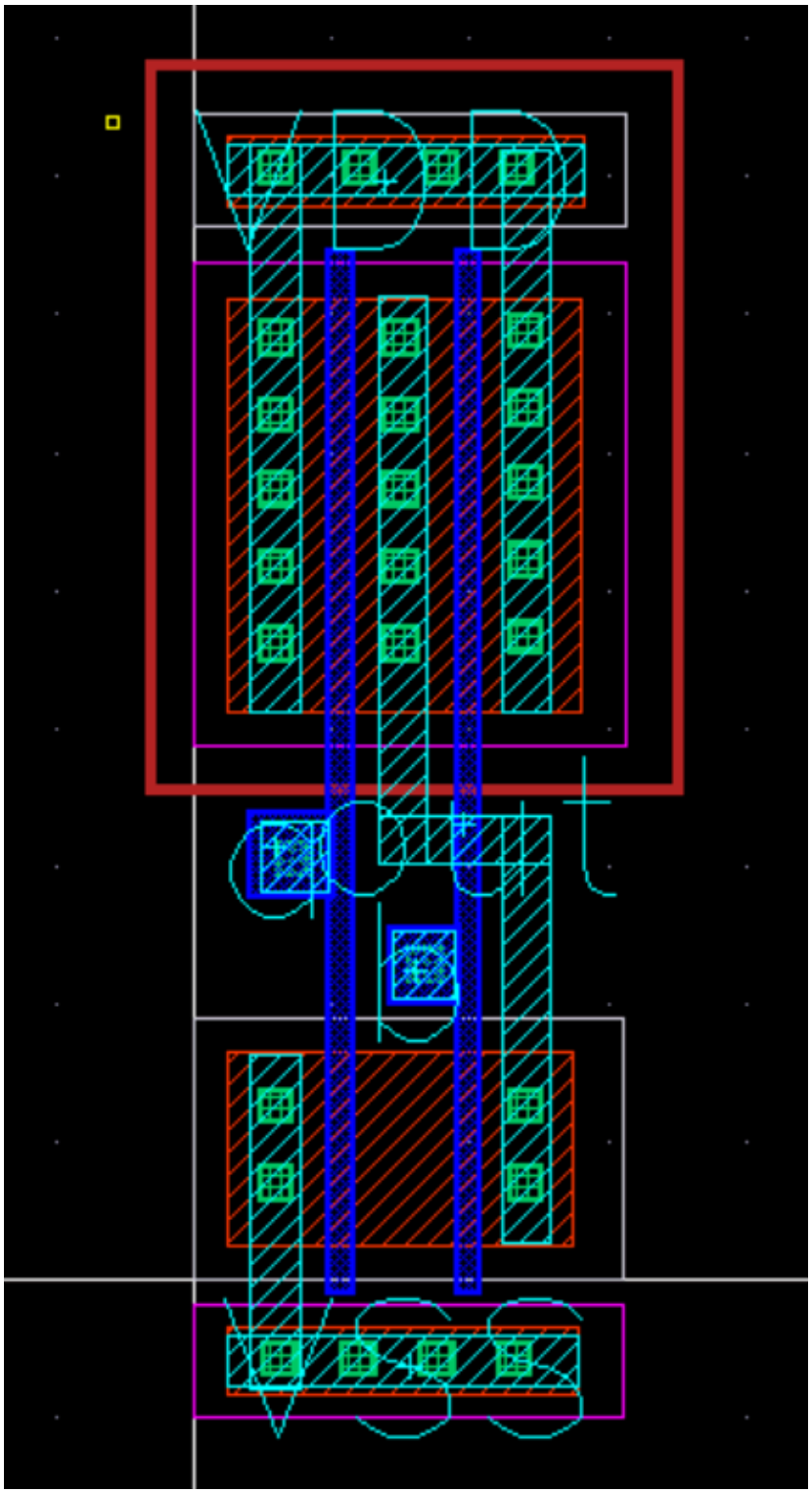
```
.subckt MUX_4x1 a b s1 s0 out VDD VSS
** special case **
** The other 2 outputs are 0 so inputs are not 2^2 **
Xmux_2x1_1 a b s0 out1 VDD VSS MUX_2x1
Xmux_2x1_2 VSS out1 s1 out VDD VSS MUX_2x1
.ends MUX_4x1
```

Inverter



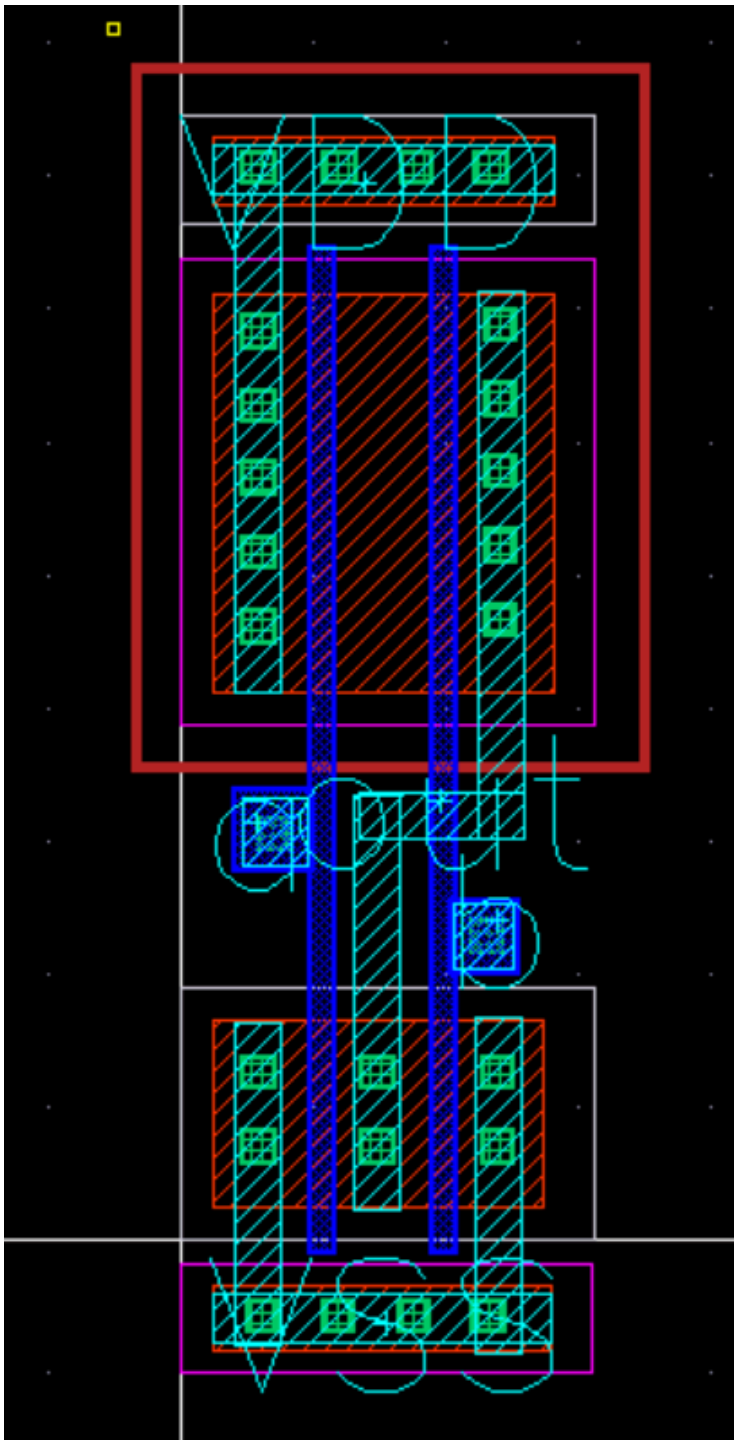
Folding techniques are used to draw the inverter layout to make the size smaller. So, the width looks like it became half of the original width.

Nand_2bit



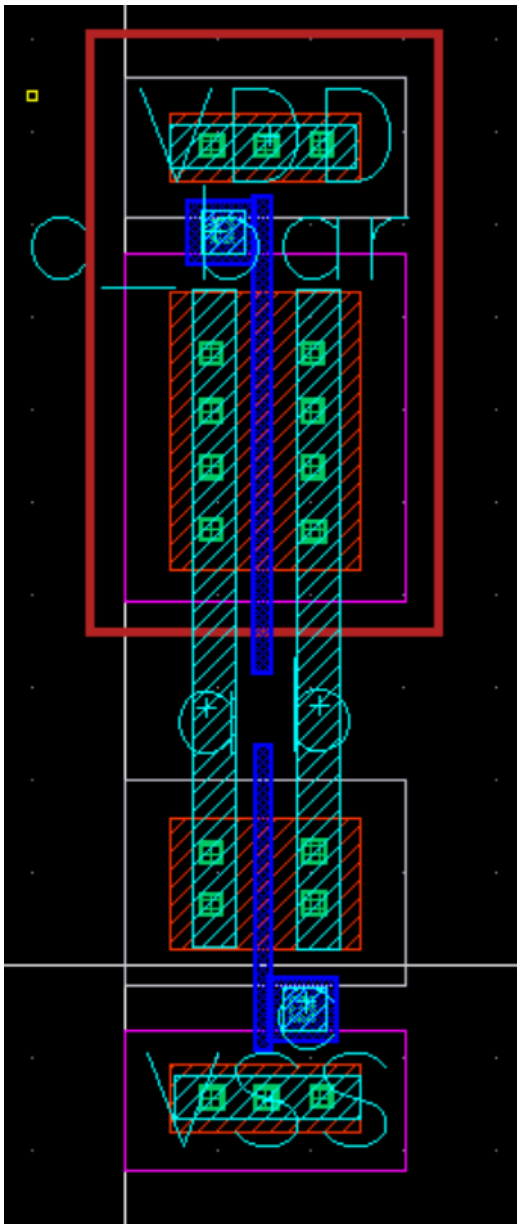
AND gate is implemented with this layout + inverter layout.

Nor_2bit



This is the implementation of the NOR gate, OR gate is NOR gate + inverter.

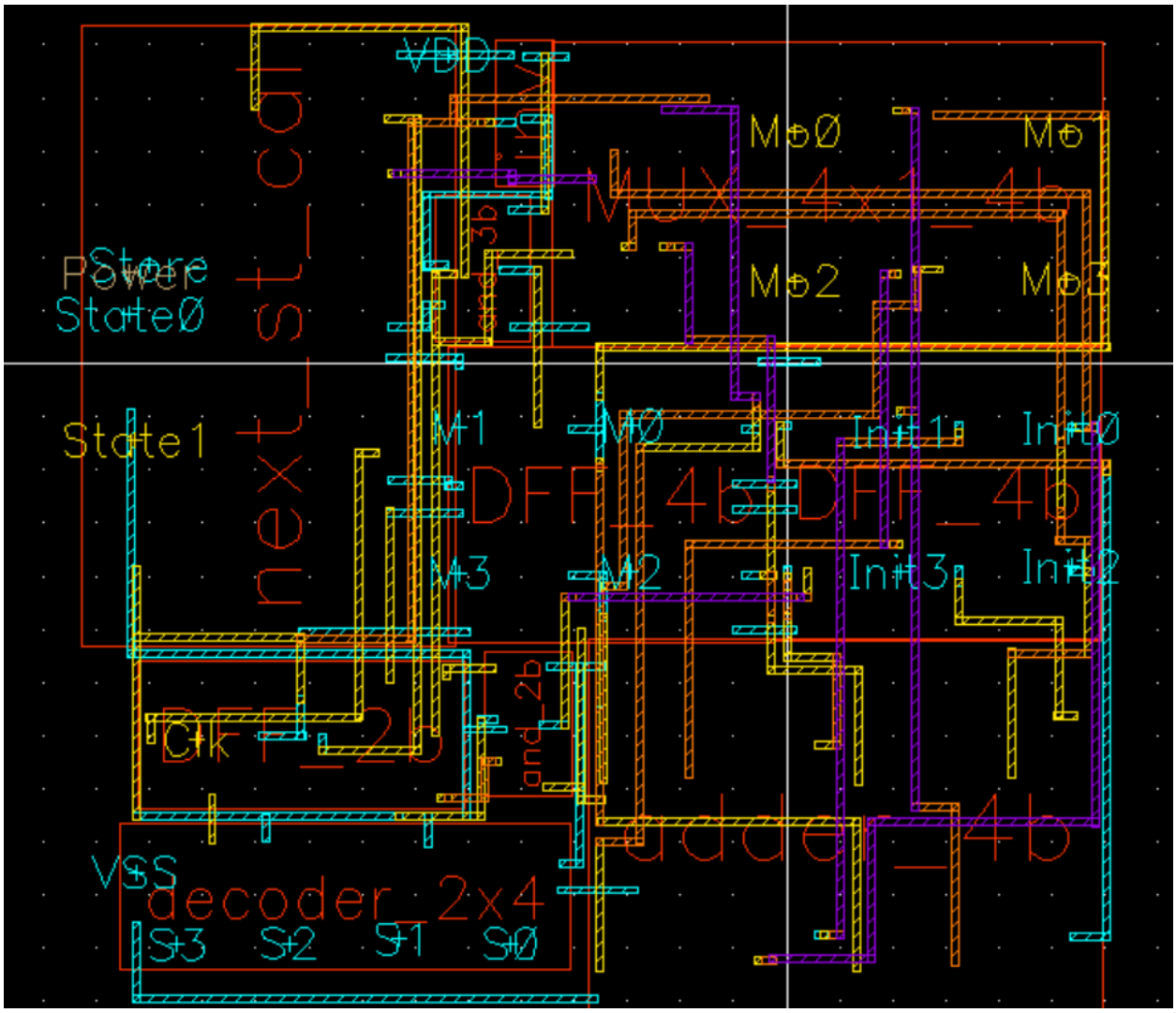
Transmission gate



There are other layouts that I do not include in the screenshot because they seem redundant, such as a 4 bit 4x1 MUX since it is just 4 4x1 mux combined together.

VIII. Hardness of this assignment

The routing is very difficult. This might be because I tried minimizing my area. I made a lot of subcircuits for it to mitigate the difficulty, such as 4 bit DFF, or 4 bit MUX. This might have an adverse effect because the MUX 4 bit drawing becomes fixed which makes it harder to route when being combined later. I do not know if this is the right approach. But regardless, my topmost hierarchy has a really messy routing too because I mix up all my metal1, 2, 3 in lower hierarchies layouts. When I combine everything together in the end, I have to open all the instance layouts, just to see if any of the metal overlaps or not. This is a picture showing how messy it is even at the top most layer.



The second hardest difficulty is to make the pre-sim wave look smooth. I had to guess how to do it by just changing the different widths of pmos and nmos. In the end, I got lucky and figured out that I only have to make the inverter bigger for some reason to solve the problem. Then I removed all my buffers and the waveform still looks nice.

However, I encountered the problem of my waveform on Mo0, Mo1, Mo2, Mo3 falling when two periods logics are 1. This is the screenshot of what I meant.

Someone also encountered it in the discussion and I'm not sure if it is possible to make the wave stay in logic 1 level because I set the Clk for the DFF for storing money and money in DFF to trigger with the state, not with the clock which I am not sure if there's a better way to do it. Maybe I have to use another MUX to have a logical if to not get

triggered on the clock edge in a certain state or something. The screenshot of the problem I meant is below.