VLSI Course Project
4-Bit CLA adder

Vedant Tejas
2023112018
vedant.tejas@research.iiit.ac.in
UG2, ECD

IIITH

I. Introduction

4 bit CLA adder adds two 4-bit numbers and gives the final 4-bit sum

However, unlike normal 4-bit Ripple Carry Adder which calculates each carry one at a time, CLA Adder calculates all the carry simultaneously, reducing the propagation delay.

1. **Construction and logic:**

We have to create three blocks. First we generate pi and gi (Propogate and Generate) for CLA block

pi = ai ⊕ bi

gi = ai · bi

Then we calculate all carries in CLA block by following the logic which is given by this equation.

$$c_{i+1} = p_i c_i + g_i$$

From this, we can calculate all carries ci ′ s from the propagates and generates (pi ′ s and gi ′ s).

$$c1 = (p0 \cdot c0) + g0$$

$$c2 = (p1 - p0 \cdot c0) + (p1 \cdot g0) + g1$$

$$c3 = (p2 \cdot p1 \cdot p0 \cdot c0) + (p2 \cdot p1 \cdot g0) + (p2 \cdot g1) + g2$$

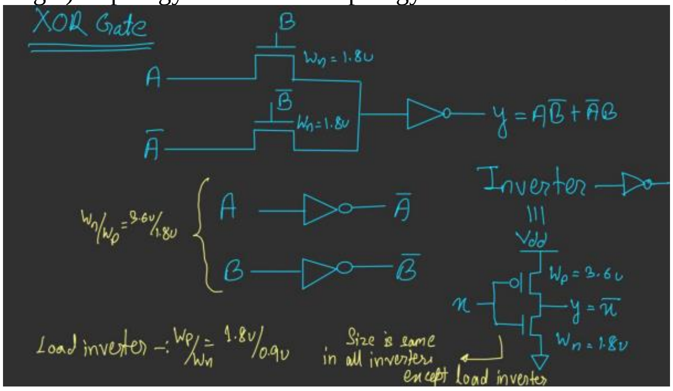
$$c4 = (p3 \cdot p2 \cdot p1 \cdot p0 \cdot c0) + (p3 \cdot p2 \cdot p1 \cdot g0) + (p3 \cdot p2 \cdot g1) + (p3 \cdot g2) + g3$$

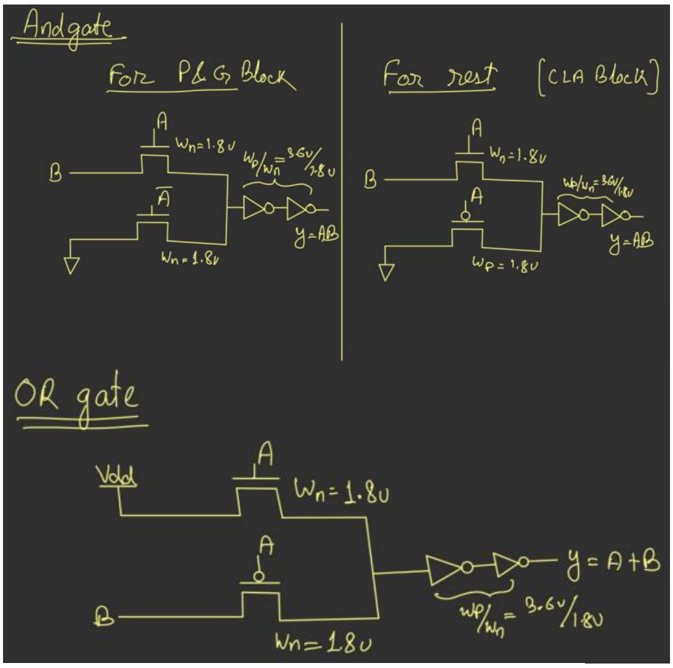
Finally we can get the final sum by following the logic

Si = pi ⊕ ci

II.DESIGN TOPOLOGY

For XOR, AND & OR gates, we use PTL (Pass Transistor Logic) Topology and CMOS Topology for Inverters.





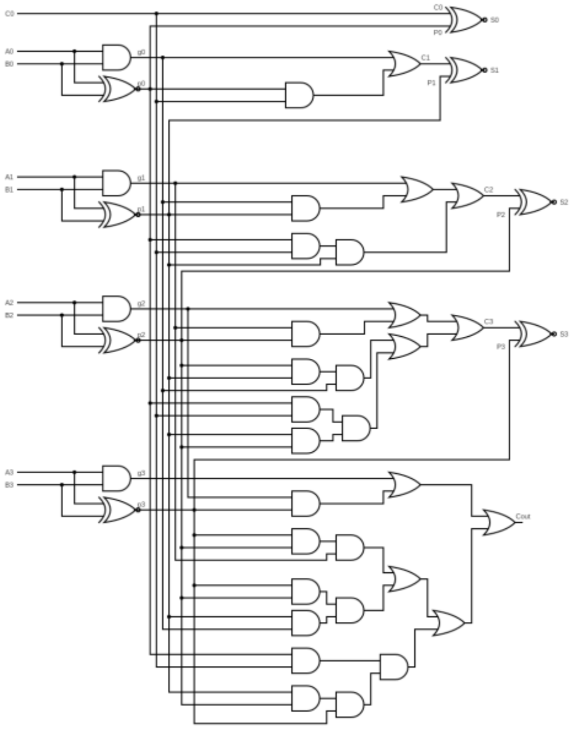
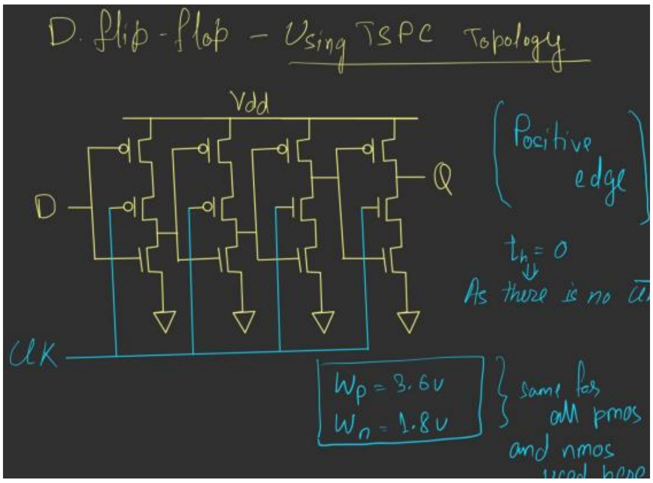


Fig. 2: Circuit for CLA Adder

D FLIP FLOP

For D flipflop, we use TSPC (True Single Phase Clock) topology. Given below is the circuit for flipflop along with its size.


D. flip-flop – Using TSPC Topology
Vdd
D
Q
Clk
Positive edge
$t_h = 0$
As there is no
$W_p = 3.6u$
$W_n = 1.8u$
Same for all pmos and nmos used here

III. PRE-LAYOUT SIMULATIONS IN NGSPICE

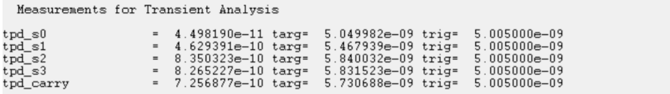
CLA ADDER

For CLA Adder, we take input as a = 1001 and b=1101 at t=5ns. The output is as follows.

Here the output is 10110

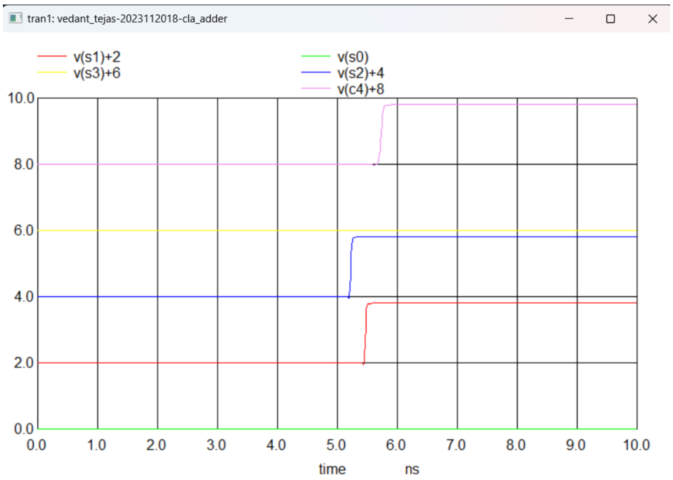
Thus this cla is working as expected

For the analysis of the propogation delays we are giving a=1111 and b=1111 as well as cin=1. From here we get output as 11111. Here we see the delays of the sum and final carry unit.



```
  Measurements for Transient Analysis

tpd_s0               =  4.498190e-11 targ=  5.049982e-09 trig=  5.005000e-09
tpd_s1               =  4.629391e-10 targ=  5.467939e-09 trig=  5.005000e-09
tpd_s2               =  8.350323e-10 targ=  5.840032e-09 trig=  5.005000e-09
tpd_s3               =  8.265227e-10 targ=  5.831523e-09 trig=  5.005000e-09
tpd_carry            =  7.256877e-10 targ=  5.730688e-09 trig=  5.005000e-09
```

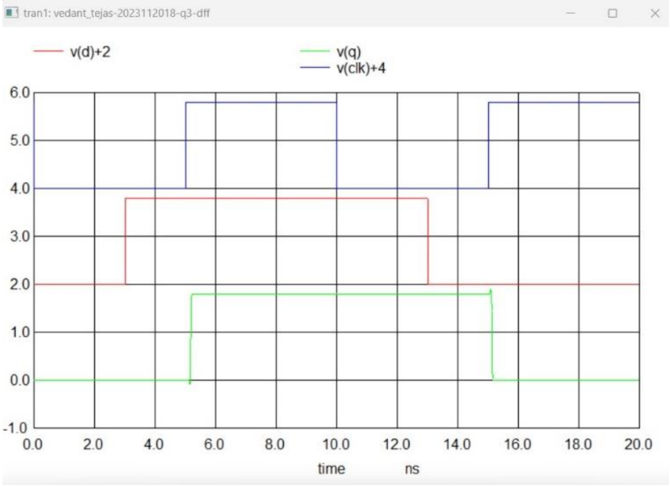
Tpd_{min}=44.9ps Tpd_{max}=83.5ps



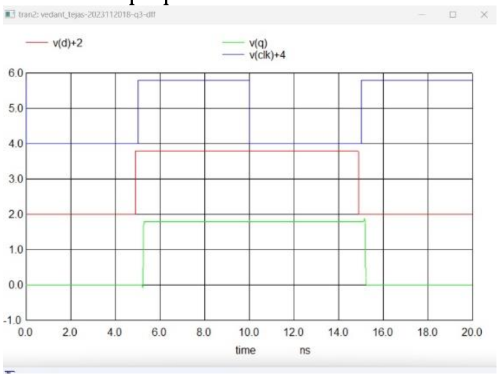
D-FLIP FLOP

For D-flipflop where assuming a positive edge, we get the following simulation:

As we can see, Q is updated to D at +ve edge only. Hence its working properly



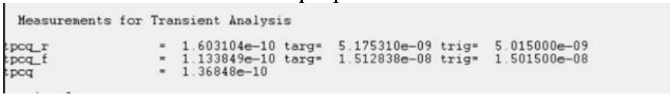
Minimum tpcq



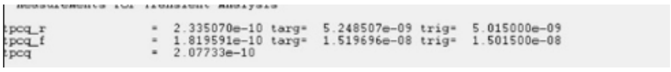
Maximum Tpcq

IV. SETUP TIME, HOLD TIME AND PROPAGATION DELAY FROM CLK TO Q IN FLIPFLOP

We have used TSPC topology for our D-FlipFlop. Since there is no use of (clk)’, we have zero hold time (t hold = 0). For setup time, we did trial and error and found it to be tsetup = 0.11ns. Any input given less than tsetup before next +ve edge, either input is not registered or output is being corrupted. For propagation delay from clk to Q (tpcq), we can see the minimum and maximum tpcq .The values are as follows:



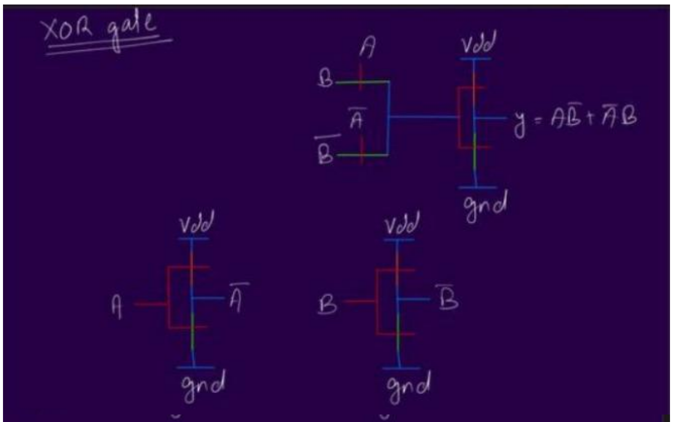
```
 Measurements for Transient Analysis

tpcq_r            =  1.603104e-10 targ=  5.175310e-09 trig=  5.015000e-09
tpcq_f            =  1.133849e-10 targ=  1.512838e-08 trig=  1.501500e-08
tpcq              =  1.36848e-10
```

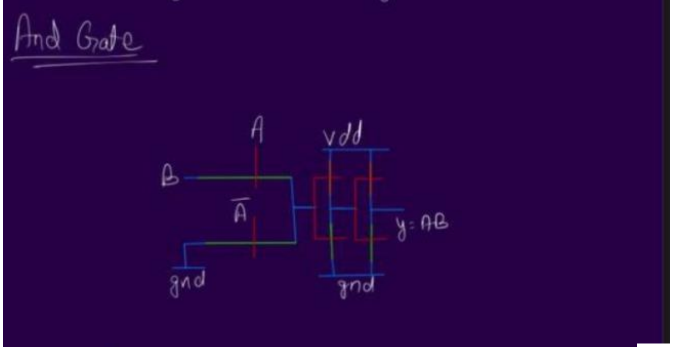


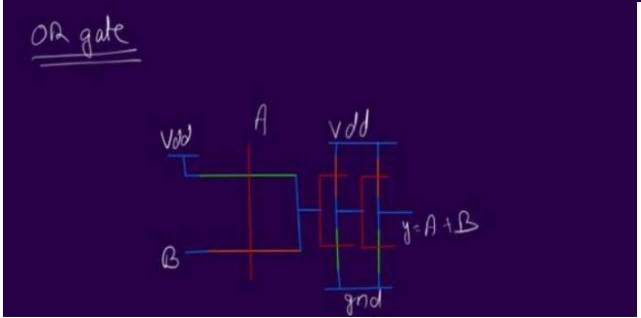
```
 Measurements for Transient Analysis

tpcq_r           =  2.335070e-10 targ=  5.248507e-09 trig=  5.015000e-09
tpcq_f           =  1.819591e-10 targ=  1.519696e-08 trig=  1.501500e-08
tpcq             =  2.07733e-10
```

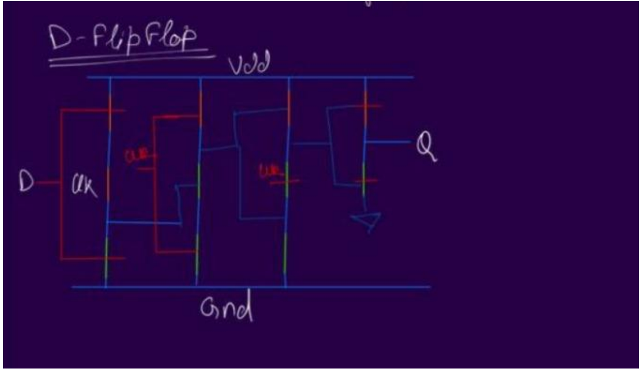
T_{pcqmin}=0.136ns T_{pcqmax}=0.207ns

V STICK DIAGRAMS FOR ALL UNIQUE GATES AND FLIPFLOPS







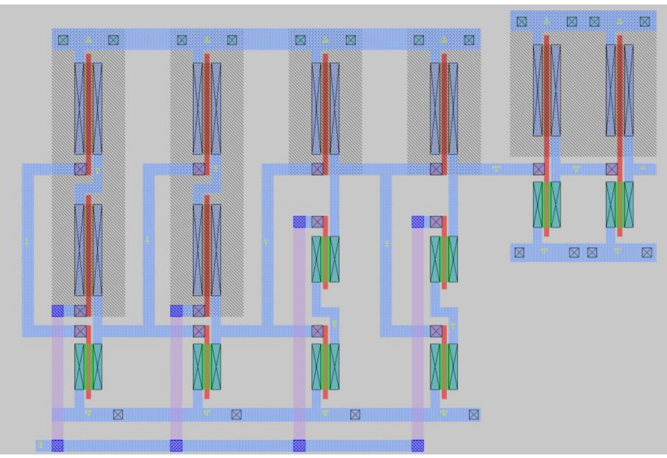


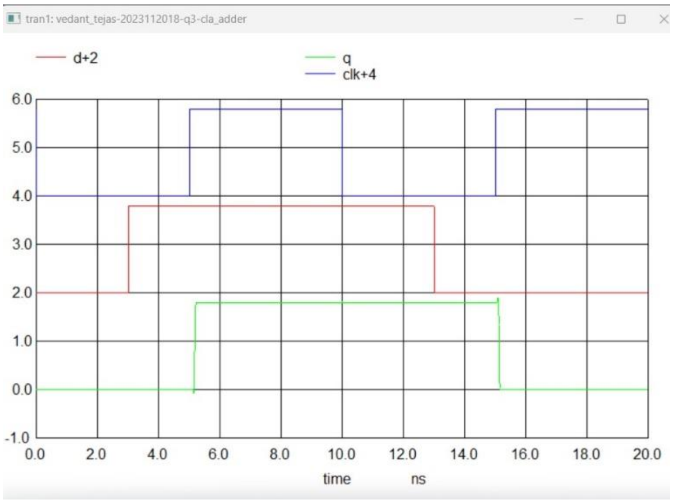
VI POST LAYOUT SIMULATIONS

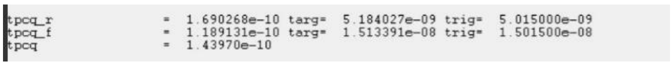
D-FlipFlop

Here we make the circuit given in MAGIC Layout and perform post-Layout simulations.

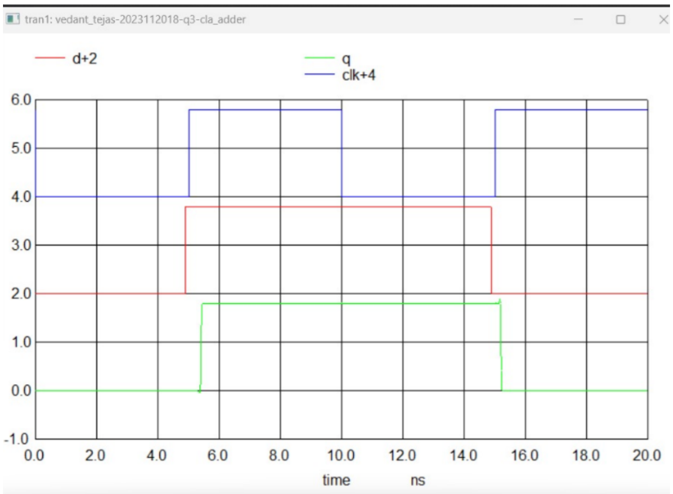
Here we can see that we get a similar output as pre-layout simulations. However, the tpcq value would be different.

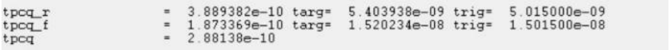






```
tpcq_r              =  1.690268e-10 targ=  5.184027e-09 trig=  5.015000e-09
tpcq_f              =  1.189131e-10 targ=  1.513391e-08 trig=  1.501500e-08
tpcq                =  1.43970e-10
```

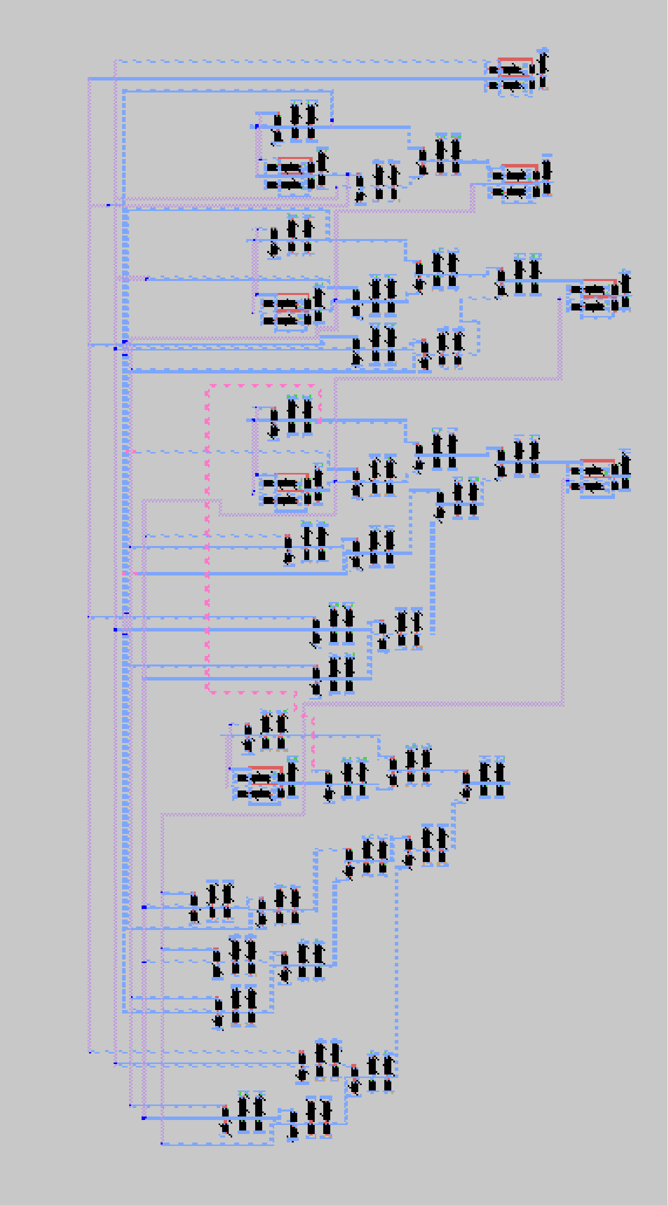




```
tpcq_r              =  3.889382e-10 targ=  5.403938e-09 trig=  5.015000e-09
tpcq_f              =  1.873369e-10 targ=  1.520234e-08 trig=  1.501500e-08
tpcq                =  2.88138e-10
```

	PRE-LAYOUT	POST LAYOUT
$T_{\text{PCQ MIN}}$	0.136NS	0.143NS
$T_{\text{PCQ MAX}}$	0.207NS	0.288NS

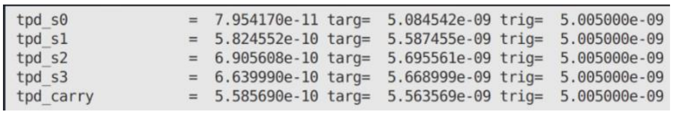
B. CLA-ADDER



Here we make the circuit given in MAGIC Layout and perform post-layout simulations. We again perform a simulation with input a = 1001 and b = 1101 at t=5ns and get the following result (next page)

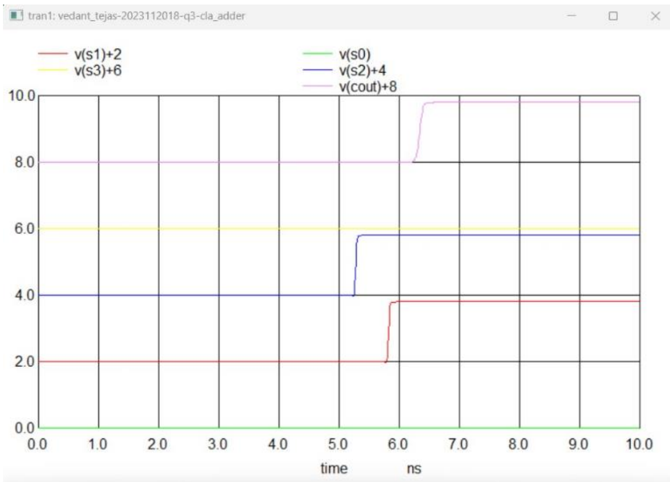
As we can see, with some delay, we get the final output s = 10110 which is the same as pre-layout simulations. By giving a and b as 1111 as well as cin = 1 for s = 1111, we get

Now comparing delays with pre-layout simulations, we get:

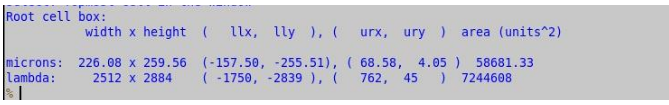


```
tpd_s0              =  7.954170e-11 targ=  5.084542e-09 trig=  5.005000e-09
tpd_s1              =  5.824552e-10 targ=  5.587455e-09 trig=  5.005000e-09
tpd_s2              =  6.905608e-10 targ=  5.695561e-09 trig=  5.005000e-09
tpd_s3              =  6.639990e-10 targ=  5.668999e-09 trig=  5.005000e-09
tpd_carry           =  5.585690e-10 targ=  5.563569e-09 trig=  5.005000e-09
```

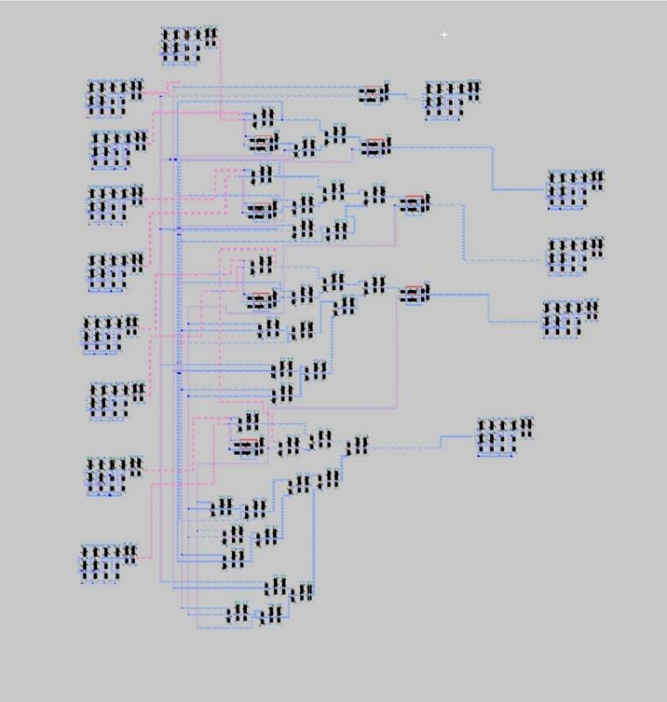
	PRE-LAYOUT	POST LAYOUT
$T_{PCQ\ MIN}$	44.9ps	79.54ps
$T_{PCQ\ MAX}$	83.5ps	69ps



VIII. FLOOR PLAN AND MAGIC LAYOUT FOR FINAL CIRCUIT

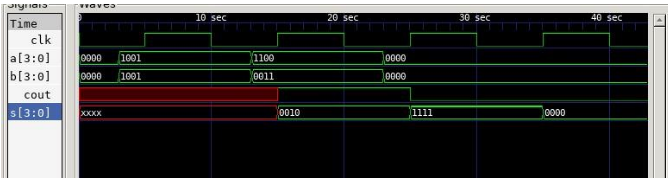


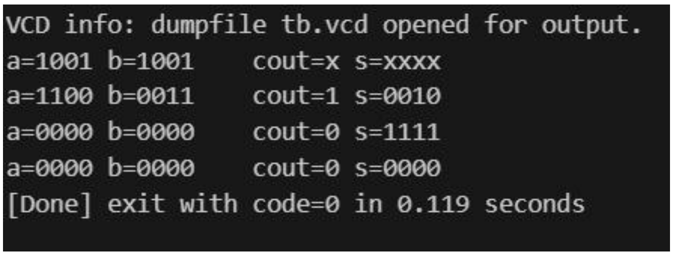
```
Root cell box:
           width x height  (   llx,  lly  ), (   urx,  ury  )  area (units^2)

microns:  226.08 x 259.56  (-157.50, -255.51), ( 68.58,  4.05 )  58681.33
lambda:     2512 x 2884    ( -1750, -2839 ), (   762,  45   )  7244608
%
```



X. VERILOG SIMULATIONS AND WAVEFORMS

Now, we create the final CLA Circuit in verilog. We give two inputs, a = 1001, b = 1101 and a = 1010, b = 01101. We get the following output:



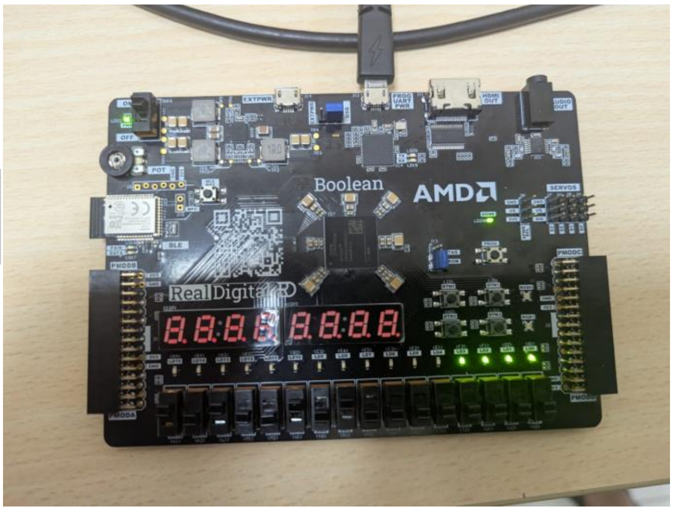


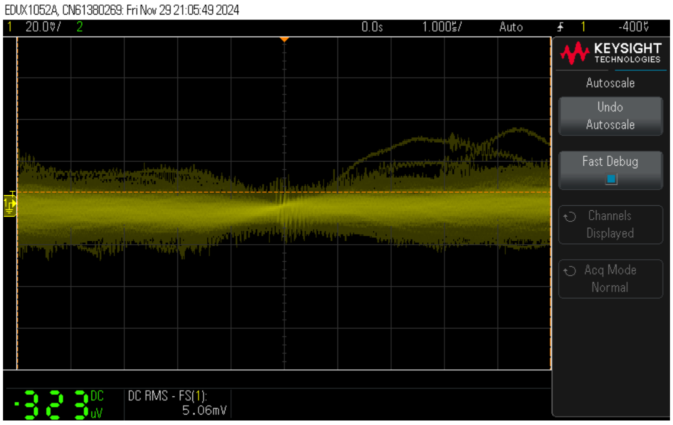
```
VCD info: dumpfile tb.vcd opened for output.
a=1001 b=1001    cout=x s=xxxx
a=1100 b=0011    cout=1 s=0010
a=0000 b=0000    cout=0 s=1111
a=0000 b=0000    cout=0 s=0000
[Done] exit with code=0 in 0.119 seconds
```

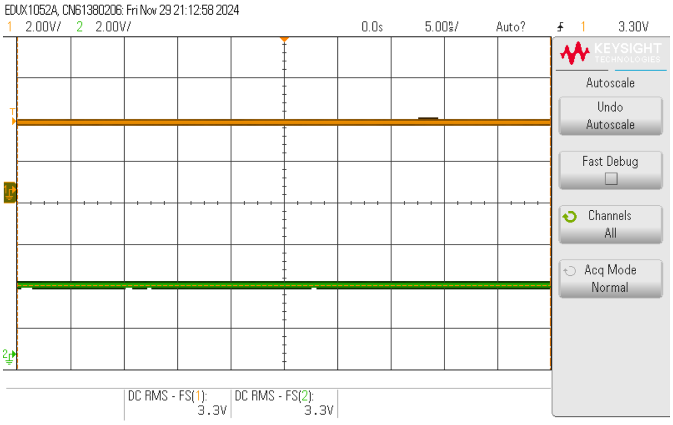
As we can see, when we give the input, we get the output in the next cycle. We get sum=10110 and sum=01111 after the next input has been given, that is, in the next +ve edge of the clock. In GTKWave, we get the following waveforms:

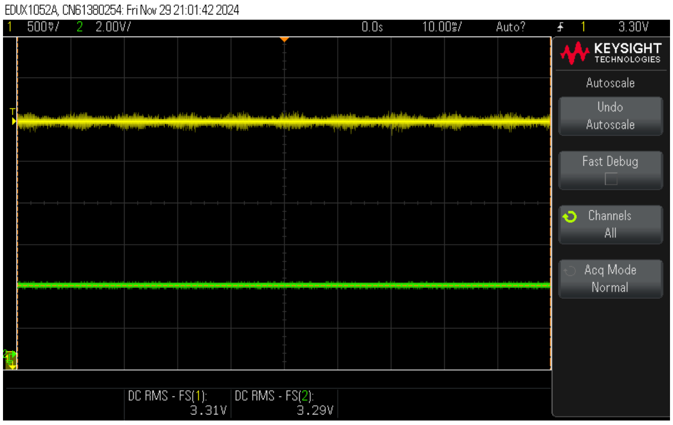
XI. FPGA AND HARDWARE SIMULATIONS

Now, we give inputs in FPGA board and get the output as shown: 1010+0101=1111









EDUX1052A, CN61380254: Fri Nov 29 21:01:42 2024

1 500mV/ 2 2.00V/ 0.0s 10.00ms/ Auto? 1 3.30V

DC RMS - FS(1): 3.31V

DC RMS - FS(2): 3.29V

As we can see, we are getting 3.31V for HIGH (bit 1) and some noise (in mV) for LOW (bit 0). Here we also are getting the sum.

XII. CONCLUSION

So, like this, we are able to design a CLA Adder which takes input at one +ve edge and gives output at next +ve edge clock. This makes sure that the CLA adder has enough time to calculate the sum bits.

REFERENCES

CMOS VLSI design Weste and Harris

Verilog HDL-Samir Palnitkar

Google

IEEE Research Papers