DIRECT DIGITAL SYNTHESIZER

Design of ROM

Anusha Ramanujam

ECE 547- Project Report
Department of Electrical and Computer Engineering
University of Maine
Orono, Maine

AMBITION: A journey of thousand miles mostly ends well!

CONTENTS

LIST OF FIGURES

List of Tables

Chapter 1

Introduction

1.1 Project Overview

Direct Digital Synthesizer (DDS) plays an important role in modern digital communications for sine wave generation. Direct Digital Frequency Synthesis has been recognized as a technology for generating highly accurate, frequency- agile (rapidly changeable frequency over a wide range) and low distortion output waveforms. At present, there are two principal forms of applications using DDS: waveform generation in *communications* and signal analysis in *industry and biomedicine[1]e.* The different components of a DDS are Accumulator, ROM and Digital to Analog converter. In this project, my design involves a 256 bit ROM which stores the sinusoidal values for DDS. It consists of four 6 to 2^6 bit tree decoders, four ROMs and 2:1 multiplexers. The multiplexers select the right ROM and their outputs are latched using D flip-flops. This ROM receives 8 bit input from a 12 bit, 50 MHz accumulator designed by Aravind Reghu. The latched outputs are then fed to a 10 bit Digital to Analog converter designed by Steve Fortune.

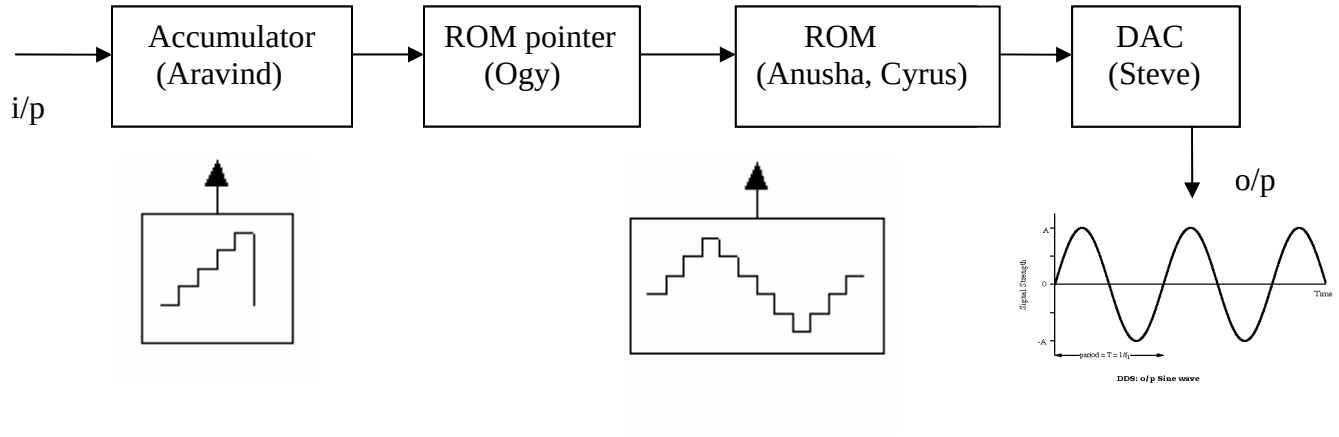


Fig 1.1 DDS guided by Prof.David Kotecki In ECE 547, Spring 2006

1.2 Objective

A DDS system generates one or more frequency from a single reference frequency using digital logic. In this project, my objective was to design a 256 bit ROM which stores the amplitude values for each phase of DDS output waveform. A 256 bit ROM could be designed by using four 64 bit ROMs which thereby minimizes the capacitance and also occupies less space on the chip.

1.3 Design Specifications

The different specifications that were given for the design of a ROM are as follows:

Name	Specification
Number of rows	256
Number of columns	9
Frequency	50 MHz
Voltage	0 to 5 V

Table 1.1 Design Specifications

1.4 Macros

MACRONUMBER	NAME
1	64 bit ROM A
2	64 bit ROM B
3	64 bit ROM C
4	64 bit ROM D
5	9 max_2 to1(2)
6	9 max_2 to 1 tall
7	ROM to DAC CONVERTER
8	DFLIPFLOP

Table 1.2 Macros

1.4 Pin Assignments

Supply net pins were assigned to the package pins with the greatest parasitic capacitance. This parasitic is actually beneficial since it provides additional bypass. Critical signals are assigned to the pins with the least parasitics. The pin assignments are listed in Table 1.3

PIN	PAD TYPE	NAME	DESCRIBTION
1	Padvdd	vdd	Vdd pin
2	Padio		
3	Padio		
4	Padio		
5	Padio		
6	padio		
7	Padio		
8	Padio		
9	Padio		
10	Padaref	out	Out pin
11	Padio	Clk	Clk pin
12	Padio		
13	Padio		
14	Padio	A4	4^{th} input from ACC
15	Padio	A5	5^{th} input from ACC
16	Padio	A6	6^{th} input from ACC
17	Padio	A7	7^{th} input from ACC
18	Padio	A8	8^{th} input from ACC
19	Padio	A9	9^{th} input from ACC
20	Padio	A10	10^{th} input from ACC
21	Padio	A11	11^{h} input from ACC
22	Padio	A3	3^{rd} input from ACC
23	Padio	A2	2^{nd} input from ACC
24	Padio	A1	1^{st} input from ACC
25	Padio	A0	0 input from ACC
26	Padio		
27	Padio		
28	Padio		
29	Padio		
30	Padio		
31	Padio		
32	Padio		
33	Padio		
34	Padio		
35	Padio		
36	Padio		
37	Padio		
38	Padio		
39	Padio		
40	Padgnd	Gnd	Gnd pin

Table 1.3 Pin Assignments

Chapter 2

Circuit Design

This chapter discusses the concept and design of the individual building blocks of the 256 bit ROM. The performance of each building block will determine the performance of the ROM and thus the ultimate performance of the DDS.
Blocks discussed in this chapter are:
1. 6 to 2^6 bit tree decoders
2. Four 64 bit ROMs
3. Sense Amplifiers
4. 2:1 Multiplexers
5. clocked D Latches

2.0 Description of Schematics

Refer to table 1.2 for all Macros used in this design.
Refer to Appendix A for full size schematics of each cell.

2.1 6 to 2^6 bit tree decoders

The 256 bit ROM receives its 8 bit input from the pipelined 12 bit accumulator. The first 6 bits (starting from the LSB) are given to the line drivers. The line driver selects and drives only one line on the ROM. It uses the 6 to 2^6 bit tree decoder in which one of the 64 bit output is pulled high depending on the 6 bit address. On the output side of the selection lines, the line is tied to a pull-down NMOS and an offset inverter connected to a regular inverter. The gate of the pull-down NMOS is tied to the signal line so any ‘high’ floating lines are sufficiently pulled down. The offset inverter allows the real high signal to be accepted, since the NMOS will be trying to pull it low. The tree consists of NMOS pass gates, with the common end tied to VDD. The configuration of the tree is an expansion of the tree in [2] to allow for 6 inputs and 64 outputs. Figure 2.1 shows the zoomed in view of the tree decoder with 6 inputs.

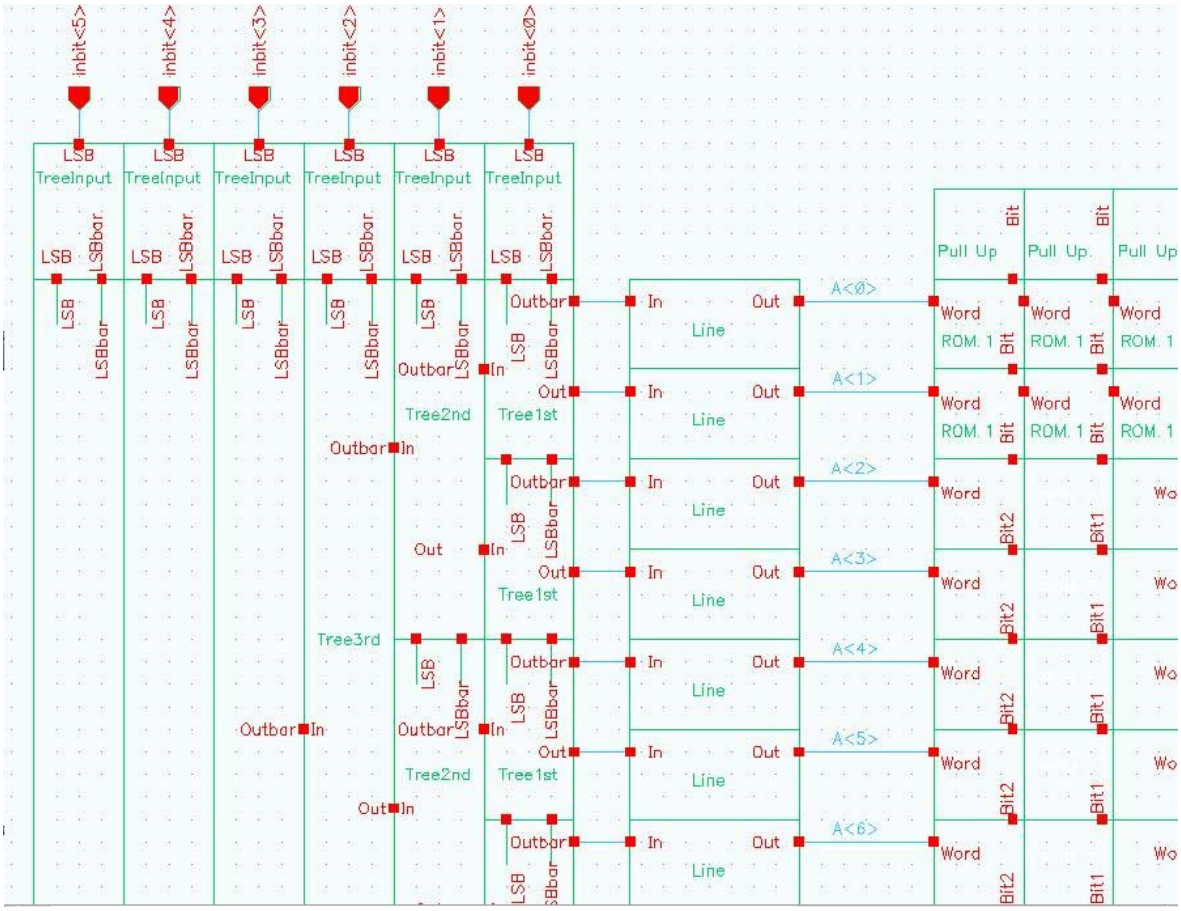


Fig 2.1 Schematic of a Tree Decoder (Zoomed in view)

2.2 Four 64 bit ROMs

Figure 2.2.1 shows the pictorial representation of 2^6 x 9 ROM organization. The four 64 bit Read Only Memories are used to store bits in a structured manner. ROMs are nonvolatile; they retain their contents when power is removed. The mask programmed ROM's in this design use one transistor per bit whose presence or absence determines 0 or 1.

In a ROM every address specifies a preprogrammed output. In this design, NOR based ROM arrays are employed

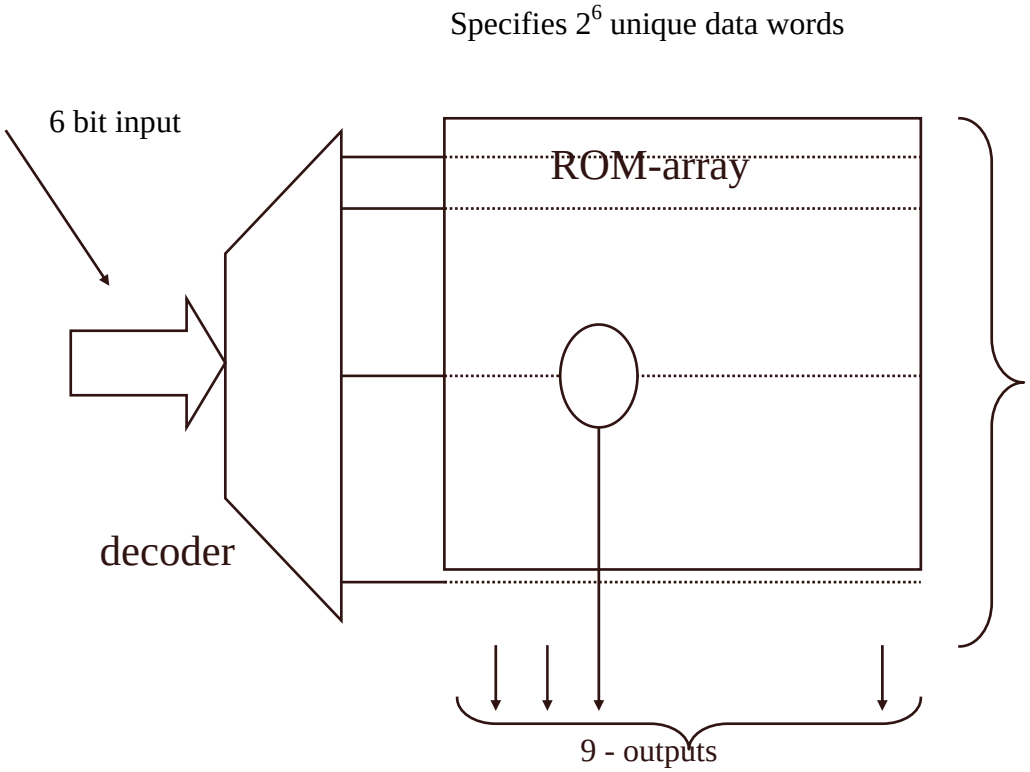


Fig 2.2.1 2^6 x 9 ROM organization

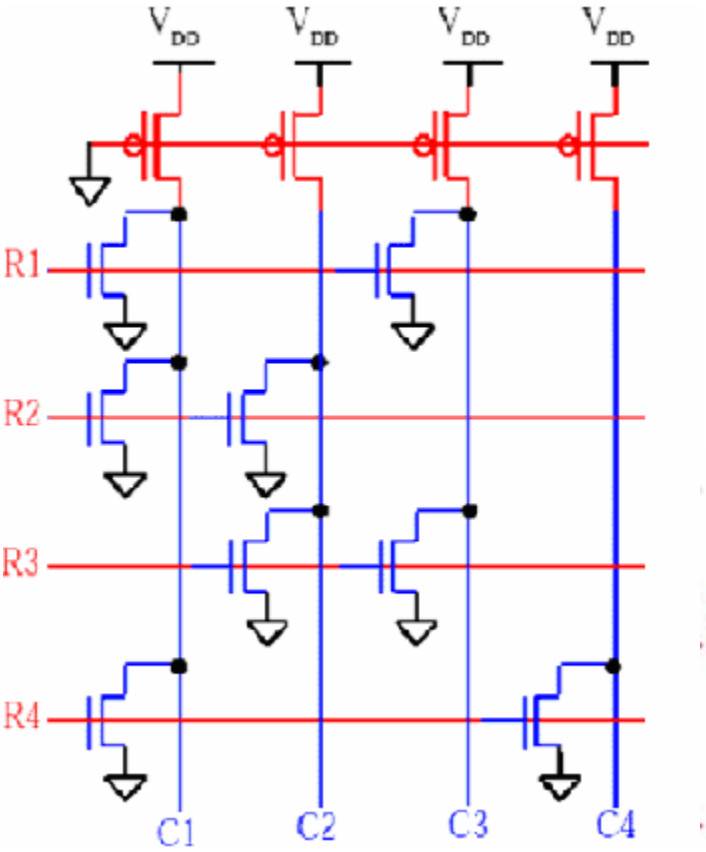
Bit 0
This cell represents a zero bit in the ROM. It contains an NMOS that pulls the line low when it is active.

Bit 1
This cell represents a one bit in the ROM. Since the ROM lines are pulled high, this cell only contains wires to continue the ROM line

The tree decoder enables the selection of one of the 64 rows. Only the selected row is at logic"1" and all other rows are at logic "0". When this particular row has bit lines with Bit 0 or Bit 1 programmed in it, then the outputs go high or low accordingly.

NOR based ROM array

The design of the 64 bit ROM has a NOR based array as illustrated in figure 2.2.2. All word lines R_i are kept at logic “0” level, except the selected line is pulled up to “1” level.



WORD LINES				BIT LINES			
R1	R2	R3	R4	C1	C2	C3	C4
1	0	0	0	0	1	0	1
0	1	0	0	0	0	1	1
0	0	1	0	1	0	0	1
0	0	0	1	0	1	1	0

Fig 2.2.2 NOR based ROM array

Programming of the ROM (Refer Appendix D)

Address of the ROM corresponds to the phase of the sine wave. The ROM contains a ¼ stroke of a sine wave instead of a full stroke to save area. The output of the Accumulator is given

to a ROM pointer which helps the ROM to generate the sine wave. The ROM has the specification of 256 rows. This means that the amount of change per address is (90/256) = 0.3516 as shown in Fog 2.2.3. The starting point in degrees is (90/256)/2 = 0.1758. This sine of 0.1758 is the value which is stored in the first location. The next value that is stored in the ROM is sin (0.1758 + (1x 0.3516)). Thus the nth value stored is sin (0.1758 + (n x 0.3516)). The Matlab programming was used to calculate all the values in binary and all the values were normalized for better accuracy.

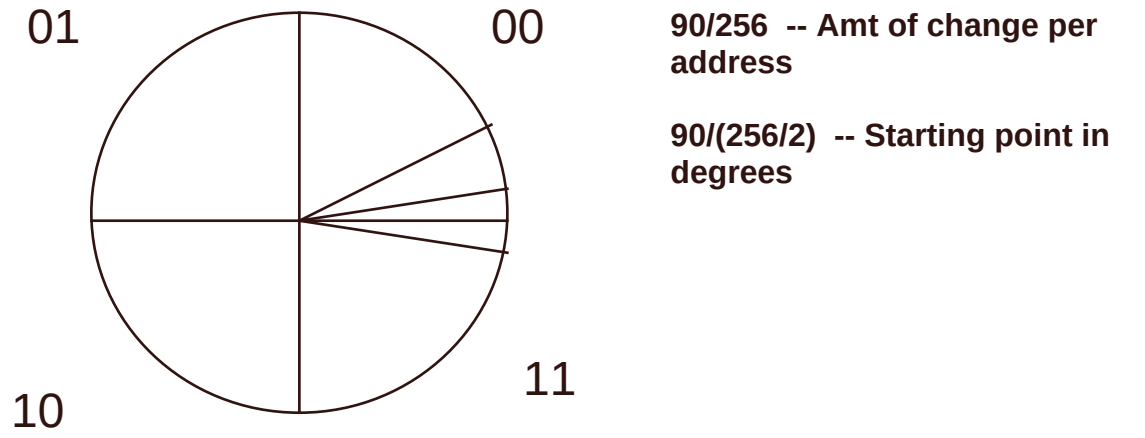


Fig 2.2.3 ROM Programming representation

2.3 Sense Amplifiers

A Sense Amplifier is an essential circuit in designing memory chips. Due to large arrays of cells, the resulting signal has a much lower voltage swing. To compensate for that swing, the output of the ROM is given to a sense amplifier which is used to amplify voltage coming out of the bit line. The schematic of the Sense Amplifier is shown in Fig 2.3.

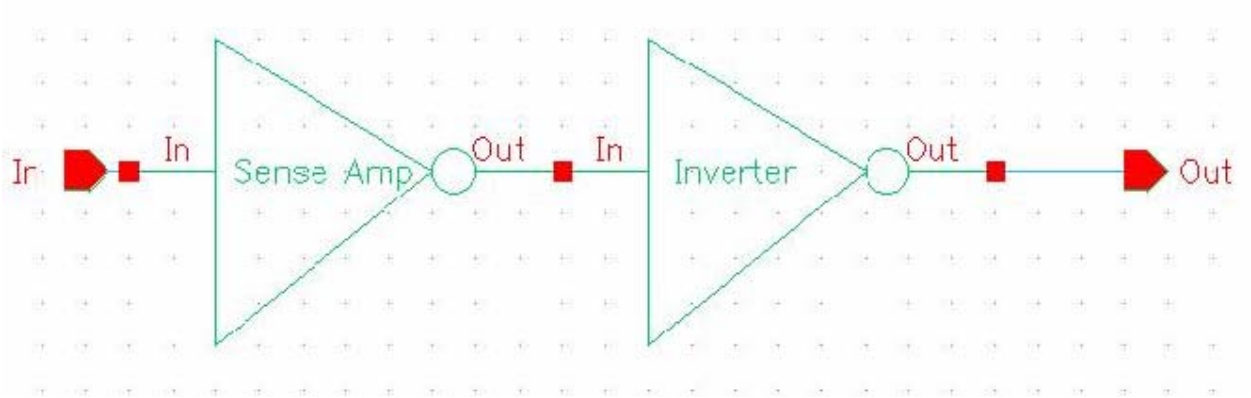


Fig 2.3 Schematic diagram of the sense Amplifier

2.4 2:1 Multiplexers

Multiplexer is a device with multiple signal inputs, one of which is selected by separate address inputs to be sent to the single output. A multiplexer will have signal inputs, control input and output. They are used in building digital semiconductors such as CPUs and graphic controllers, and also in communications.

In this project, the selection of the four 64 bit ROMs is done using multiplexers. The MSB and the next most significant bit that are received from the accumulator act like control inputs for the multiplexers. The multiplexers are constructed using Transmission (TX) gates. The symbol for the multiplexer is as shown in Fig 2.4.1.The TX gates are particularly used for path selection. When the select signal is HIGH one input is passed and when it is LOW the other input is passed. The application of the transmission gate as the path selector can be used to implement multiplexers.

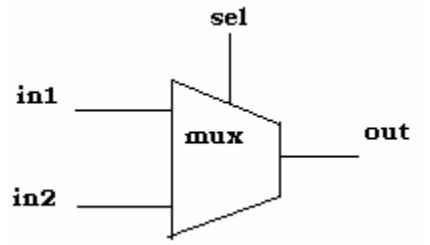


Fig 2.4.1 Symbol of 2:1 Multiplexer

Fig 2.4.2 represents a pictorial representation of the four 64 bit ROMs with multiplexers.The four 64 bit ROMs are marked as A, B, C and D. The 9 outputs from each of the ROMs A and B are given to nine 2:1 multiplexers connected in a parallel manner. Similarly, the 9 outputs from each of the ROMs C and D are given to another set of nine 2:1 multiplexers connected parallel to each other. The next most significant bit from the accumulator to the ROMs is used as the select line N for all the 2:1 multiplexers. This acts like a control input which selects (ROM A or ROM B) and (ROM C or ROM D). The MSB that is given to the ROM acts like the control input M which enables the selection of only one ROM at a time. Table 2.1 shows the selection of ROM based on input bits M and N.

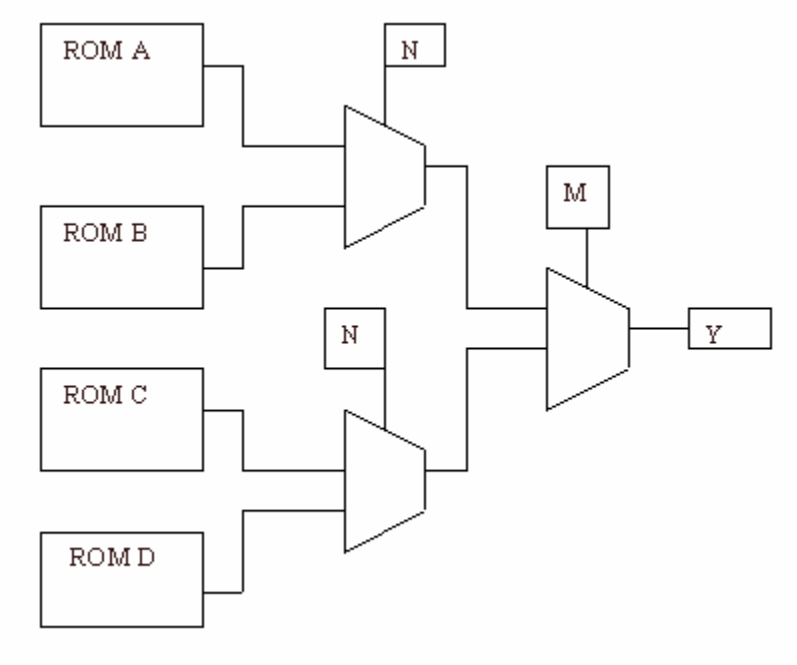


Fig 2.4.2 Pictorial representation of 64 bit ROMs with multiplexers

M	N	Y(ROM selected)
0	0	D
0	1	C
1	0	B
1	1	A

Table 2.1 ROM selection based on input bits M and N

2.5 Clocked D Flip-flops

The outputs from the multiplexer are latched using clocked D flip-flops. The edge triggered D flip-flop can be easily constructed from the RS flip-flop. One essential point about the D flip-flop is that when the clock input falls to logic 0 and the outputs can change state, the Q output always takes on the state of the D input at the moment of the clock edge. This circuit has two D latches in a master- slave configuration driven by a clock. The different logic circuits used in this design are 2 inverters, 4 AND gates and 4 NOR gates. The sizing of the transistors was done for each of this digital logic. Fig 2.5.1 shows the schematic view of the D flip-flop used for latching.

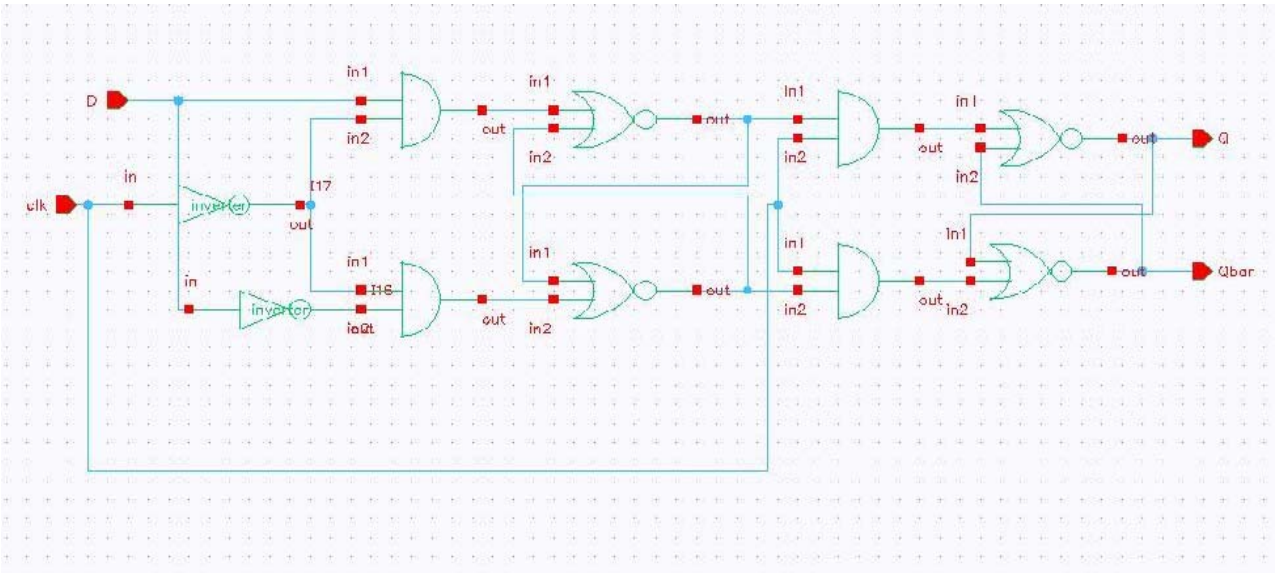


Fig 2.5.1 Schematic view of D flip-flop

Chapter 3

Circuit Performance

Simulation results

3.1 Transmission Gate

The transmission gate has two transistors: one N channel transistor and one P channel transistor. Each one is capable of transmitting one of the levels accurately but neither device is capable of transmitting both high and low voltages. These two transistors are connected in parallel. The reason to use these two transistors is that they pass logic lows and logic highs. Transmission gates act as tristate buffers.
Fig 3.1.1 shows the test circuit of TX gate.When the select signal is high the output is high and when the select signal goes low the output goes low. The figure 3.1.2 demonstrates that the output is the same as the input when the transmission gate is enabled and that is it is floating when it is disabled.

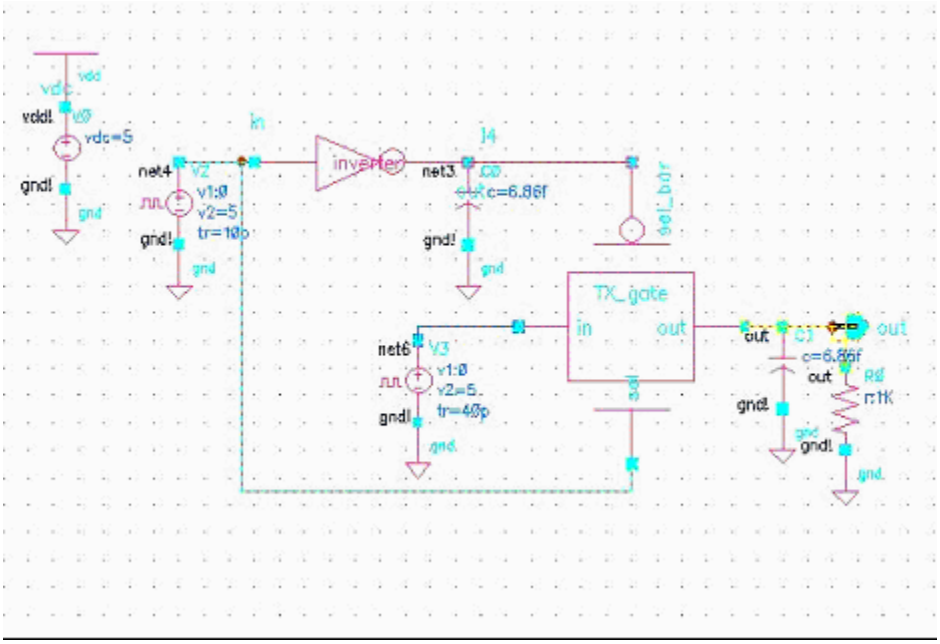


Fig 3.1.1 Test circuit of Transmission gate

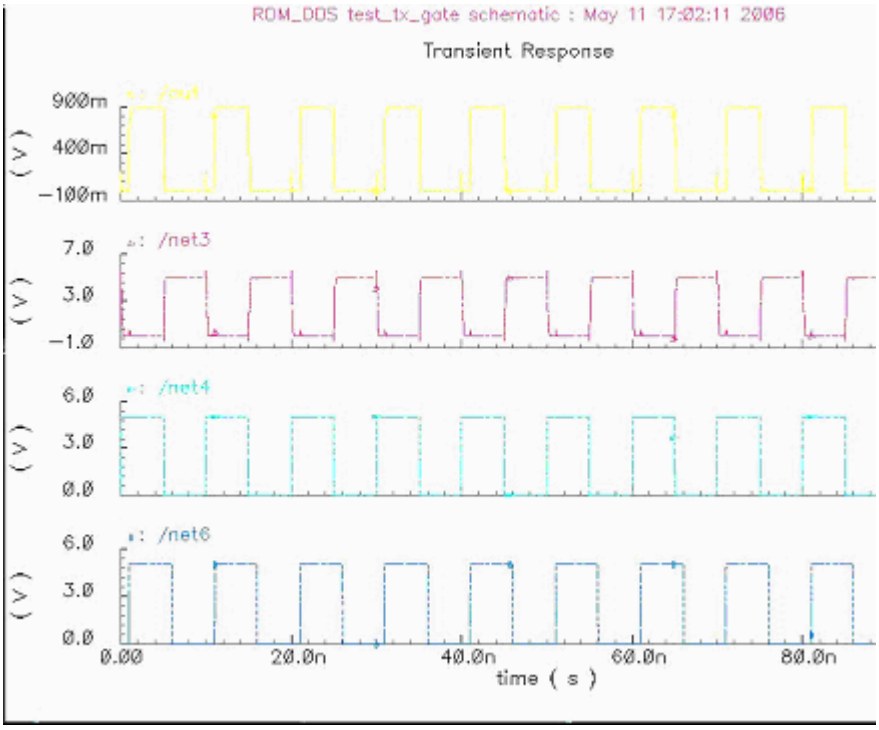


Fig 3.1.2 Output waveform of Transmission gate

The advantage of the CMOS transfer gate is that the degree of capacitive feedthrough is reduced. This is because it is clocked with the true and complement values of the enable signal.

3.2 2:1 Multiplexer

The 2:1 multiplexer is constructed using 2 Transmission gates (Fig 3.2.1). The idea of path selector can be used to implement multiplexers. Logically, the output of the path selector [3] can be written as

Z = AS +BSbar

When the selector signal S is high, A is passed to the output while a low on S passes B to the output. Fig 3.2.2shows the output waveform of 2:1 multiplexer.

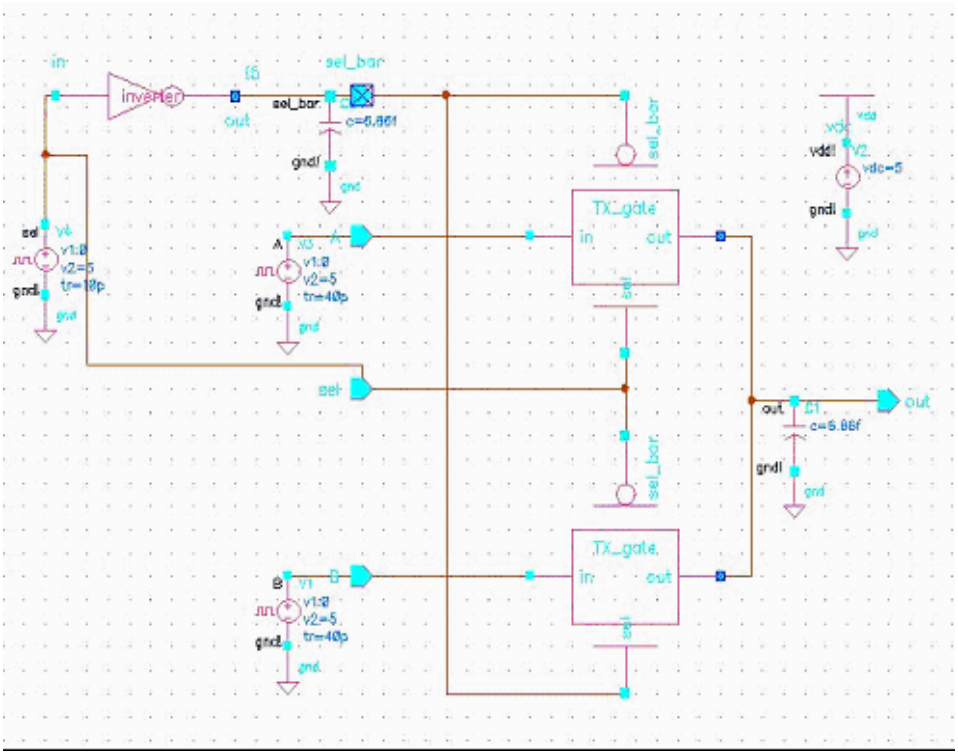


Fig 3.2.1 Test Circuit of 2:1 Multiplexer

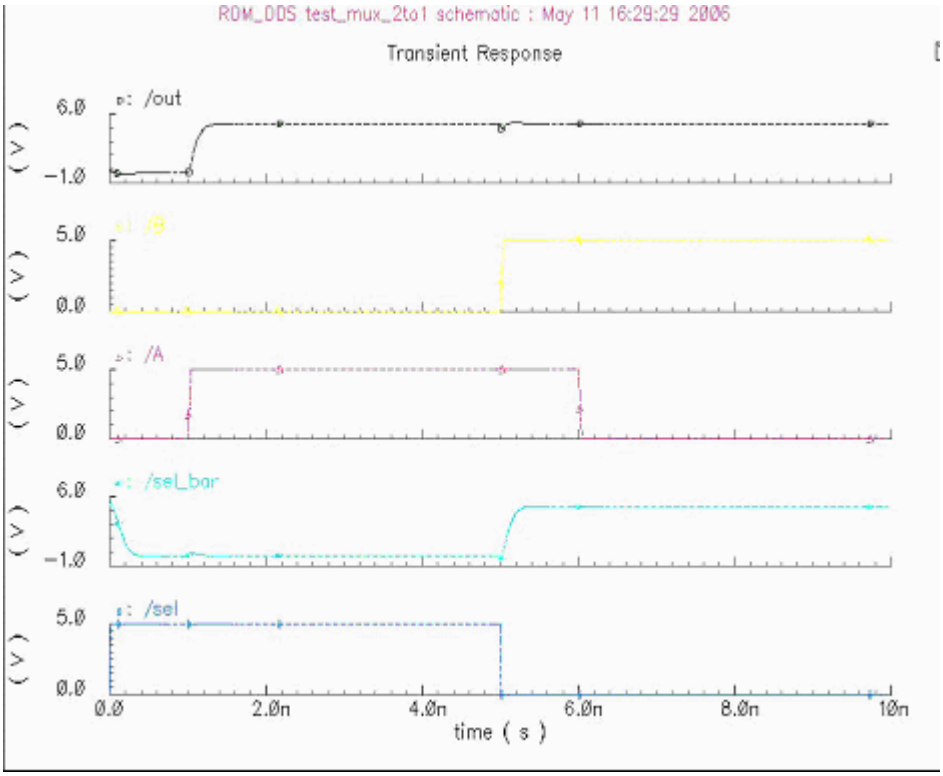


Fig 3.2.2 Output waveform of 2:1 multiplexer

3.3 Two input NAND gate

NAND gates consist of two parallel PMOS devices and two series NMOS devices. Test circuit of NAND gate is shown in Fig 3.2.1.Sizing the four transistors is based on achieving the same delay as the inverter when driving the same load capacitance under worst case input conditions [4]. All four devices in the NAND gate are 2W devices. When both the inputs A and B are high then the output of the NAND gate is zero. When both the inputs A and B are high or when one of the inputs is high, then the input is high. This could be proved by the simulation result in Fig 3.2.2.

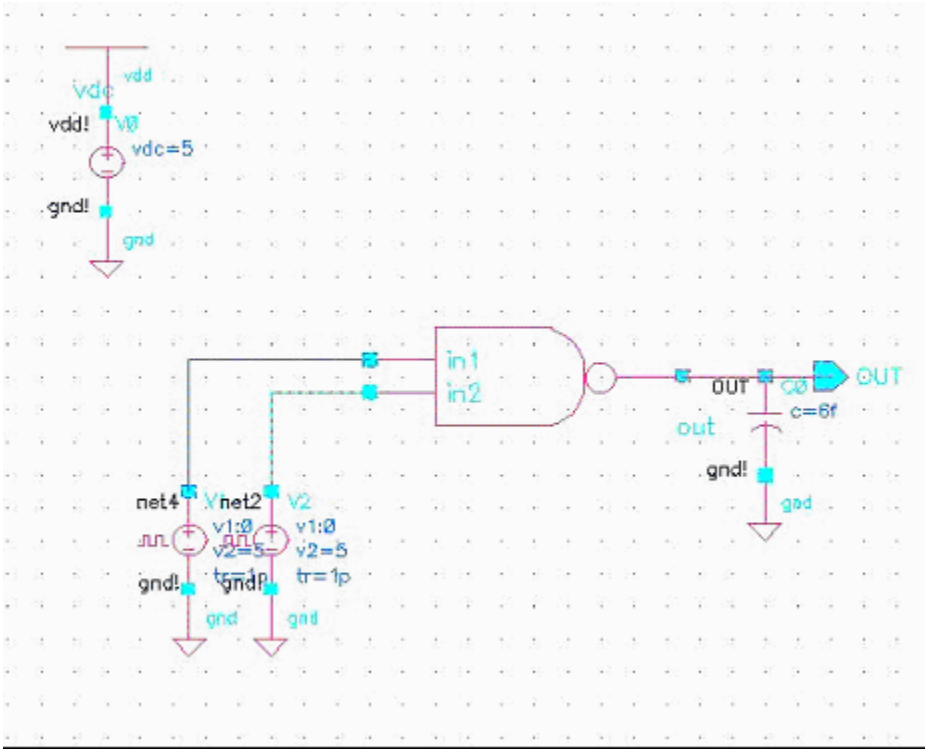


Fig 3.3.1 Test circuit of 2 input NAND gate

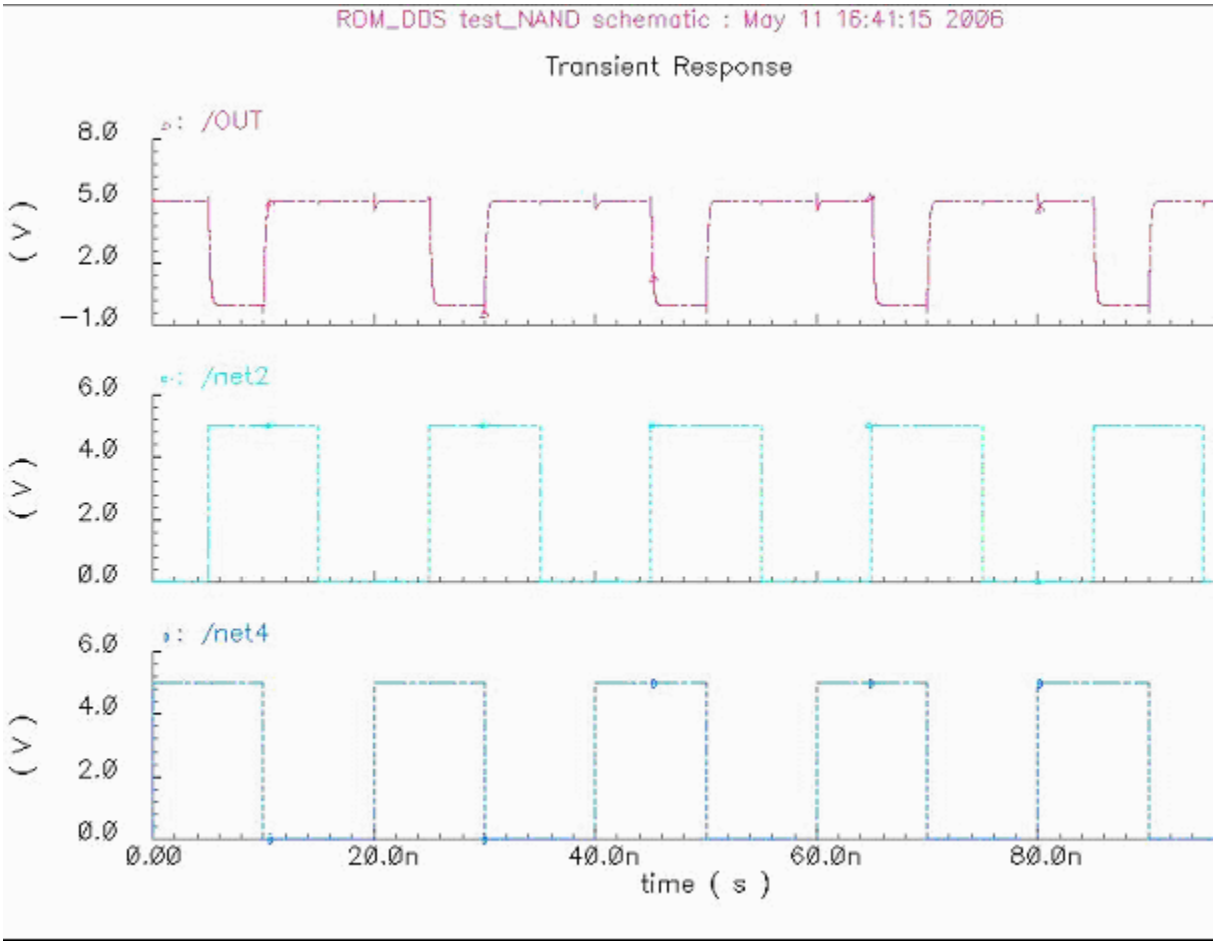


Fig 3.2.2 Output waveform of NAND circuit

3.4 64 bit ROM

The 64 bit ROMs were tested (Fig 3.4.1) by giving the output of the accumulator to the input of the tree decoder. The test result was that the tree decoder helped the circuit to select the right row and the outputs were zero. The outputs were zero and one corresponding to that row. Fig 3.4.2, 3.4.3, 3.4.4 show the output waveforms of 64 bit ROM A, ROM B,ROM C, ROM D respectively. The outputs of the multiplexer 3 are latched and the waveform is shown in Fig 3.4.5.

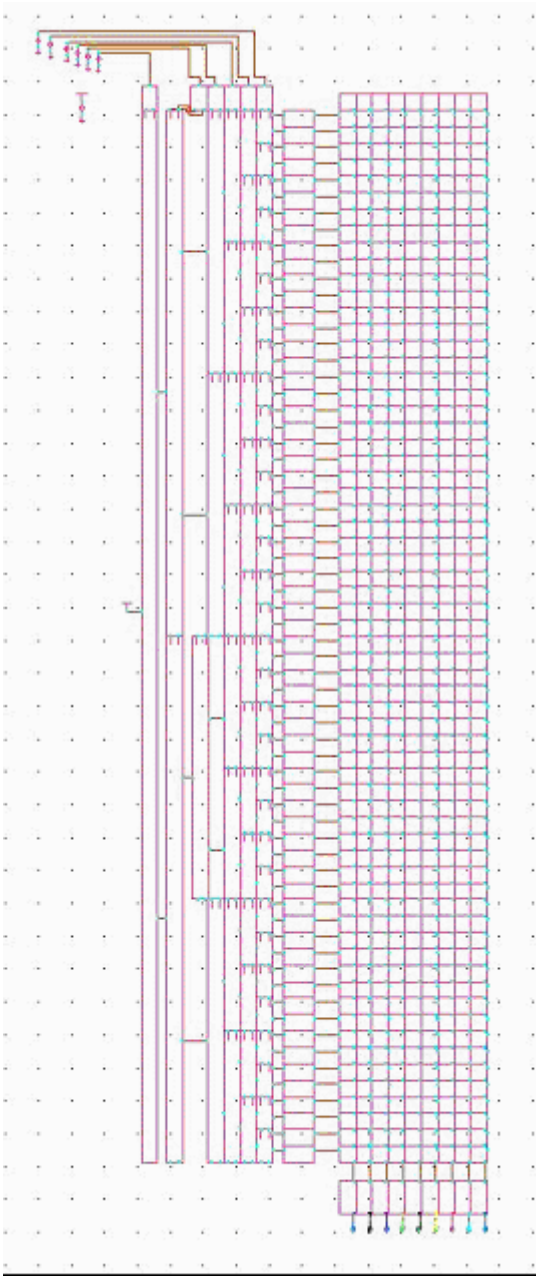


Fig 3.4.1 Test of 64 bit ROM A

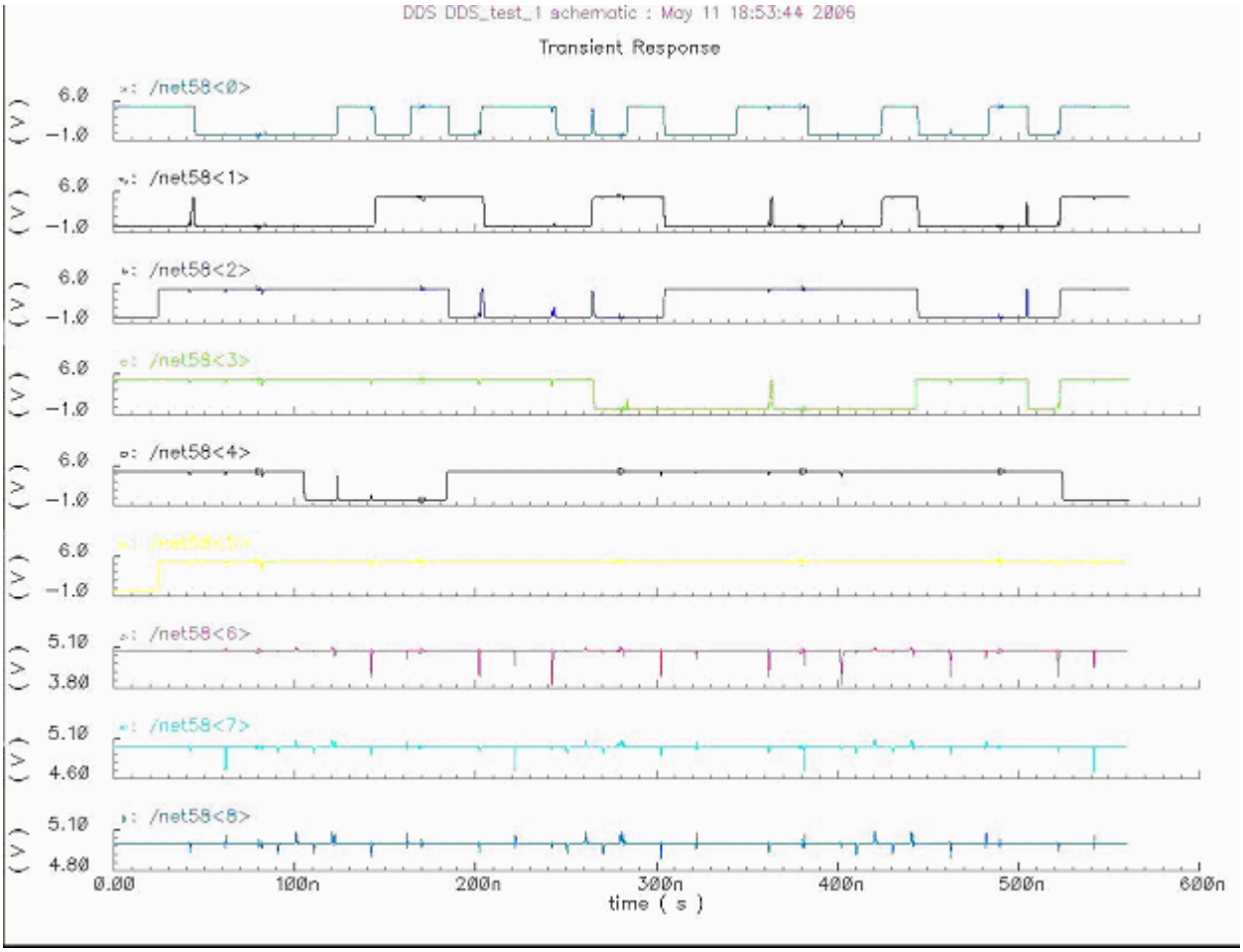


Fig 3.4.2 Output waveform of 64 bit ROM A

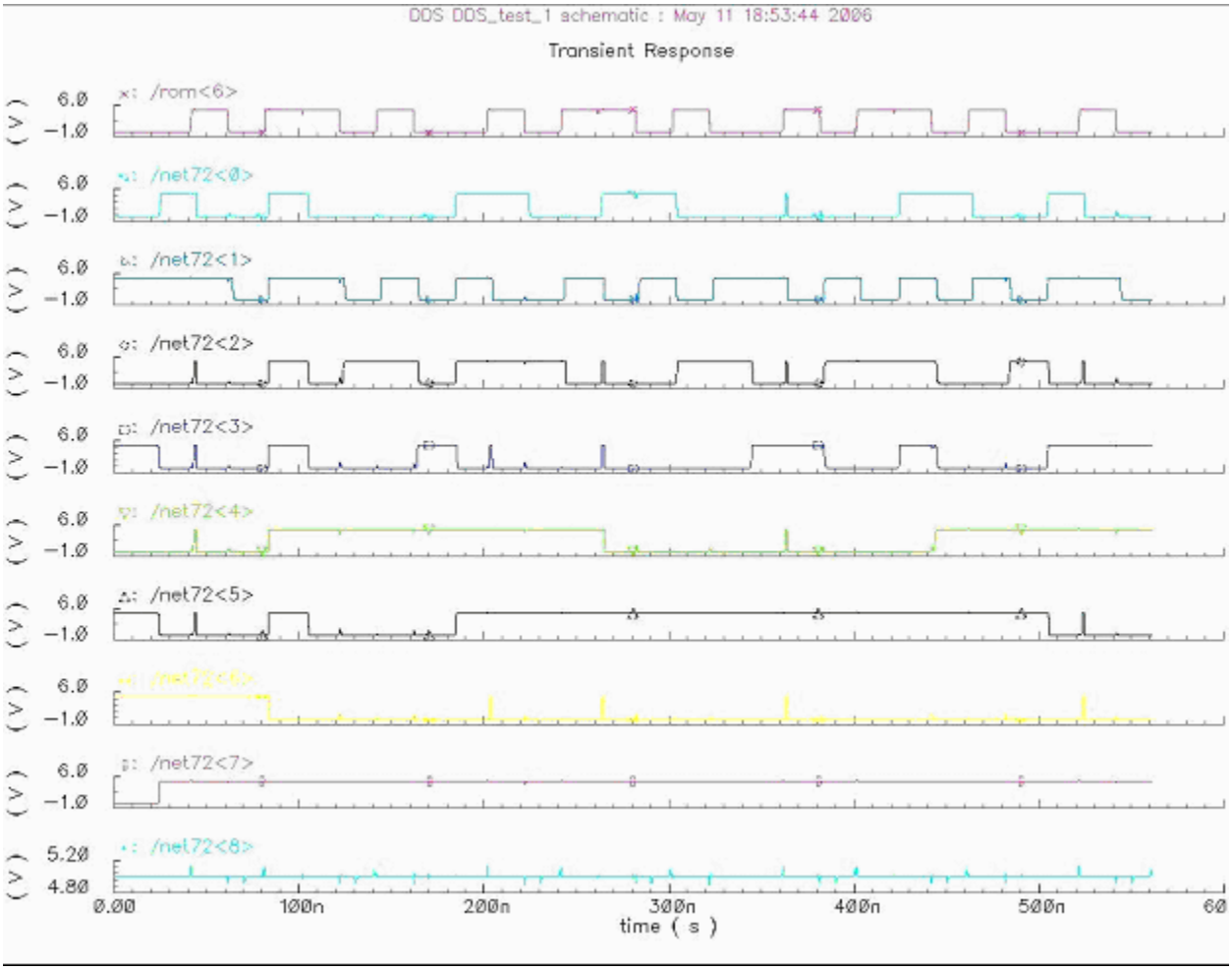


Fig 3.4.3 Output waveform of 64 bit ROM B

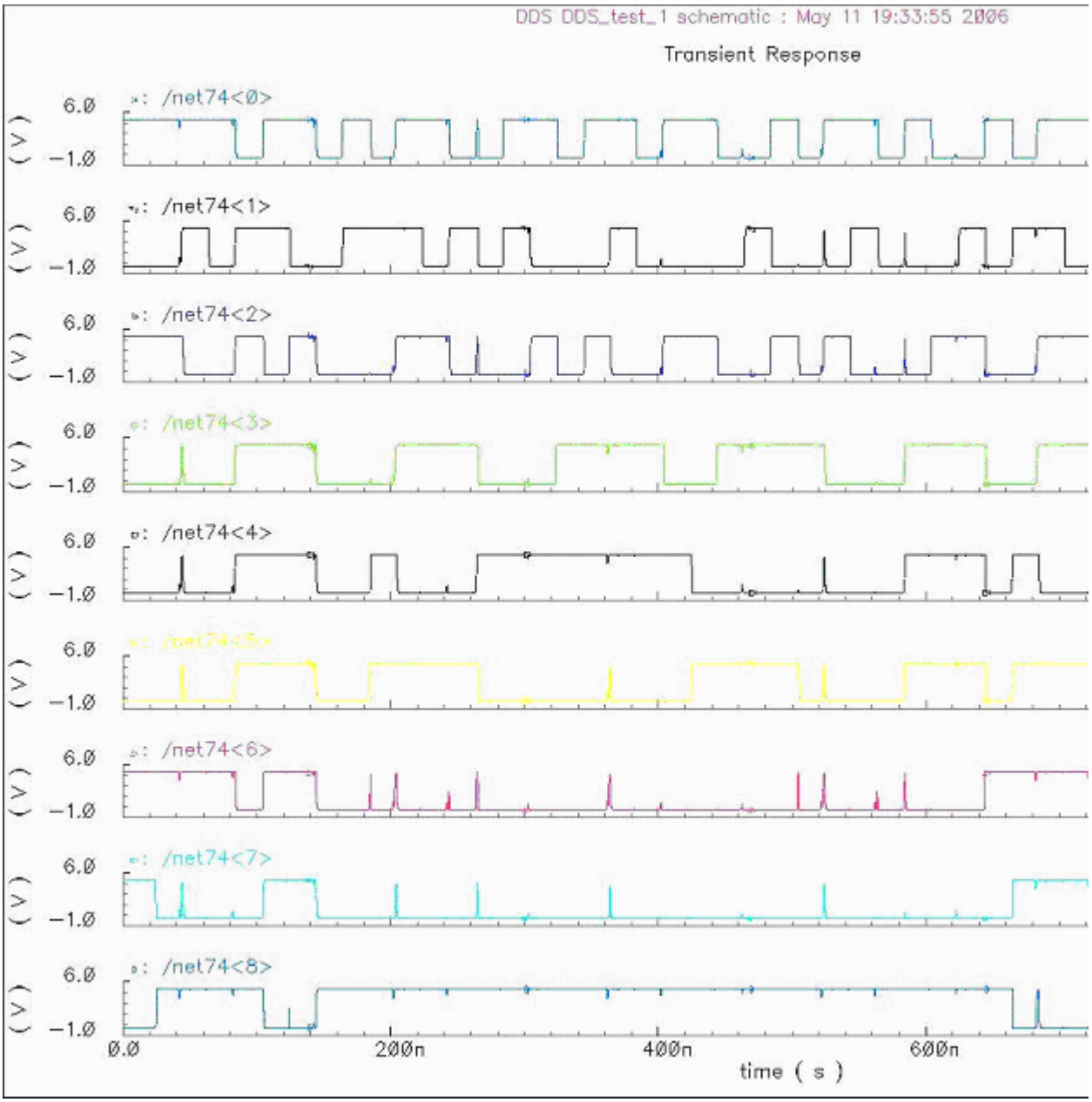


Fig 3.4.4 Output waveform of 64 bit ROM C

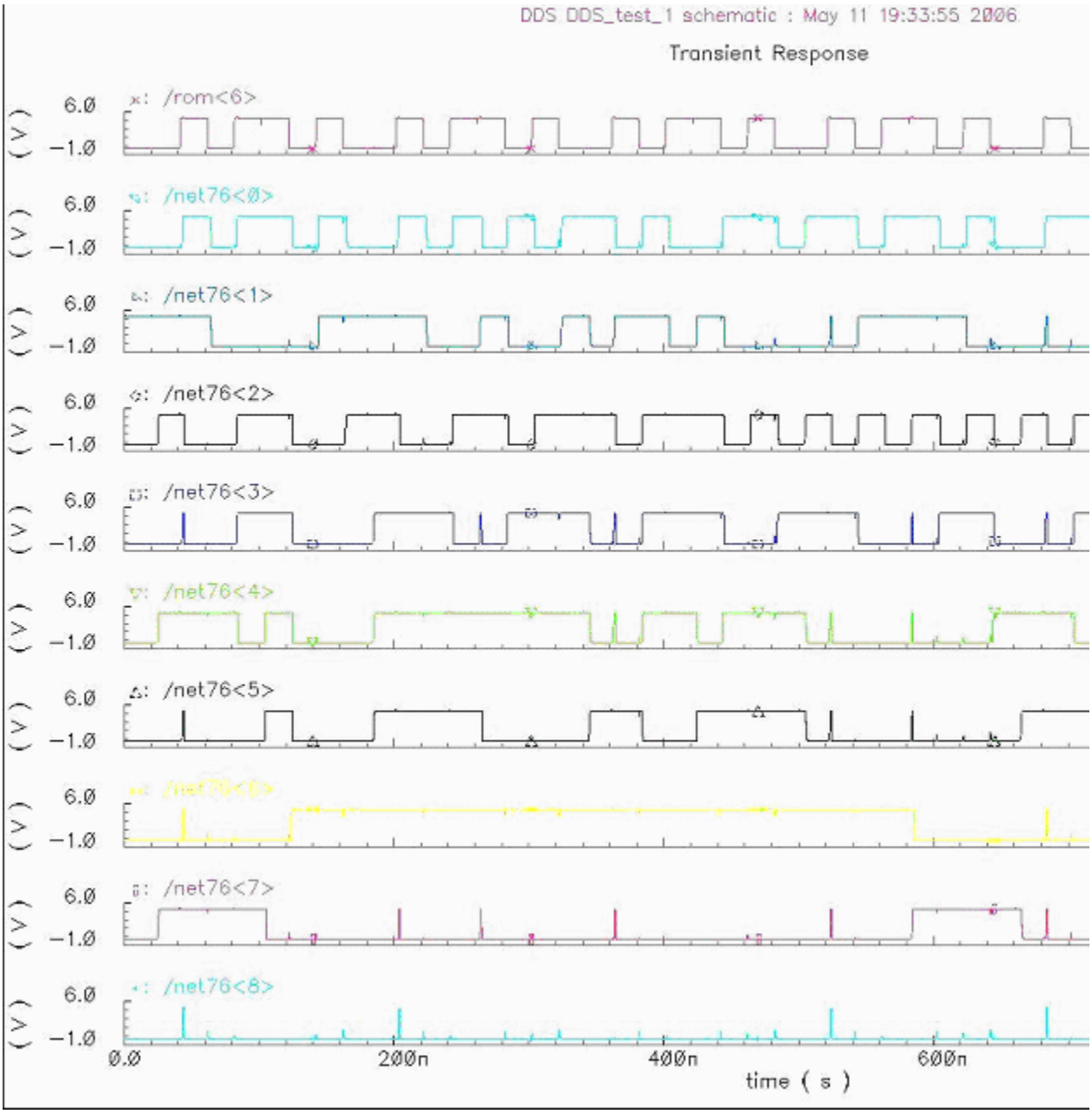


Fig 3.4.5 Output waveform of 64 bit ROM D

3.4 Multiplexers

Fig 3.4.1 shows the overall ROM schematic. The outputs of ROM A and ROM B are given to multiplexer 1 and the in-bit 6 selects the ROM. Similarly the outputs of Rom C and ROM D are given to multiplexer 2 and the in-bit 6 selects the ROM. In bit 7 helps in the final Rom selection. The test results show that the multiplexers perform the correct selection of ROMs. Fig 3.4.2, 3.4.3, 3.4.4 show the output waveforms of multiplexers 1,2 and 3 respectively.

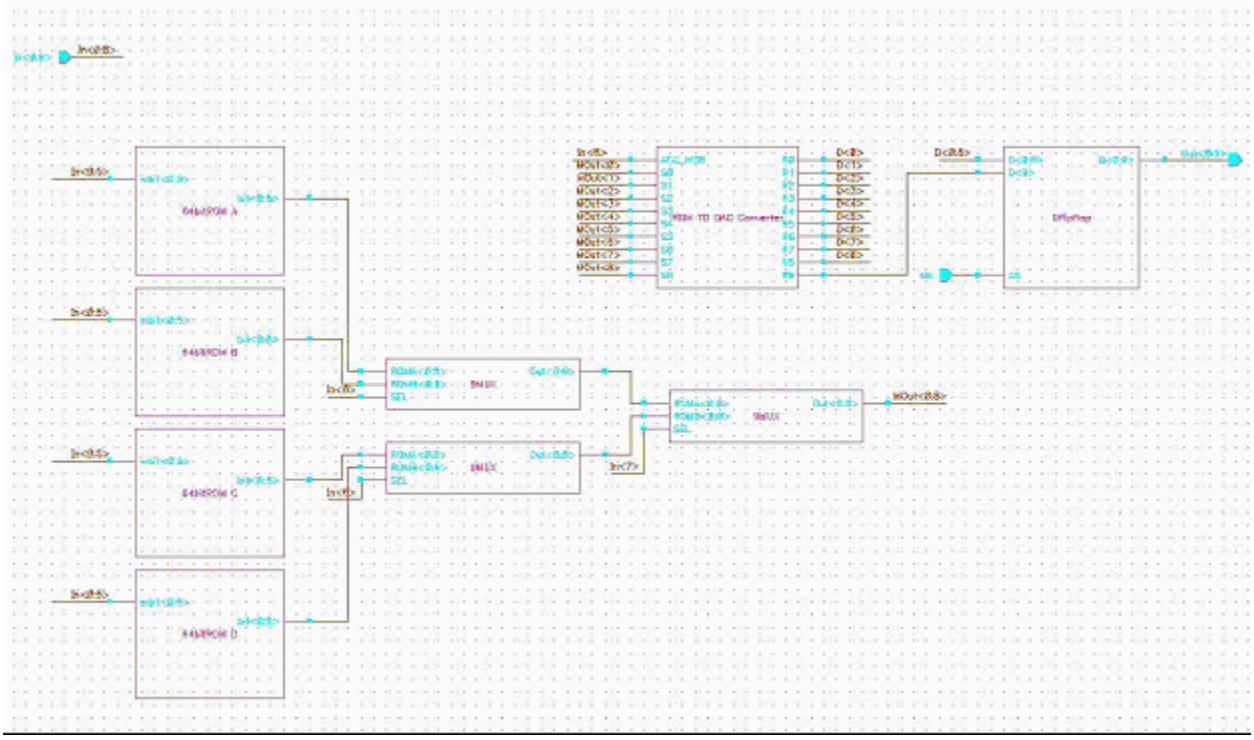


Fig 3.4.6 Overall circuit of 256 ROM

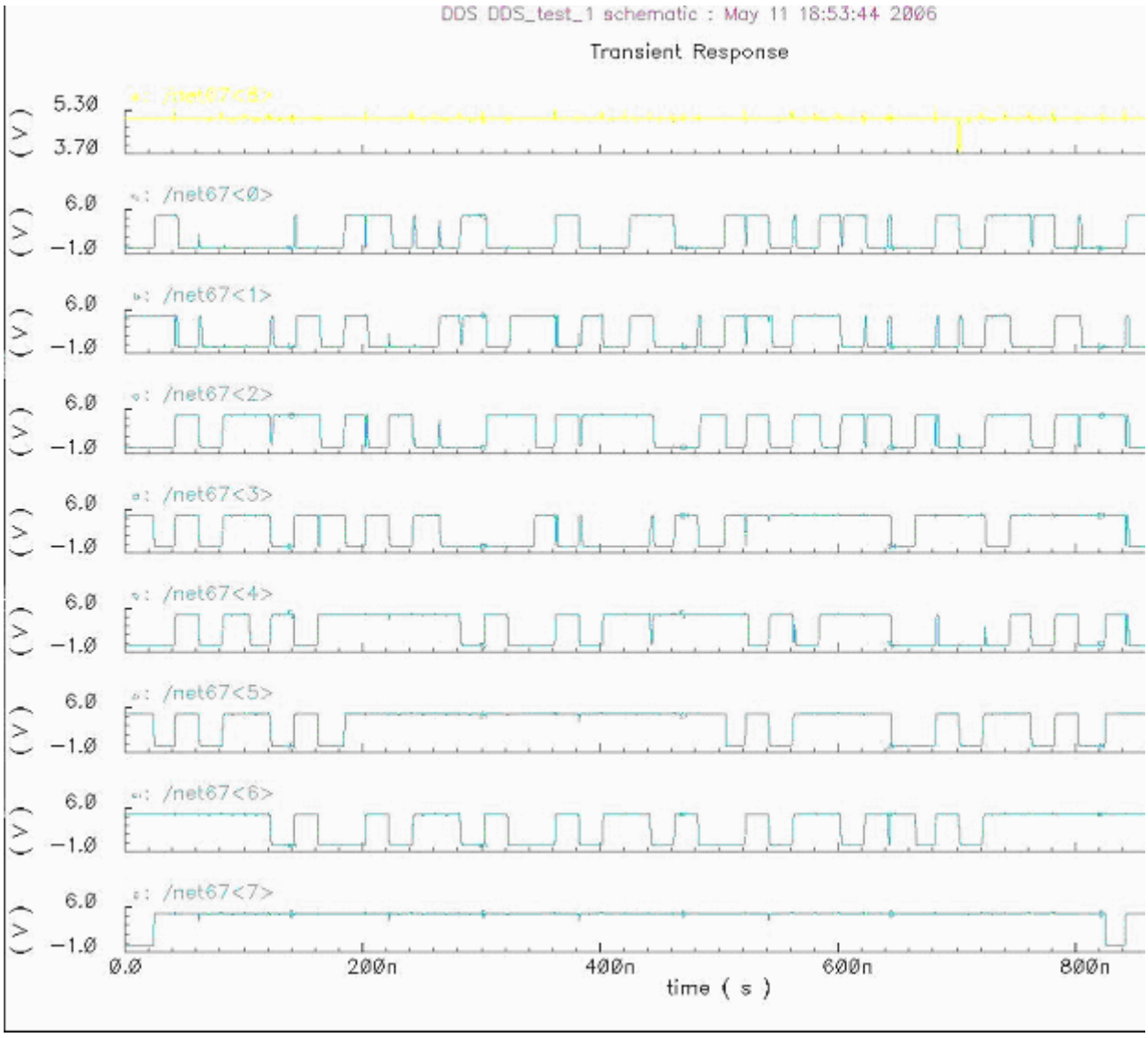


Fig 3.4.7 Output waveform of Multiplexer 1

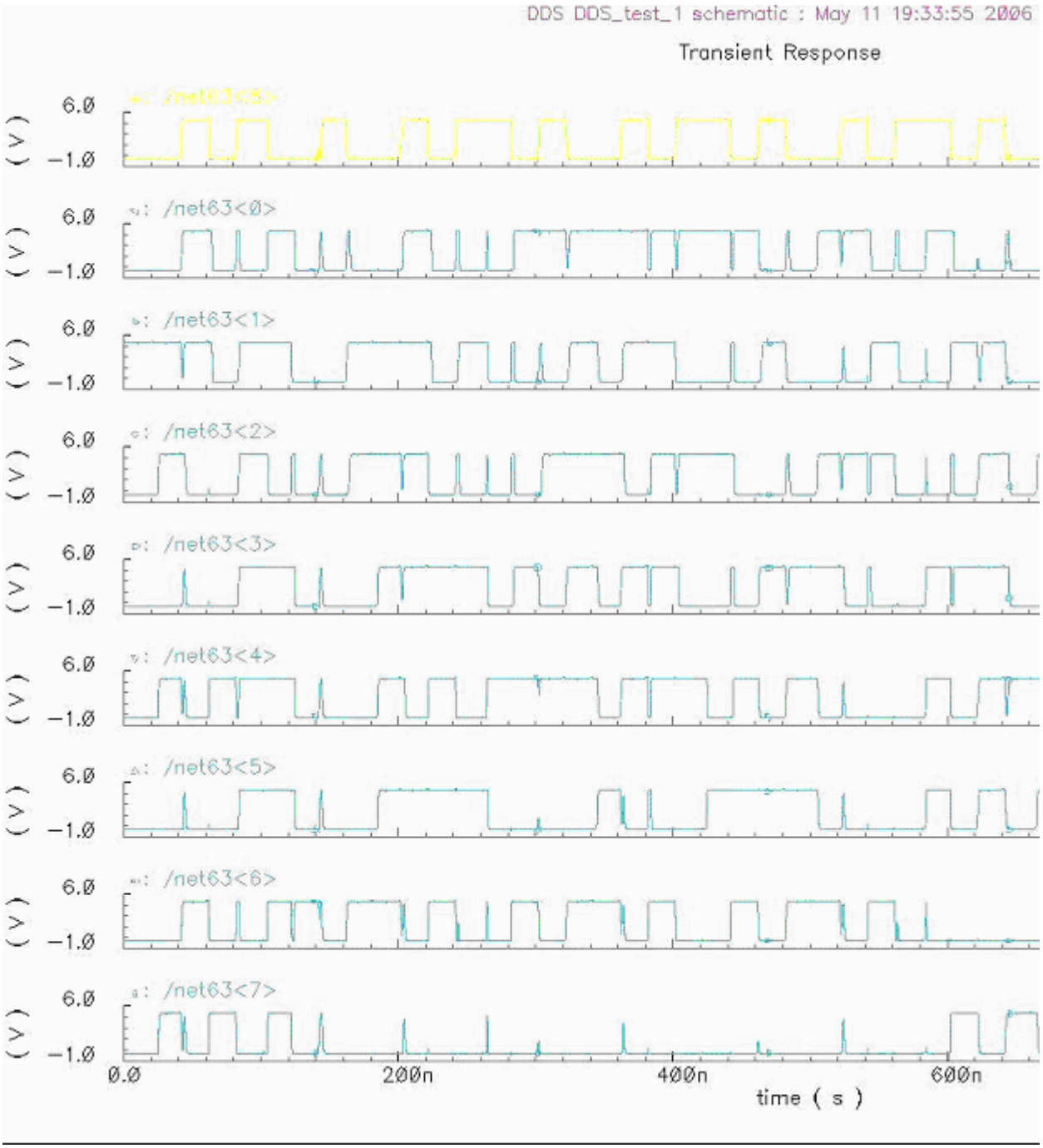


Fig 3.4.8 Output waveform of Multiplexer 2

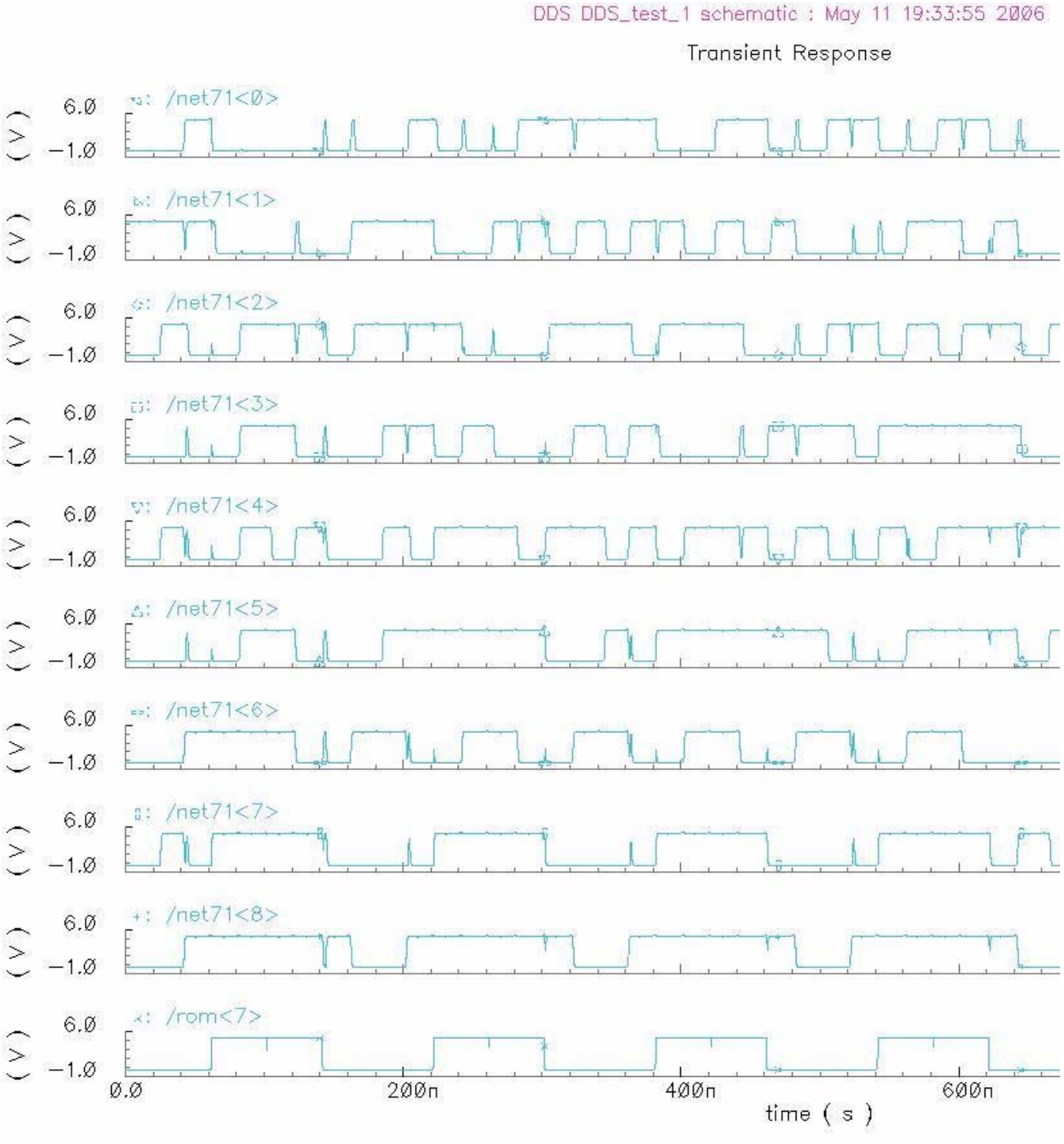


Fig 3.4.9 Output waveform of Multiplexer 3

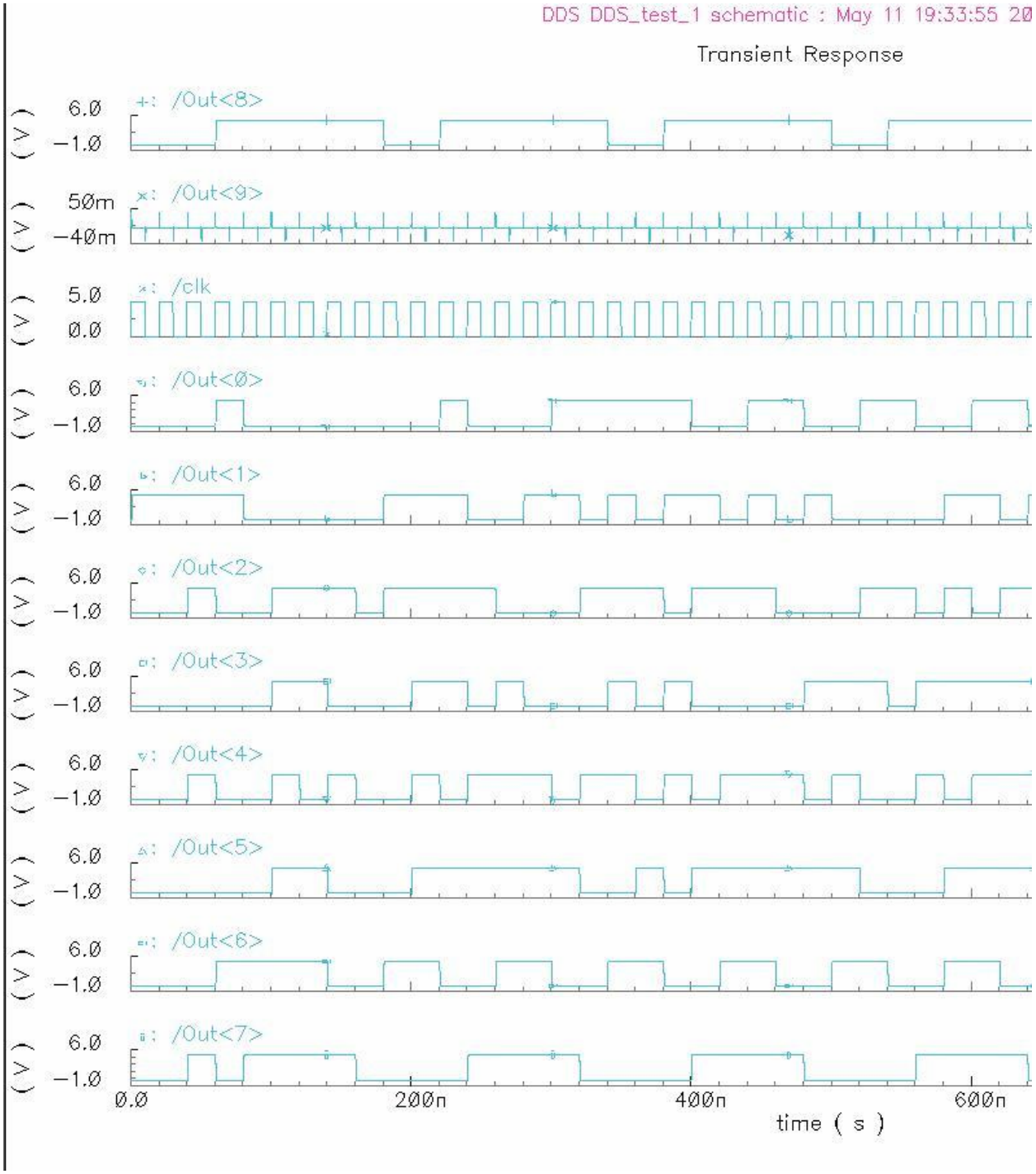


Fig 3.4.10 Latched output waveform of 256 ROM

3.5 Sine Waveform

The overall chip circuit is shown in fig 3.5.1. The latched outputs of the ROM are given to the Digital to analog converter through a ROM to DAC converter. The output waveform was a sine wave but it had some glitches (Fig 3.5.2). When the circuit was tested with D flip-flops after the ROM to DAC converter, we obtained a pure sine waveform (Fig 3.5.3).

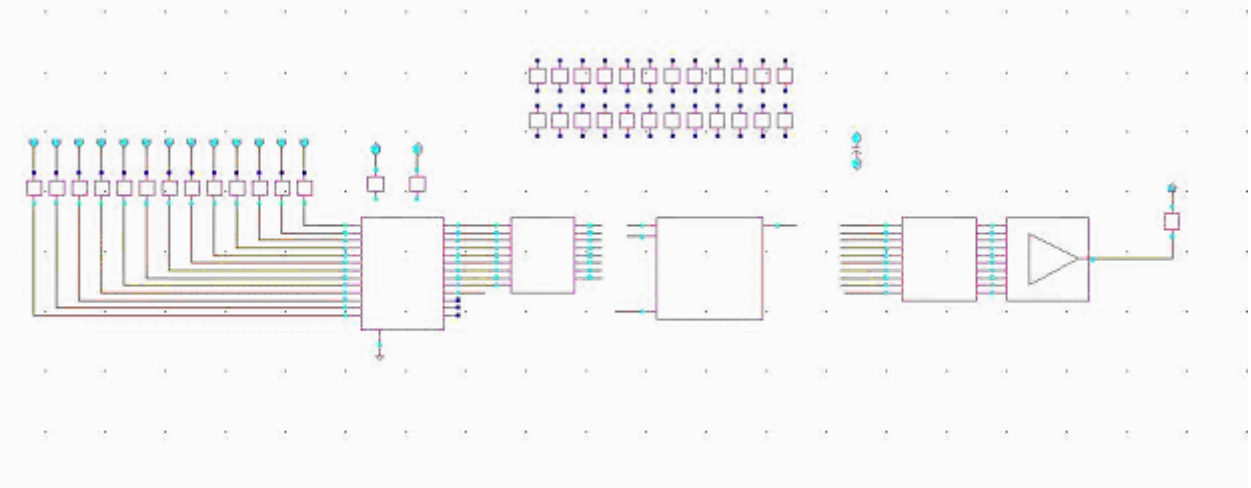


Fig 3.5.1 DDS overall circuit

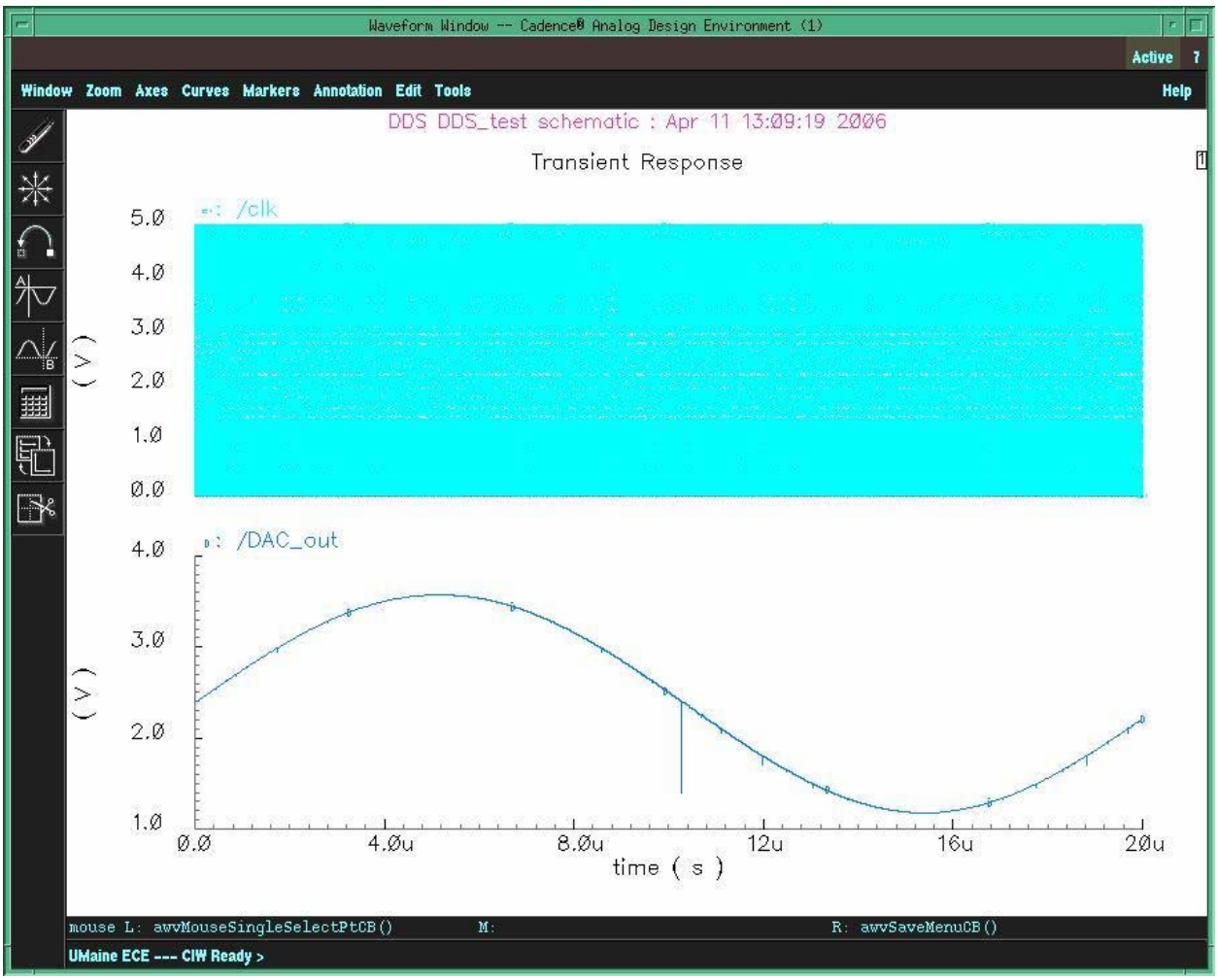


Fig 3.5.2 Sine waveform with glitches

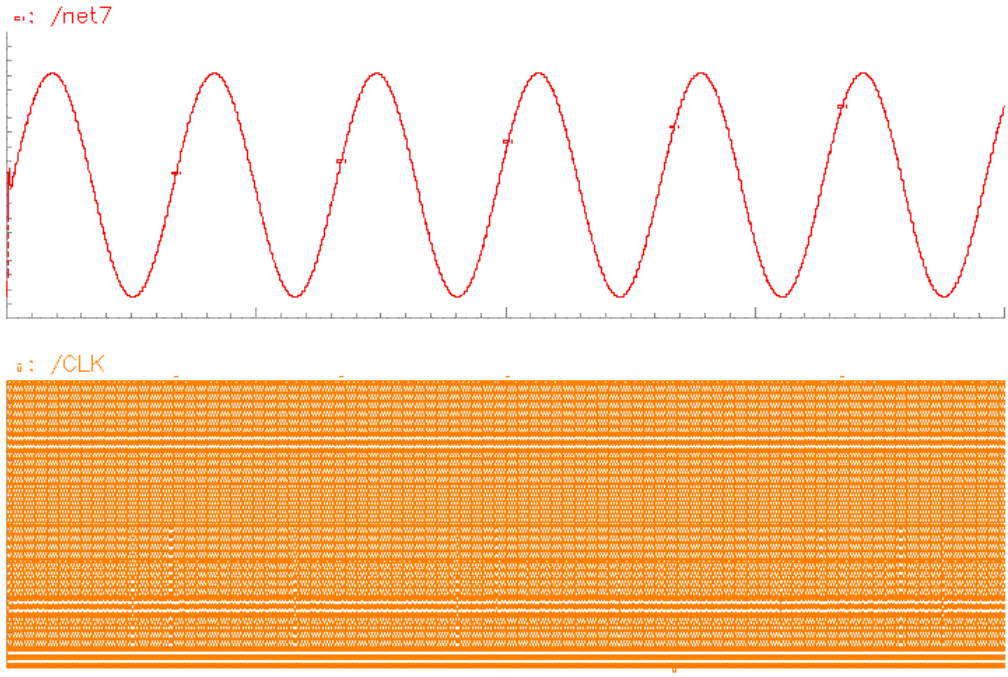


Fig 3.5.3 **Sinewaveform** of DDS circuit

Chapter 4

Physical Design

This chapter discusses the pin assignment, constraints and considerations taken into account while doing a layout and floor planning of the final chip.

4.1 Pin Assignment

The die size that was available for the current design was 0.153” x 0.133” and it was packaged as a 40 pin DIP. The physical design of the final chip and floor planning of the die can be found in Appendix B and Appendix C. Depending on the location some pins have higher parasitic impedance than others which has a negative effect on the signal transmission. Pins were assigned based on the sensitivity of the signals and electrical characteristic of the package used. Table 4.1 contains the electrical characteristic of the 40 pin DIP. For the pin assignment refer to Table 1.4 in Chapter 1. Pins with the lowest parasitic impedance were used for the critical signals, such as output input signals and clock, in order to minimize the noise that may be introduced by the pin trace.

Pin	R (ohm)	L (nH)	C (pF)	t_{of} (ps)
1,20,21,40	0.217	8.18	5.32	209
2,19,22,39	0.177	7.92	4.39	187
3,18,23,38	0.154	7.34	3.37	157
4,17,24,37	0.110	6.48	2.34	1213
5,16,25,36	0.103	5.69	2.16	111
6,15,26,35	0.0661	4.37	1.43	79.0
7,14,27,34	0.0646	4.54	1.48	81.9
8,13,28,33	0.0498	3.69	1.05	62.3
9,12,29,32	0.0378	3.54	0.863	55.3
10,11,30,31	0.0247	3.15	0.660	45.6

Table 4.1 Electrical Characteristic of 40 Pin DIP

4.2 Floorplanning Issues

The ROM occupies the major part of the DDS chip because of the numerous components. Since the ROM acts like the middle component, receiving its input from the accumulator and giving its output to the Digital to Analog converter, it was best to do the layout at the bottom rightmost corner of the chip. Metal 1 was used for all the VDD and GND connections and Metal 3 was used for the input, output and interconnections.

4.3 Component Layout and Outer Guard Ring

The overall chip size (layout area) was 900 x 900 microns of which the ROM alone measures 800 x 497 microns. In general, it was a goal to have the guard rings around cells and ample power connections. The layout of all components share some general characteristics like to reduce the cell size, minimize the noise and introduce large number of contacts. Multipliers and fingers were used wherever appropriate to help reduce the cell size. Ground (GND) and VDD rings were implemented to minimize the noise coupling between components. Also, to satisfy the design rules additional poly had to be added.

Chapter 5

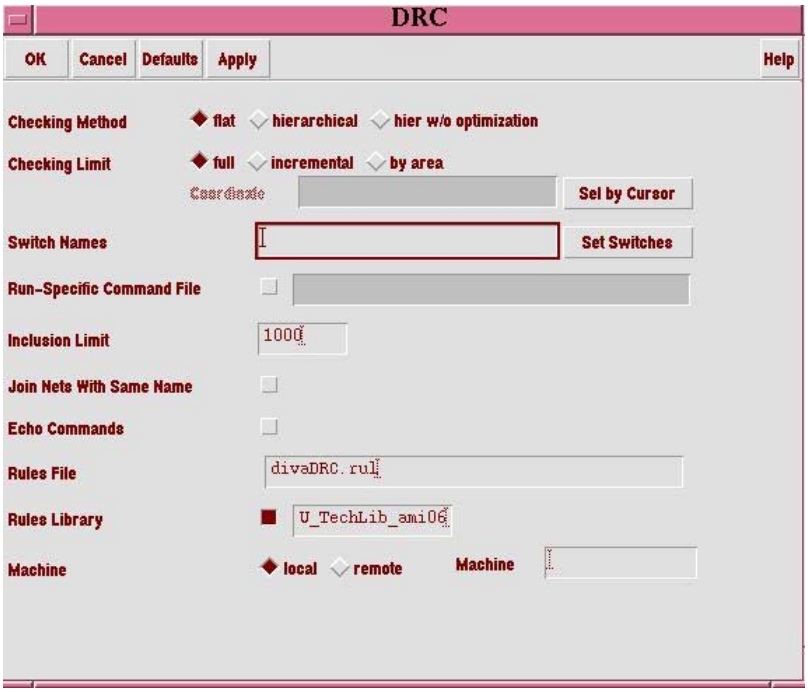
Design Verification

5.1 Introduction

Physical design verification can be accomplished with two software checks: *Design Rule Checking (DRC)* and *Layout versus Schematics (LVS).* DRC verifies that the layout does not violate a manufacturing rules and LVS verifies that the physical layout logically matches the schematic in the number of terminals, components, component sizes and interconnection. After the DRC is done for the layout, it is extracted and then succeeded by LVS. After design and layout verification, the analog extracted schematics could be created to verify the circuit performance with parasitics included.

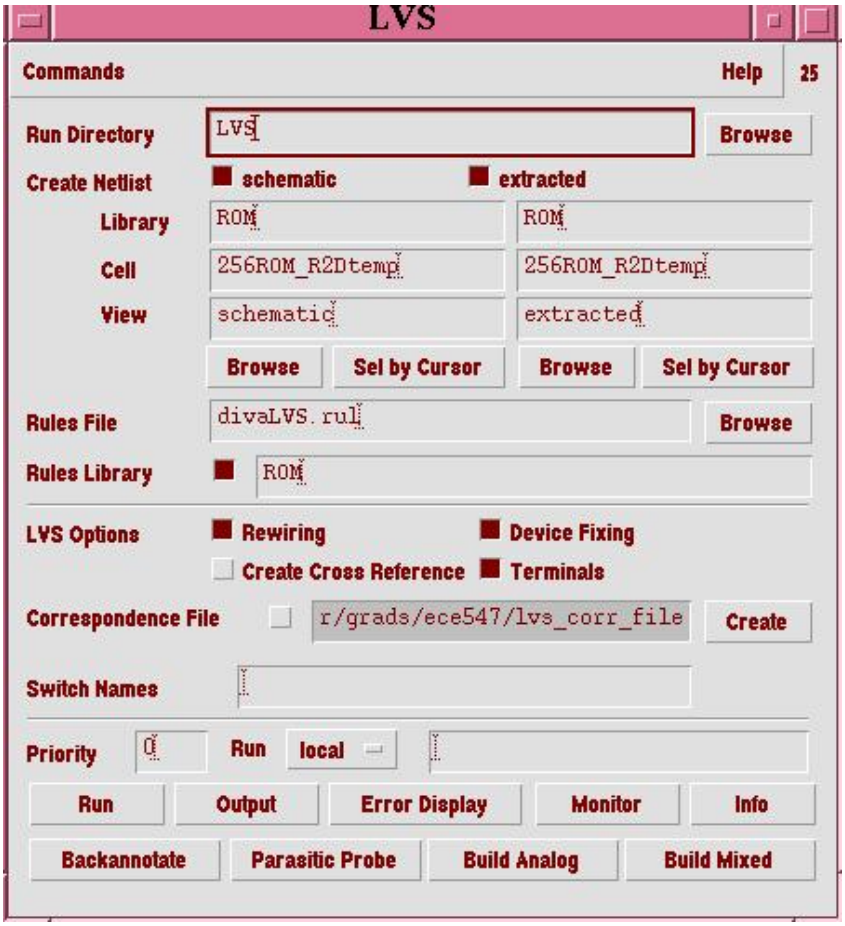
5.2 Design Rule Verification (DRC)

DRC and *LVS* were performed on the individual components before running verification on the circuit level and finally on the top-level layout (complete chip).
Top-level layout was *DRC* clean, meaning there were no errors. DRC could be performed in an incremental or full manner.



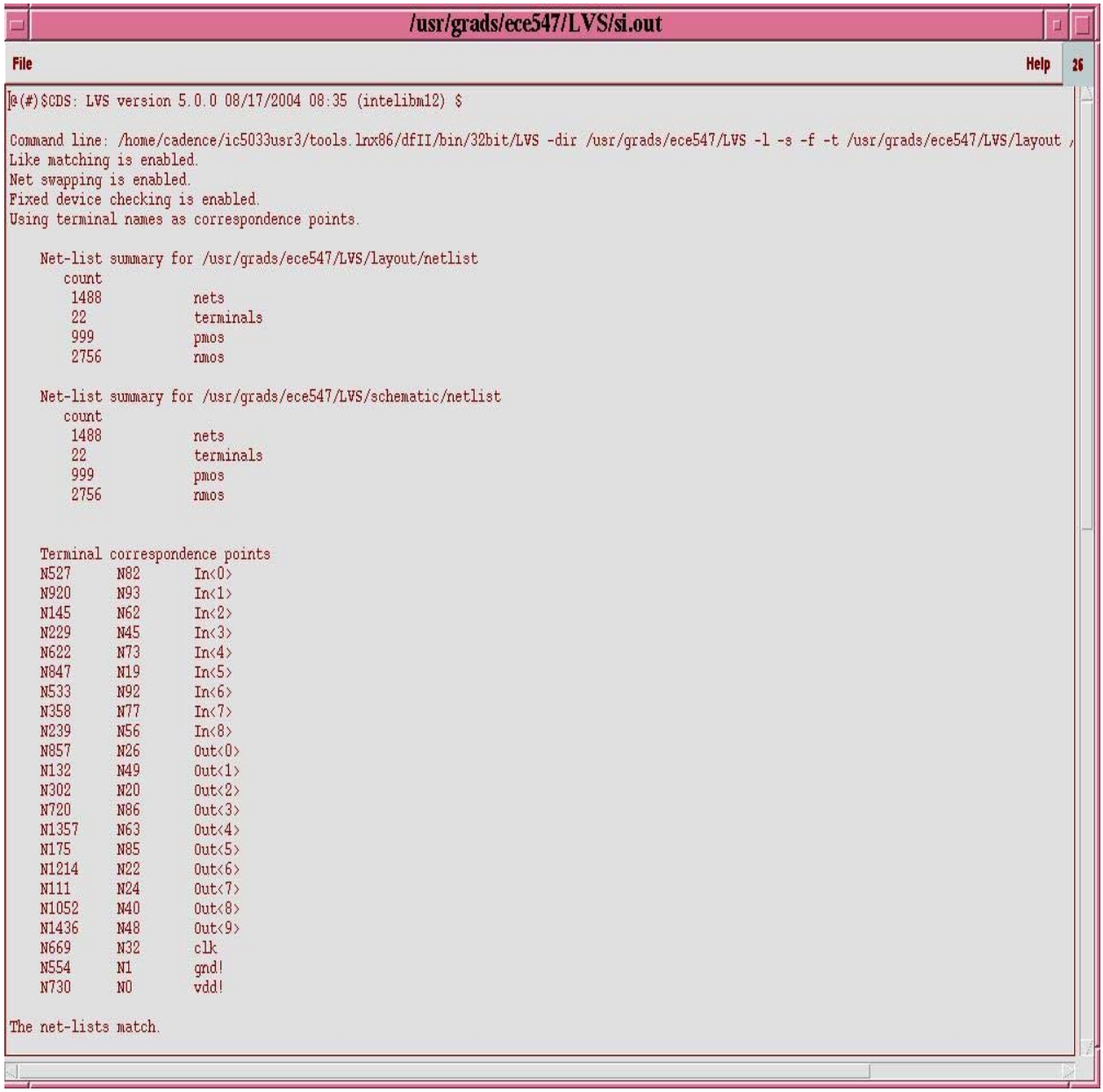
5.3 Layout versus Schematics Check (LVS)

LVS verifies that the physical layout logically matches schematics. As mentioned before, *LVS* verification was performed on the individual components before running top-level layout verification.



Once the LVS has succeeded the output could be seen from the Ouput button else we can open the Error display button.

The Top- level layout of the circuit design does pass LVS.


/usr/grads/ece547/LVS/si.out
File
Help
26
@(#)$CDS: LVS version 5.0.0 08/17/2004 08:35 (intelibm12) $
Command line: /home/cadence/ic5033usr3/tools.lnx86/dfII/bin/32bit/LVS -dir /usr/grads/ece547/LVS -l -s -f -t /usr/grads/ece547/LVS/layout /
Like matching is enabled.
Net swapping is enabled.
Fixed device checking is enabled.
Using terminal names as correspondence points.
Net-list summary for /usr/grads/ece547/LVS/layout/netlist
count
1488 nets
22 terminals
999 pmos
2756 nmos
Net-list summary for /usr/grads/ece547/LVS/schematic/netlist
count
1488 nets
22 terminals
999 pmos
2756 nmos
Terminal correspondence points
N527 N82 In<0>
N920 N93 In<1>
N145 N62 In<2>
N229 N45 In<3>
N622 N73 In<4>
N847 N19 In<5>
N533 N92 In<6>
N358 N77 In<7>
N239 N56 In<8>
N857 N26 Out<0>
N132 N49 Out<1>
N302 N20 Out<2>
N720 N86 Out<3>
N1357 N63 Out<4>
N175 N85 Out<5>
N1214 N22 Out<6>
N111 N24 Out<7>
N1052 N40 Out<8>
N1436 N48 Out<9>
N669 N32 clk
N554 N1 gnd!
N730 N0 vdd!
The net-lists match.

File

```
    N669      N32       clk
    N554      N1        gnd!
    N730      N0        vdd!

The net-lists match.

                             layout  schematic
                                instances
        un-matched              0       0
        rewired                 0       0
        size errors             0       0
        pruned                  0       0
        active                  3755    3755
        total                   3755    3755

                                  nets
        un-matched              0       0
        merged                  0       0
        pruned                  0       0
        active                  1488    1488
        total                   1488    1488

                                terminals
        un-matched              0       0
        matched but
        different type          0       0
        total                   22      22


Probe files from /usr/grads/ece547/LVS/schematic

devbad.out:

netbad.out:

mergenet.out:

termbad.out:

prunenet.out:

prunedev.out:

audit.out:


Probe files from /usr/grads/ece547/LVS/layout

devbad.out:

netbad.out:

mergenet.out:

termbad.out:

prunenet.out:

prunedev.out:

audit.out:
```

Chapter 6

Conclusion

Thus the 256 bit ROM was designed and tested. The separation of 256 ROM into four 64 bit ROMs saved a lot of space and also reduced capacitance. The tree decoder enabled the selection of only one of the 64 rows. The multiplexers were first designed with NAND gates and it was found that there was better performance using Transmission gates. The D latches at the output of the ROM enabled us to obtain an edge triggered output. From the simulation it was found that placing the ROM to DAC converter after the D latches produced a sine waveform with glitches. However, the glitches could be removed when the output of the ROM to DAC converter is latched. The designed ROM met all the necessary specifications: 256 rows, 9 columns, 50 MHz frequency and a voltage range of 0 to 5 Volts. The overall layout of the DDS also met the size requirements. The entire schematic, symbol and layout of the chip are in APPENDIX C.

Acknowledgements

I would like to express my heartfelt gratitude to my course instructor, Prof.David Kotecki for his constant support and guidance in completion of this paper. I thank my advisors Prof.Rosemary Smith and Prof.Scott Collins for their invaluable assistance in all my endeavors. I would also like to thank my loving parents, my sister and all my friends for their precious support and constant encouragement.

Bibliography

[1] http://www.analog.com/library/analogDialogue/archives/30-3/single_chip.html

[2] "Class D Audio Amplifier Front End circuit" Wayne Slade, Steven Turner, 2002

[3] CMOS circuits, Jacob Baker

[4] http://www.eecg.toronto.edu/~roman/teaching/1388/2004/finalProj/2004_ECE1388_FP_www/Quadrature_Direct_Digital_Synthesizer/index.html

APPENDIX A

SCHEMATICS

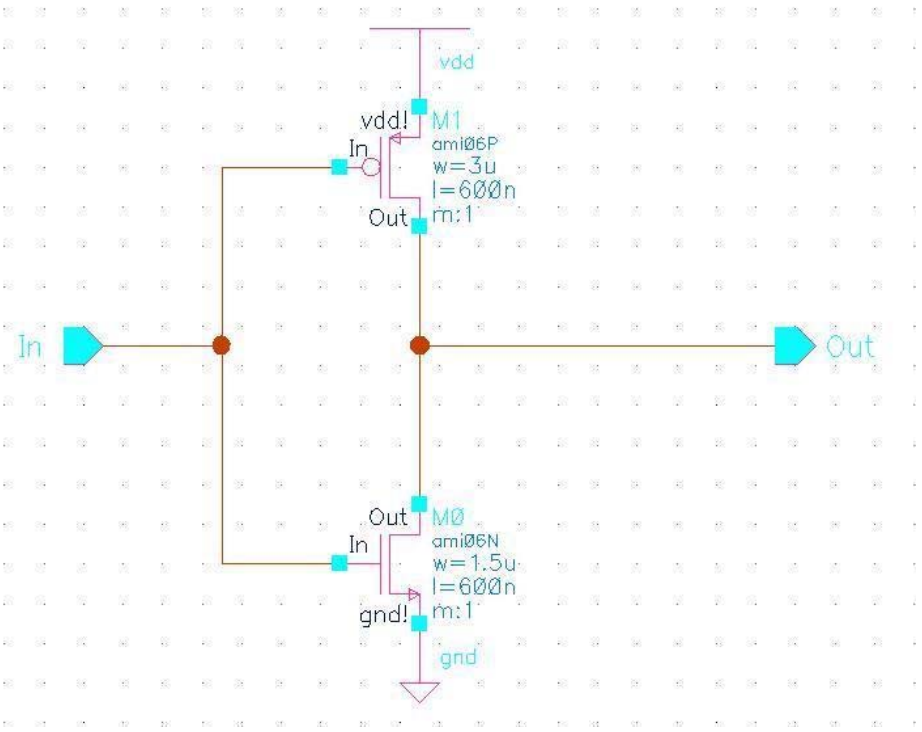


Fig A.1 Inverter Schematic

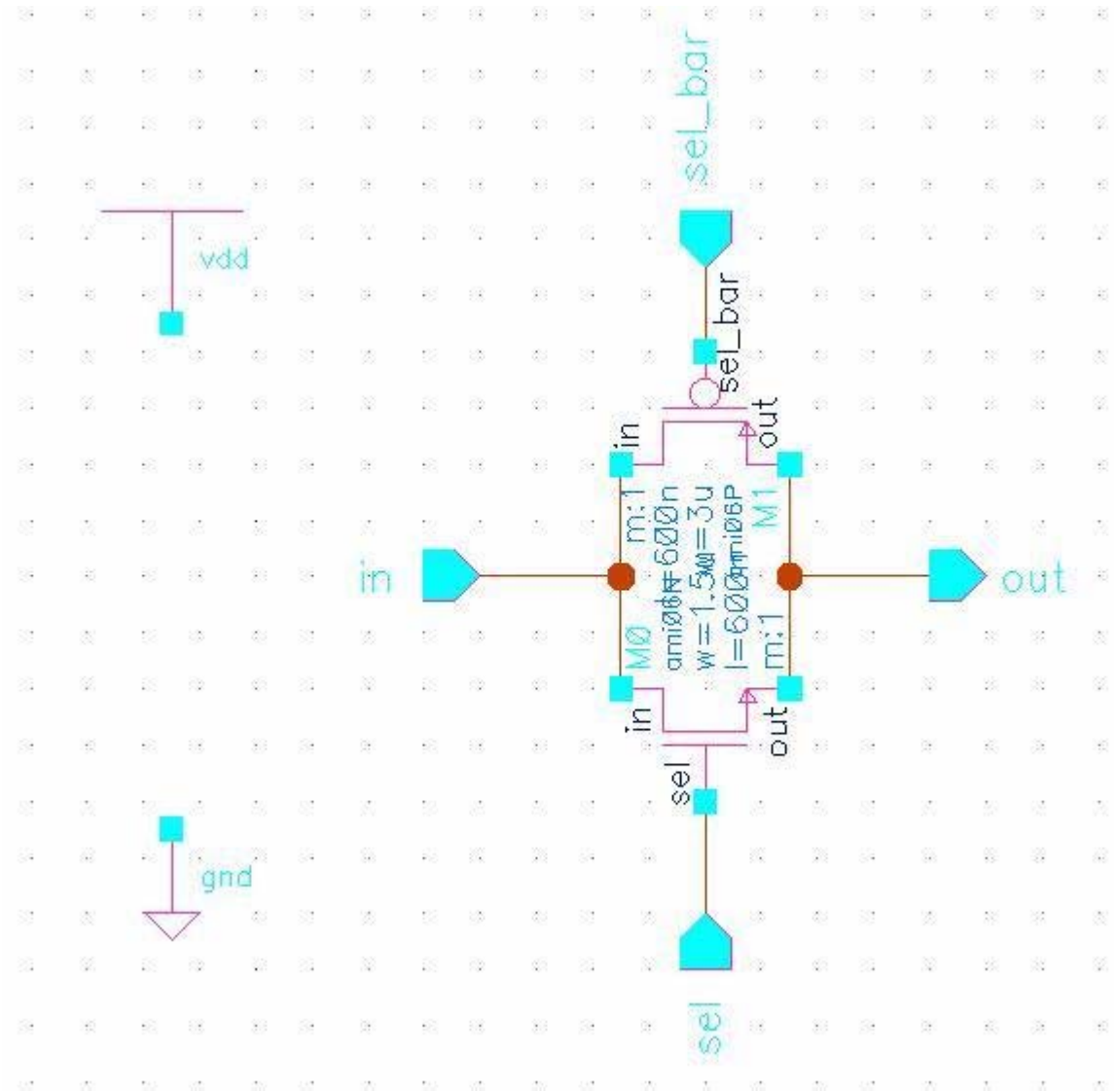


Fig A.2 Transmission Gate

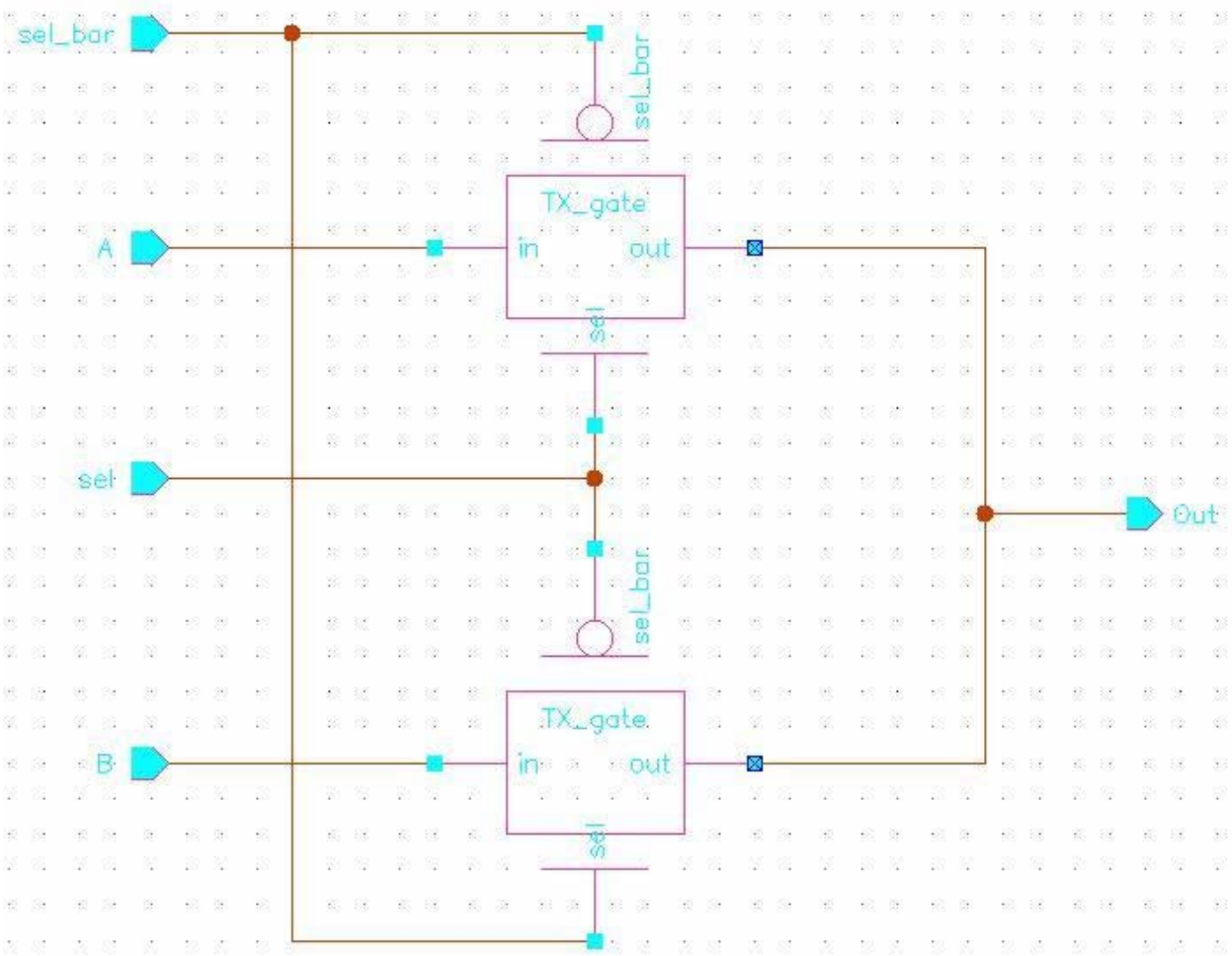


Fig A.3 Transmission Gate

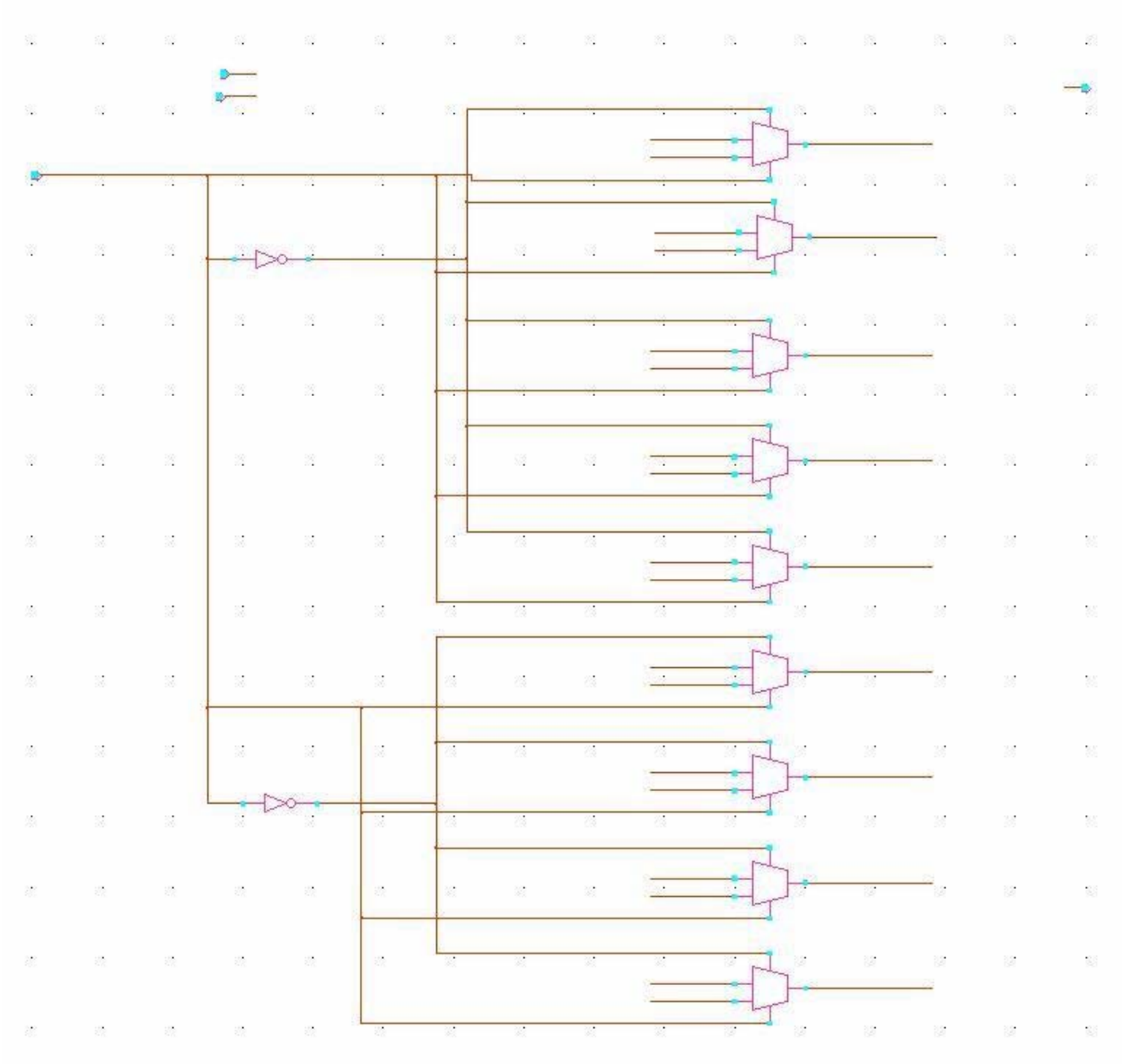


Fig A.4 9mux_2to1

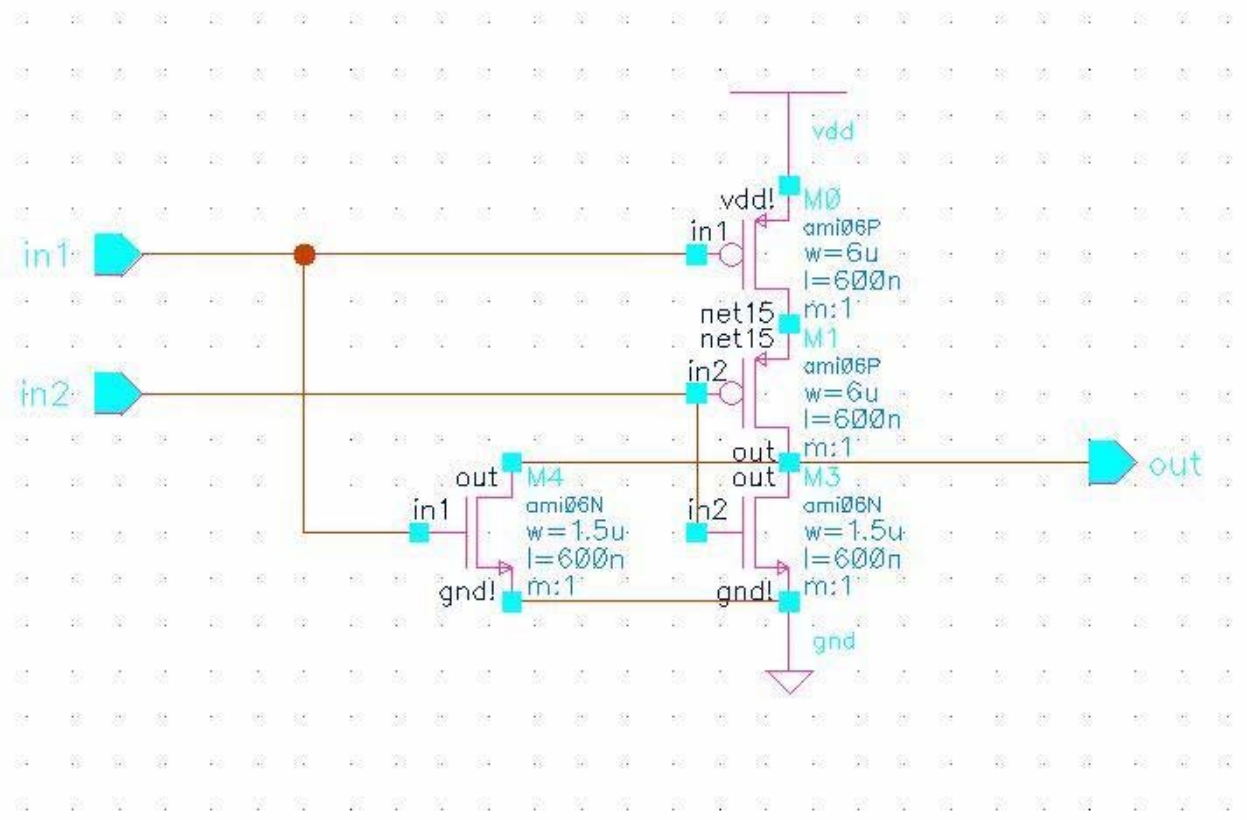


Fig A.5 NOR gate

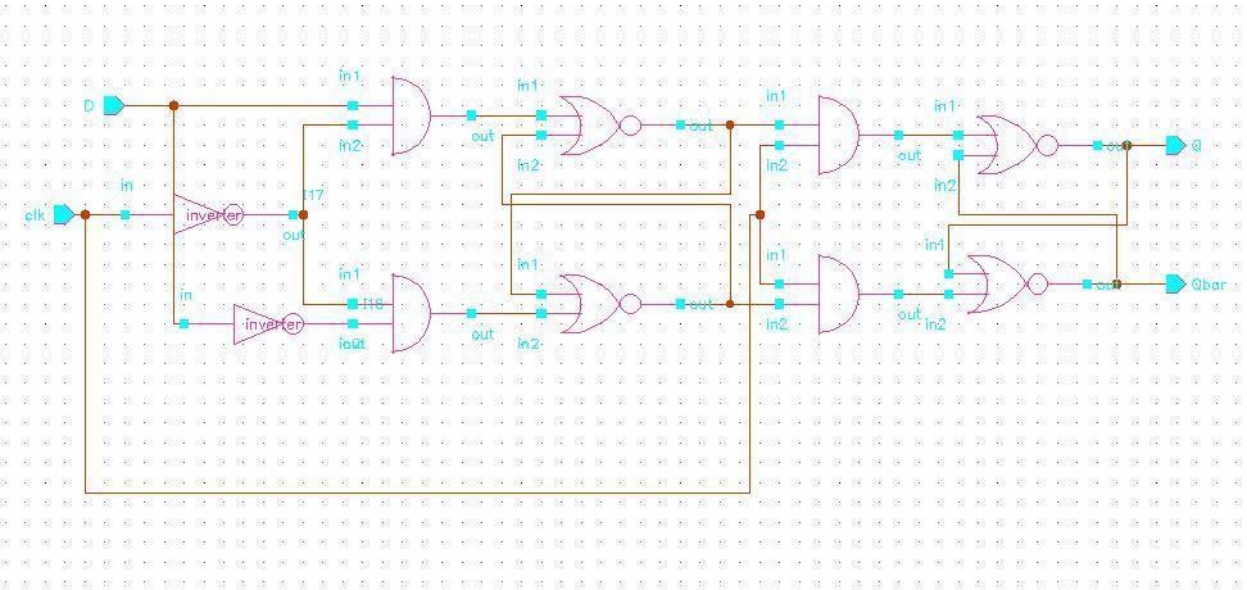


Fig A.6 DFF

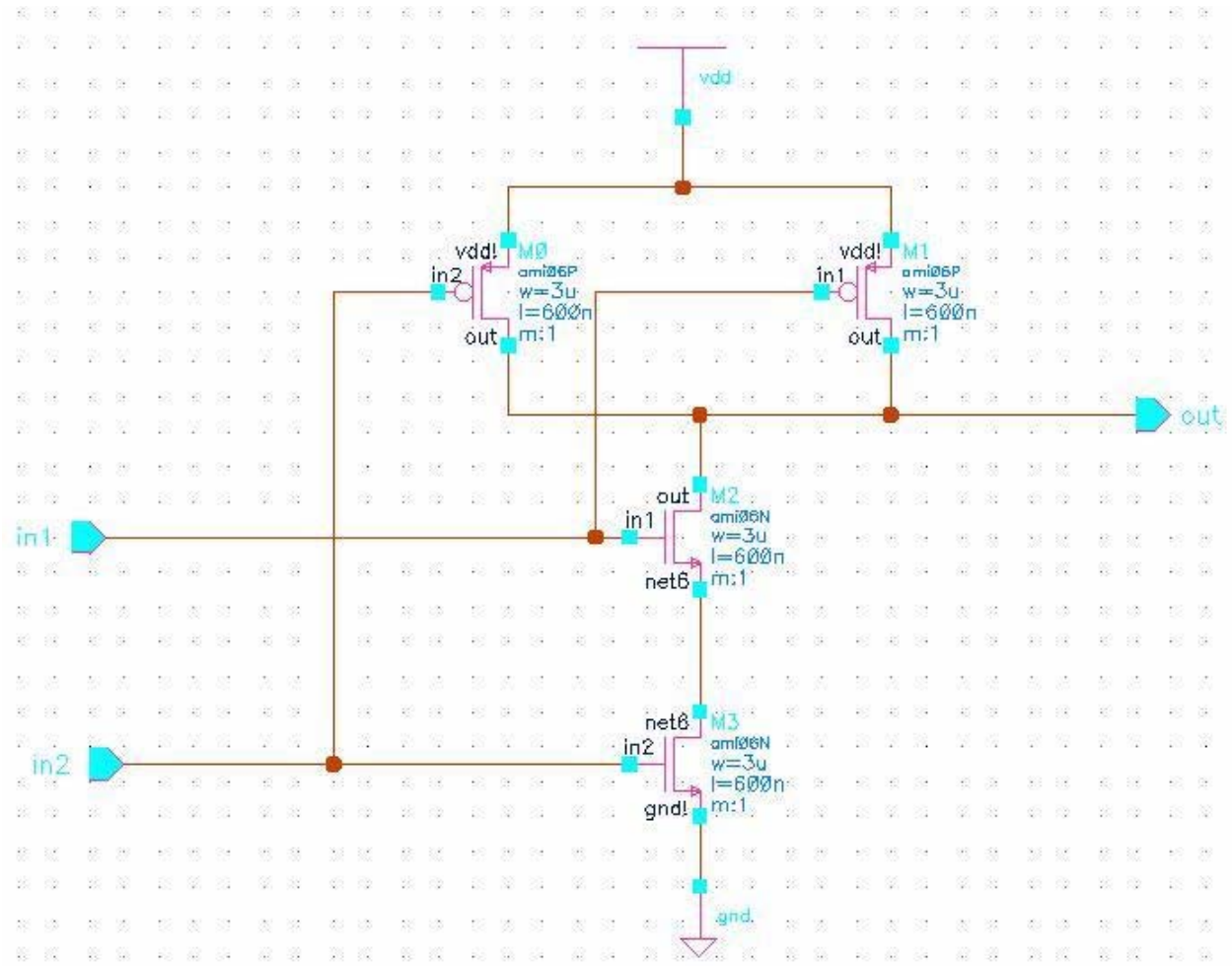


Fig A.7 NAND gate

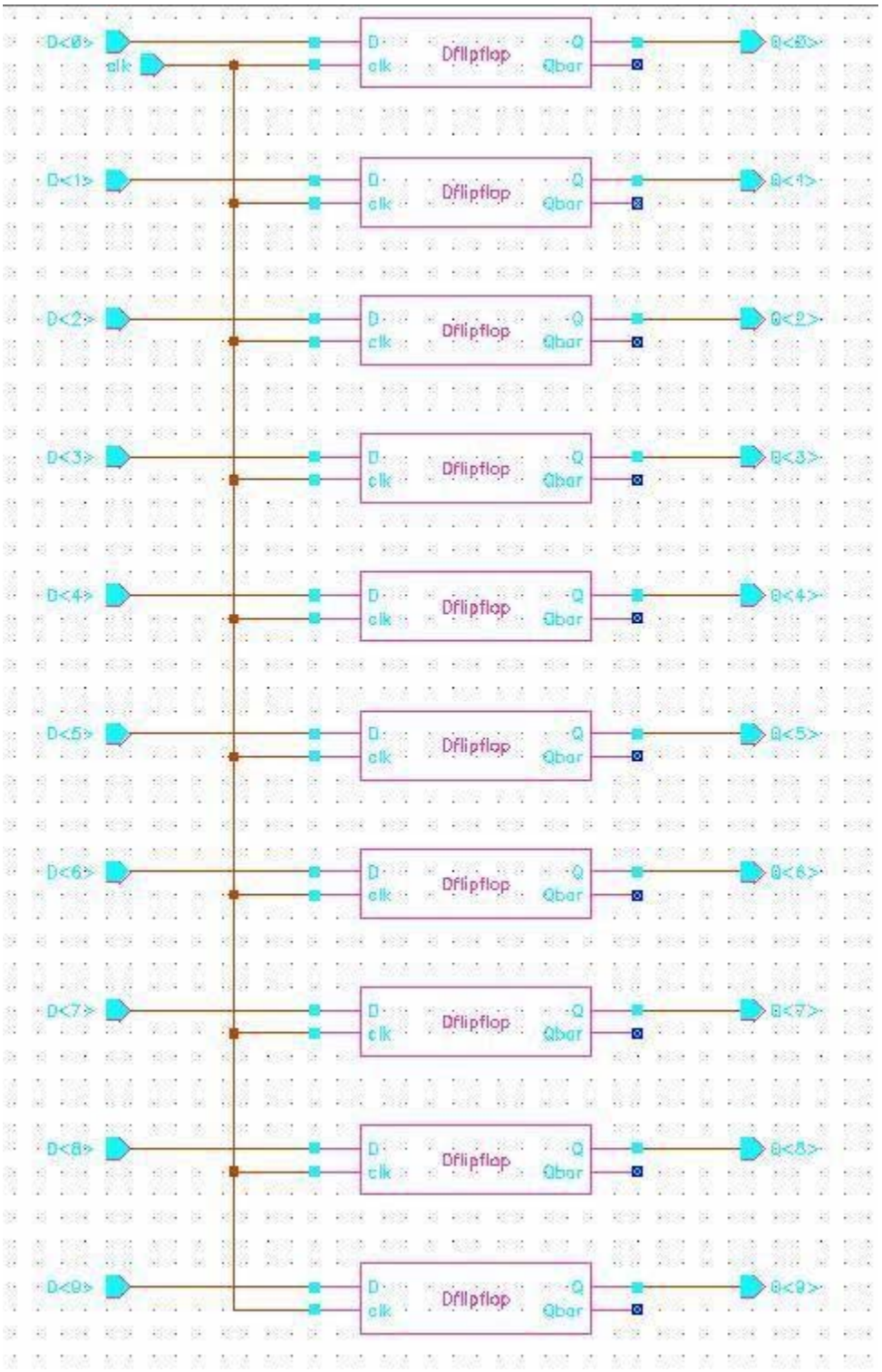

D<0>
clk
D
clk
Dflipflop
Q
Qbar
Q<0>
D<1>
Q<1>
D<2>
Q<2>
D<3>
Q<3>
D<4>
Q<4>
D<5>
Q<5>
D<6>
Q<6>
D<7>
Q<7>
D<8>
Q<8>
D<9>
Q<9>

Fig A.8 10 DFF

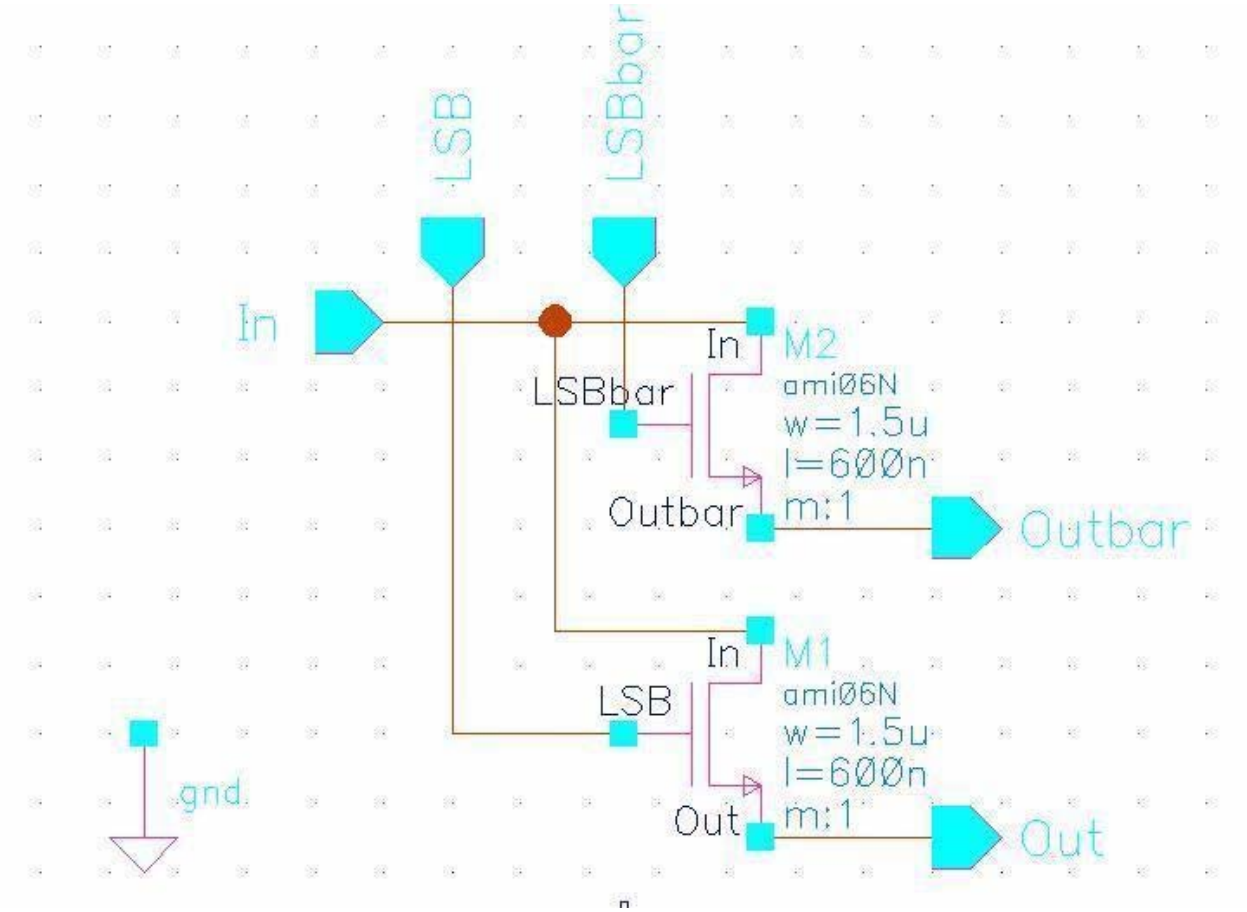


Fig A.9 Tree first schematic

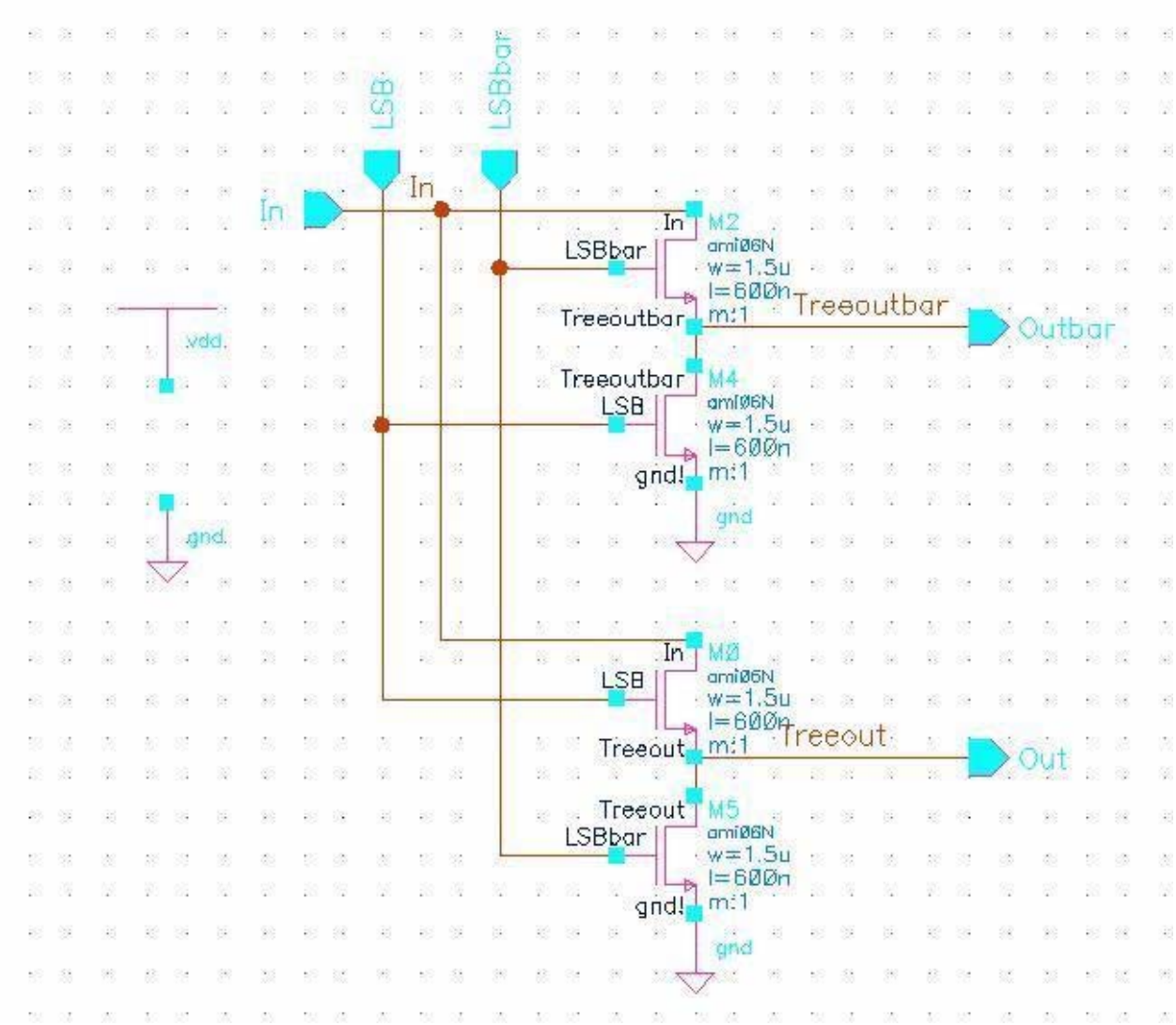


Fig A.10 Tree 4 th schematic

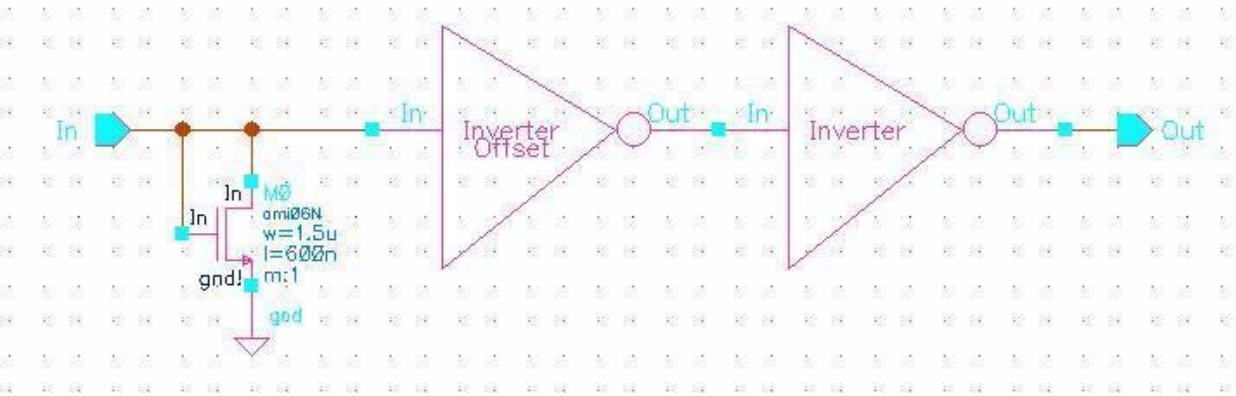


Fig A.11 Tree line schematic

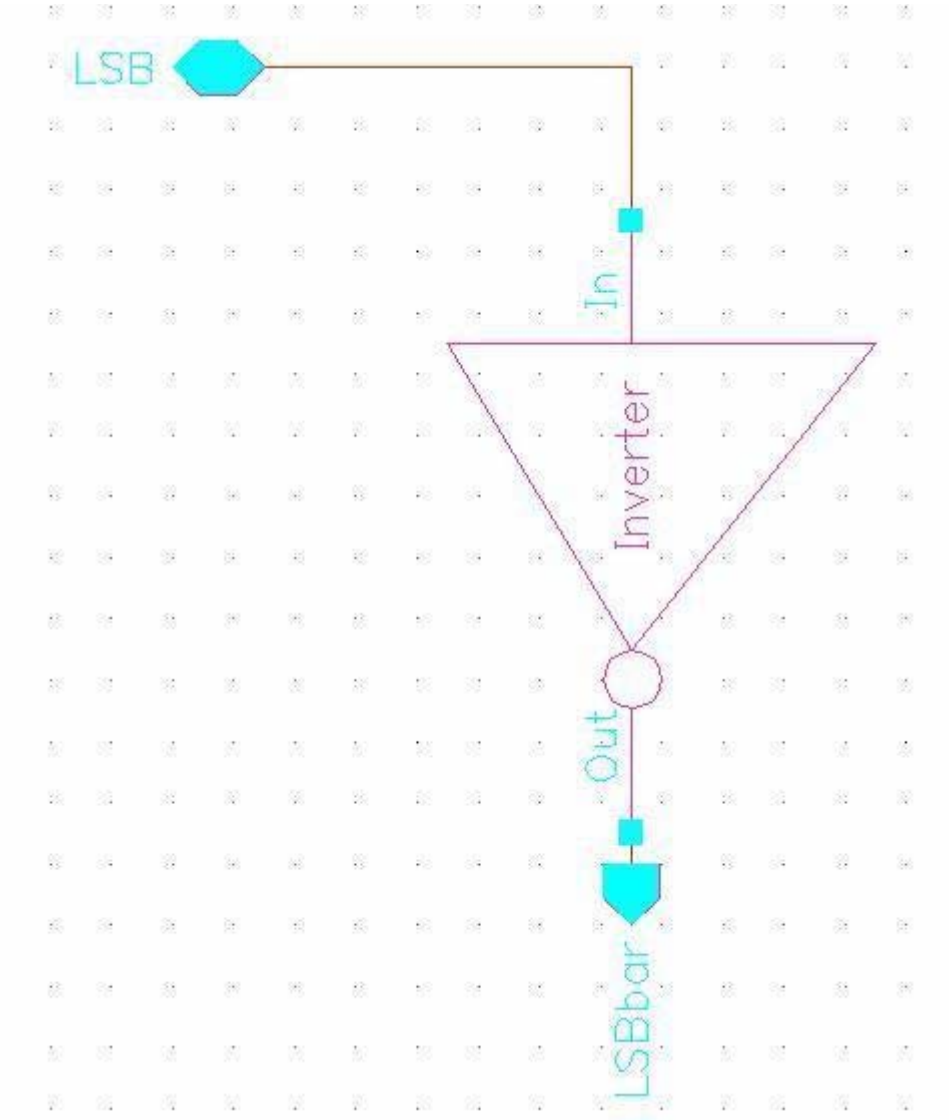


Fig A.12 Tree input schematic

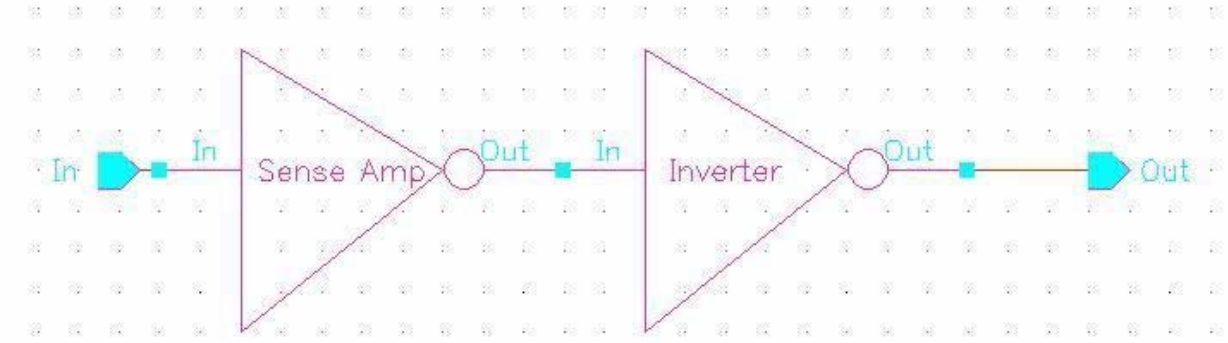


Fig A13 Sense amplifier schematic

APPENDIX B

LAYOUT

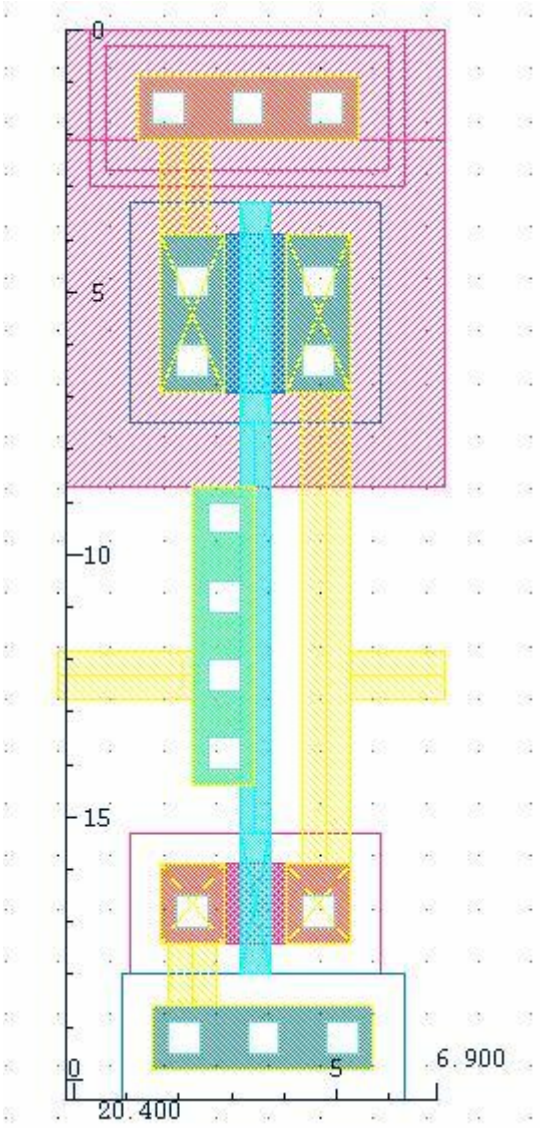


Fig B.1 Inverter Layout

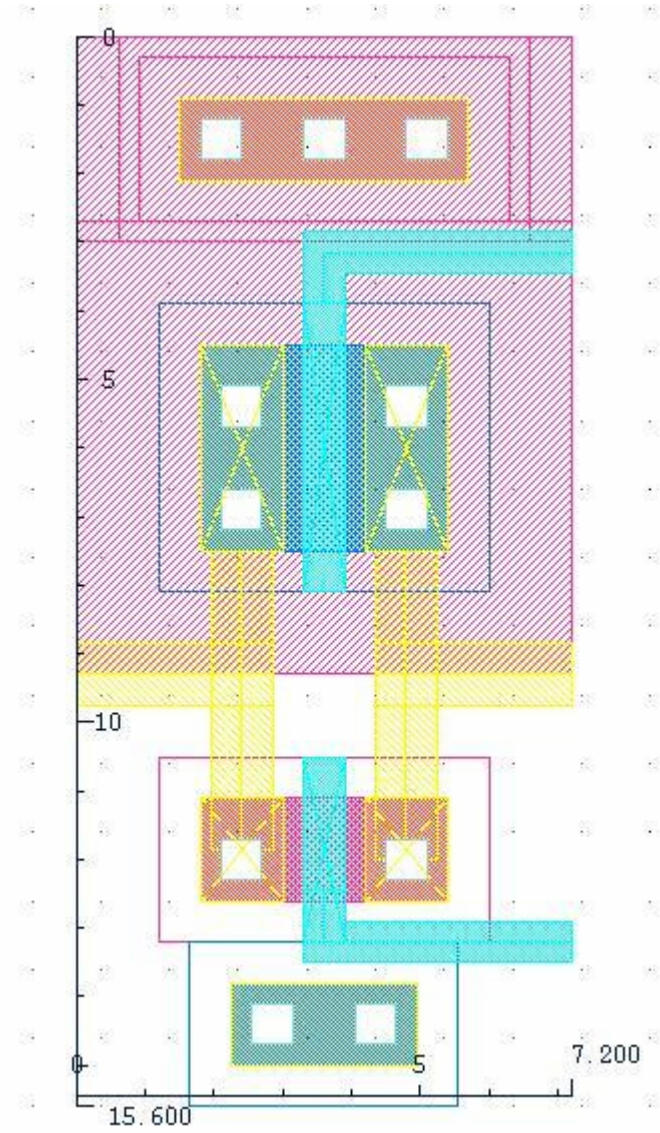


Fig B.2 Tx gate layout

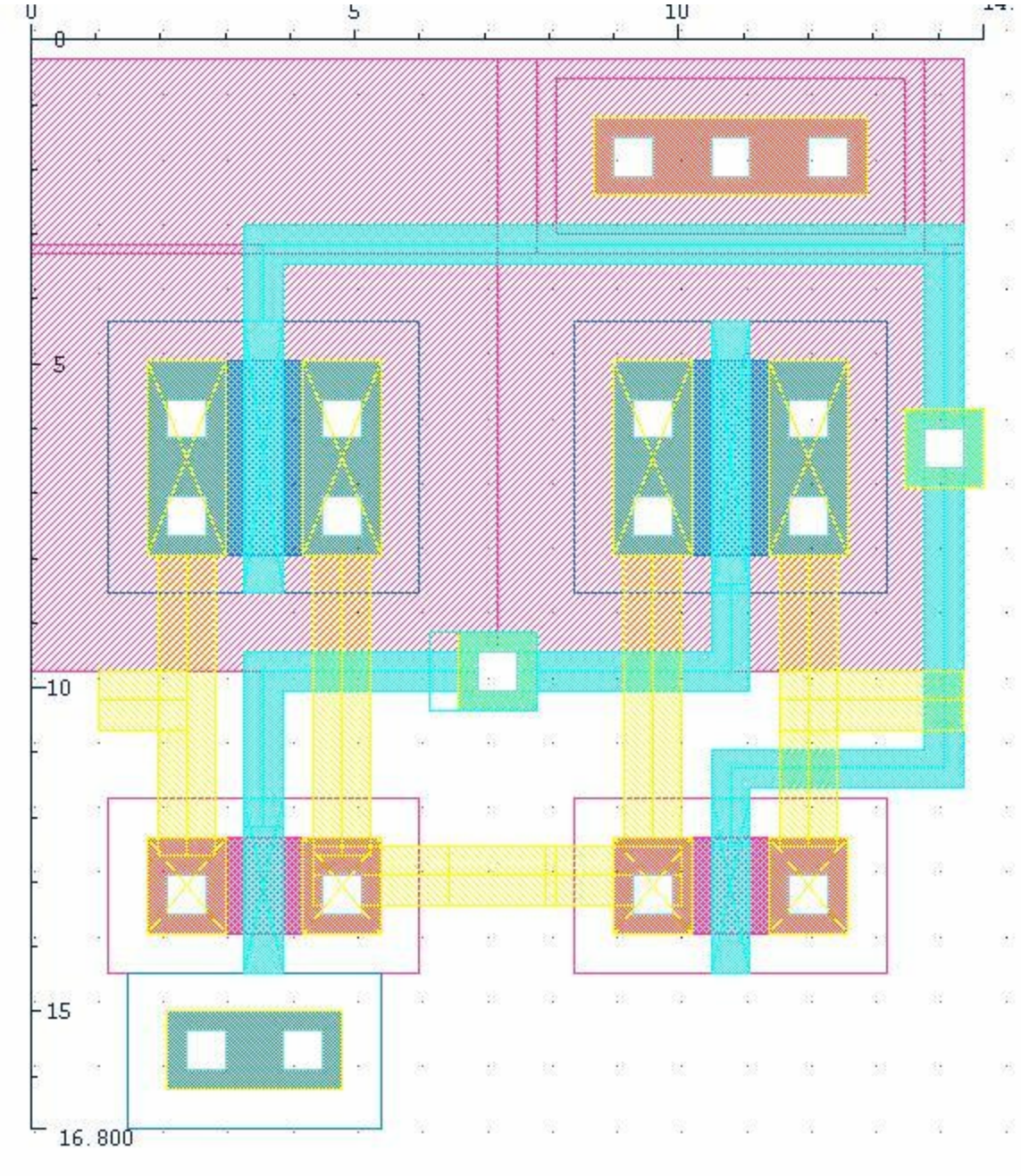


Fig B.3 Mux_2to1

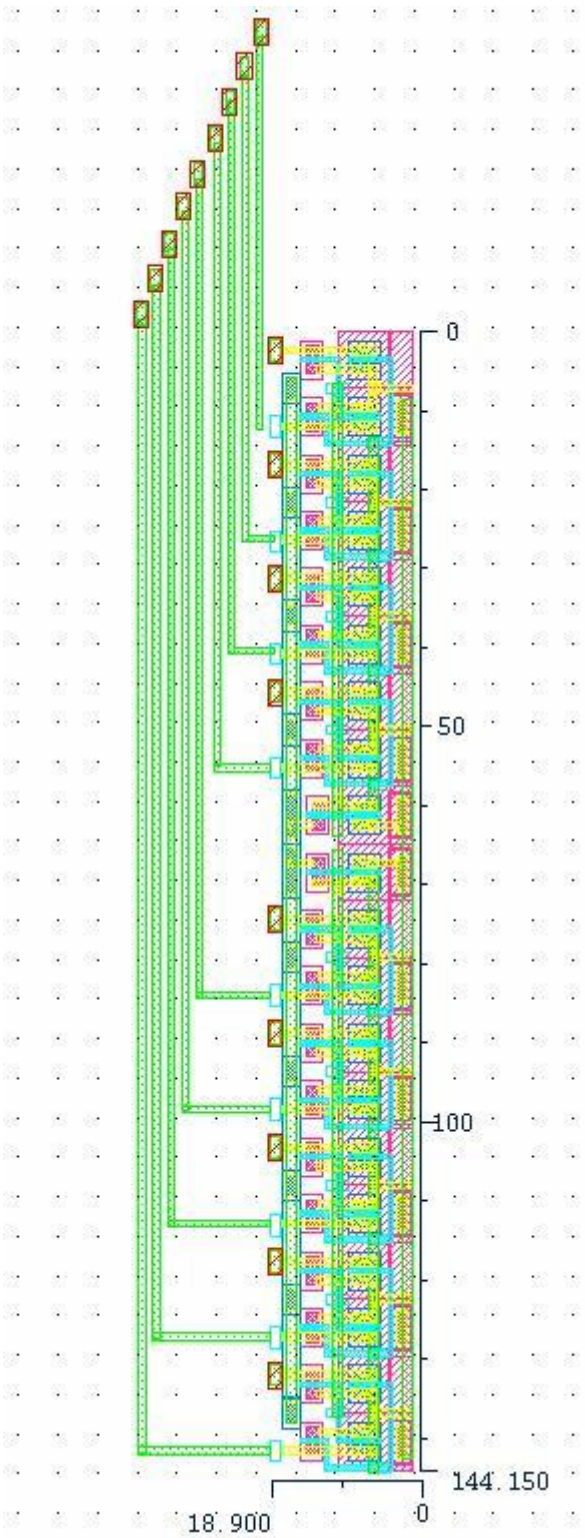


Fig B.4 9 Mux_2to1 layout

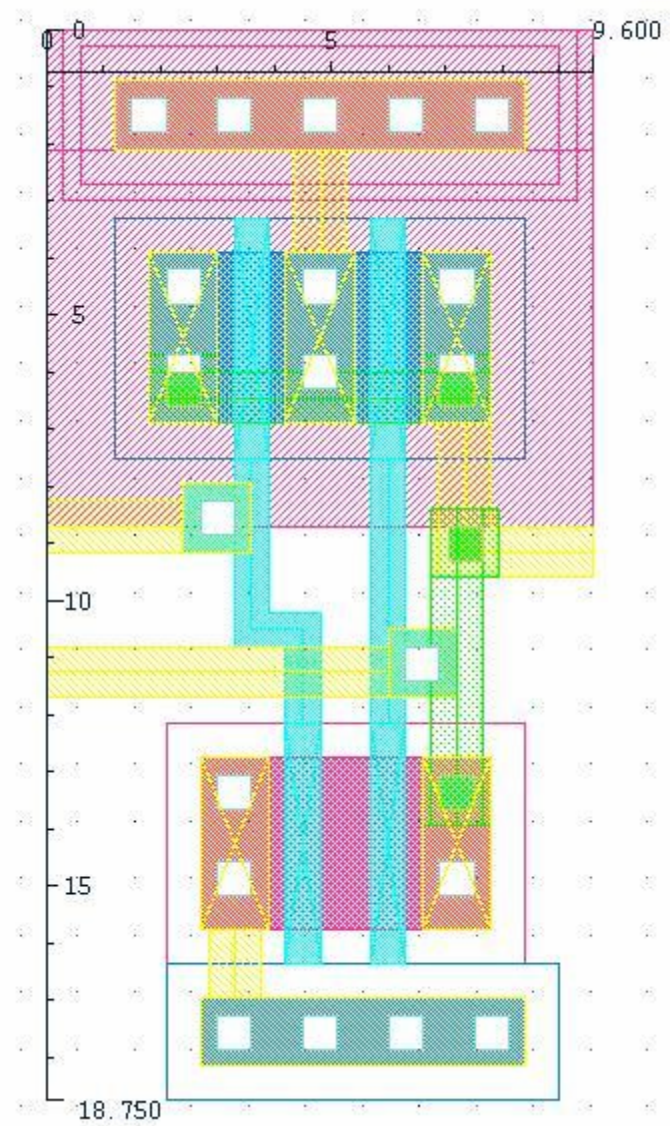


Fig B.5 NAND gate layout

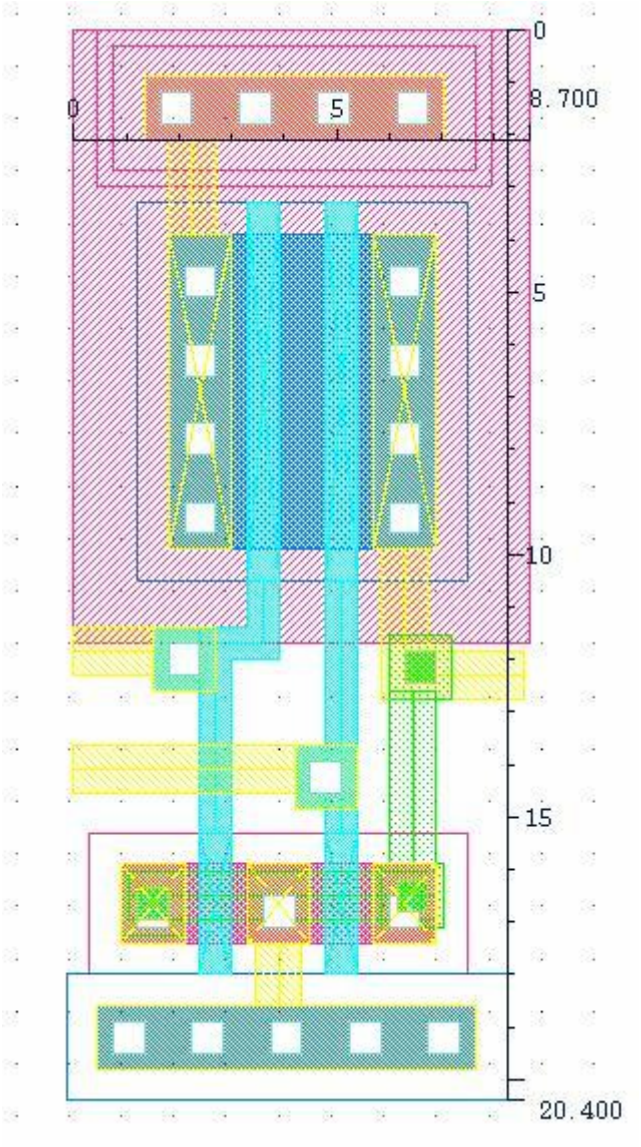


Fig B.6 NOR layout

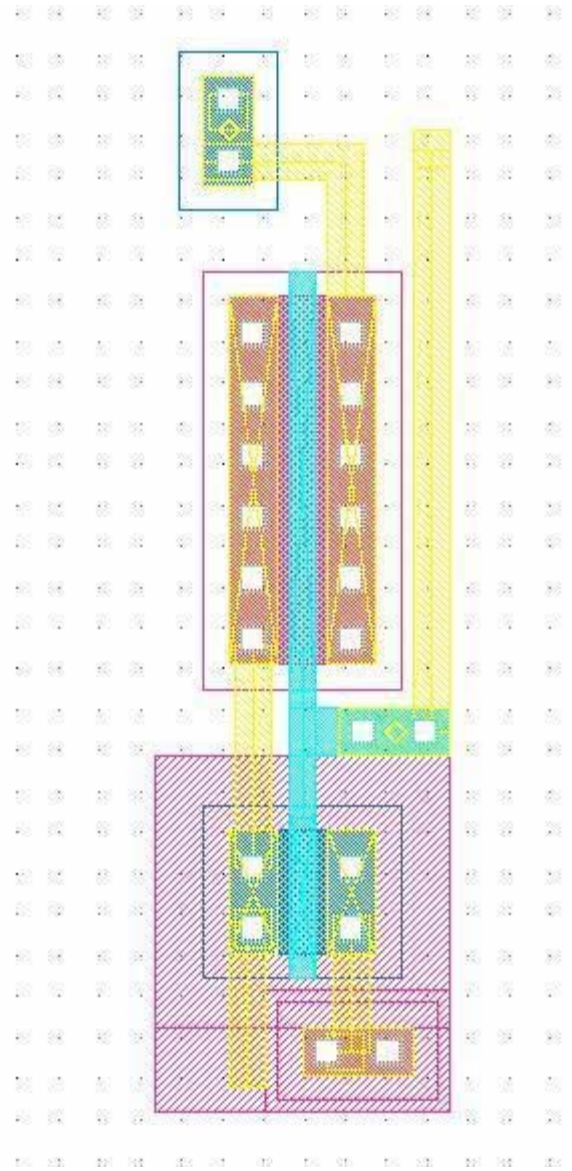


Fig B.7 Inverter offset Layout

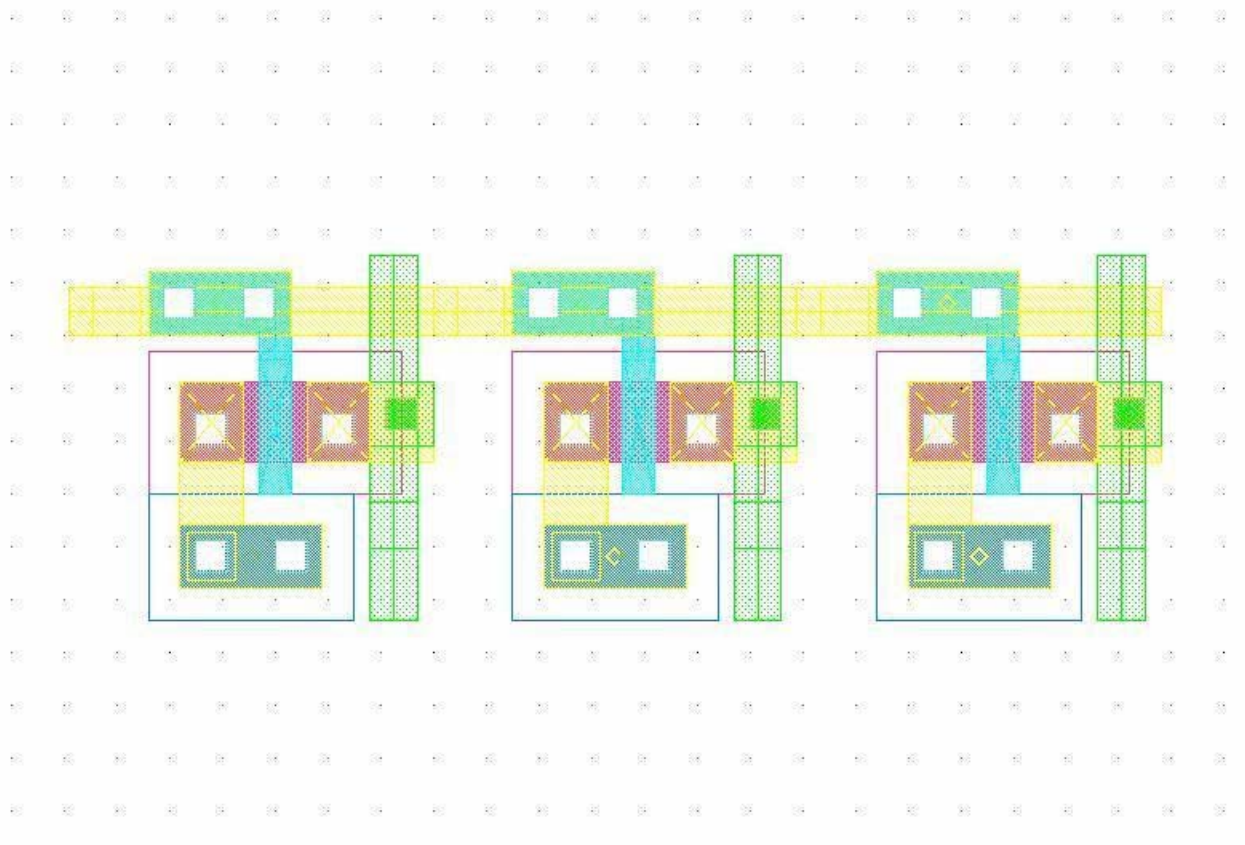


Fig B.8 ROM zero layout

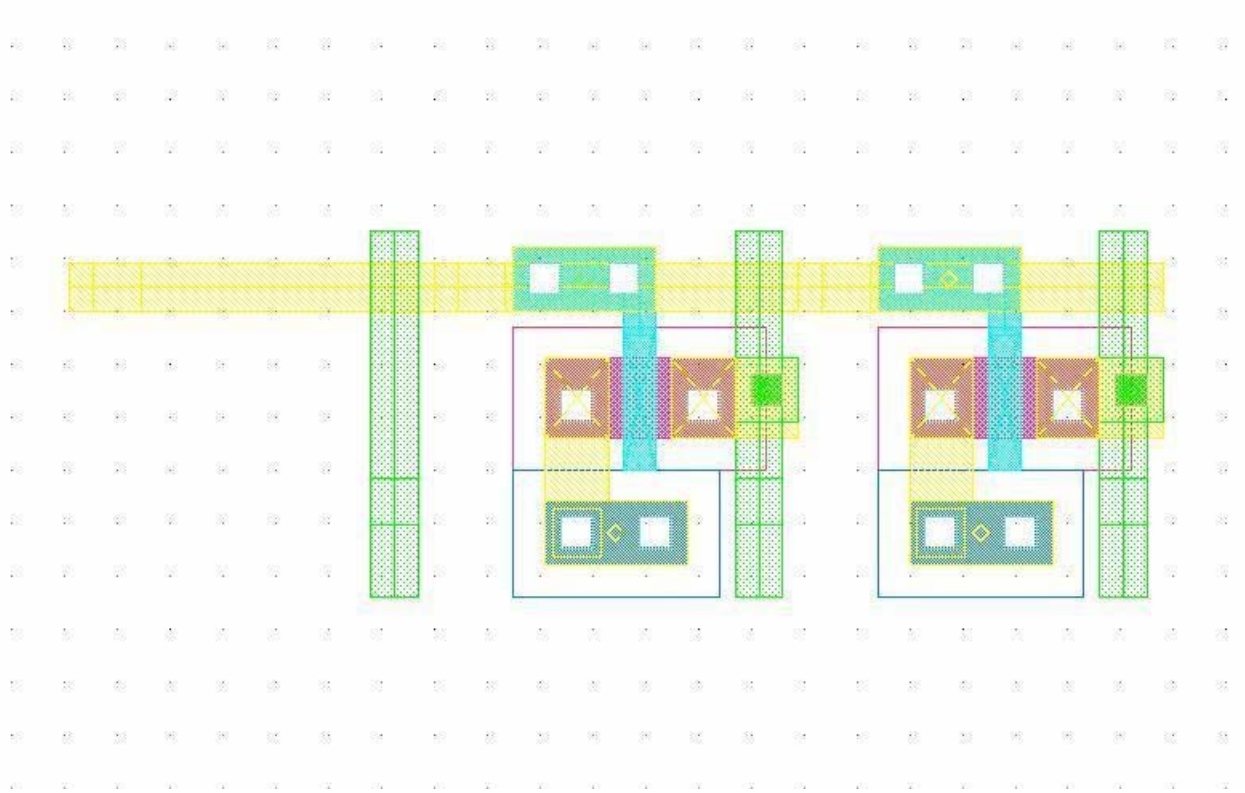


Fig B.9 ROM 4 Layout

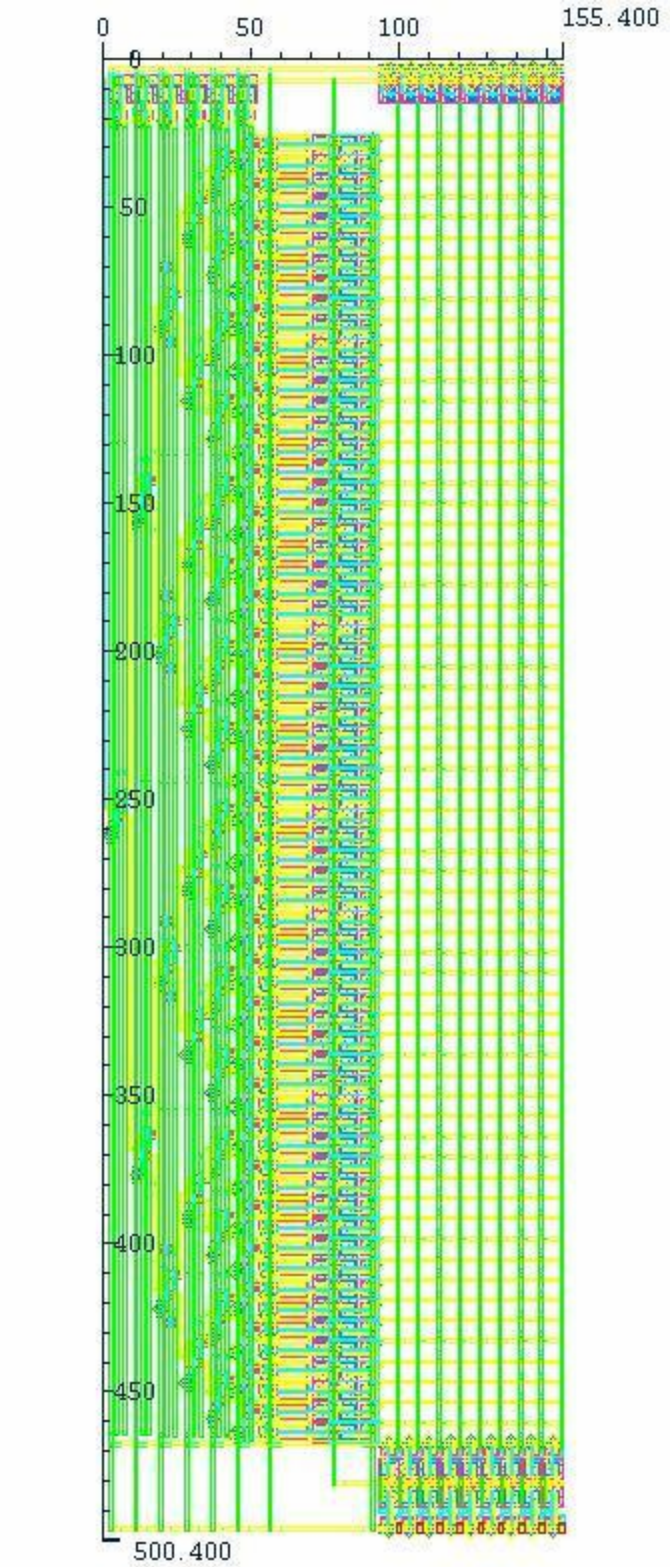


Fig. B.10 64 bit ROM layout

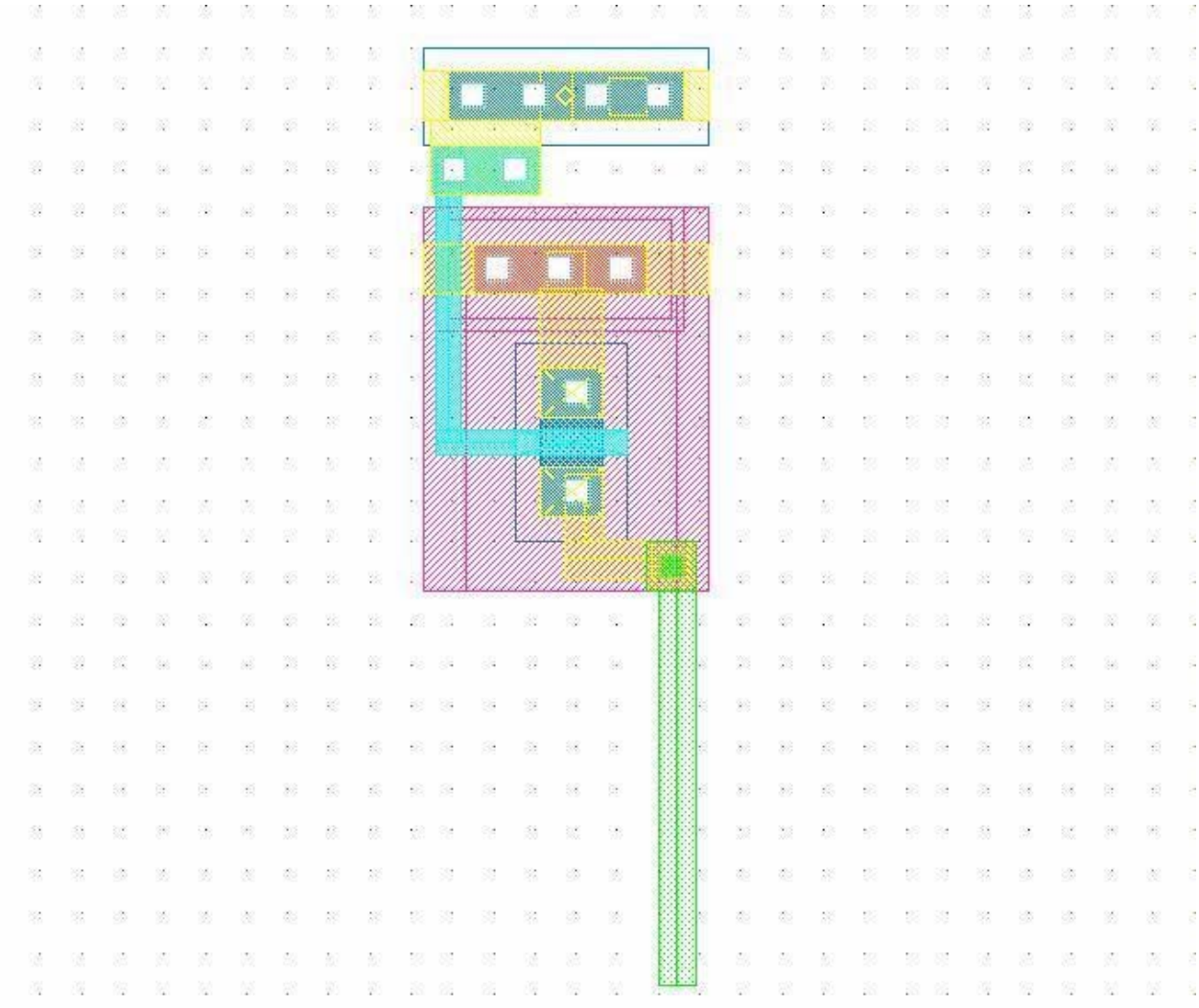


Fig.B.11. ROM pull up layout

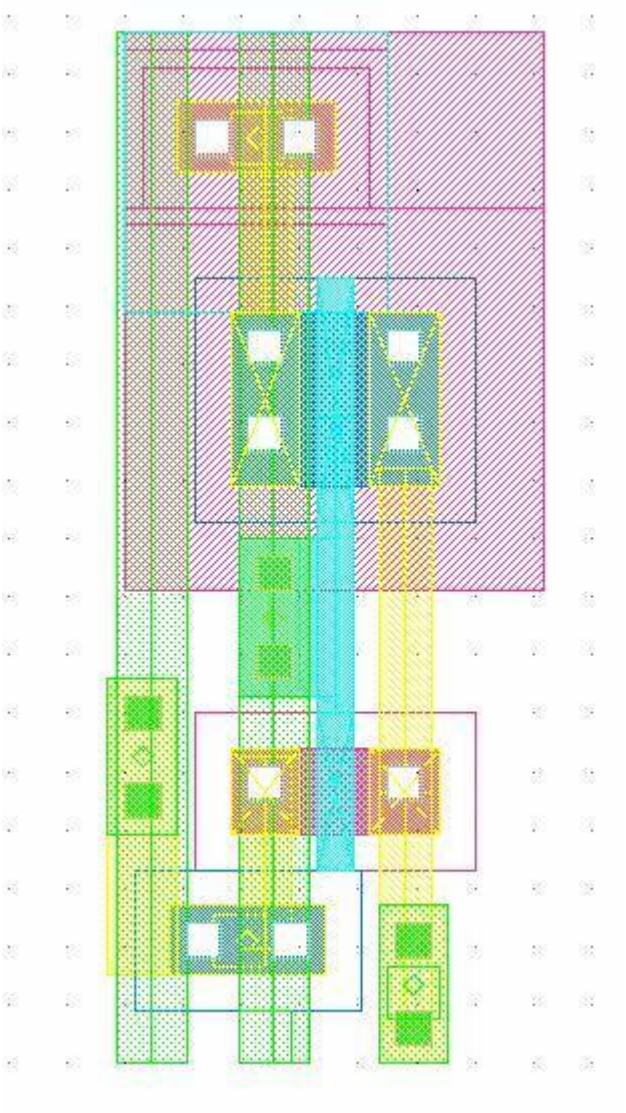


Fig B.12 Tree input layout

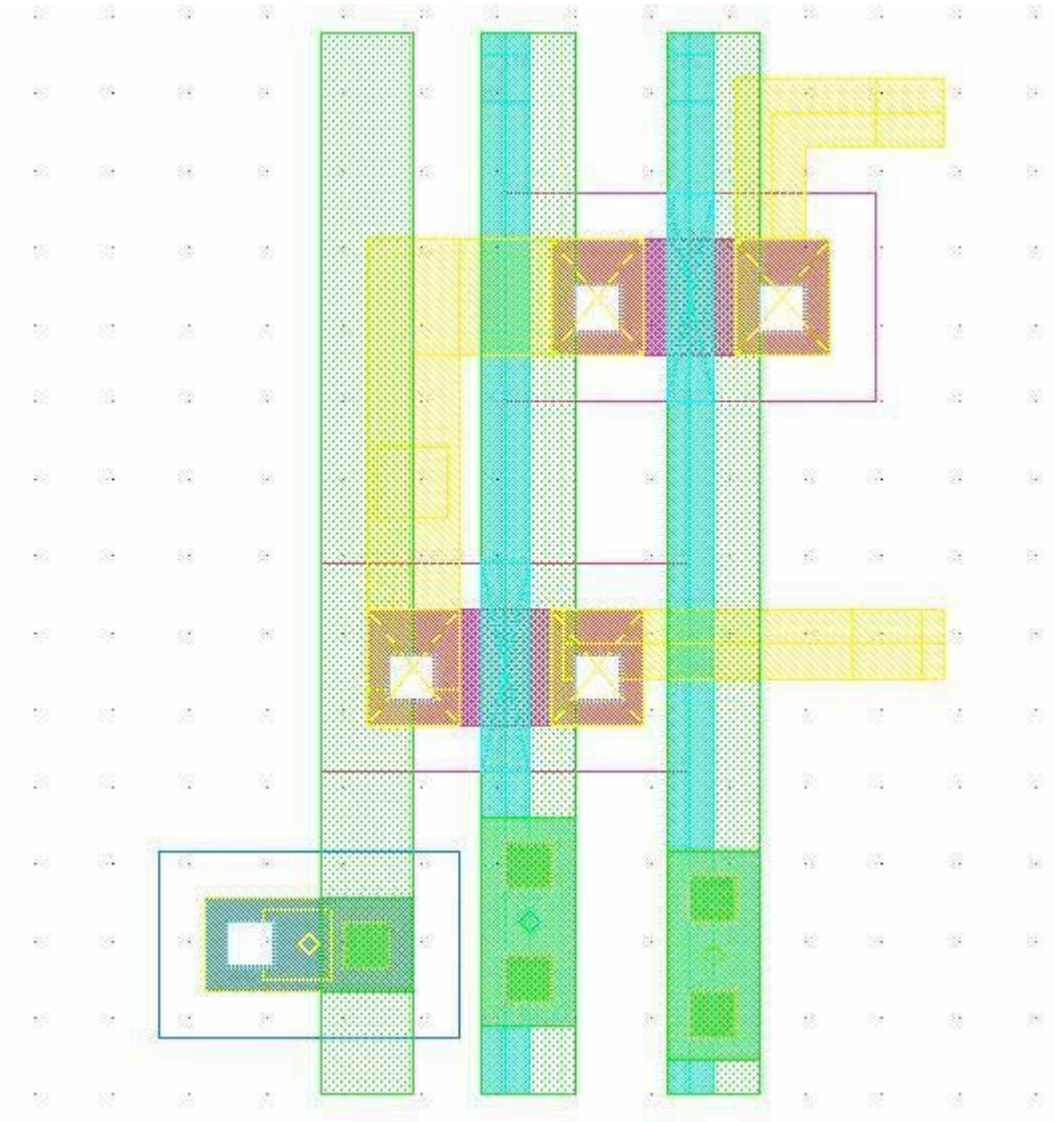


Fig. B.13 Tree 1st layout

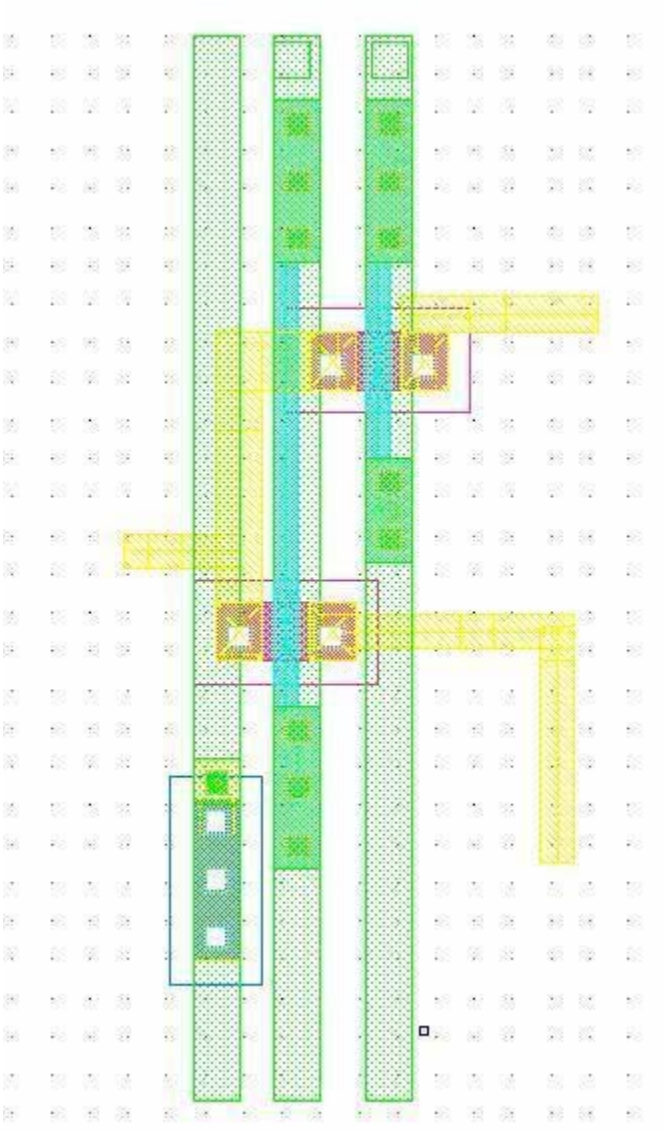


Fig B.13 Tree 2nd layout

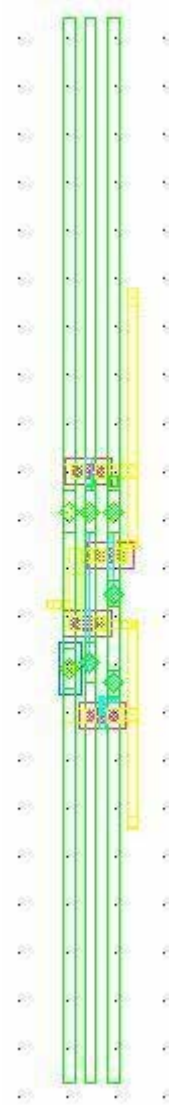


Fig B 14 Tree 4th layout

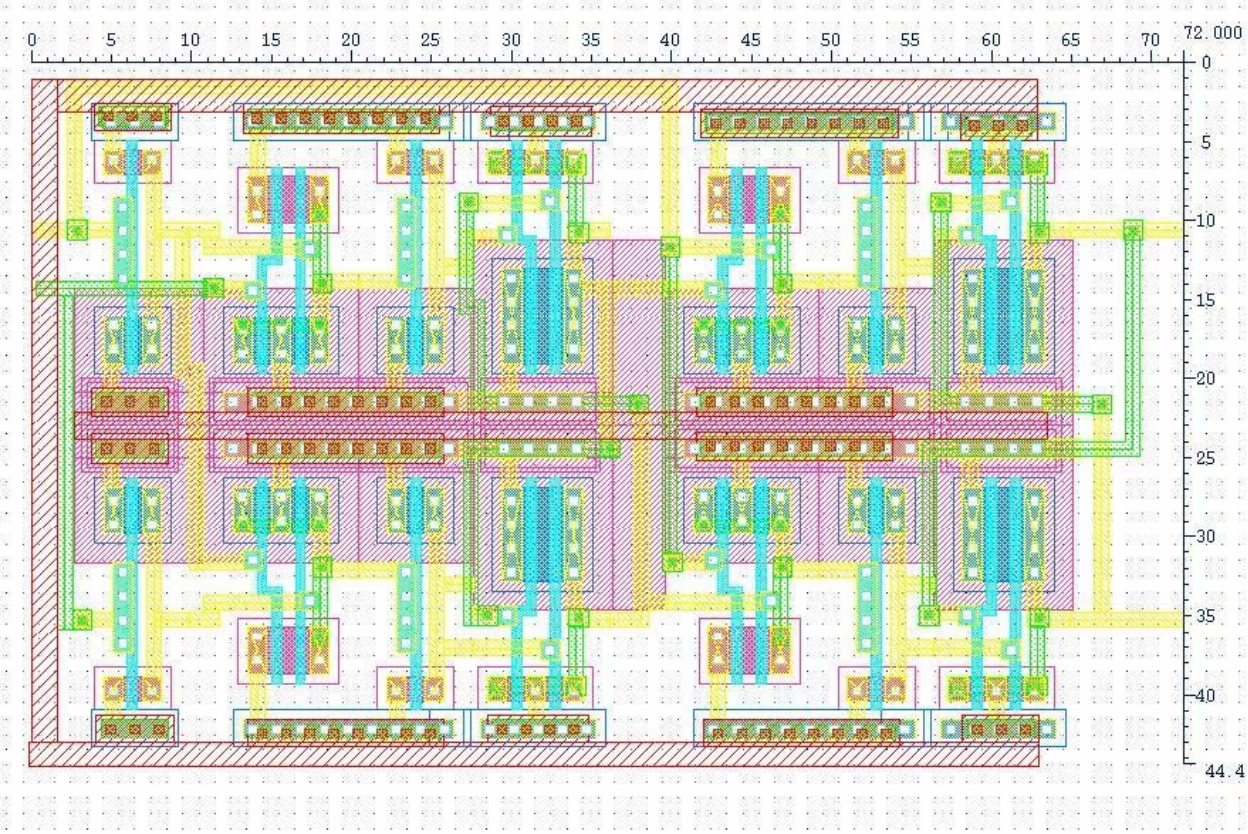


Fig B.15 D flip flop

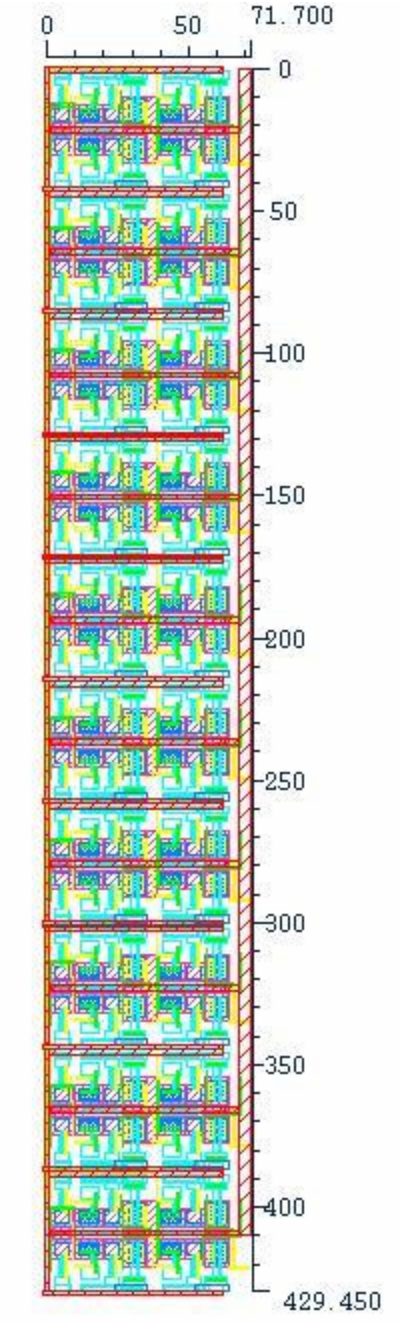


Fig B.16 9 D flip flop layout

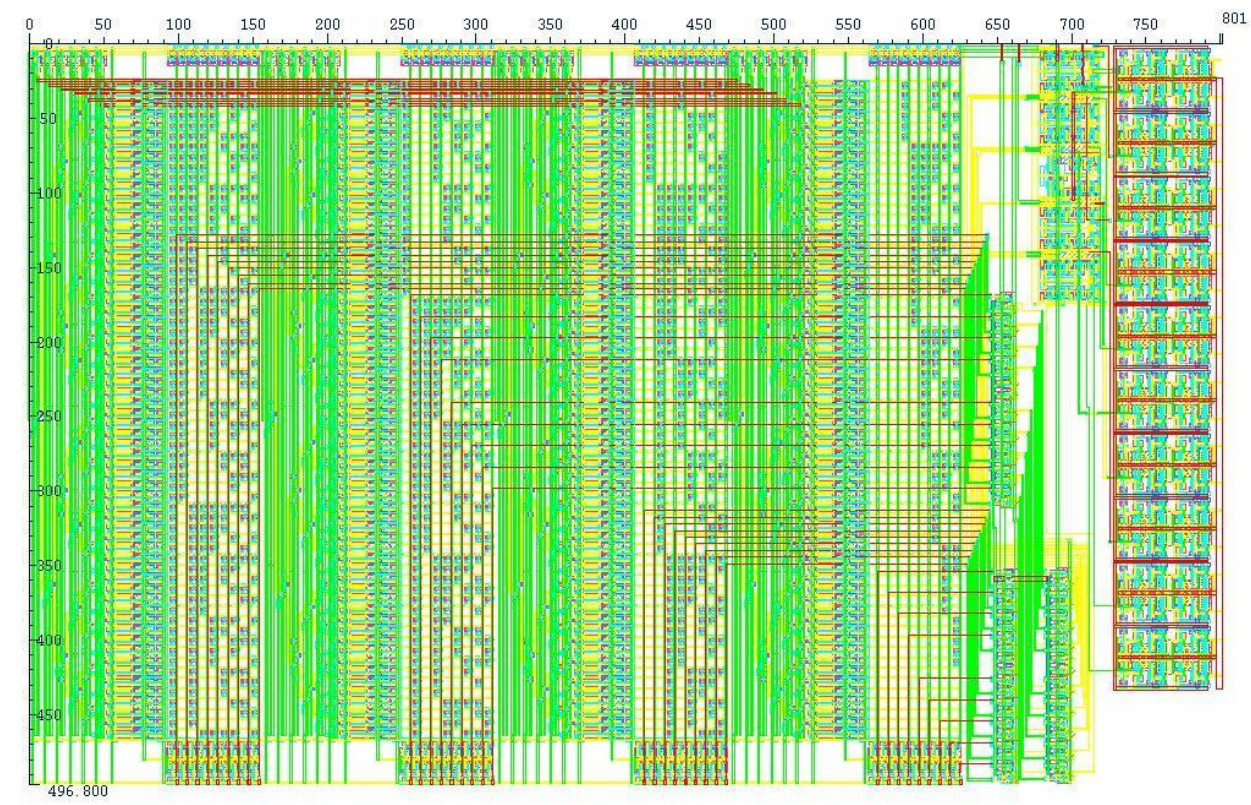


Fig B.17 256 ROM layout

APPENDIX C

DDS CHIP

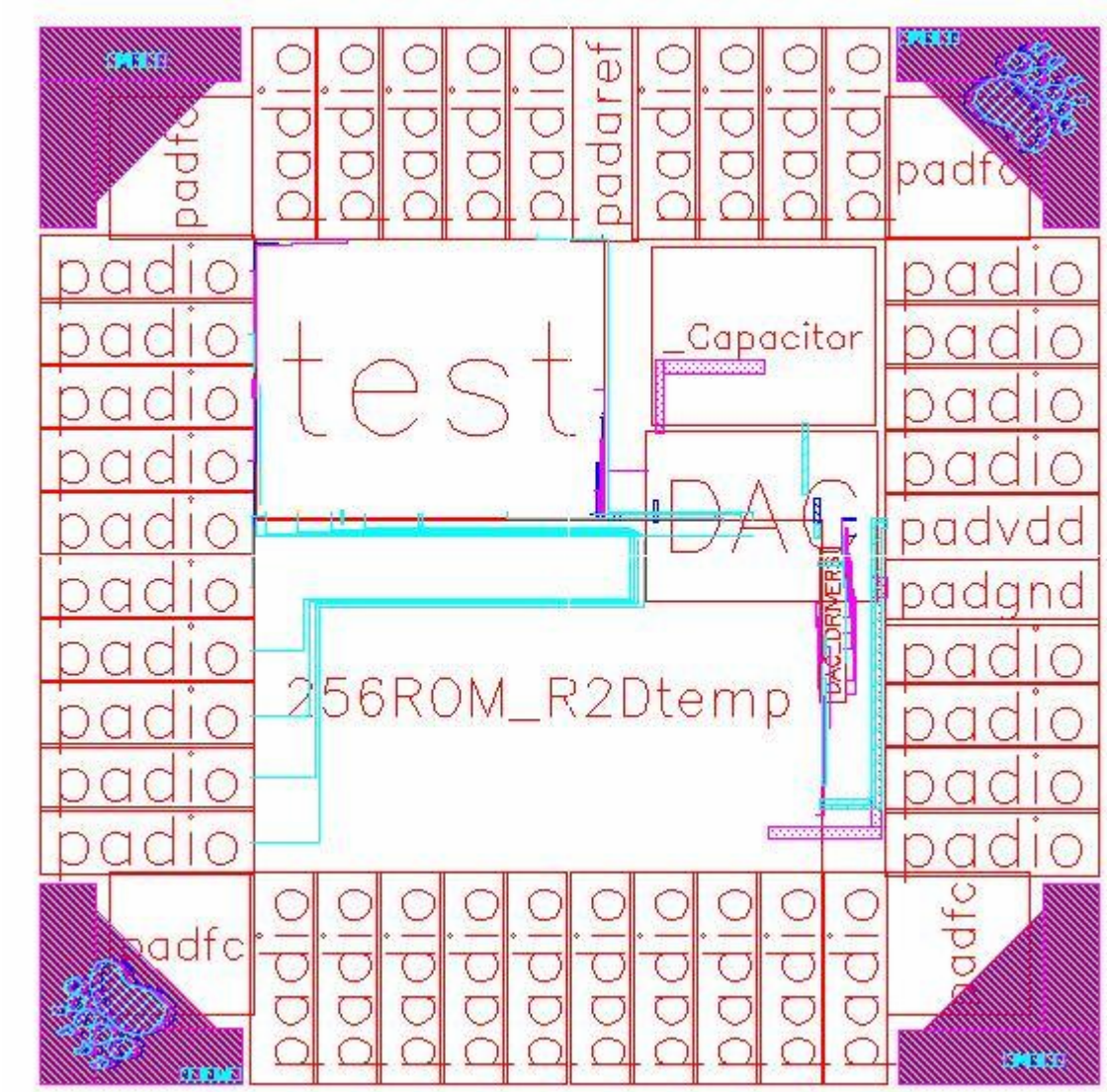

padfc
padio
padaref
test
_Capacitor
DAC
DAC_DRIVER
256ROM_R2Dtemp
padvdd
padgnd

Fig C.1 DDS

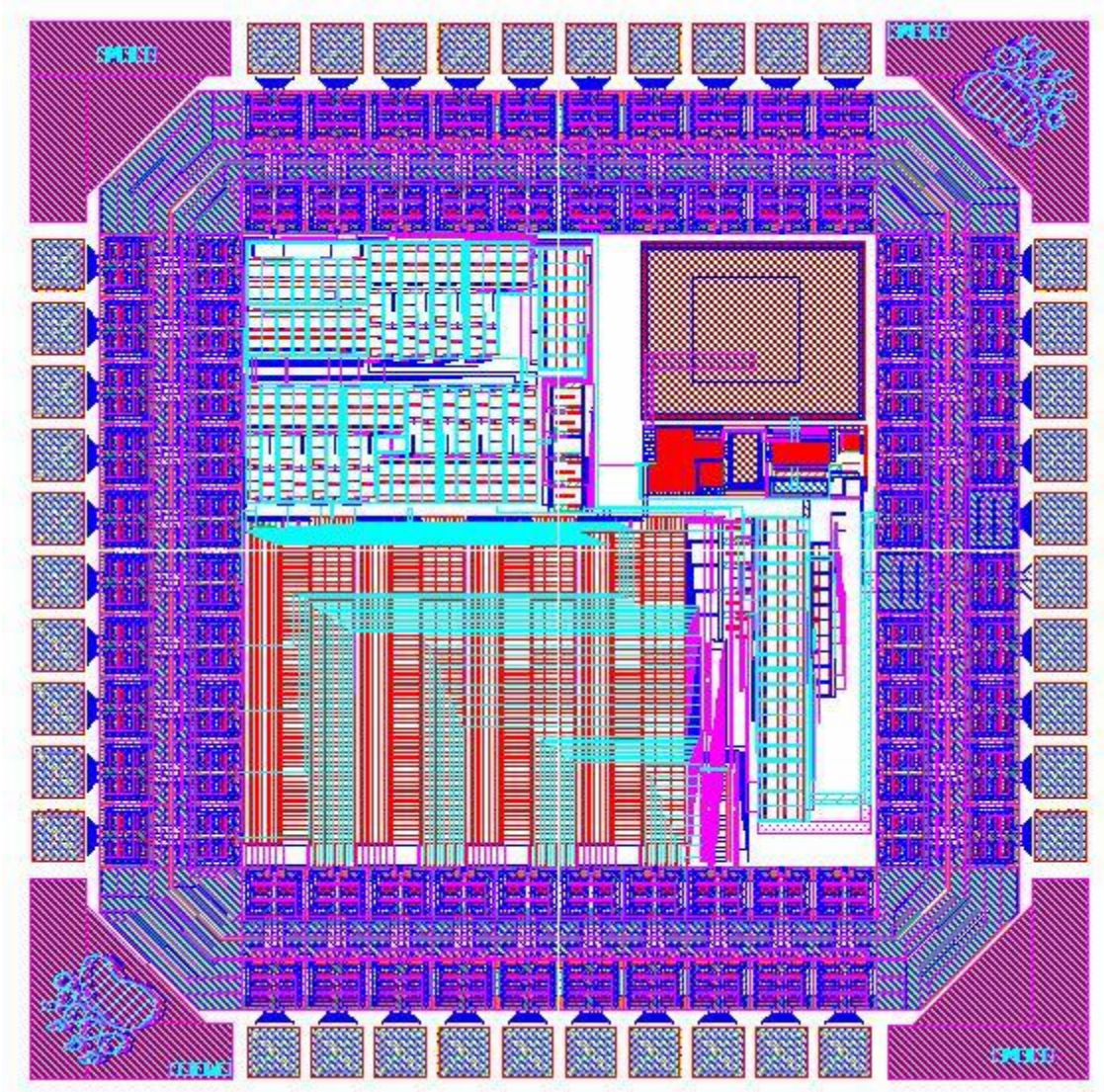


Fig C.2 Top level layout of DDS

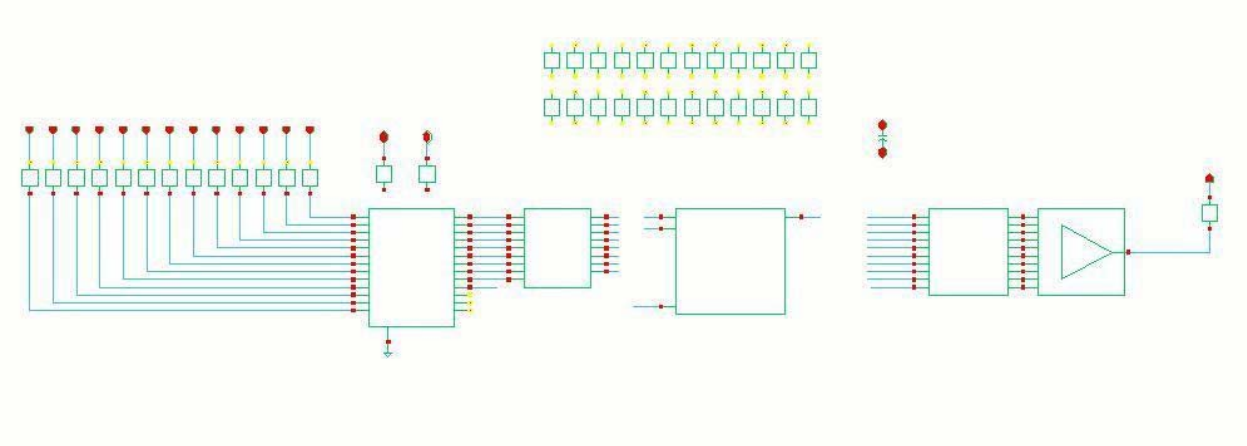


Fig C.3 Top level schematic of DDS

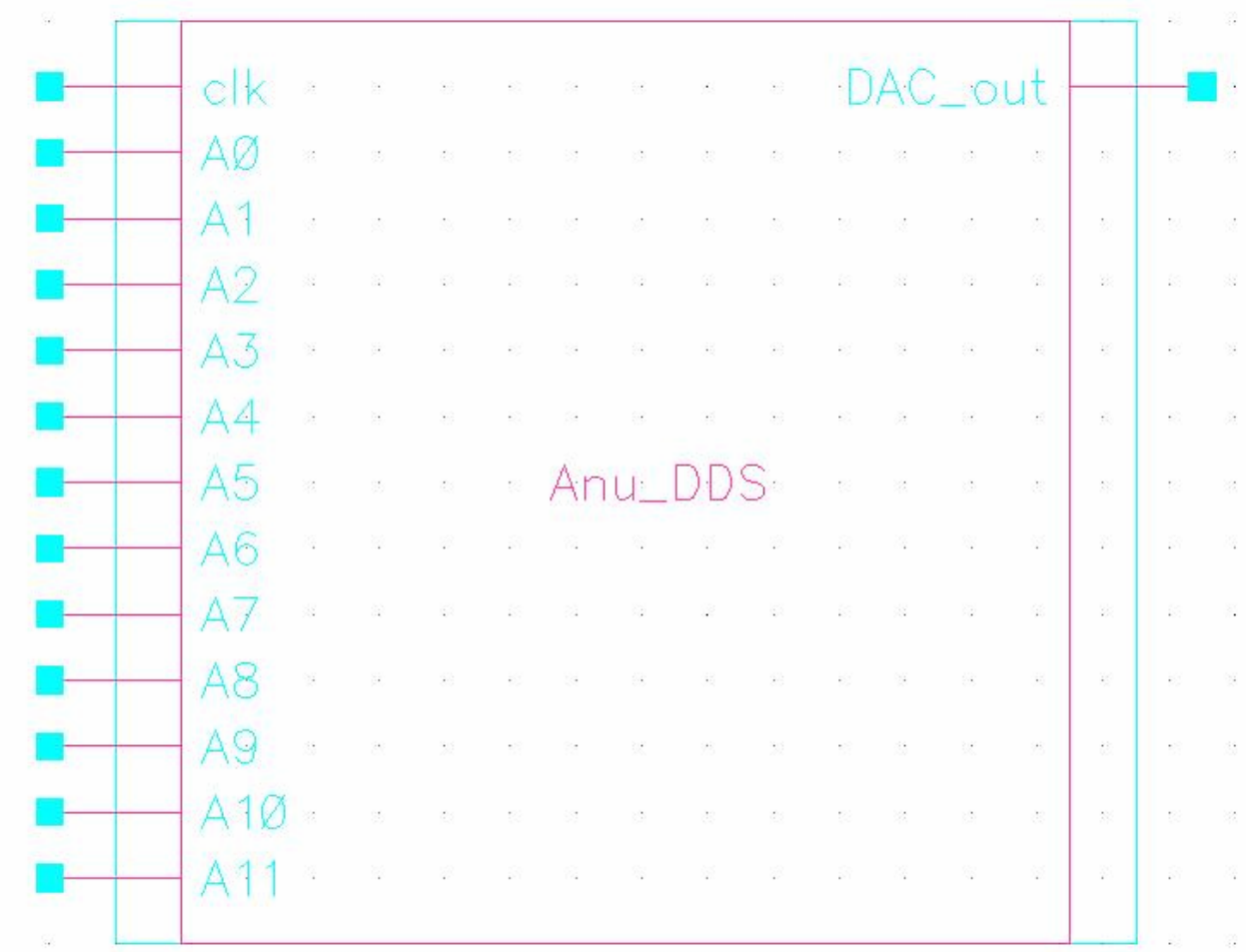


Fig C.4 Top level symbol of DDS

APPENDIX D

VALUES OF PROGRAMMED 256 ROM

ROM D	ROM C	ROMB	ROM A
000000010	011000101	101101010	111011001
000000101	011001000	101101101	111011010
000001011	011001011	101101111	111011011
000001000	011001110	101110001	111011100
000001110	011010001	101110011	111011101
000010001	011010011	101110101	111011110
000010100	011010110	101110111	111100000
000011000	011011001	101111010	111100001
000011011	011011100	101111100	111100010
000011110	011011111	101111110	111100011
000100001	011100010	110000000	111100100
000100100	011100100	110000010	111100101
000100111	011100111	110000100	111100110
000101010	011101010	110000110	111100111
000101101	011101101	110001000	111101000
000110001	011101111	110001010	111101001
000110100	011110010	110001100	111101001
000110111	011110101	110001110	111101010
000111010	011111000	110010000	111101011
000111101	011111011	110010010	111101100
001000000	011111101	110010100	111101101
001000110	100000000	110010110	111101110
001000110	100000011	110011000	111101111
001001001	100000101	110011010	111101111
001001101	100001000	110011011	111110000
001010000	100001011	110011101	111110001
001010011	100001101	110011111	111110010
001010110	100010000	110100001	111110010
001011001	100010011	110100011	111110011
001011100	100010101	110100100	111110100
001011111	100011000	110100110	111110100
001100010	100011011	110101000	111110101
001100101	100011101	110101010	111110101
001101000	100100000	110101011	111110110
001101011	100100010	110101101	111110111
001101110	100100101	110101111	111110111
001110001	100101000	110110001	111111000
001110101	100101010	110110010	111111000
001111000	100101101	110110100	111111001
001111011	100101111	110110101	111111001
001111110	100110010	110110111	111111010
010000001	100110100	110111001	111111010
010001010	100110111	110111010	111111011
010000111	100111001	110111100	111111011
010001010	100111100	110111101	111111011

010001101	100111110	110111111	111111100
010010000	101000001	111000000	111111100
010010011	101000011	111000010	111111100
010010110	101000101	111000011	111111101
010011001	101001000	111000101	111111101
010011100	101001010	111000110	111111101
010011111	101001101	111001000	111111101
010100010	101001111	111001001	111111110
010100101	101010001	111001011	111111110
010101000	101010100	111001100	111111110
010101011	101010110	111001101	111111110
010101110	101011000	111001111	111111110
010110001	101011011	111010000	111111111
010110100	101011101	111010001	111111111
010110110	101011111	111010011	111111111
010111001	101100001	111010100	111111111
010111100	101100100	111010101	111111111
010111111	101100110	111010110	111111111
011000010	101101000	111011000	111111111