ENERGY EFFICIENT VLSI DESIGN

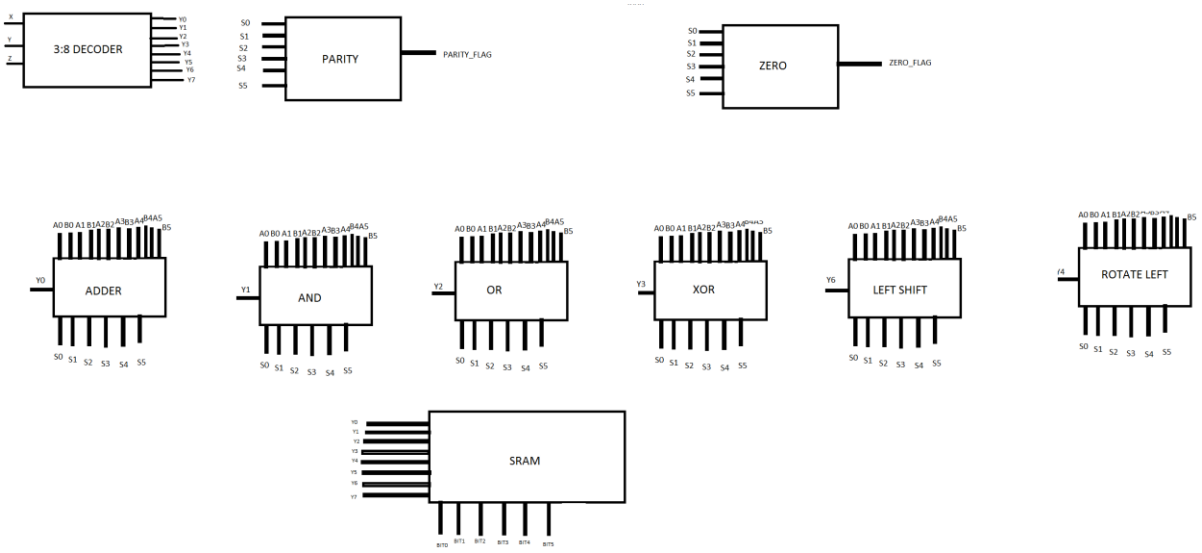
FINAL PROJECT-2019

BY

HERAMB SAWANT

BLOCK DIAGRAM OF FULL ALU

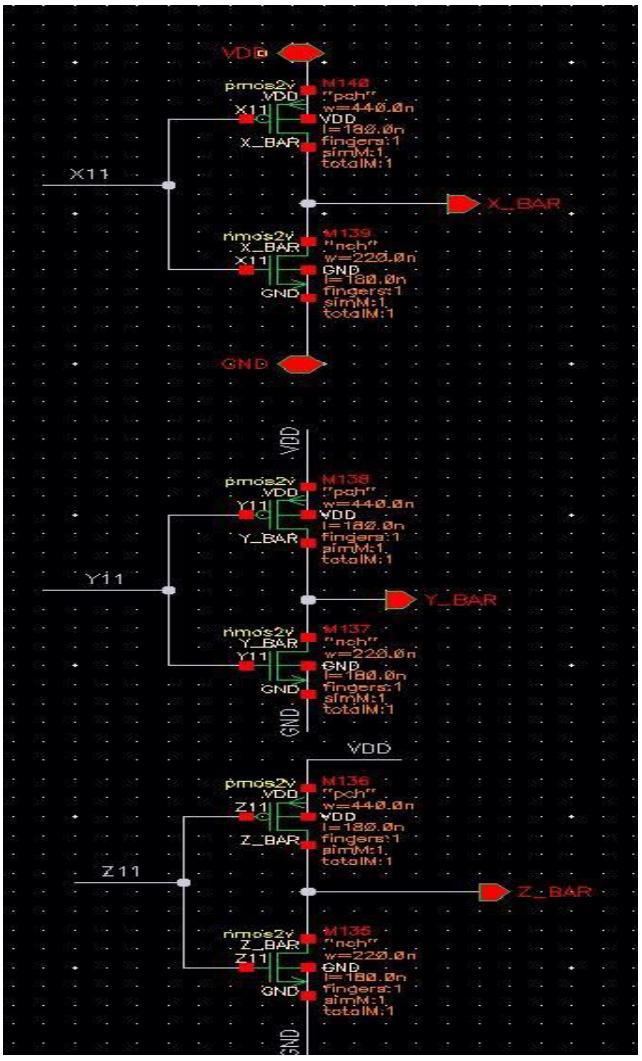
At the beginning I have three control signals named X, Y and Z which controls a specific operation at a time. Outputs of control logic are Y0, Y1, Y2, Y3, Y4, Y5, Y6, and Y7. I have 2 Six bits inputs A0, A1, A2, A3, A4, A5 and B0, B1, B2, B3, B4, B5. Finally I have six output bits common for every block, and also 4 flags as required named ZERO_FLAG, PARITY_FLAG, CARRY_FLAG, and OVERFLOW_FLAG. For all the logic block (adder, and, etc.) I have considered W/L for PMOS as 440nm/180nm and W/L for NMOS as 220nm/180nm.

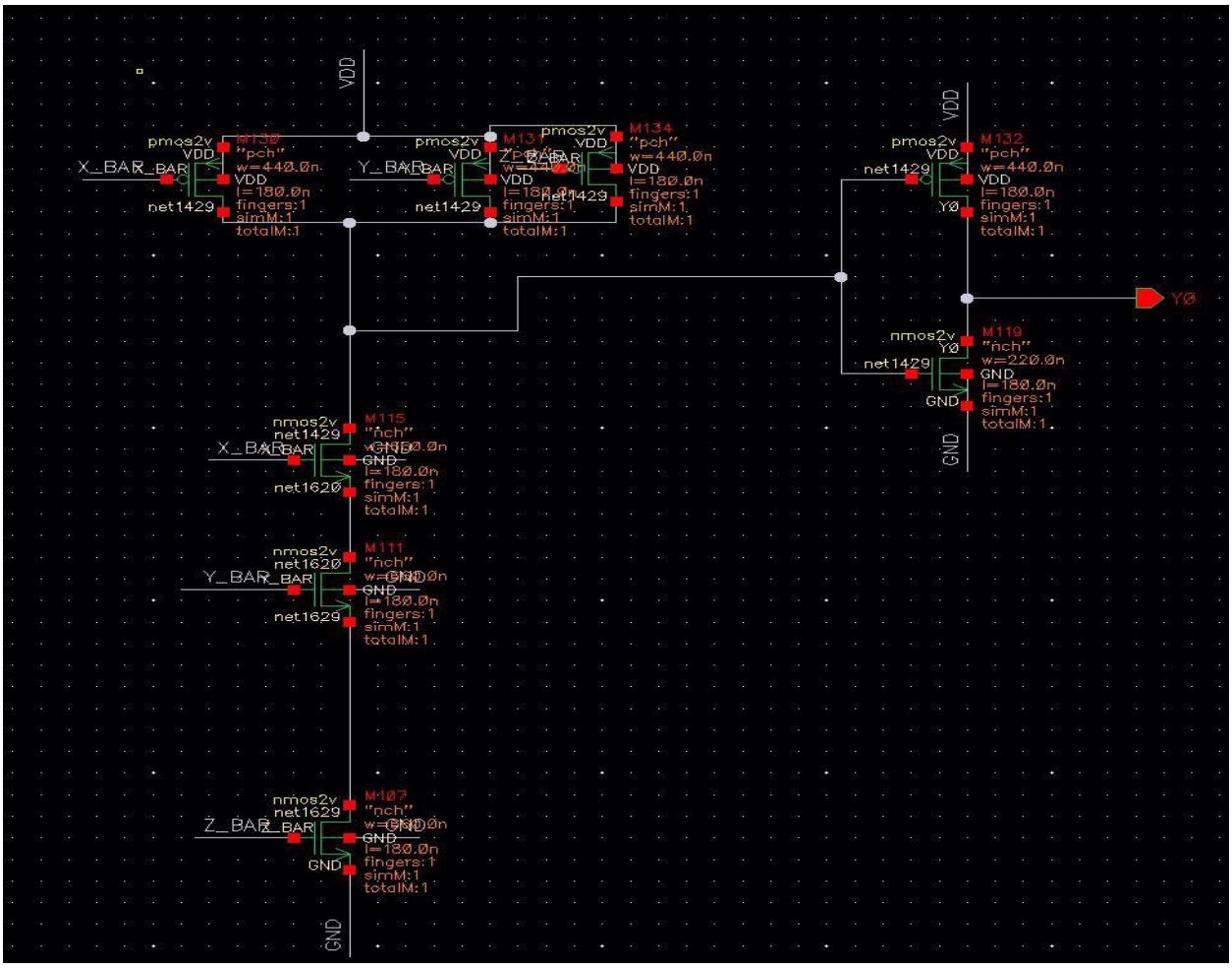


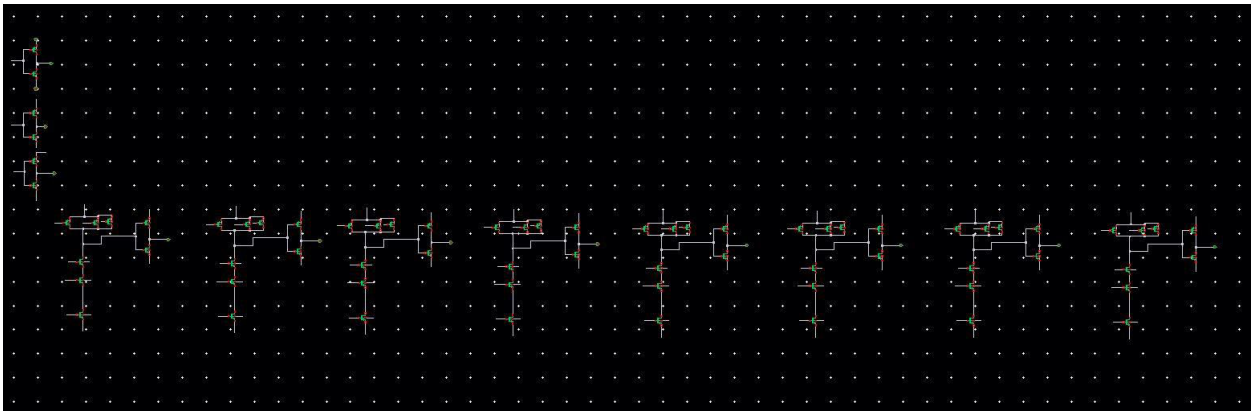
1- CONTROL LOGIC

A – For control logic I used 3:8 decoder. 3:8 decoder uses 3 input bits and make one output high at a time. In my design, I used Y0 output for adder, Y1 output for LOGICAL AND, Y2 output for LOGICAL OR, Y3 output for LOGICAL XOR, Y4 output for ROTATE LEFT, Y6 output for SHIFT LEFT and Y7 for POWER SAVING.

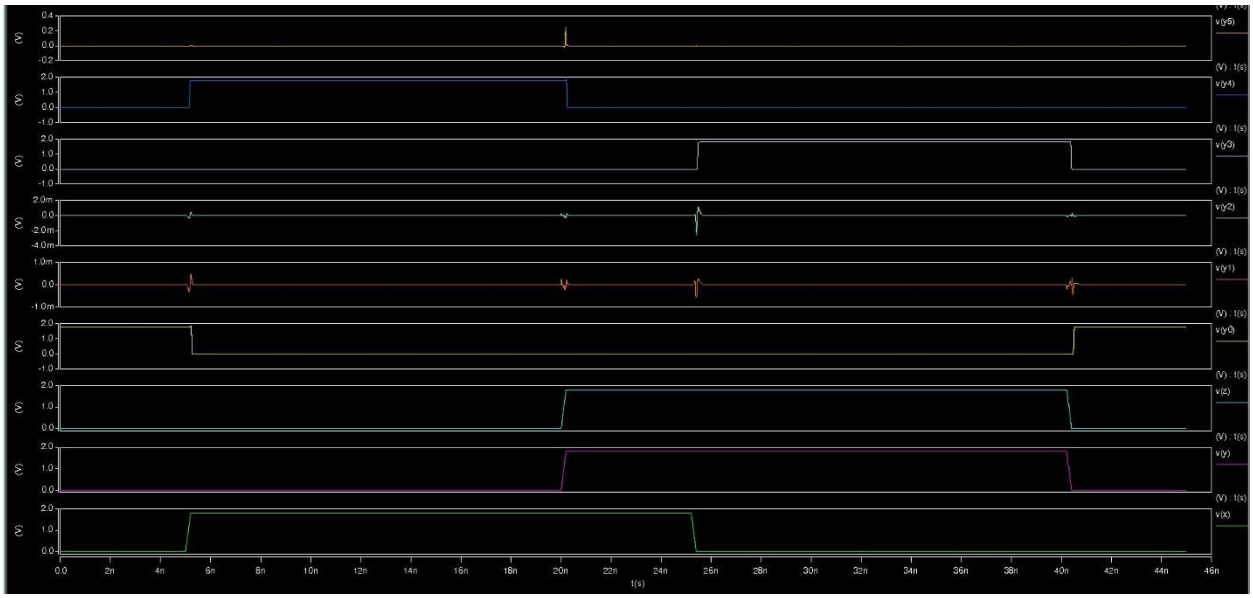
B – SCHEMATICS- I am attaching my schematics below. First I am attaching the inverters which I used to get inverted output of control signals. Then I am attaching a single bit logic diagram to give a clear picture, all other 7 bits are of same pattern. I am using 3 input AND gate, and finally I am attaching full schematics of my control logic.

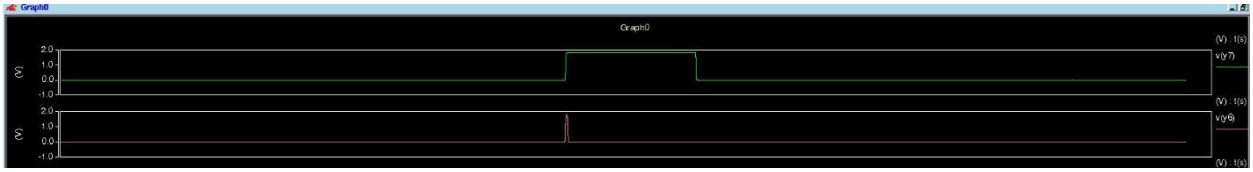



VDD
pmos2v
"pch"
w=440.0n
l=180.0n
fingers:1
simM:1
totalM:1
X_BAR
Y_BAR
Z_BAR
net1429
M134
M132
M119
"nch"
w=220.0n
GND
YØ
nmos2v
M115
M111
M107
net1620
net1629



C – WORKING – As we can observe from output graph, at a given 3 inputs only one output is high. I have not mentioned all the 8 outputs but couple of outputs to show proper functionality.

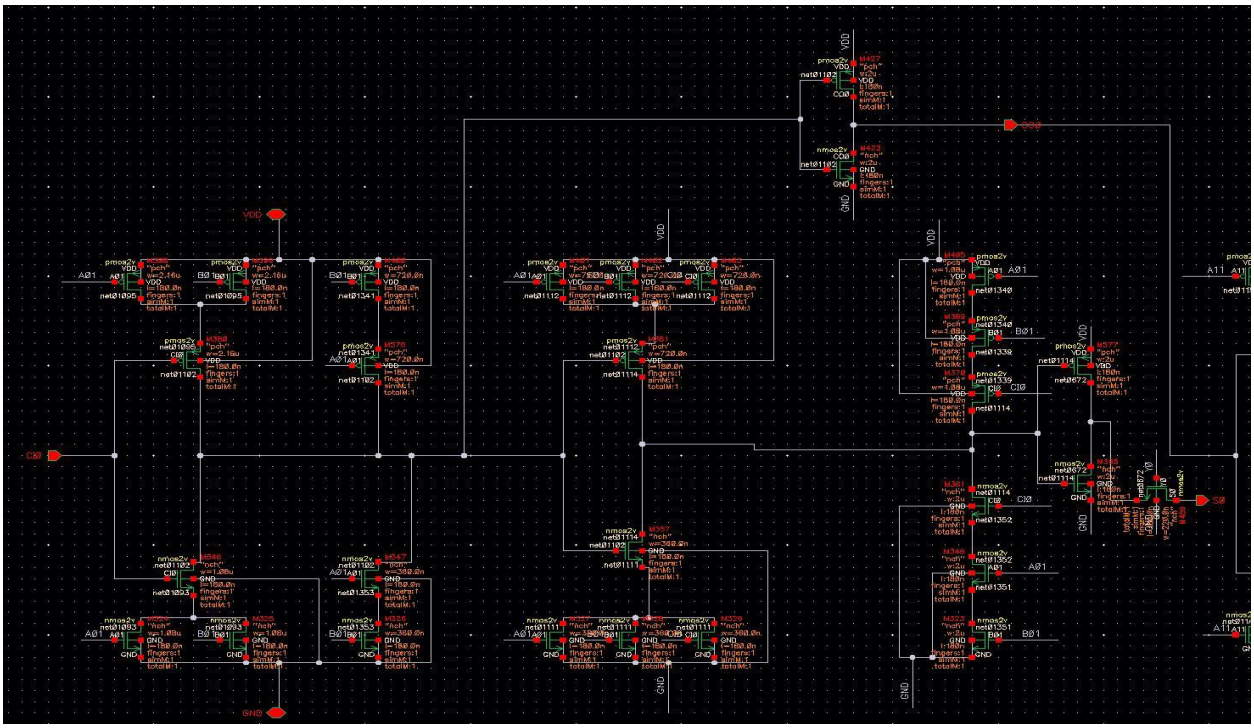


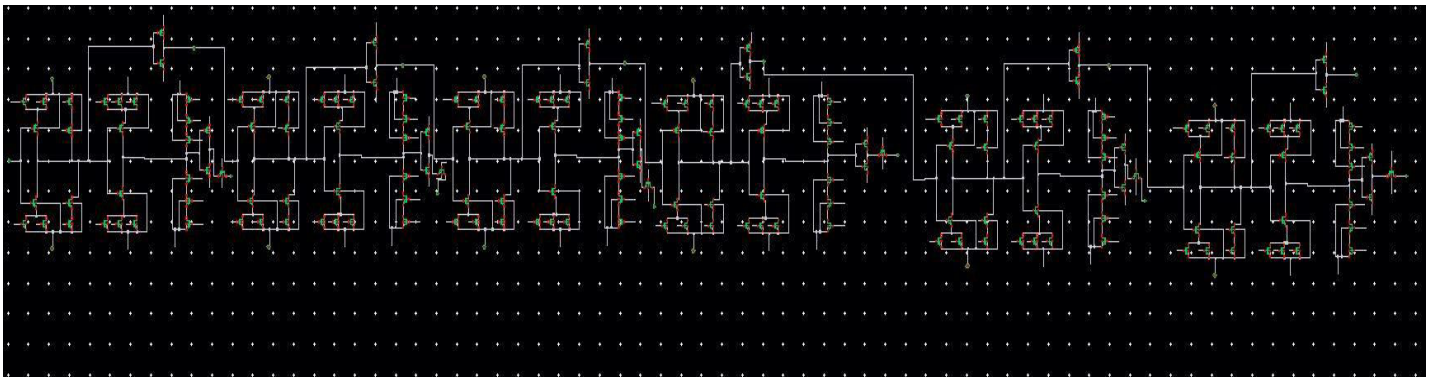


2- ADDER

A - As per the requirement adder block should work at control signals 000. As my control bits become 000, Y0 output signal of my decoder goes high which starts my adder block. I am connecting a pass transistor before the output pins of every block which ensures that only one block is functioning at a time. For clearer picture I am attaching schematics of one bit adder and the same schematics is repeated for every bit. At the end of ADDER section I will add full 6 bit schematics for reference. For this project I am using ripple carry adder. I have added an inverter at the end of carry out to ensure that non-inverted carry propagates.

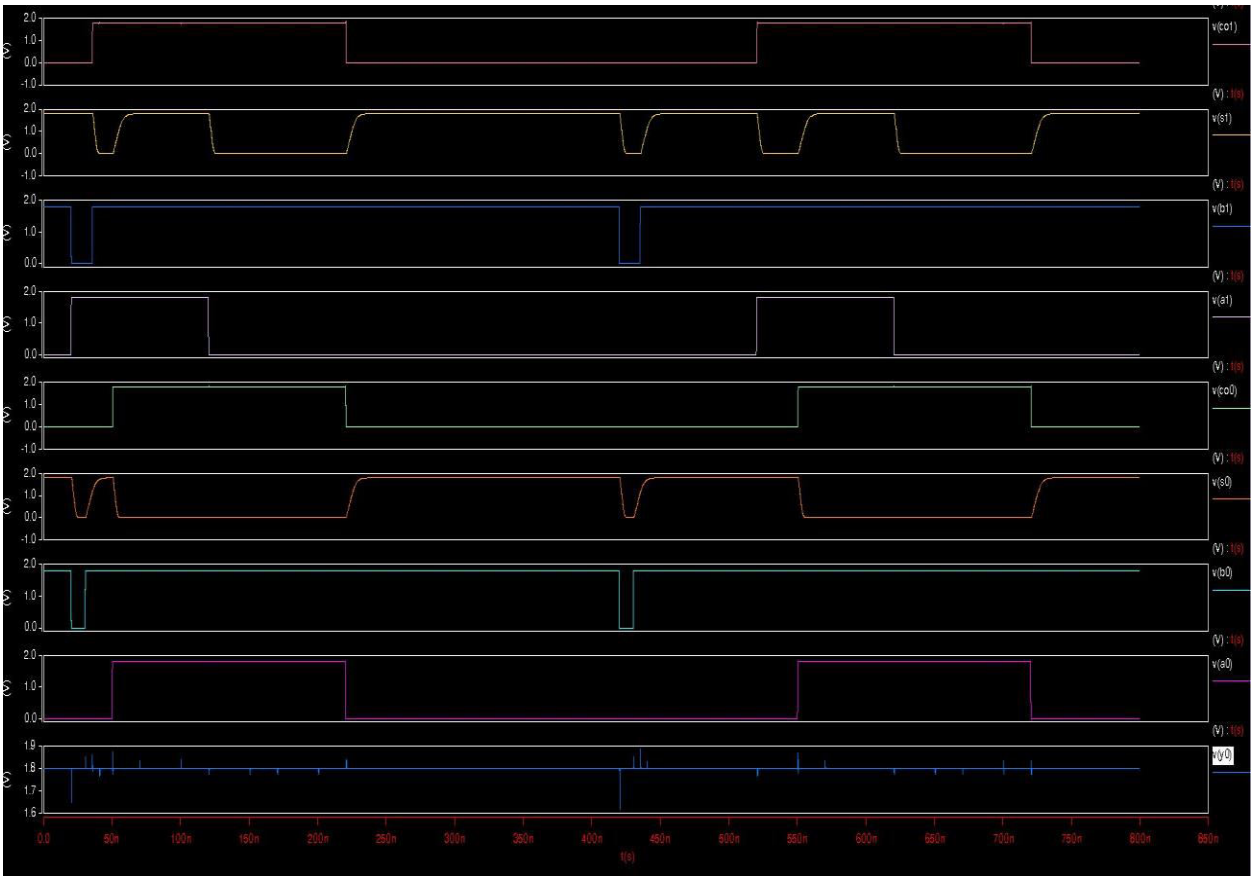
B - SCHEMATICS

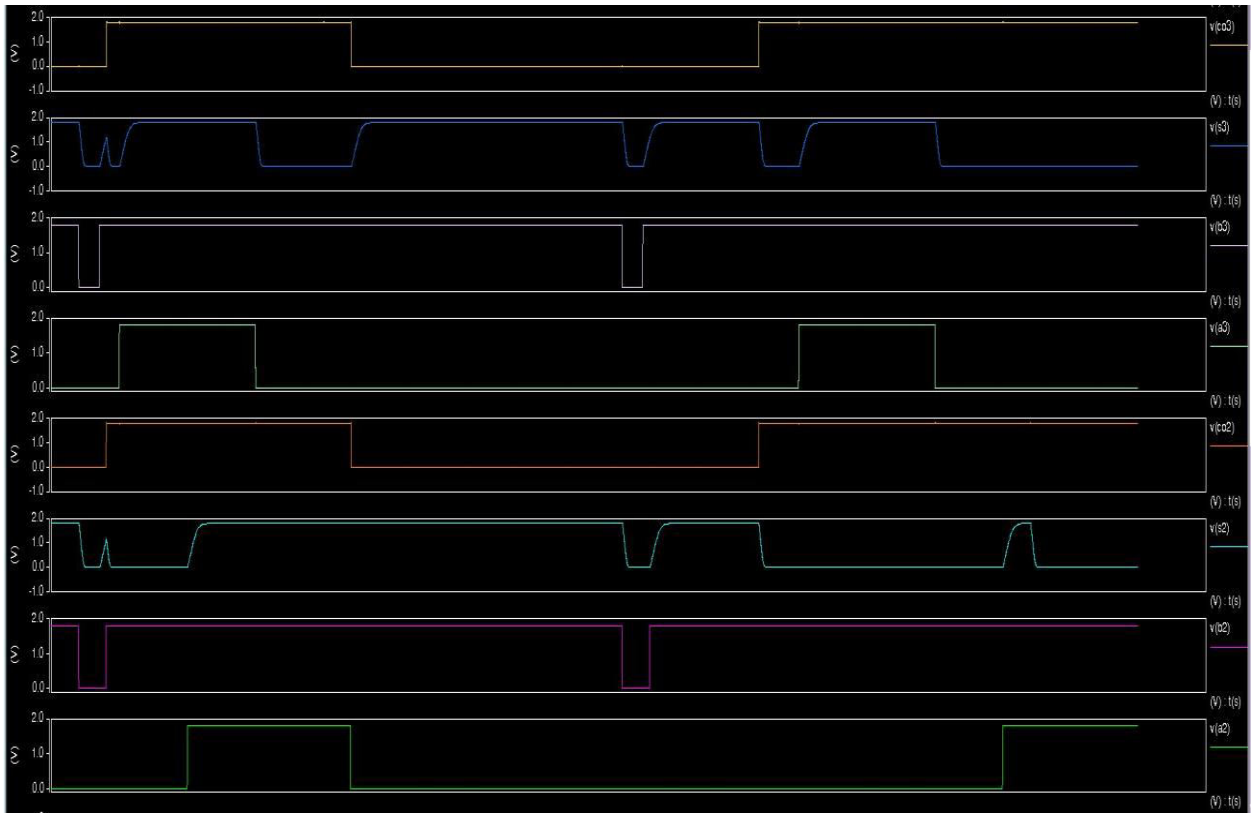


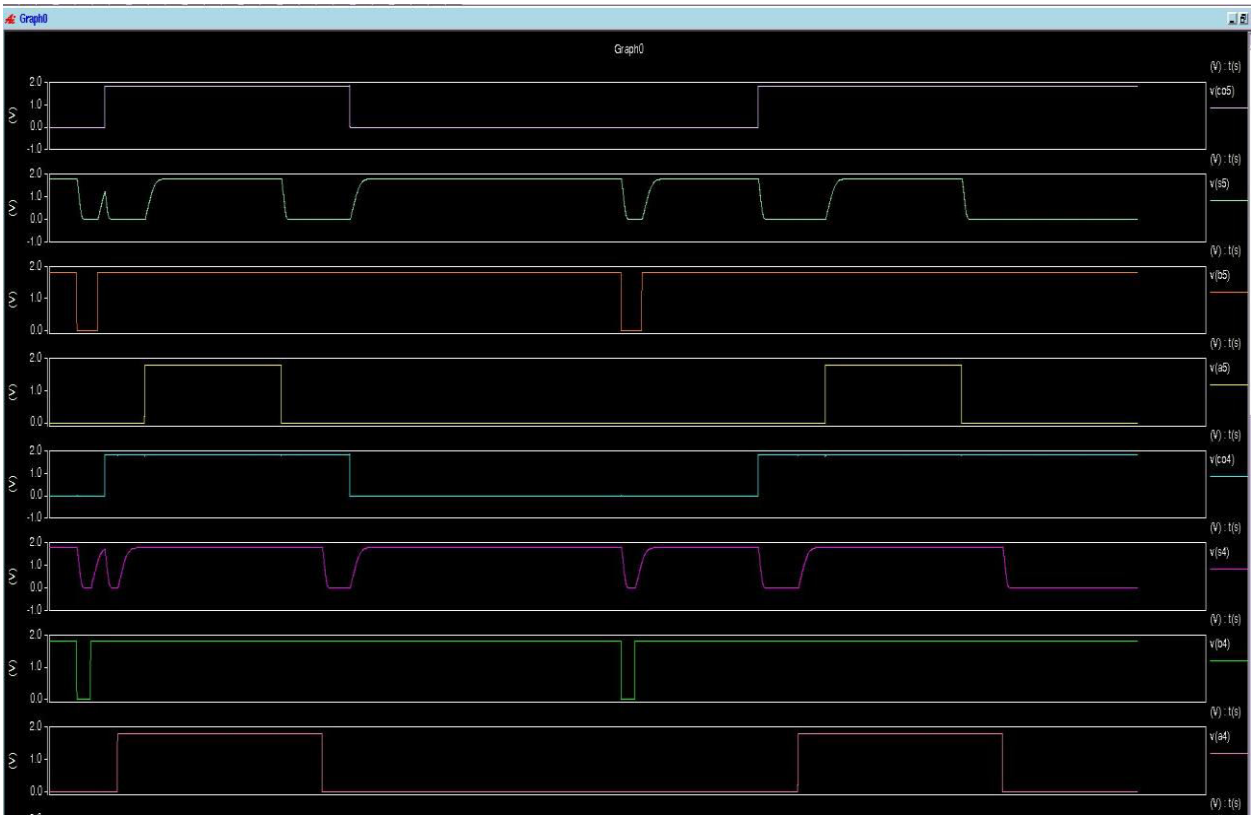


C – WORKING

I am attaching waveforms which proves the correct functionality of my adder block. I have not performed operation of all possible combinations, but the waveforms are sufficient to understand the proper working of my adder block.




v(co3)
v(s3)
v(b3)
v(a3)
v(co2)
v(s2)
v(b2)
v(a2)
(V)
(V) : t(s)


Graph0
Graph0
v(co5)
v(s5)
v(b5)
v(a5)
v(co4)
v(s4)
v(b4)
v(a4)
(V)
(V) : t(s)

D – If I were to redesign this block I will like to redesign it with look ahead carry block as it has less delay. I have done minimum sizing to optimize my design.

E – WORST CASE DELAY, ENERGY CONSUMPTION AND EDP OF ADDER

Initial state = A0 B0 A1 B1 A2 B2 A3 B3 A4 B4 A5 B5 = 0 0 0 0 0 0 0 0 0 0 0 0

Final state = A0 B0 A1 B1 A2 B2 A3 B3 A4 B4 A5 B5 = 1 1 0 1 0 1 0 1 0 1 0 1

For calculation of EDP I used the formula from textbook which says

EDP = (CL*(VDD)^2)/2) * tp

PARAMETER	VALUE
DELAY(LOW TO HIGH)	8.153E-09
ENERGY(LOW TO HIGH)	5.78E-04
EDP	2.64E-20

F - WORST CASE DELAY, ENERGY CONSUMPTION AND EDP OF ADDER FOR 130NM.

For scaling I am using 130nm. Following are the readings for 130nm technology.

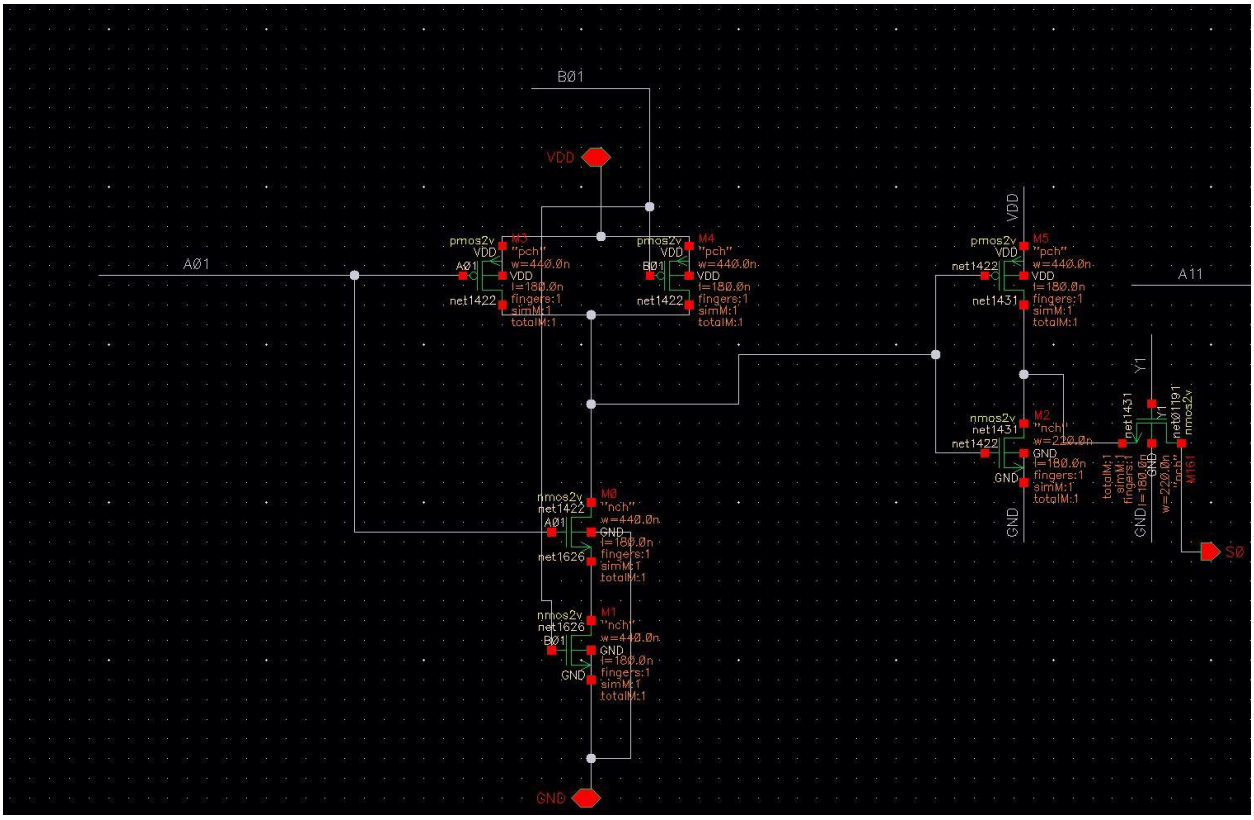
PARAMETER	VALUE
DELAY(LOW TO HIGH)	6.497E-09
ENERGY(LOW TO HIGH)	3.75E-04
EDP	2.105e-20

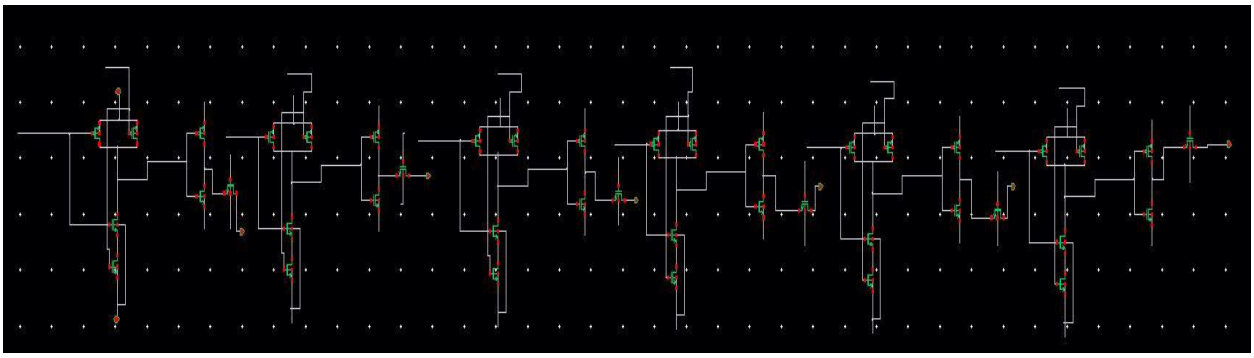
3- LOGICAL AND

A – For logical AND operation I am using normal 2 input AND gate.

B - SCHEMATICS

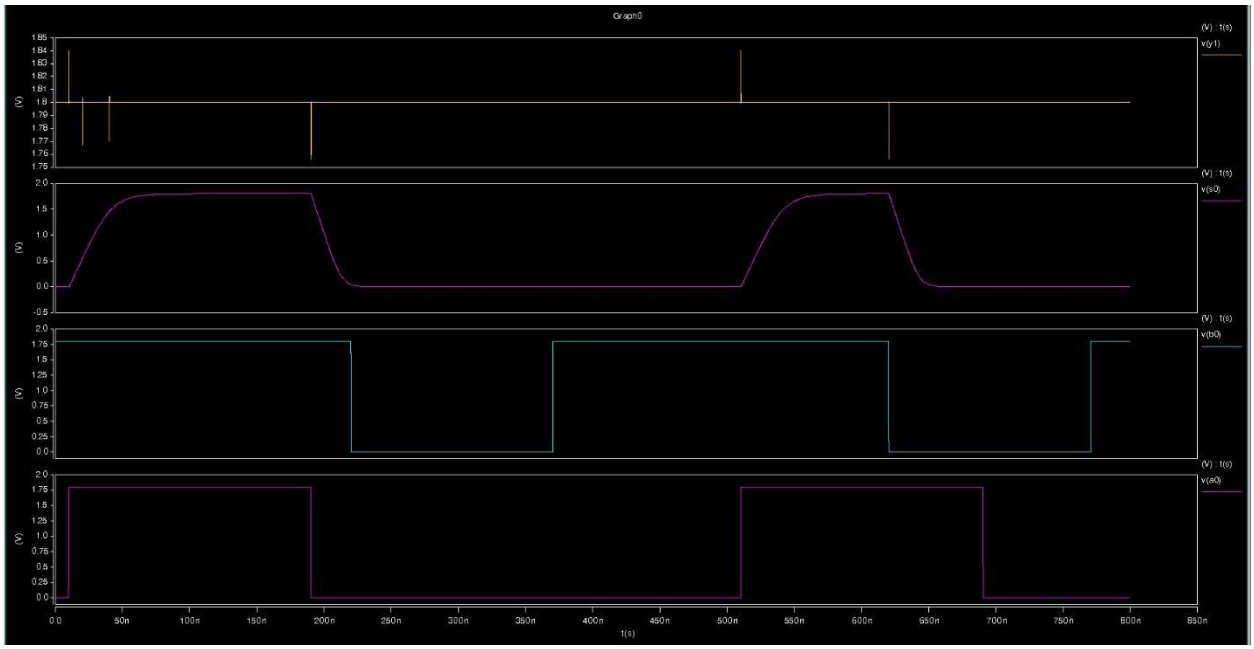
First I am attaching one bit AND gate schematics, for rest 5 bits the structure of AND gate is similar to first bit and then I am attaching full schematics. Pass transistor at the output ensures that only LOGICAL AND is working when Y1 signal from decoder is high.





C – WORKING

For showing the working, I am attaching waveforms which confirms correct functionality of my LOGICAL AND. The slope in the output curve is because of 2pf capacitor at the output. I am attaching output of only first bit because from second bit onwards I have used same schematics, so outputs will be same for every stage.



D – To optimize my AND block, I have reduced fan in by converting it to 2 input AND gate instead of 3 input AND gate.

E - WORST CASE DELAY, ENERGY CONSUMPTION AND EDP OF AND GATE

Initial state = A0 B0 A1 B1 A2 B2 A3 B3 A4 B4 A5 B5 = 0 0 0 0 0 0 0 0 0 0 0 0

Final state = A0 B0 A1 B1 A2 B2 A3 B3 A4 B4 A5 B5 = 1 1 1 1 1 1 1 1 1 1 1 1

I am using EDP formula as

EDP = (CL*(VDD)^2)/2) * tp

PARAMETER	VALUE
DELAY(LOW TO HIGH)	2.97E-09
ENERGY(LOW TO HIGH)	3.422E-04
EDP	9.59E-21

F - WORST CASE DELAY, ENERGY CONSUMPTION AND EDP OF AND GATE FOR 130nm

As we can observe after scaling, EDP decreases.

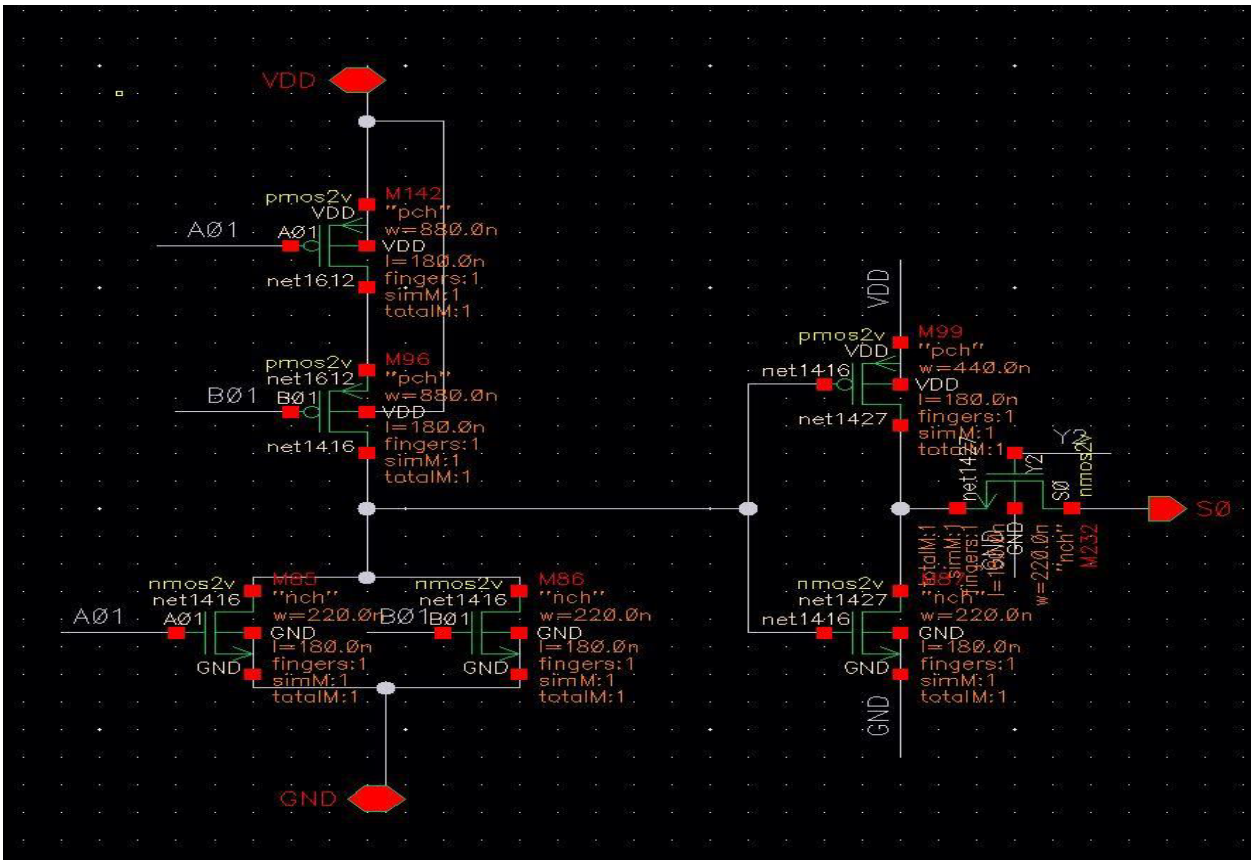
PARAMETER	VALUE
DELAY(LOW TO HIGH)	1.911E-09
ENERGY(LOW TO HIGH)	1.503E-04
EDP	6.1916E-21

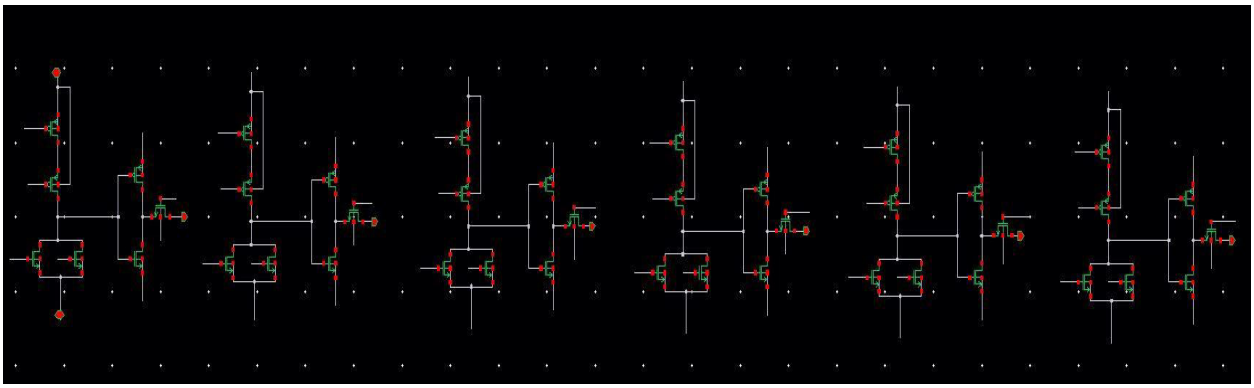
4- LOGICAL OR

A -For LOGICAL OR operation I am using basic 2 input OR gate.

B – SCHEMATICS

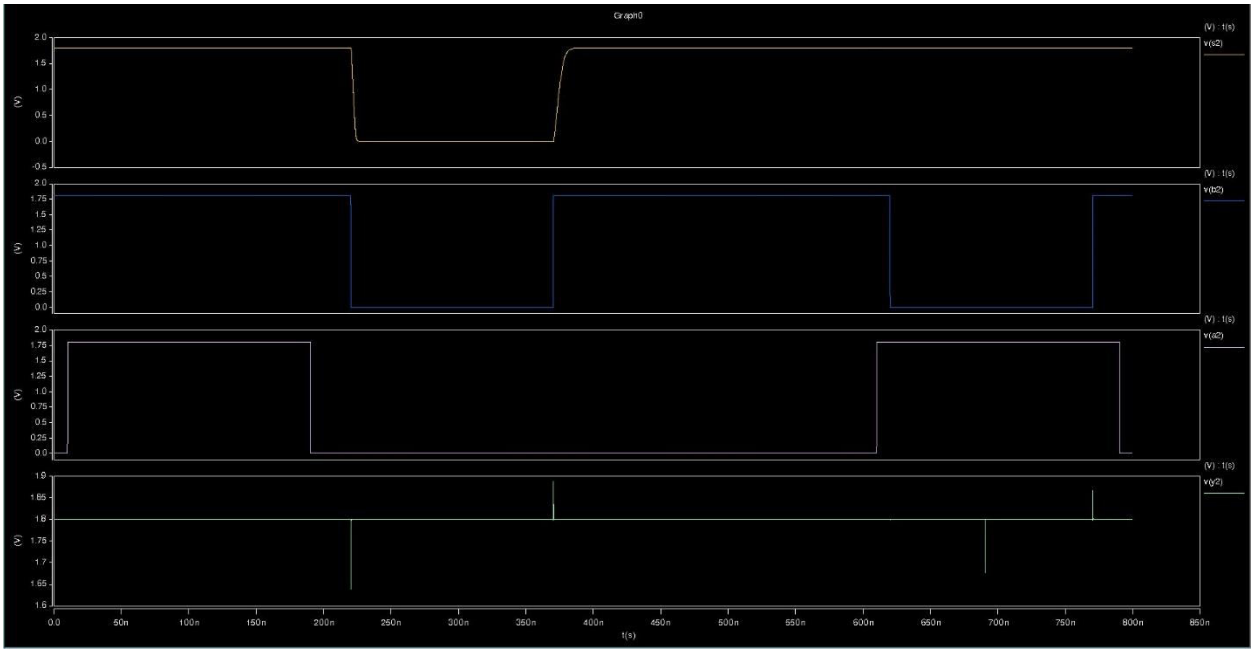
First I am attaching schematics of 1 bit. I have used same schematics for rest of the bits. A pass transistor at the end ensures that only OR gate is working when Y2 logic is high.





C – WORKING

I am attaching waveforms which is sufficient to show the functioning. I am attaching functioning of 1 bit. Functioning for rest of beats is same.



D – I have reduced fan-in by reducing number of inputs to OR gate. I have used 2 input OR gate instead of 3 input OR gate to reduce fan-in.

E - WORST CASE DELAY, ENERGY CONSUMPTION AND EDP OF OR GATE

Initial state = A0 B0 A1 B1 A2 B2 A3 B3 A4 B4 A5 B5 = 0 0 0 0 0 0 0 0 0 0 0 0

Final state = A0 B0 A1 B1 A2 B2 A3 B3 A4 B4 A5 B5 = 1 1 1 1 1 1 1 1 1 1 1 1

PARAMETER	VALUES
DELAY(LOW TO HIGH)	2.97E-09
ENERGY(LOW TO HIGH)	1.98E-04
EDP	9.6228E-21

F - WORST CASE DELAY, ENERGY CONSUMPTION AND EDP OF OR GATE FOR 130nm

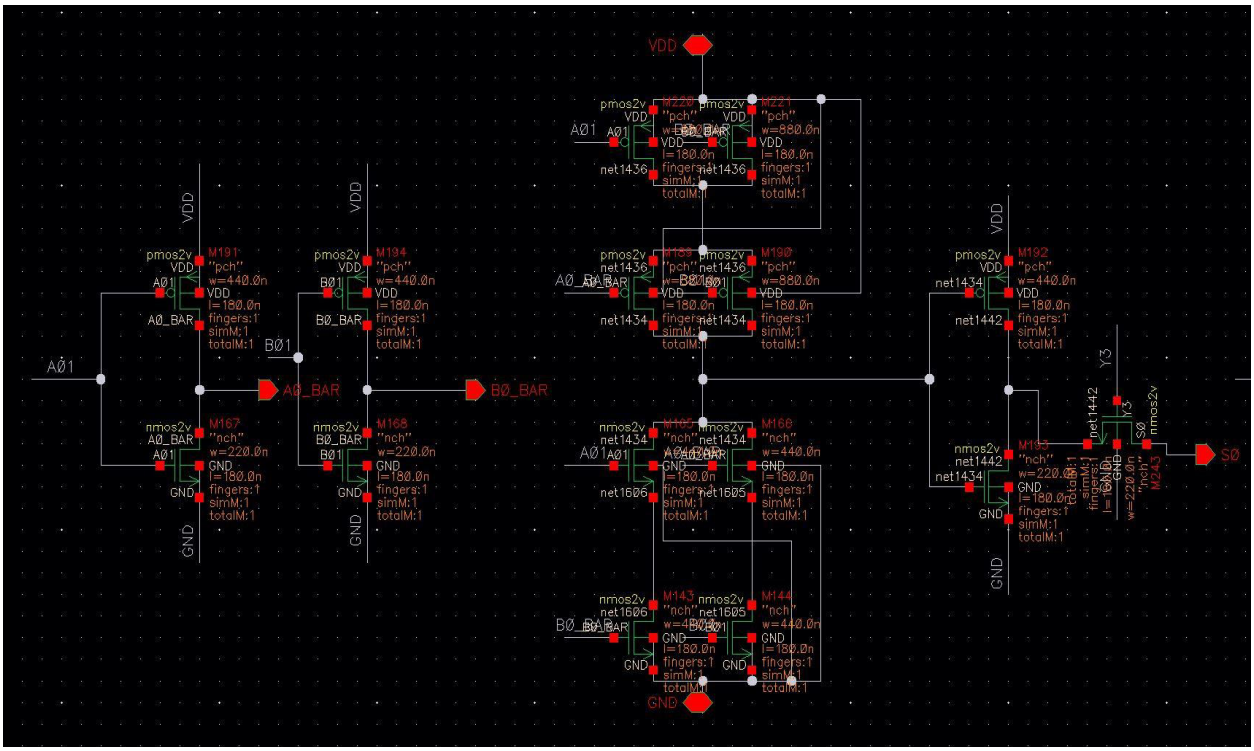
PARAMETER	VALUE
DELAY(LOW TO HIGH)	1.83E-09
ENERGY(LOW TOHIGH)	2.871E-04
EDP	5.9292E-21

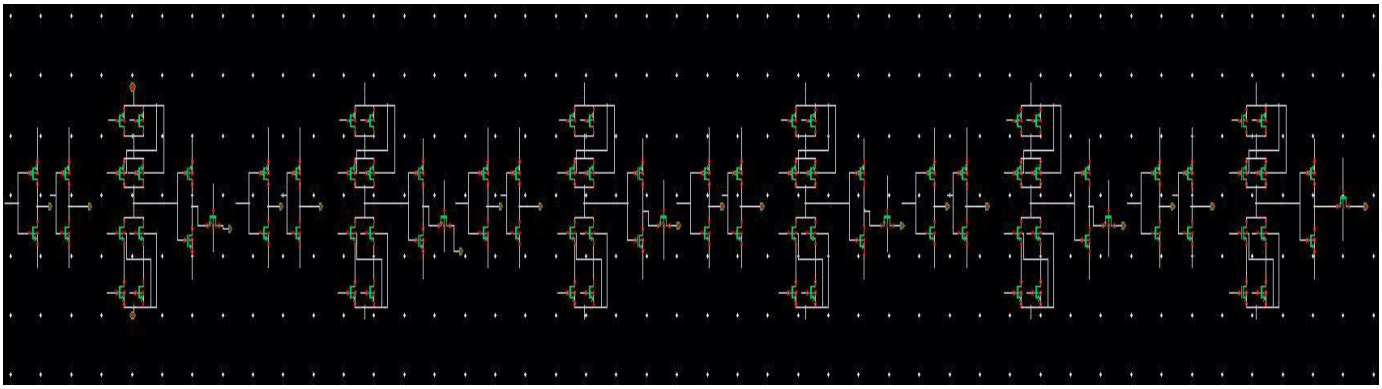
5- LOGICAL XOR

A – I have used normal 2 input XOR gate (A.B_BAR + B.A_BAR).

B – SCHEMATICS

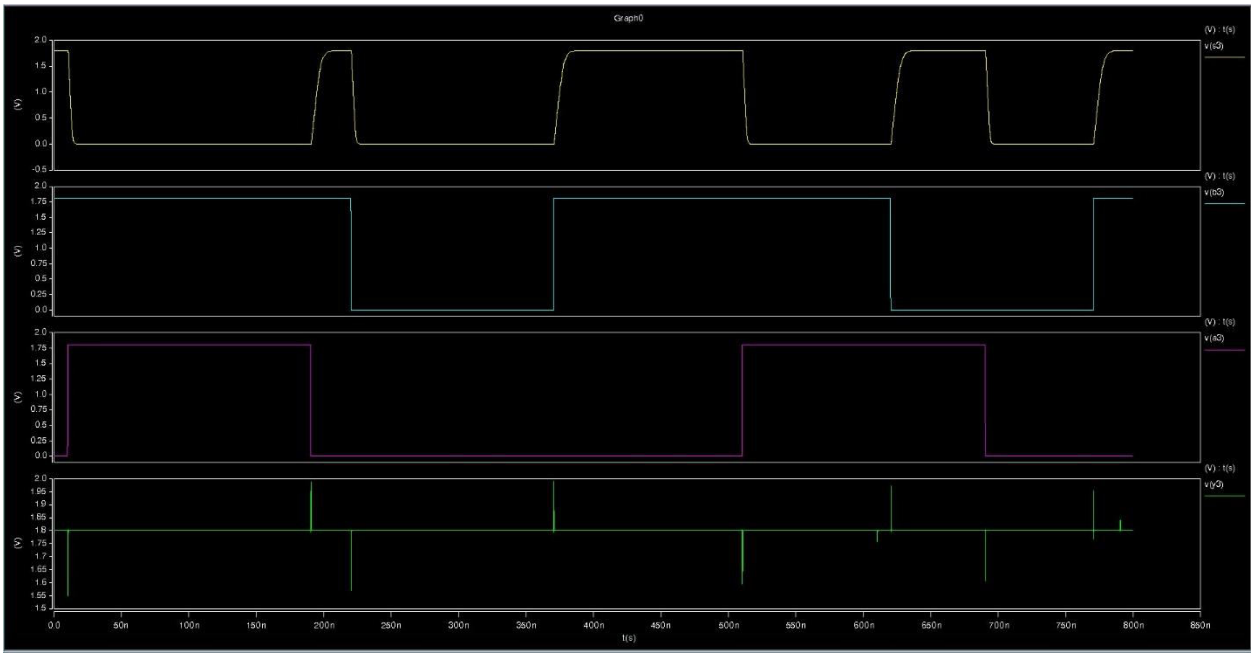
I am attaching schematics of 1 bit for clearer picture. Rest all bits follow same schematics.





C – WORKING

I am showing one bit operation of XOR gate, and rest of bits function in the similar manner.



D - I did minimum sizing in XOR gate to optimize the block.

E - WORST CASE DELAY, ENERGY CONSUMPTION AND EDP OF XOR GATE

Initial state = A0 B0 A1 B1 A2 B2 A3 B3 A4 B4 A5 B5 = 0 0 1 1 0 0 1 1 0 0 1 1

Final state = A0 B0 A1 B1 A2 B2 A3 B3 A4 B4 A5 B5 = 0 1 0 1 0 1 0 1 0 1 0 1

PARAMETER	DELAY
DELAY(LOW TO HIGH)	3.087E-09
ENERGY(LOW TO HIGH)	1.986E-04
EDP	1.0001E-20

F - WORST CASE DELAY, ENERGY CONSUMPTION AND EDP OF XOR GATE FOR 130nm

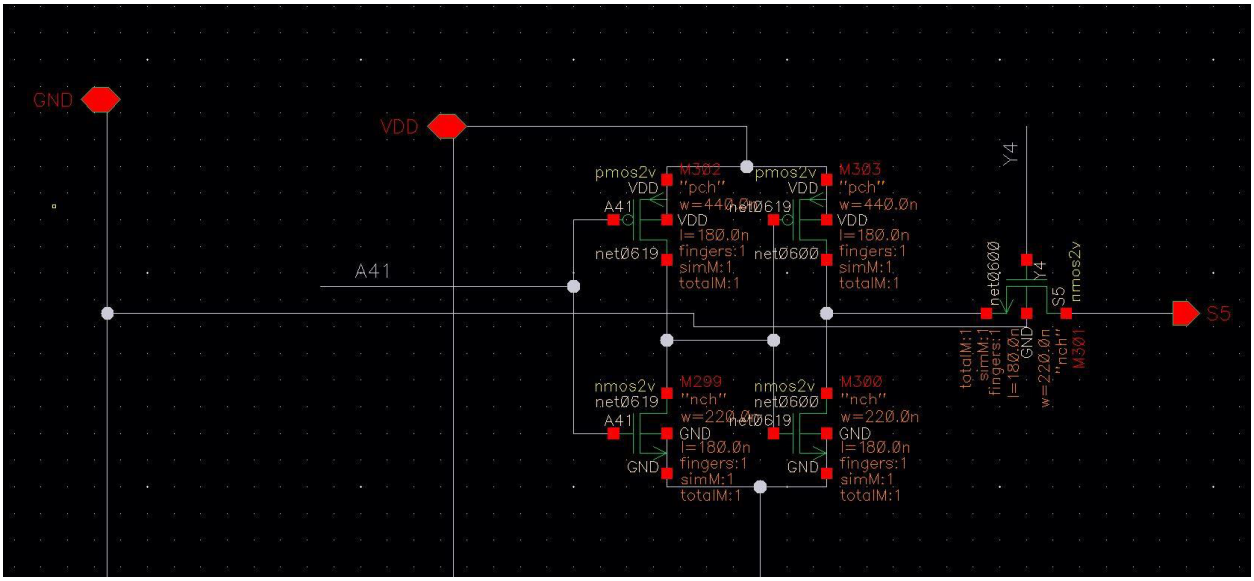
PARAMETER	VALUE
DELAY	1.935E-09
ENERGY	2.303E-04
EDP	6.2694E-21

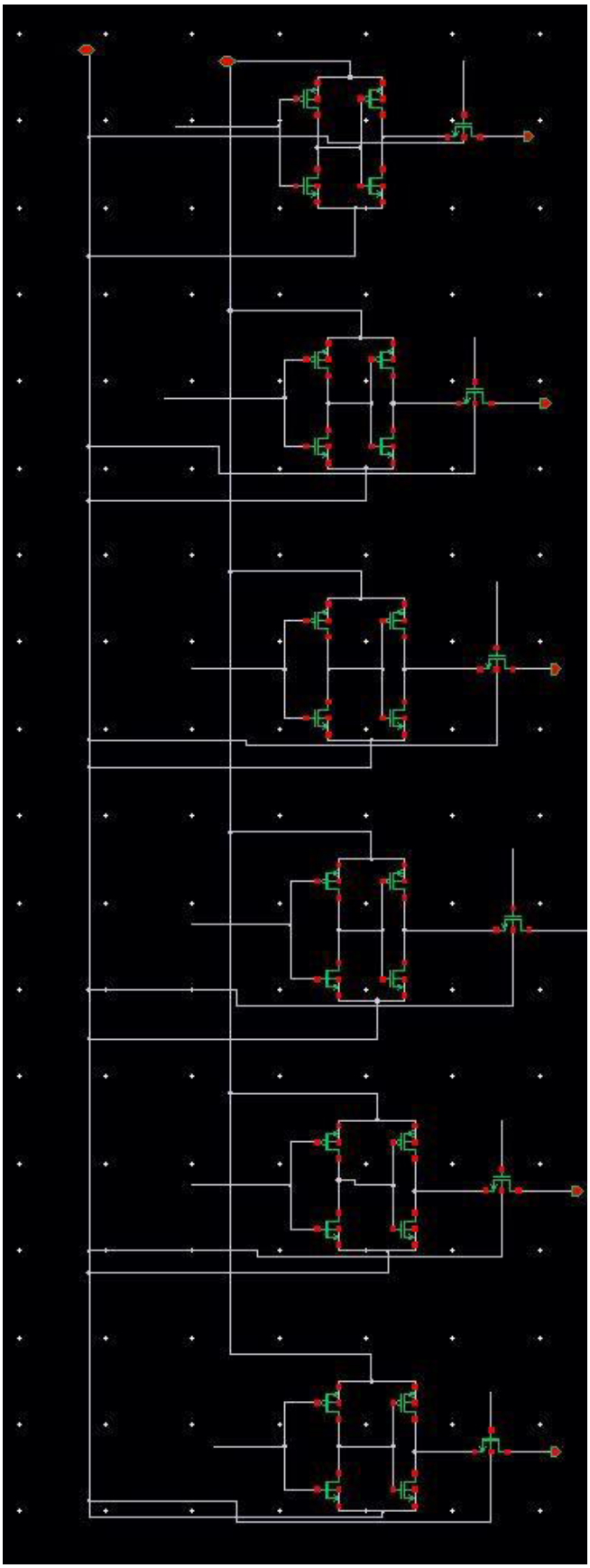
6- ROTATE LEFT

A – I have connected single hard wire since we only have to do single rotate. Inputs and outputs are controlled by pass transistor.

B – SCHEMATICS

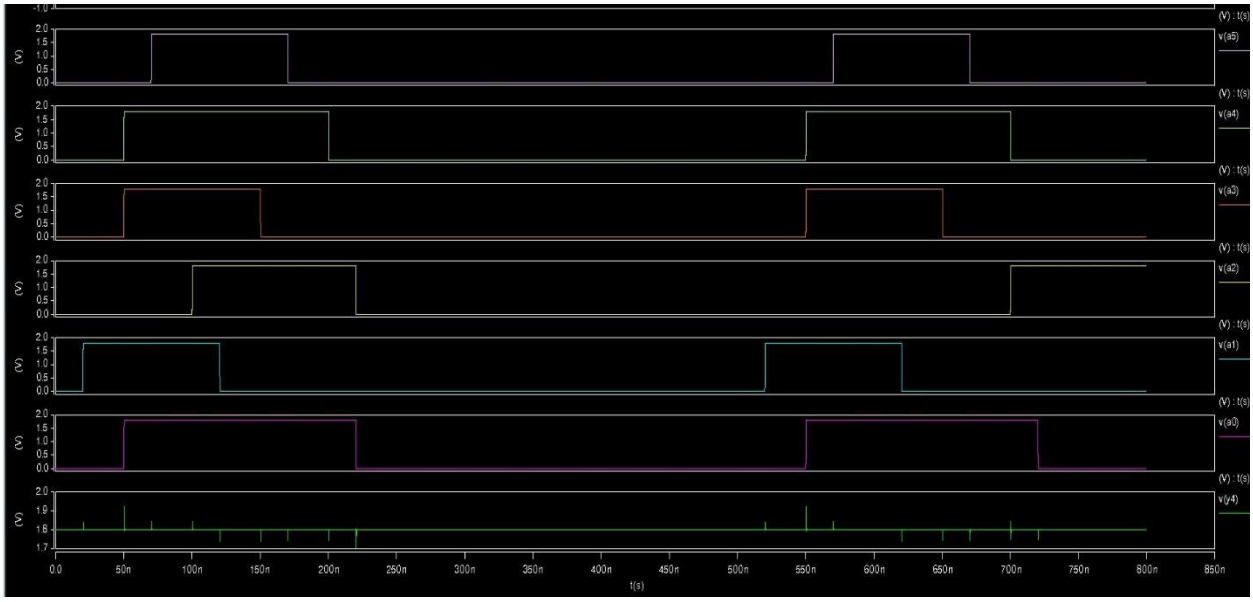
First I am adding one bit schematics to have a closer look and then I am attaching full schematics. Rest all bits follows same schematics.

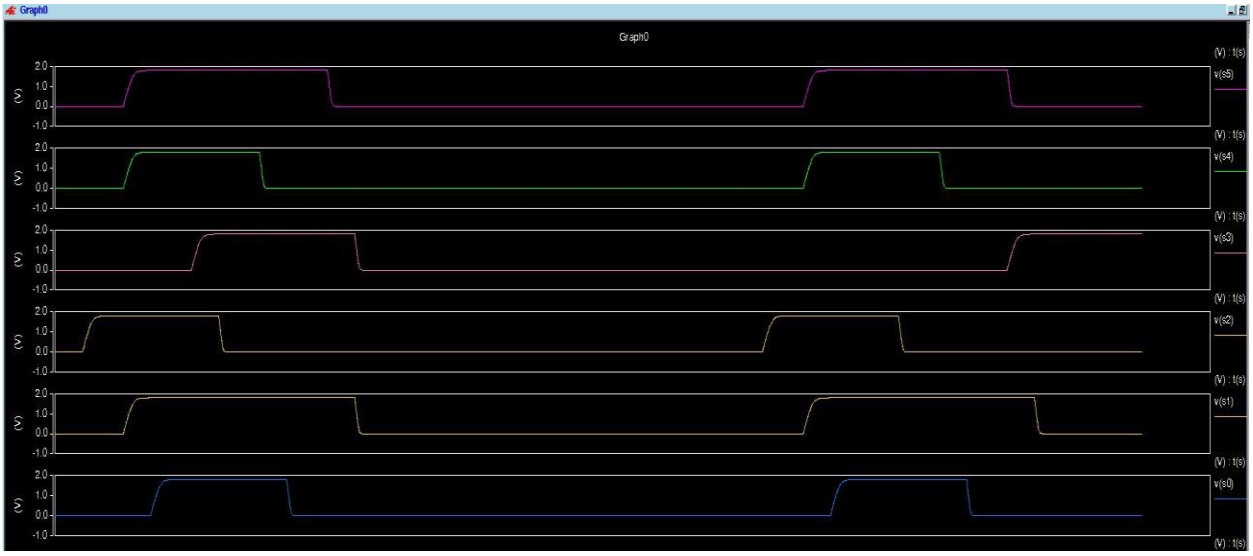




C – WORKING

Graphs showing correct functionality are as follows. First I am attaching all input bits and then rotated bits.





D – I reduced number of transistors by just connecting hard wire to inputs and outputs with buffer in between.

E - WORST CASE DELAY, ENERGY CONSUMPTION AND EDP OF ROTATE LEFT

I have considered worst as

Initial state = A0 A1 A2 A3 A4 A5 = 0 1 0 1 0 1

Final state = A0 A1 A2 A3 A4 A5 =1 0 1 0 1 0

PARAMETER	VALUE
DELAY	2.66E-09
ENERGY	3.89E-04
EDP	8.61E-21

F - WORST CASE DELAY, ENERGY CONSUMPTION AND EDP OF ROTATE LEFT FOR 130nm

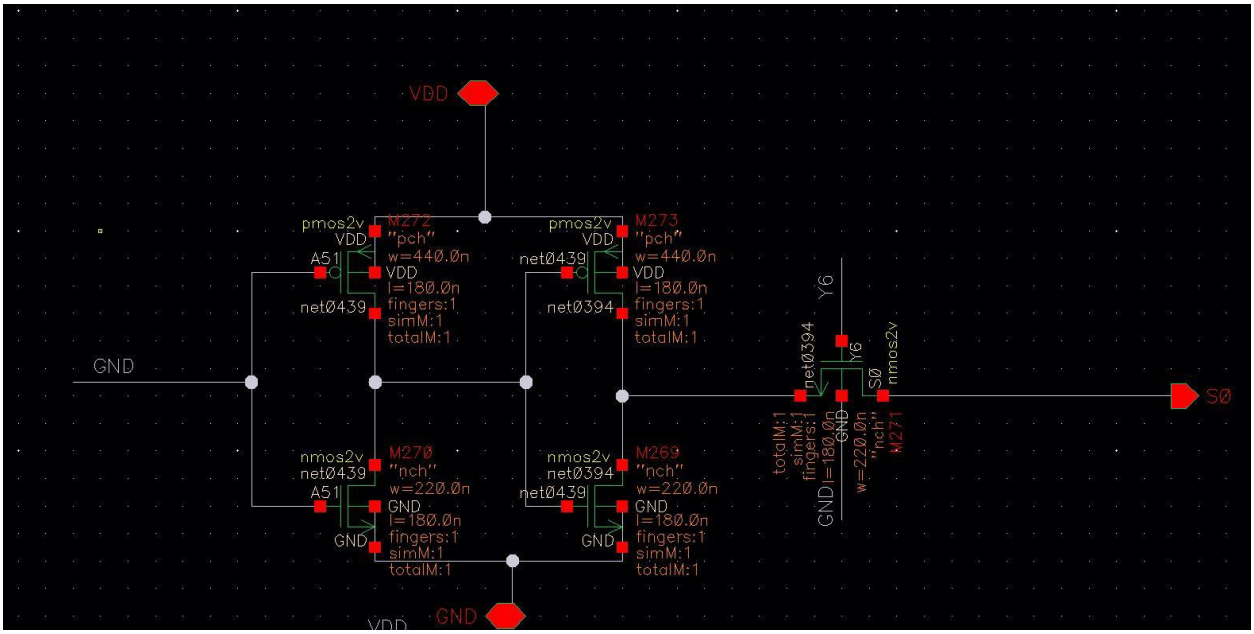
PARAMETER	VALUE
DELAY	1.75E-09
ENERGY	3.15E-04
EDP	5.67E-21

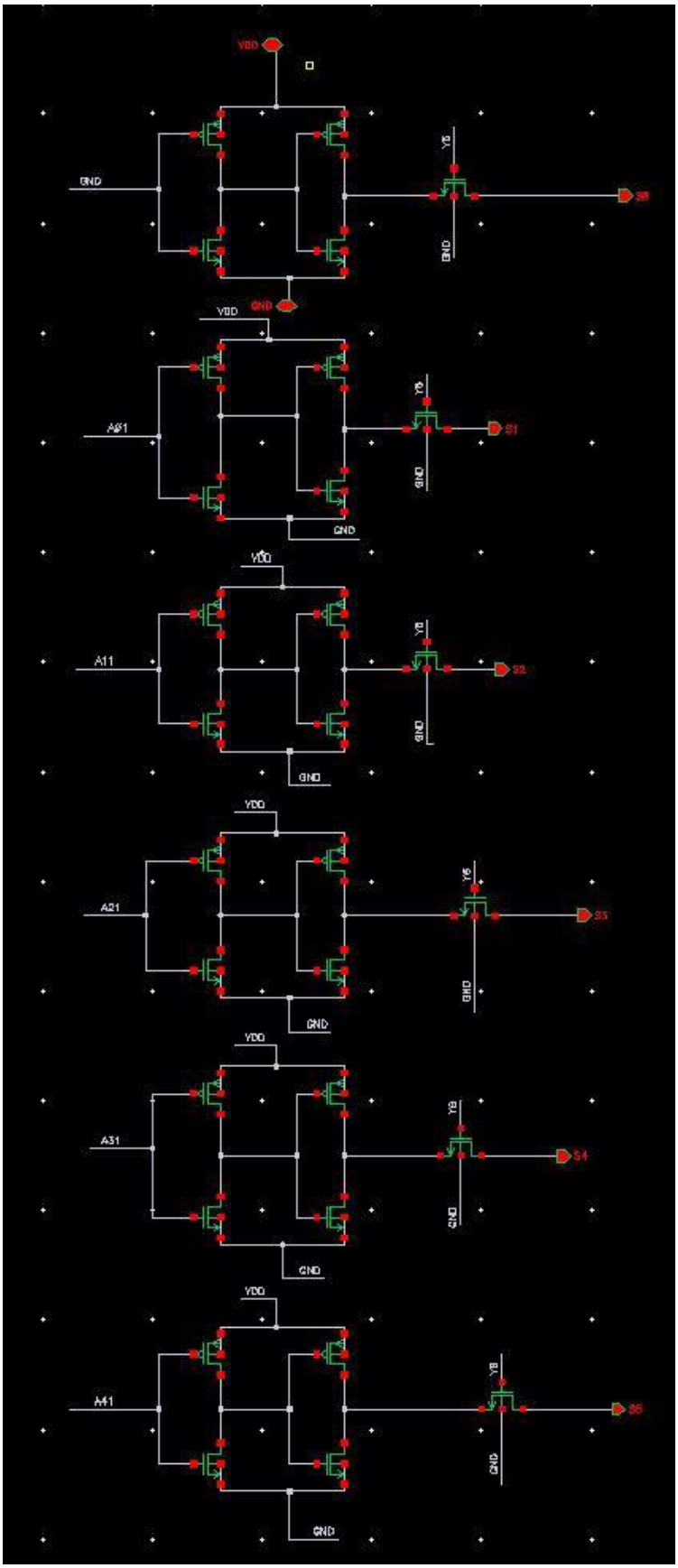
7- LOGICAL SHIFT

A – For logical shift I have implemented same schematics as of rotate left, just first bit of input is grounded as we need to append it with zero when the shift operation is going on.

B – SCHEMATICS

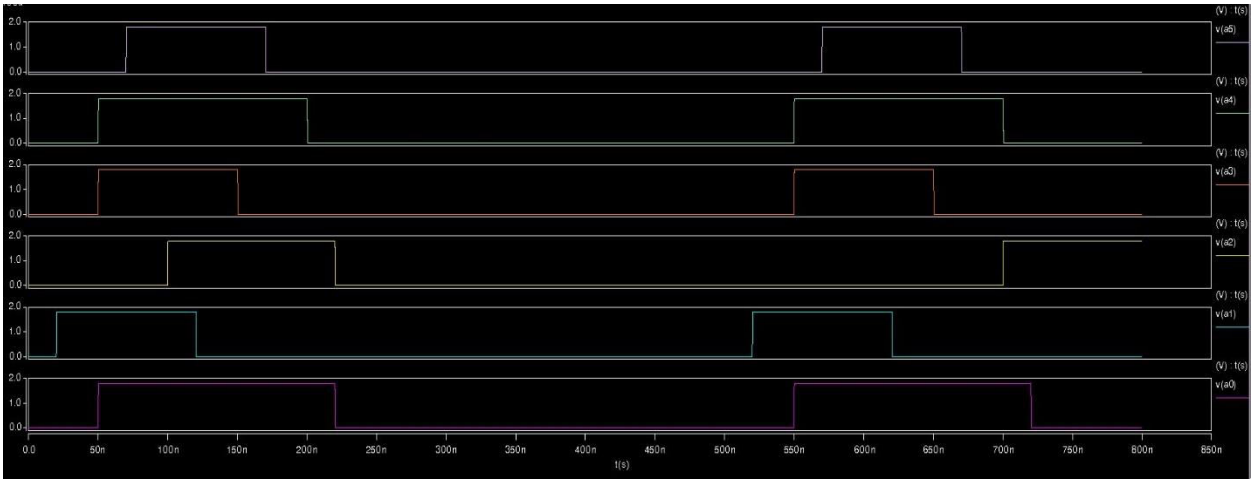
First I am attaching one bit schematics and then the full schematics.

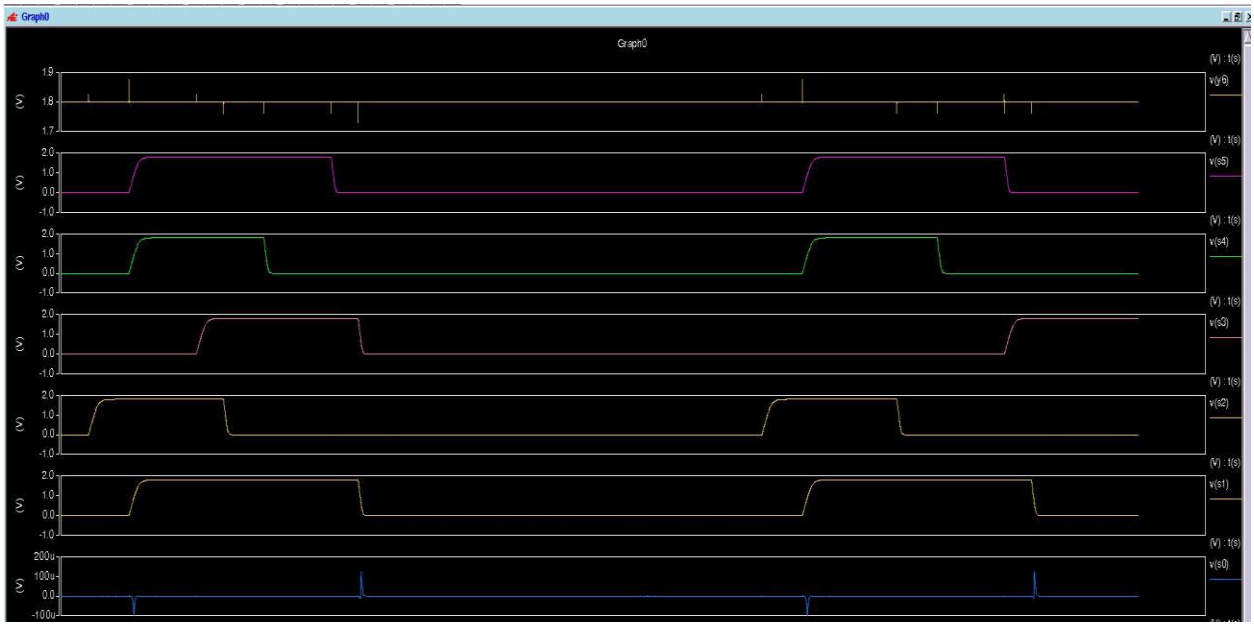



VDD
GND
A01
A11
A21
A31
A41
S0
S1
S2
S3
S4
S5

C – WORKING

I am attaching waveforms to prove the proper functionality.





D – By implementing shifter on hard wire I have reduced the number of transistors.

E - WORST CASE DELAY, ENERGY CONSUMPTION AND EDP OF SHIFT LEFT

I have considered worst as

Initial state = A0 A1 A2 A3 A4 A5 = 0 1 0 1 0 1

Final state = A0 A1 A2 A3 A4 A5 =1 0 1 0 1 0

PARAMETERS	VALUE
DELAY	2.63E-09
ENERGY	3.889E-04
EDP	8.52E-21

E - WORST CASE DELAY, ENERGY CONSUMPTION AND EDP OF SHIFT LEFT 130NM

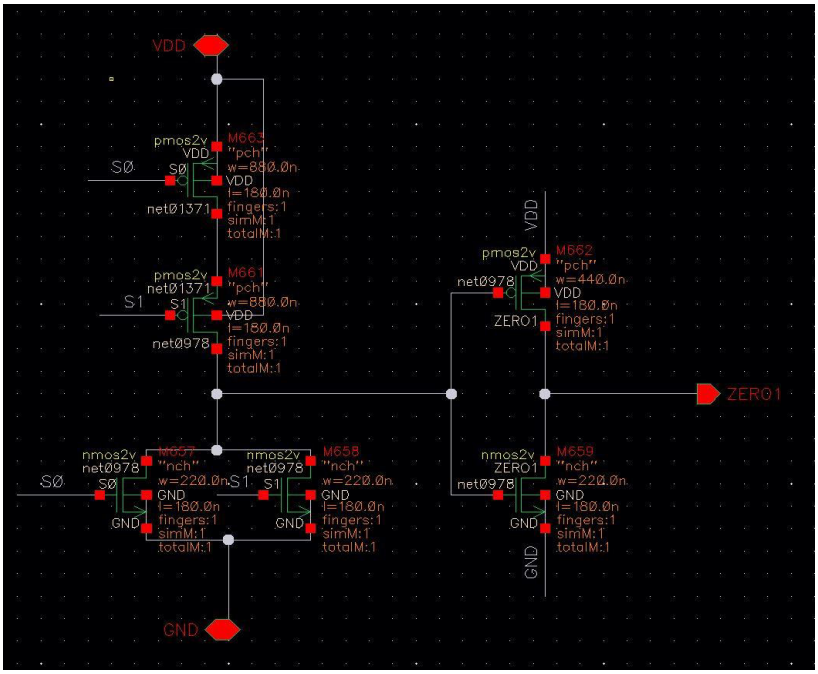
PARAMETER	VALUE
DELAY	1.75E-09
ENERGY	3.15E-04
EDP	5.67E-21

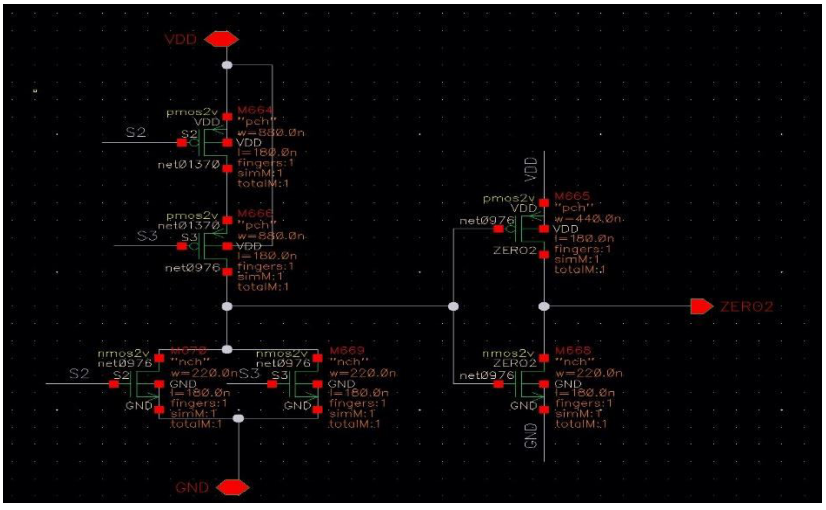
8- FLAGS

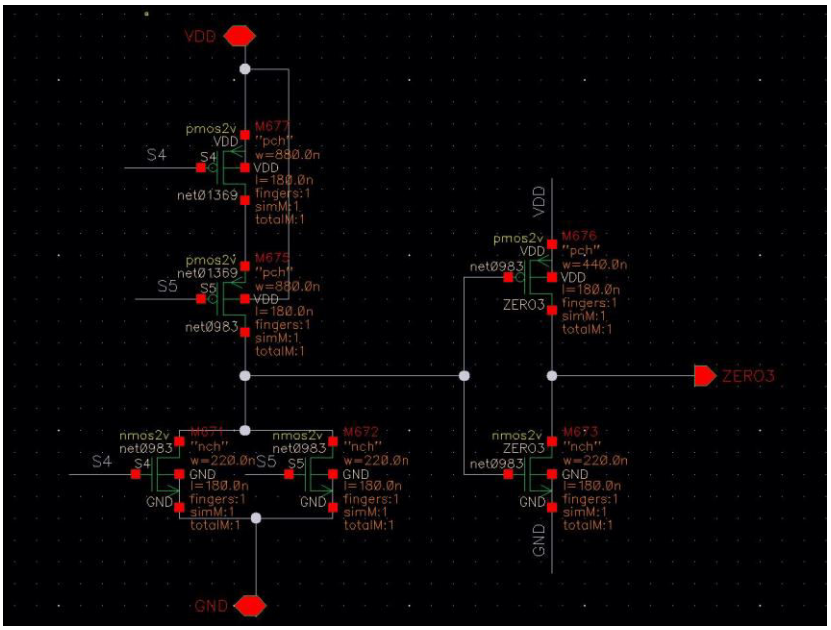
A – ZERO FLAG

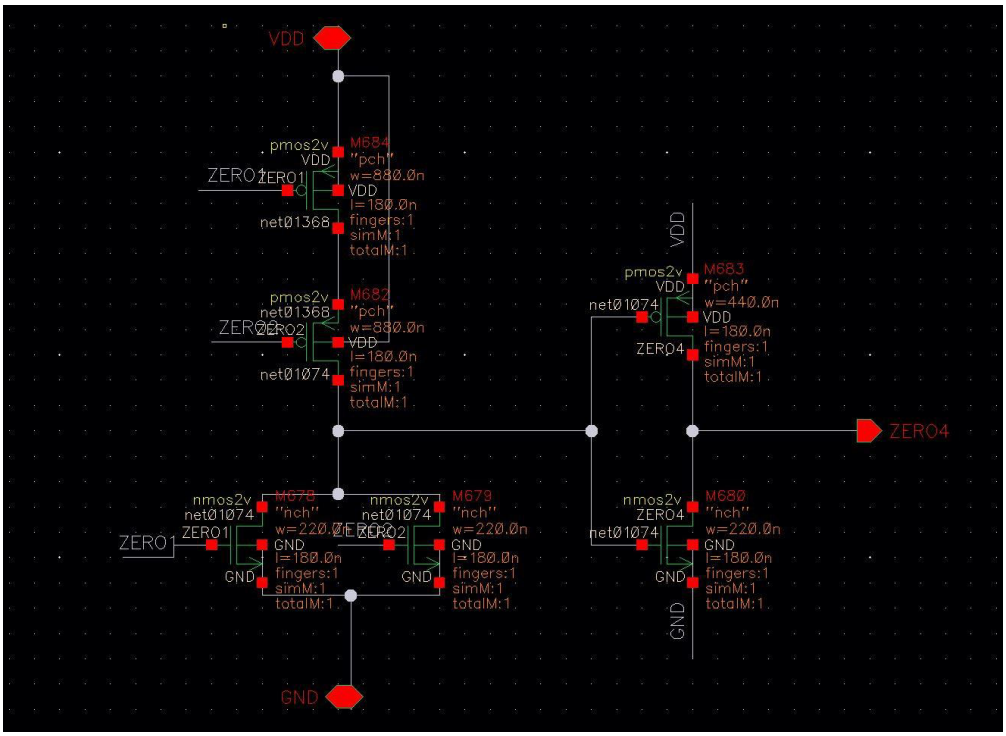
i- Schematics of zero flag is as follows

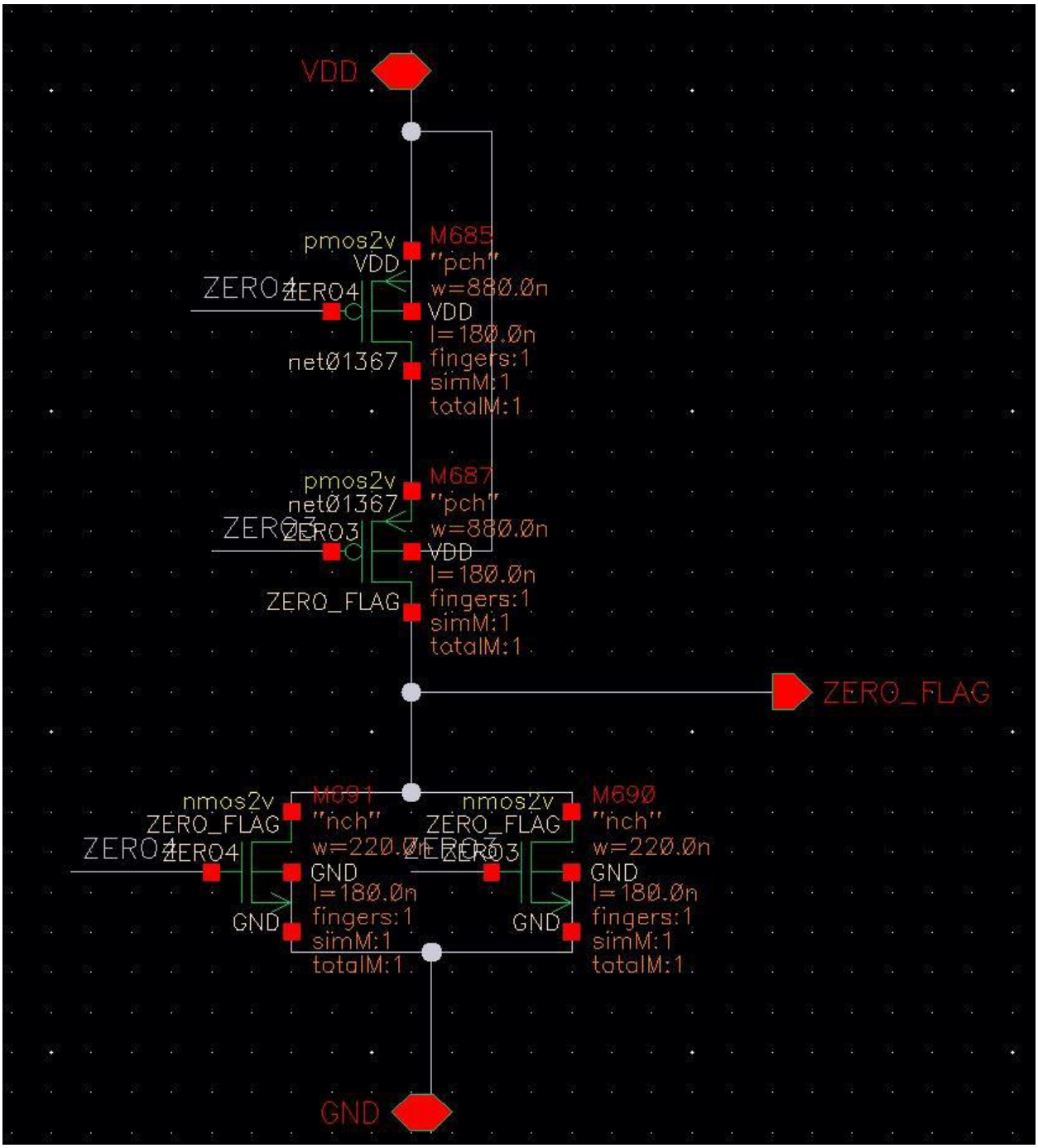
For zero flag, I am using basic OR gate. All the six output bits of ALU are fed to 3 different OR gates and if all the outputs are zero, then zero_flag bit goes high. I have split six input OR gate into three 2-input OR gates.





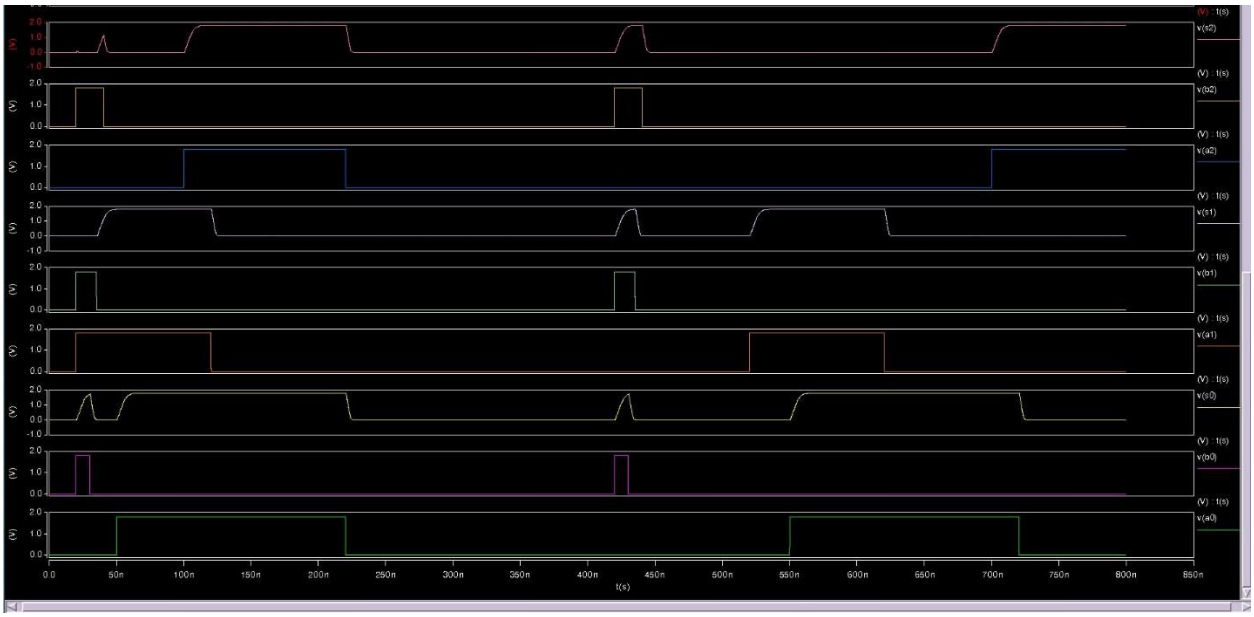

VDD
pmos2v
M677
"pch"
w=880.0n
S4
VDD
l=180.0n
fingers:1
net01369
simM:1
totalM:1
M675
S5
net0983
M676
net0983
ZERO3
w=440.0n
M671
"nch"
w=220.0n
nmos2v
GND
M672
M673
ZERO3
ZERO3
GND

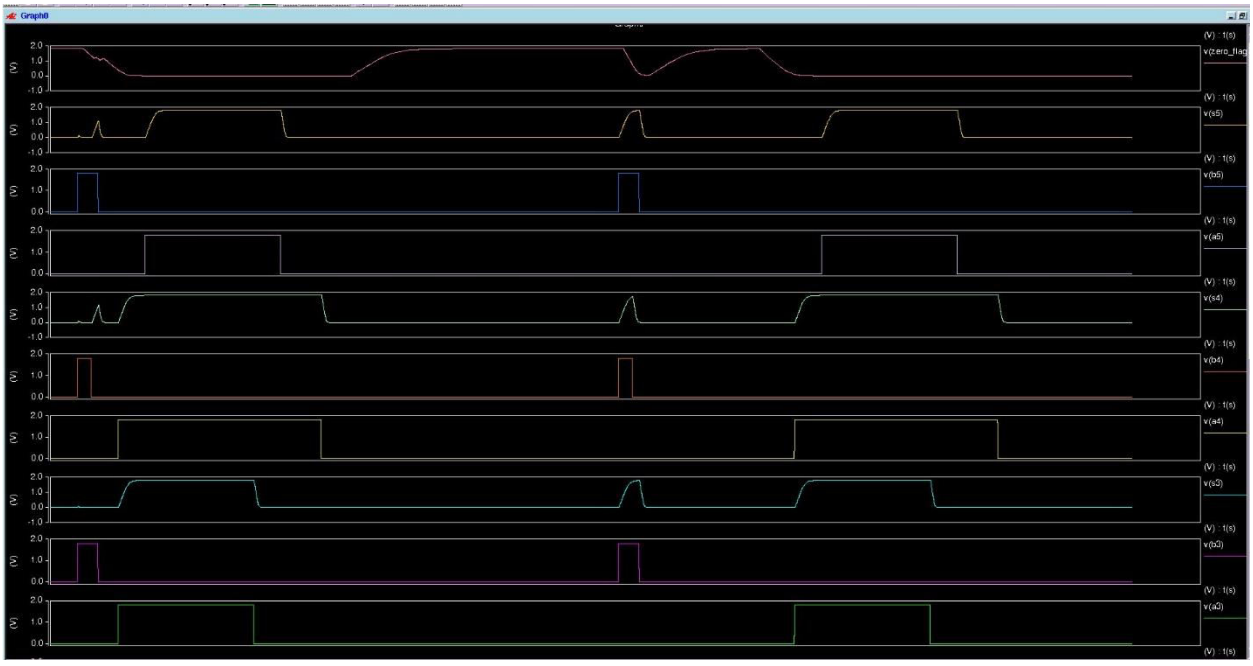

VDD
pmos2v
M684
"pch"
w=880.0n
ZERO1
VDD
l=180.0n
fingers:1
net01368
simM:1
totalM:1
M682
net01368
ZERO2
net01074
M683
net01074
ZERO4
w=440.0n
ZERO4
ZERO4
M678
"nch"
w=220.0n
nmos2v
GND
M679
net01074
ZERO1
ZERO2
M680
ZERO1
GND


VDD
pmos2v
VDD
M685
"pch"
w=880.0n
ZERO4
ZERO4
VDD
l=180.0n
net01367
fingers:1
simM:1
totalM:1
pmos2v
net01367
M687
"pch"
ZERO3
ZERO3
w=880.0n
VDD
l=180.0n
ZERO_FLAG
fingers:1
simM:1
totalM:1
ZERO_FLAG
nmos2v
ZERO_FLAG
M691
"nch"
w=220.0n
ZERO4
ZERO4
GND
l=180.0n
GND
fingers:1
simM:1
totalM:1
nmos2v
ZERO_FLAG
M690
"nch"
ZERO3
ZERO3
w=220.0n
GND
l=180.0n
GND
fingers:1
simM:1
totalM:1
GND

ii- Functionality

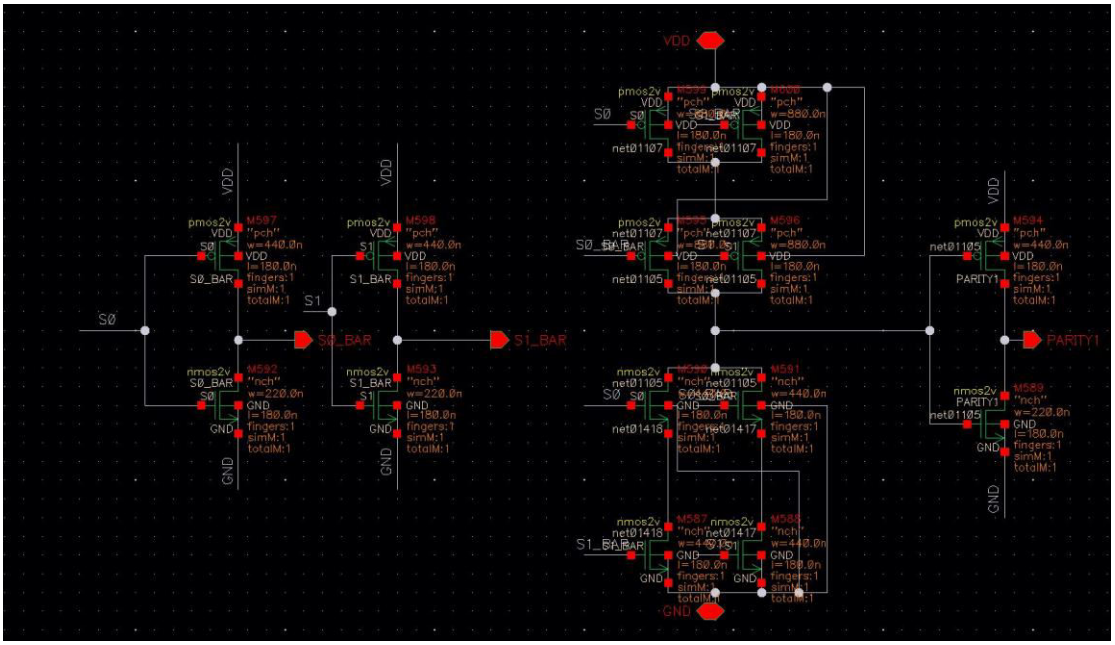
Graph below shows how zero_flag is set high when all the outputs from ALU are zero.

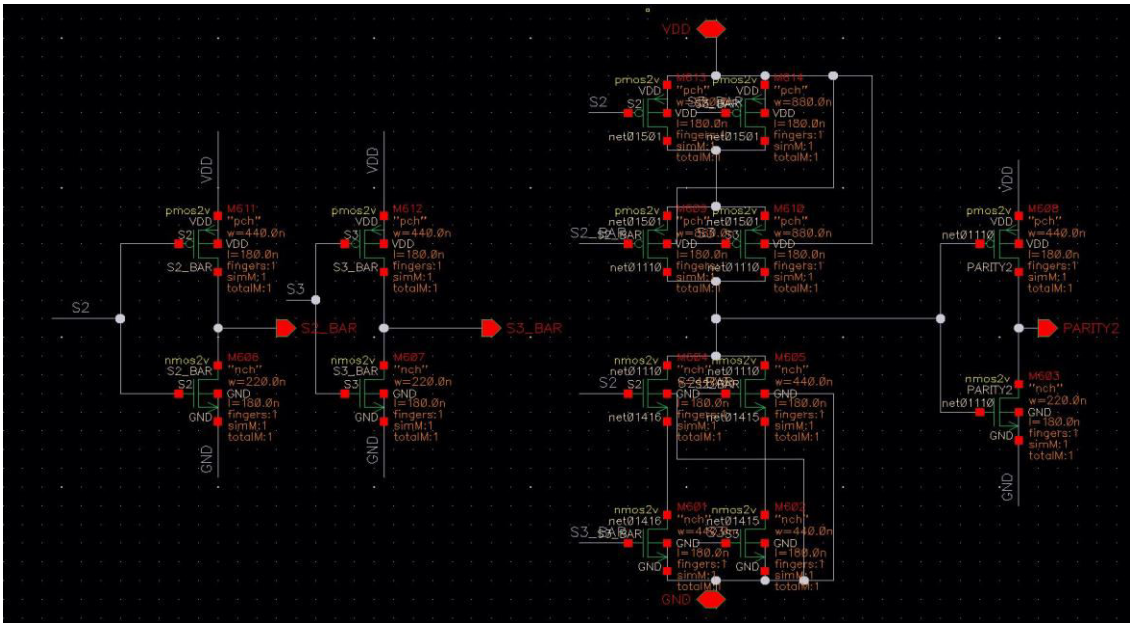


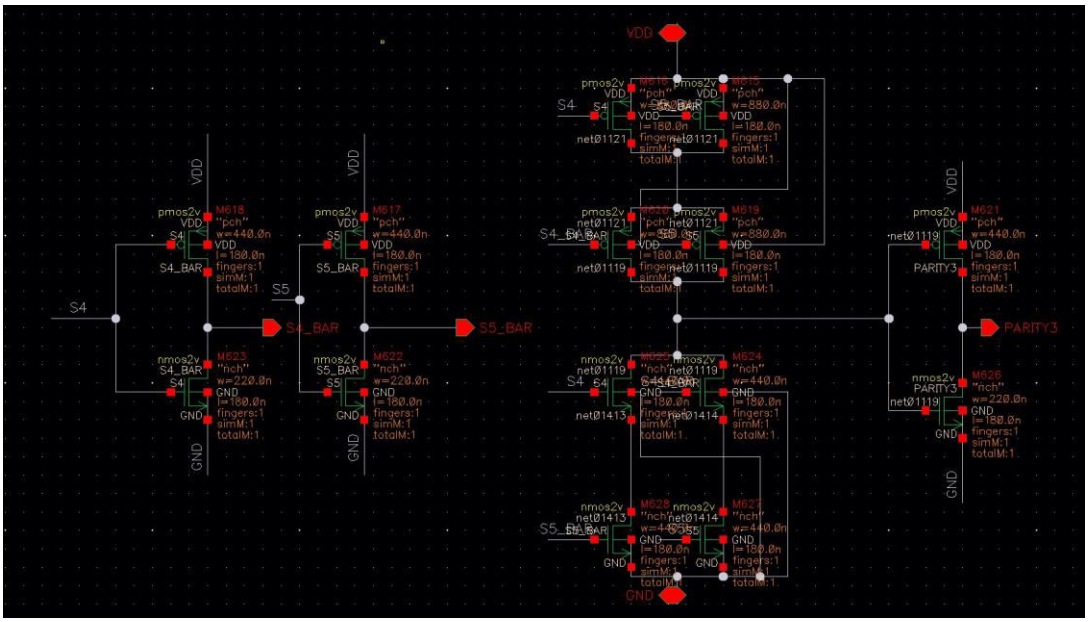


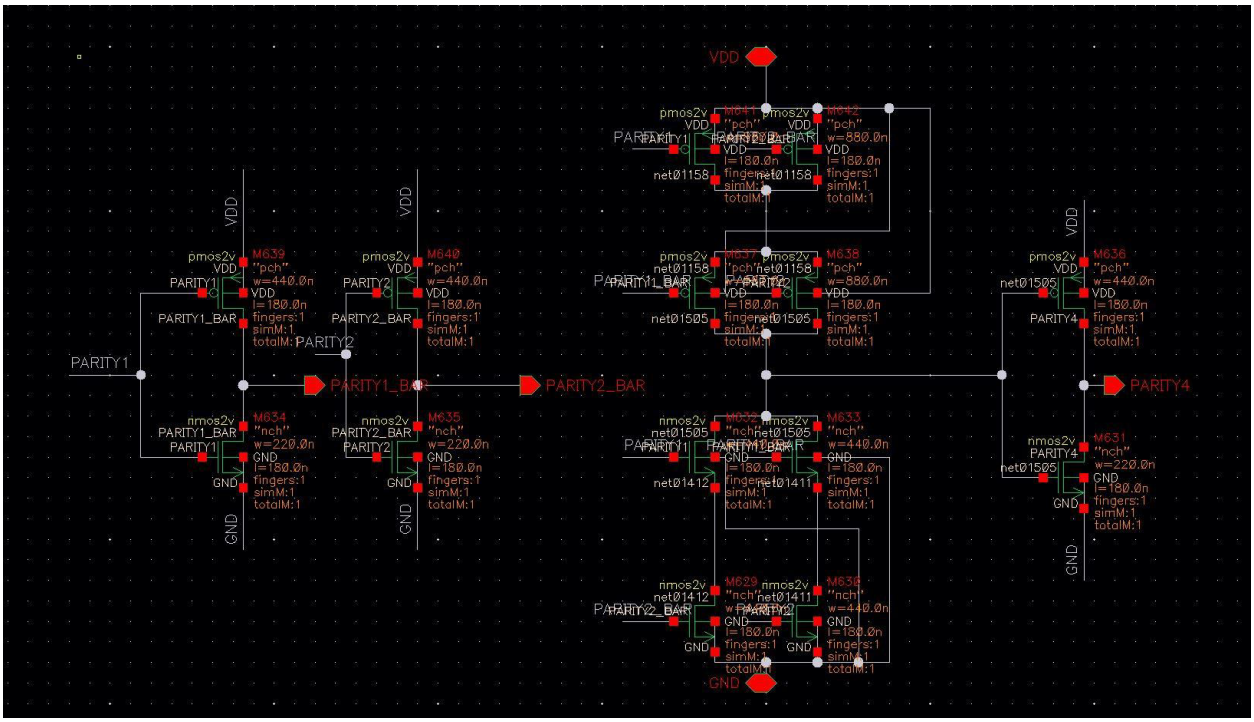
B – PARITY FLAG

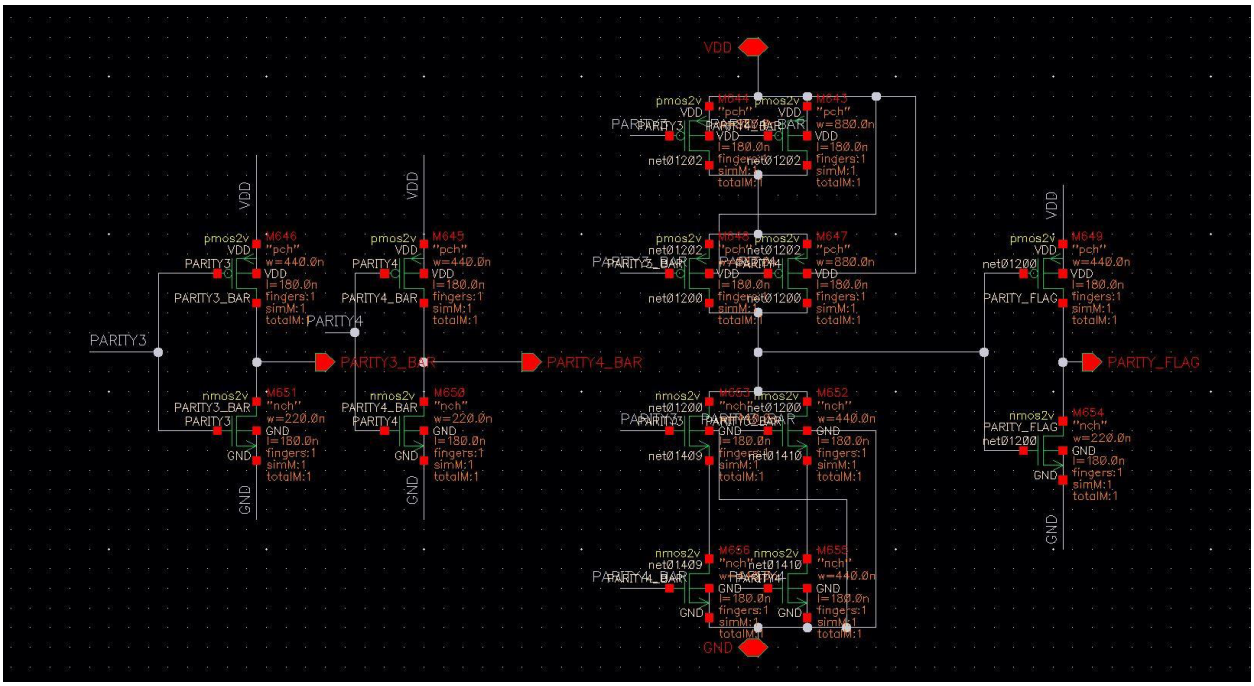
i- Schematics of parity flag is as follows. I have used basic 2 input XOR gate. I have used same concept for parity as I used above for zero flag.





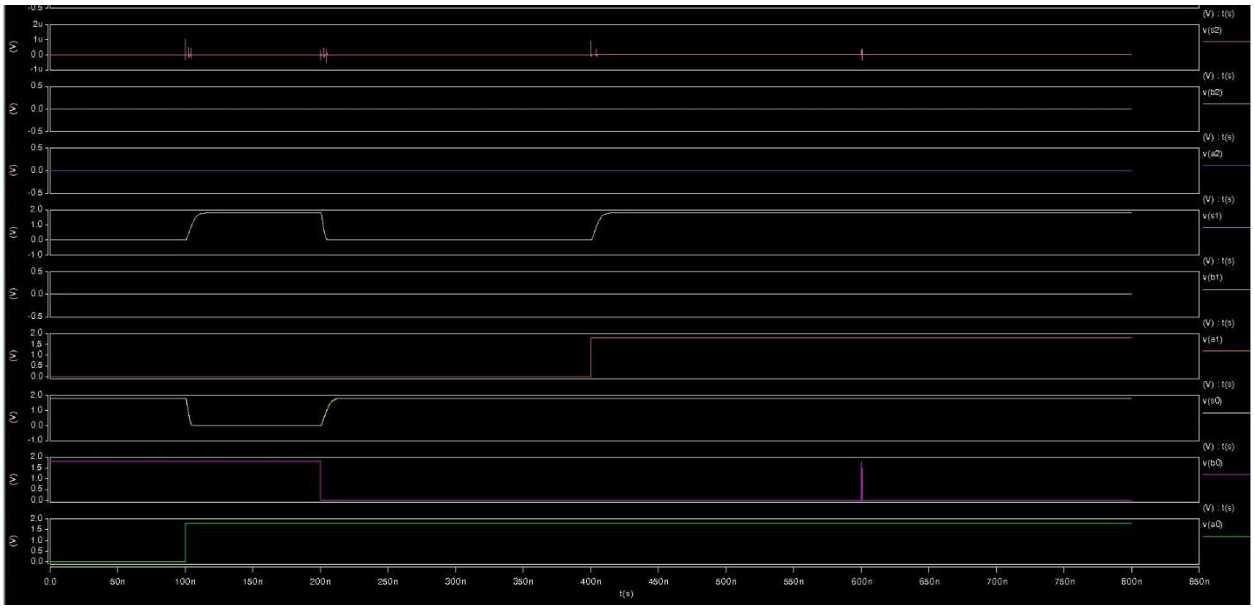

VDD
S4
S5
S4_BAR
S5_BAR
PARITY3
GND
pmos2v
nmos2v
M618
M617
M619
M620
M621
M622
M623
M624
M625
M626
M627
M628
"pch"
"nch"
w=440.0n
w=880.0n
w=220.0n
l=180.0n
fingers:1
simM:1
totalM:1
net01121
net01119
net01413
net01414

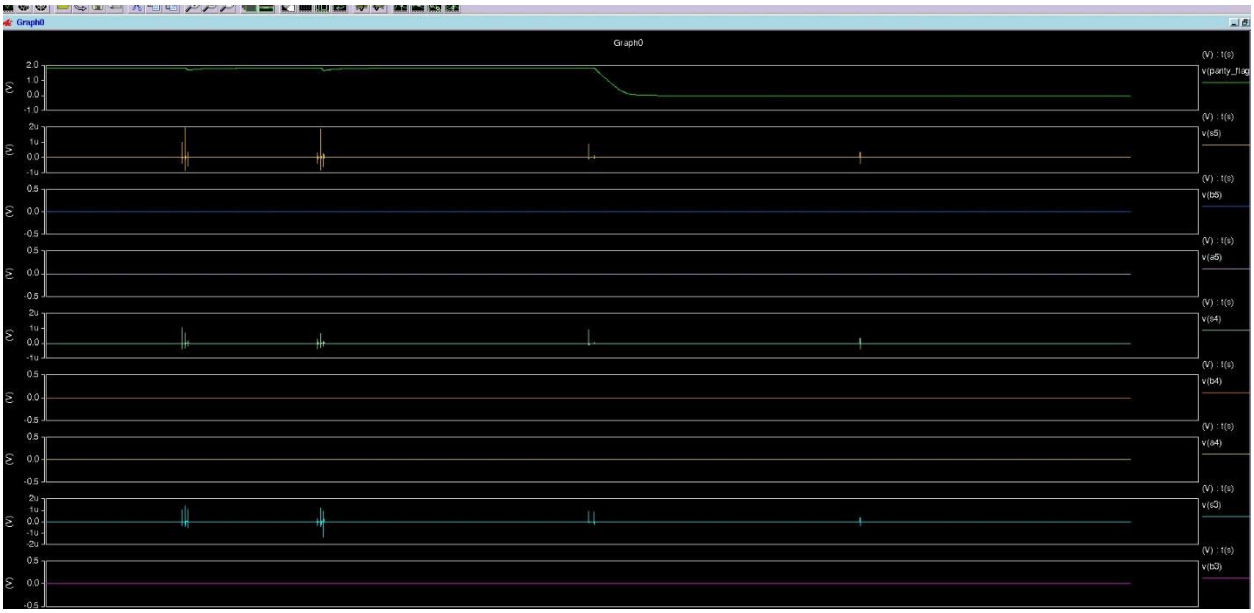

VDD
PARITY1
PARITY2
PARITY1_BAR
PARITY2_BAR
PARITY4
GND
pmos2v
nmos2v
M639
M640
M641
M642
M637
M638
M636
M634
M635
M632
M633
M629
M630
M631
"pch"
"nch"
w=440.0n
w=880.0n
w=220.0n
l=180.0n
fingers:1
simM:1
totalM:1
net01158
net01505
net01412
net01411



ii- WORKING

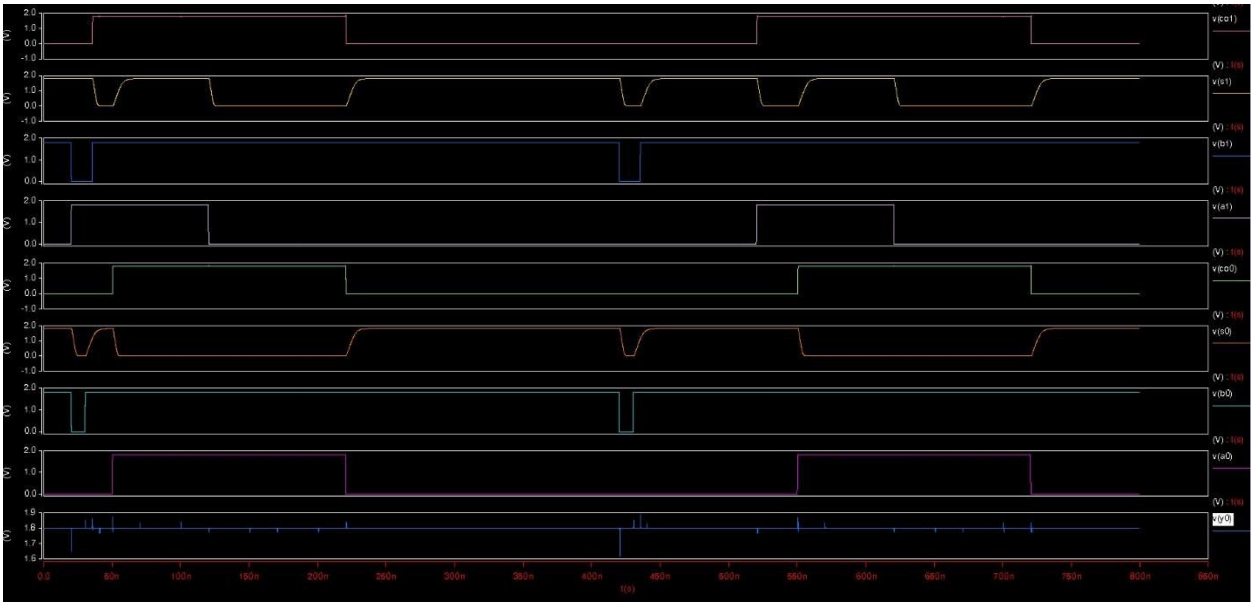
Following graphs shows the proper functioning of parity flag.

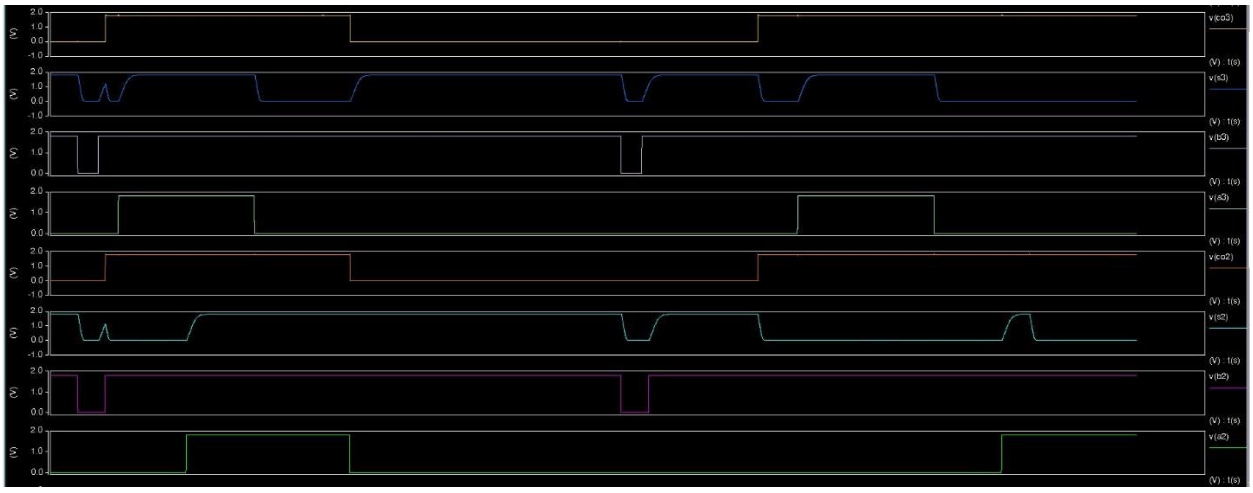


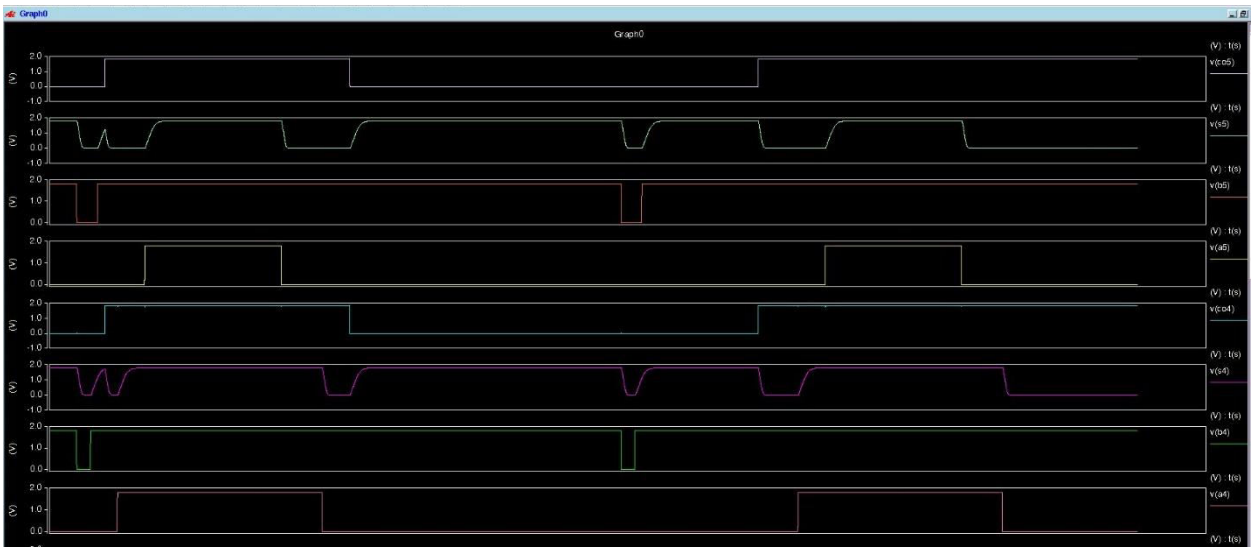

Graph0
Graph0
(V) : t(s)
v(parity_flag)
v(s5)
v(b5)
v(a5)
v(b4)
v(a4)
v(s3)
v(b3)

C – CARRY FLAG & OVERFLOW FLAG

I have not implemented any special schematics for carry flag and overflow flag. This is because we can detect carry flag and overflow flag by looking at the last carry out of the adder block. If last carry out signal goes high then there is a carry at the final output stage. Same concept applies for overflow flag. If last carry bit is high that means overflow has occurred. I am adding a simulation graph which shows that last carry bit (CO5 in my case) is generated in short carry flag and overflow flags are set.







OBSERVATION –

i- When I simulated my schematics with 130nm, delay, energy and EDP dropped down. In all the blocks the drop was nearly consistent but was not exactly same.

ii- In my ALU I observed that the worst delay I faced from inputs to any of the outputs is for my adder block. I am attaching a table to show the worst case delay for my ALU.

Initial state = A0 B0 A1 B1 A2 B2 A3 B3 A4 B4 A5 B5 = 0 0 0 0 0 0 0 0 0 0 0 0

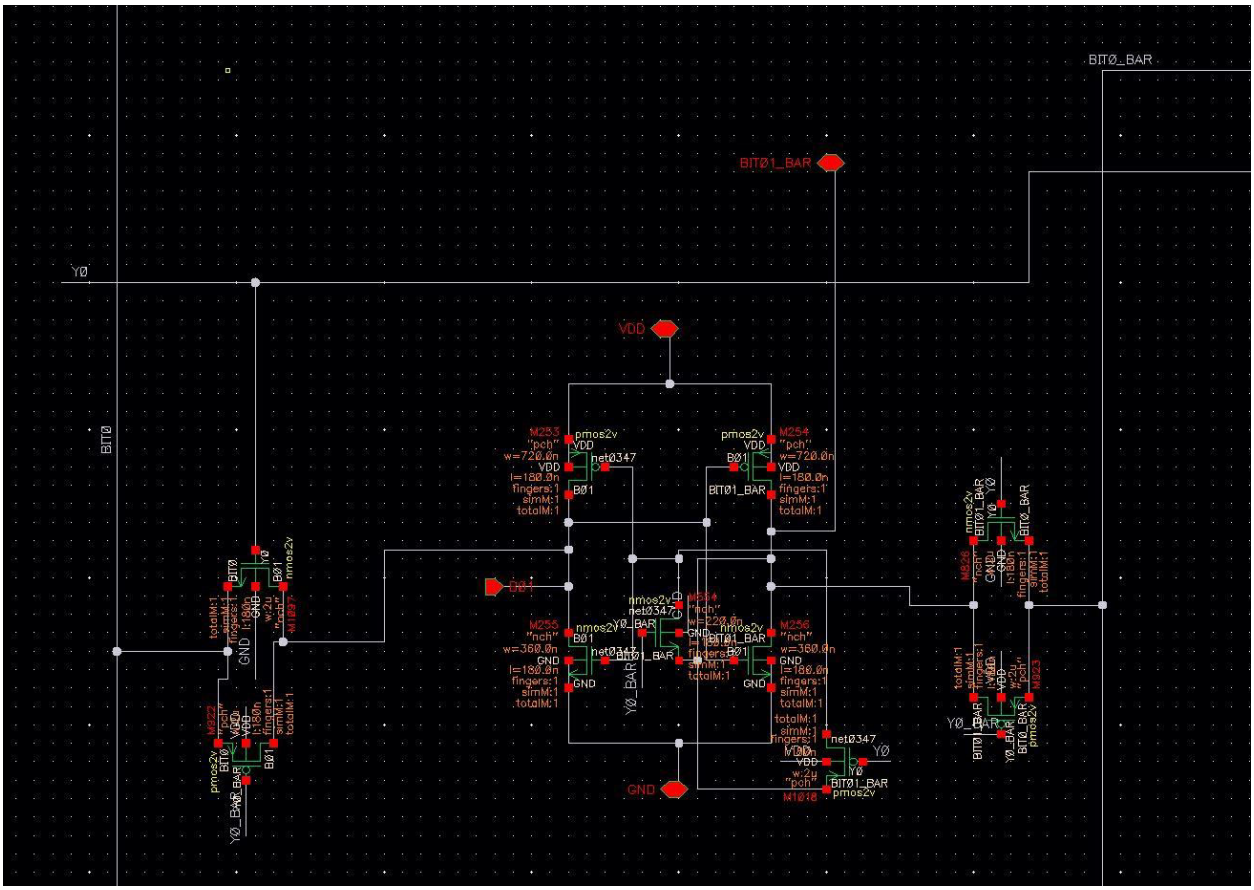
Final state = A0 B0 A1 B1 A2 B2 A3 B3 A4 B4 A5 B5 = 1 1 0 1 0 1 0 1 0 1 0 1

PARAMETER	VALUE
DELAY	8.15E-09

9- SRAM

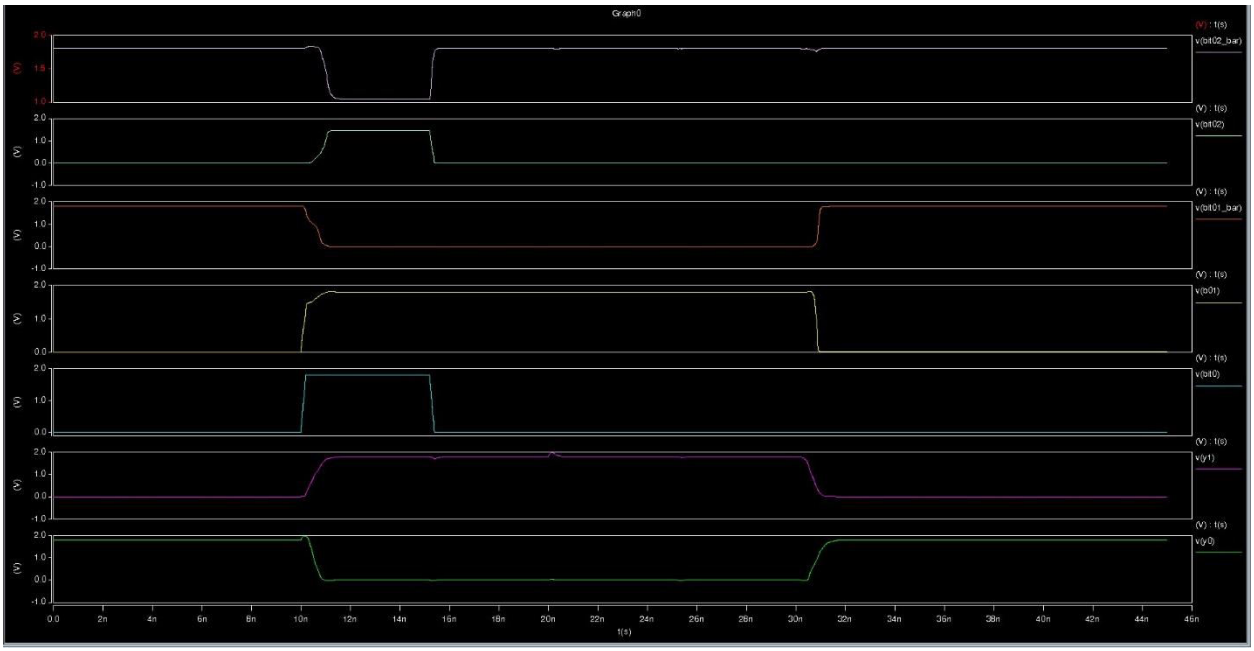
A – SCHEMATICS

In my SRAM, I am using same control logic which I used in my ALU operation which is 3:8 decoder for row decoding. I did not used any column decoder because bits are being parallel load into SRAM at the same time. 6X8 SRAM cell is as follows. First I am attaching 1 bit SRAM cell and then I am attaching full 6X8 SRAM cell array.



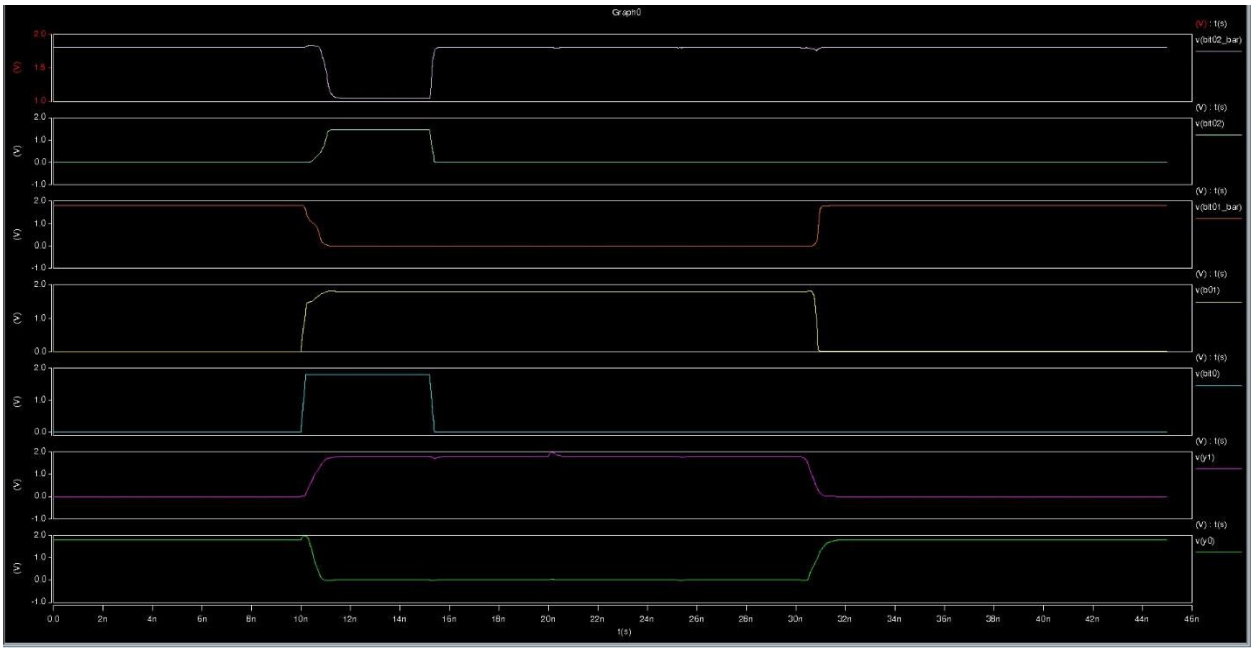
B – WRITE OPERATION

I am attaching the graph which shows successful write operation into SRAM cell. Below graphs also shows writing data into different rows at a time using row decoder.



C – READ OPERATION

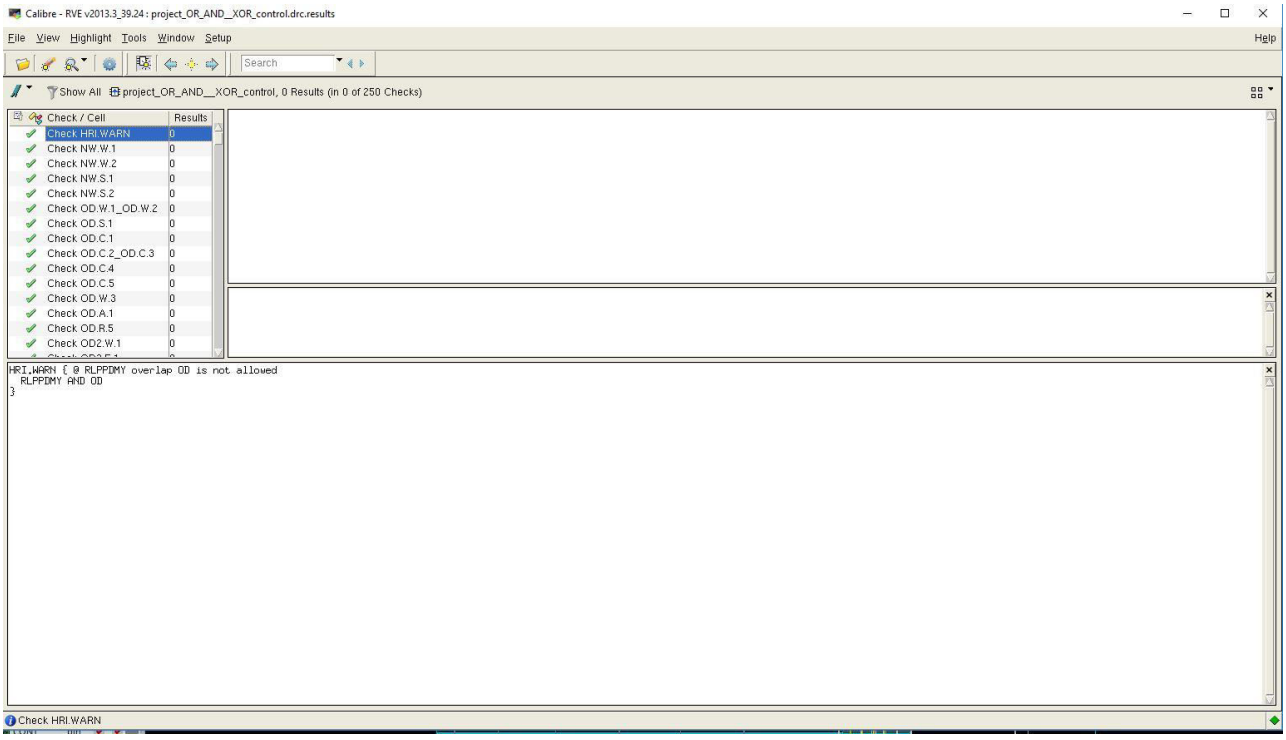
I am attaching a graph which shows that even if we are writing a data into second row, we can still read data from first row as it retains the previously stored data. I was not able to implement sense amplifier or any sort of reading logic, but I think the way I used is the simplest way and can be used to read data from SRAM.



10- LAYOUT

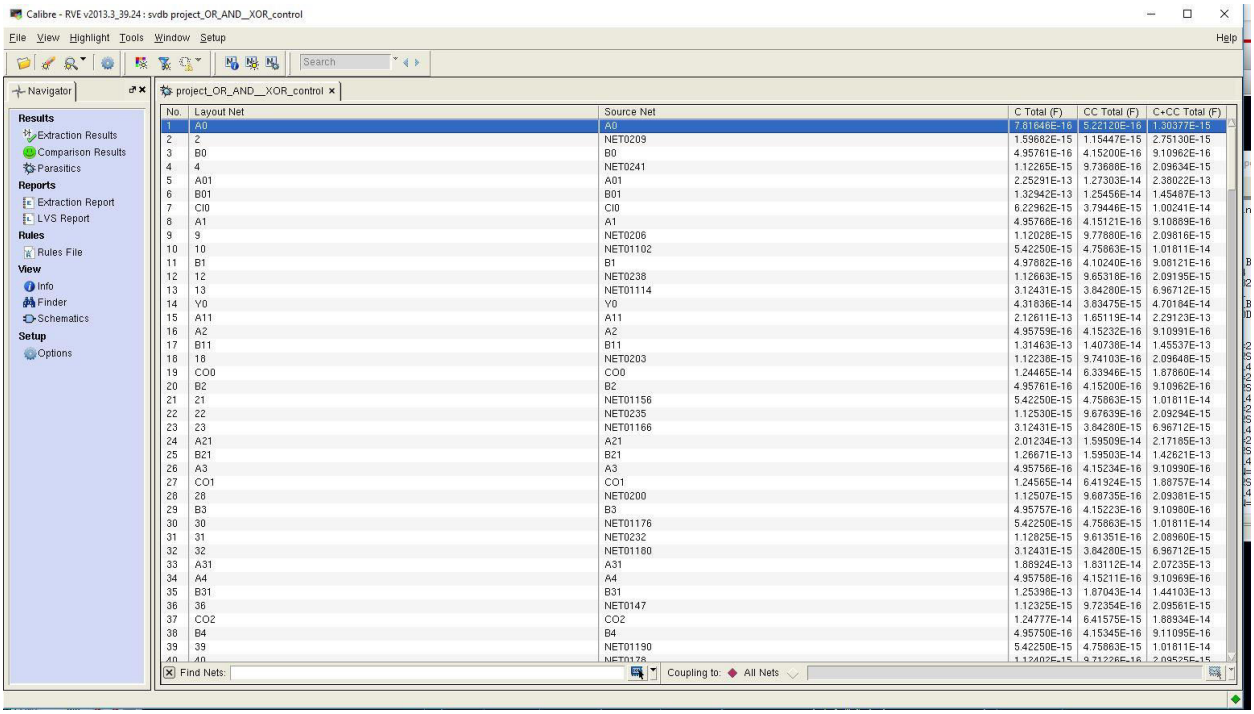
A – DRC CLEAR RESULT

DRC clear result of my layout is as follows



B – PEX CLEAR RESULT

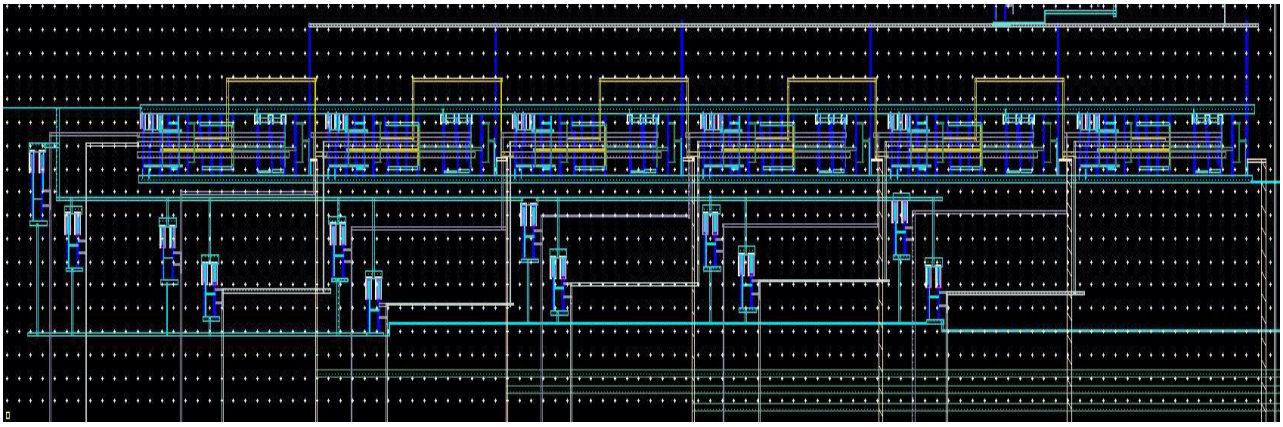
PEX clear result of my layout is as follows.



No.	Layout Net	Source Net	C Total (F)	CC Total (F)	C+CC Total (F)
1	A0	A0	7.81648E-16	5.22120E-16	1.30377E-15
2	2	NET0209	1.59682E-15	1.15447E-15	2.75130E-15
3	B0	B0	4.95761E-16	4.15200E-16	9.10962E-16
4	4	NET0241	1.12265E-15	9.73688E-16	2.09634E-15
5	A01	A01	2.25291E-13	1.27303E-14	2.38022E-13
6	B01	B01	1.32942E-13	1.25456E-14	1.45487E-13
7	CI0	CI0	6.22962E-15	3.79446E-15	1.00241E-14
8	A1	A1	4.95768E-16	4.15121E-16	9.10889E-16
9	9	NET0206	1.12028E-15	9.77880E-16	2.09816E-15
10	10	NET01102	5.42250E-15	4.75863E-15	1.01811E-14
11	B1	B1	4.97882E-16	4.10240E-16	9.08121E-16
12	12	NET0238	1.12663E-15	9.65318E-16	2.09195E-15
13	13	NET01114	3.12431E-15	3.84280E-15	6.96712E-15
14	Y0	Y0	4.31836E-14	3.83475E-15	4.70184E-14
15	A11	A11	2.12611E-13	1.65119E-14	2.29123E-13
16	A2	A2	4.95759E-16	4.15232E-16	9.10991E-16
17	B11	B11	1.31463E-13	1.40738E-14	1.45537E-13
18	18	NET0203	1.12238E-15	9.74103E-16	2.09648E-15
19	CO0	CO0	1.24465E-14	6.33946E-15	1.87860E-14
20	B2	B2	4.95761E-16	4.15200E-16	9.10962E-16
21	21	NET01156	5.42250E-15	4.75863E-15	1.01811E-14
22	22	NET0235	1.12530E-15	9.67639E-16	2.09294E-15
23	23	NET01166	3.12431E-15	3.84280E-15	6.96712E-15
24	A21	A21	2.01234E-13	1.59509E-14	2.17185E-13
25	B21	B21	1.26671E-13	1.59503E-14	1.42621E-13
26	A3	A3	4.95756E-16	4.15234E-16	9.10990E-16
27	CO1	CO1	1.24565E-14	6.41924E-15	1.88757E-14
28	28	NET0200	1.12507E-15	9.68735E-16	2.09381E-15
29	B3	B3	4.95757E-16	4.15223E-16	9.10980E-16
30	30	NET01176	5.42250E-15	4.75863E-15	1.01811E-14
31	31	NET0232	1.12825E-15	9.61351E-16	2.08960E-15
32	32	NET01180	3.12431E-15	3.84280E-15	6.96712E-15
33	A31	A31	1.88924E-13	1.83112E-14	2.07235E-13
34	A4	A4	4.95758E-16	4.15211E-16	9.10969E-16
35	B31	B31	1.25398E-13	1.87043E-14	1.44103E-13
36	36	NET0147	1.12325E-15	9.72354E-16	2.09561E-15
37	CO2	CO2	1.24777E-14	6.41575E-15	1.88934E-14
38	B4	B4	4.95750E-16	4.15345E-16	9.11095E-16
39	39	NET01190	5.42250E-15	4.75863E-15	1.01811E-14
40	40	NET0178	1.12402E-15	9.71226E-16	2.09525E-15

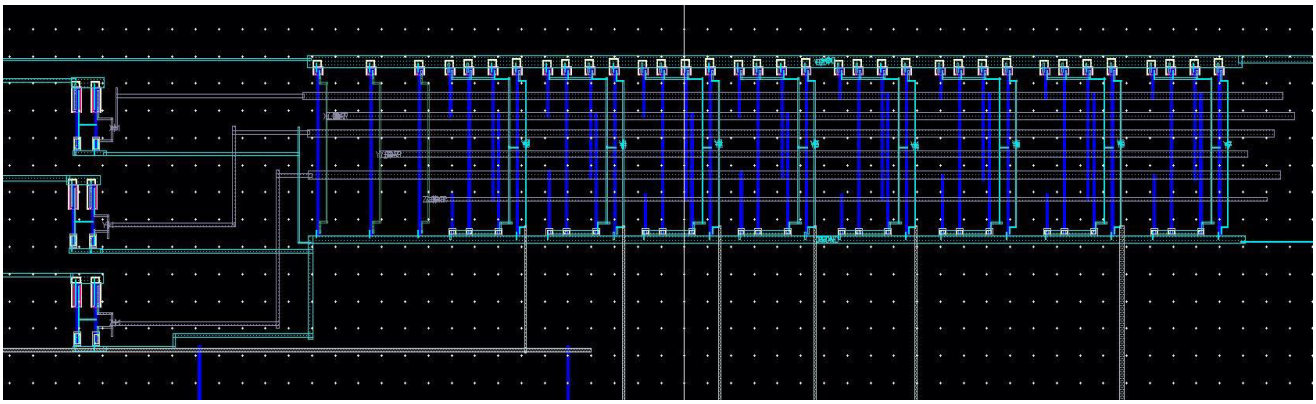
C – ADDER LAYOUT

ADDER layout is as follows



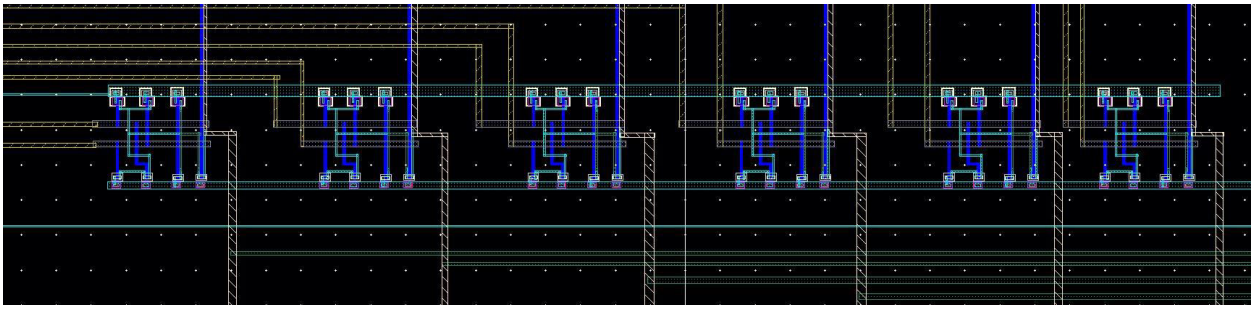
D – CONTROL LOGIC LAYOUT

Control logic layout is as follows

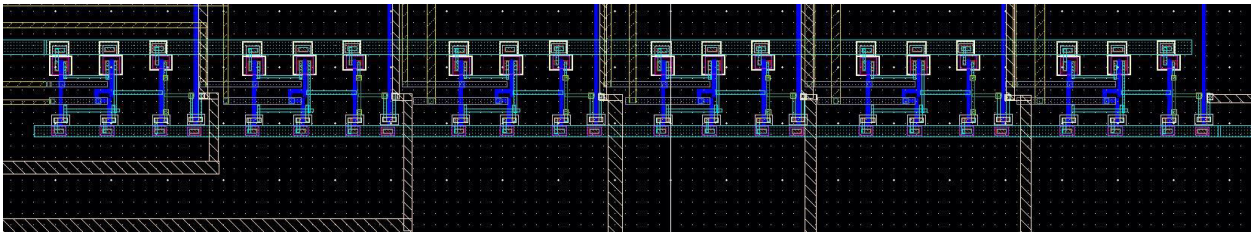


E - LOGIC AND LAYOUT

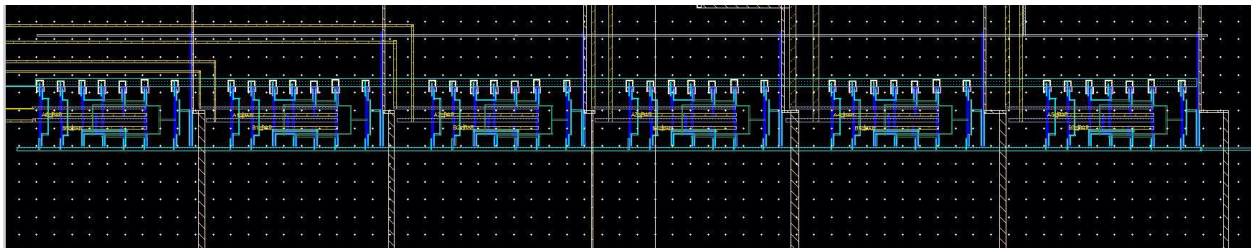
AND gate layout is as follows



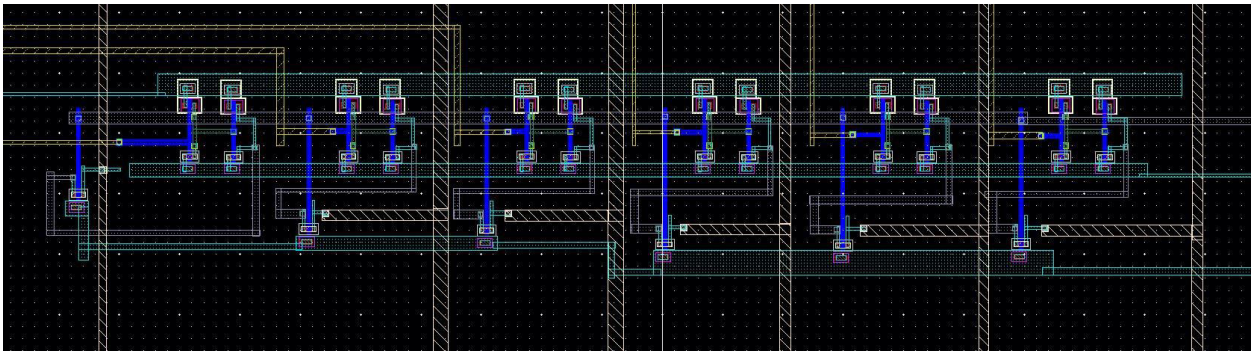
F – LOGIC OR LAYOUT



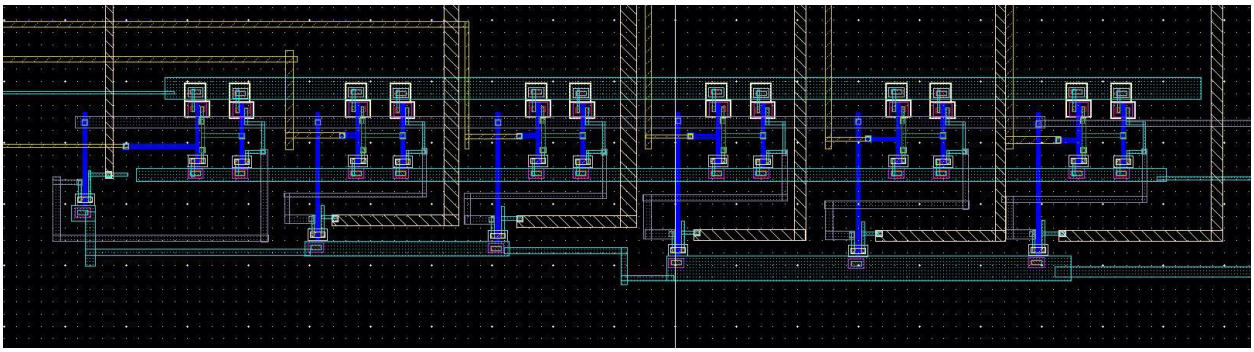
G – LOGIC XOR LAYOUT



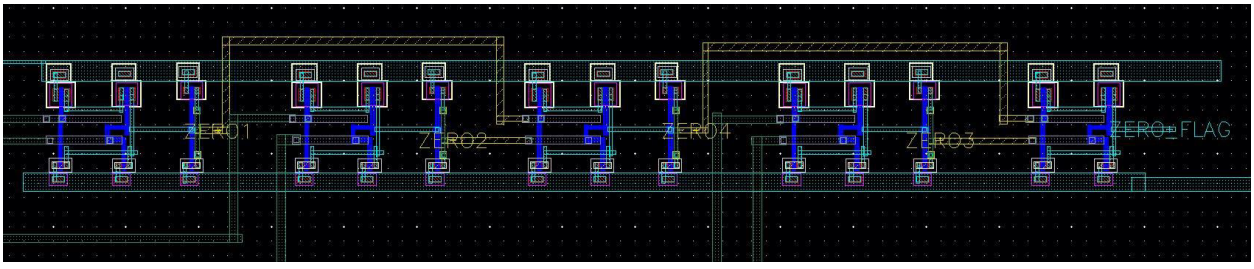
H – ROTATE LEFT LAYOUT



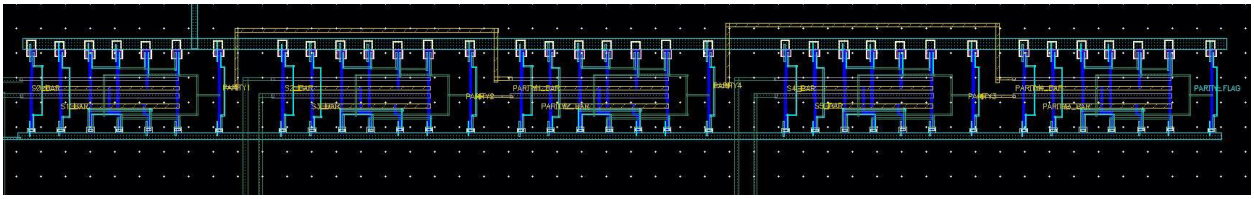
I – SHIFT LEFT LAYOUT



J – ZERO FLAG LAYOUT



K – PARITY FLAG LAYOUT



L – FULL ALU LAYOUT