ECCE – 625: Digital Integrated Circuit Design
Course Project
Report

Aaron Tekleab
100043374

Submitted to Dr. Baker Mohammad

Dec 11, 2022

ECCE 625
Integrated Digital Circuit Design
Project
Dr. Baker Mohammad

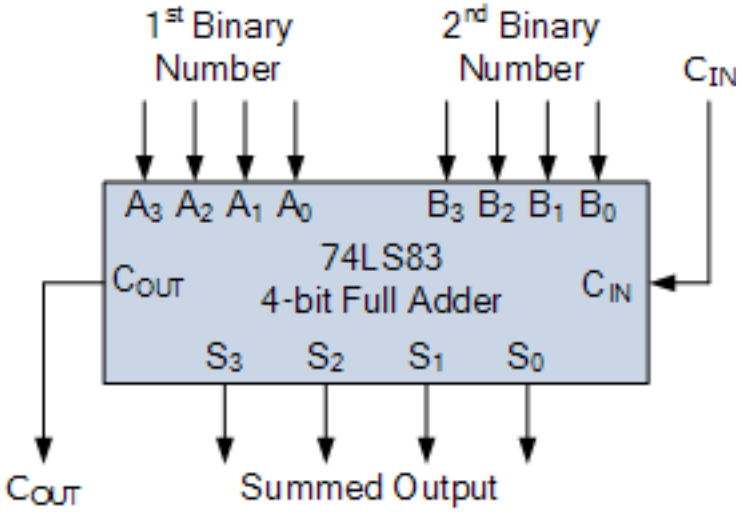
For all the questions use 65nm low power process files from Global foundry and cadence tools.

The goal of this project is to use the design you have done in homework to build a block and analyze the important metrics in digital circuit design.

Design a 4-bit binary adder where the 2 operands will be coming from a flip-flop and the result will be flopped in a flip flop as well. Use TT corner for the process, 1.2v and 25C for all the analysis. You are asked to report:

1- Maximum frequency for your design
2- Area for your design (add the width of all transistors used in your design)
3- Active power assuming 50% of the input bits switch in the design
4- Leakage power for your design at 25C
5- Generate your clock using ring oscillator and study clock skew

Your design report should show the logic optimization and schematic in addition to the simulation results from cadence



Contents

Table of Figures

i) The flipflop Design

The sizing of the flip flop was done assuming a 20C at the output. The design that was implemented can be shown below. The inverters at the input and output nodes are used to isolate the input and output respectively. ϕ1 and ϕ2 are the same clock in my design.

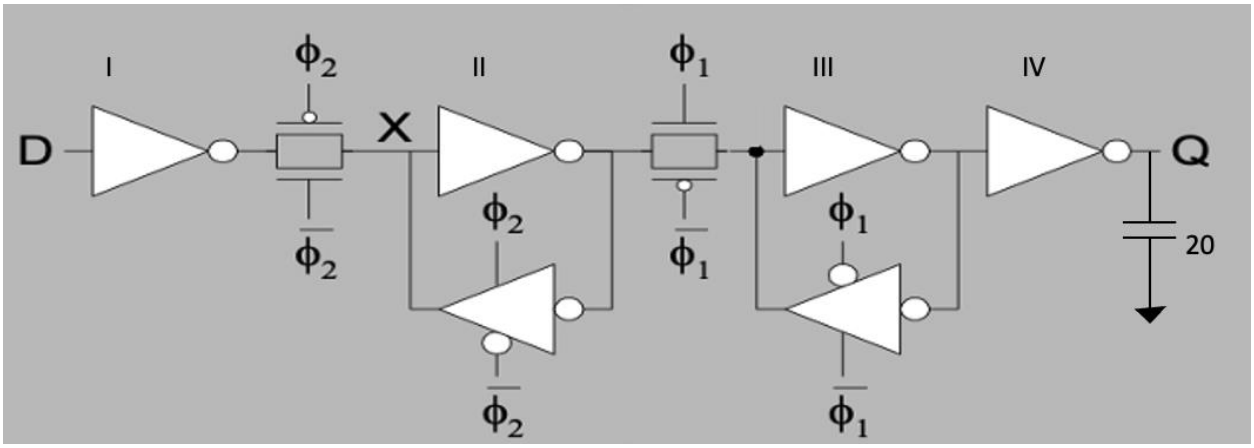


Figure 1: Flipflop structure

The best stage effort was assumed to be 3.33. Working backwards the Cin of each stage can be found using the following formula.

$$Cin = Cout * \frac{G}{f}$$

Where G is the logical effort of each stage (1 for inverter and 2 for tristate) and f is the best stage effort of the respective stage. The feedback loops are only using for driving so the minimum size is used for the transistors. And the pass gates and invertors on the forward path are assumed as tristate and the size is shared between them. Using the above formula, the Cin of inverter (IV) is 6. This means the pmos and nmos will have 4 to 2 size ratio. Inverter (III) and the pass-gate are considered as a tristate thus G = 2. Cout is 6 plus the Cin of the feedback input which is 2. Thus, Cout is 8 and taking the stage effort to be 3.33 results into 4.8. However, for easier sizing the Cin is taken to be 6 which gives stage effort to be 2.67. The pmos and nmos will have 4 to 2 ratio size. This is divided by the pass gate and the inverter which results into 2 by 1 pmos to nmos size ratio. For inverter (II), Cout is 8 and taking f to be 2.67 results into the same result as inverter (III). For inverter (I), the Cout is 6 and since the basis pmos to nmos ratio is assumed to be 2:1, inverter (I) is given Cin equal to 3 which results into stage effort of 2. The overall stage effort of the design is calculated using the following formula:

$$f^N = \prod_{i=1}^{N} g_i h_i$$

Where g and h are stage i respective logical effort and electrical effort respectively. Stage effort f, for stages (I-IV) are 2, 2.67, 2.67, and 3.33 respectively. **Thus, the best stage effort is 2.625 and this is in the acceptable range i.e., 2.4 – 6.**

The actual design on cadence is shown below

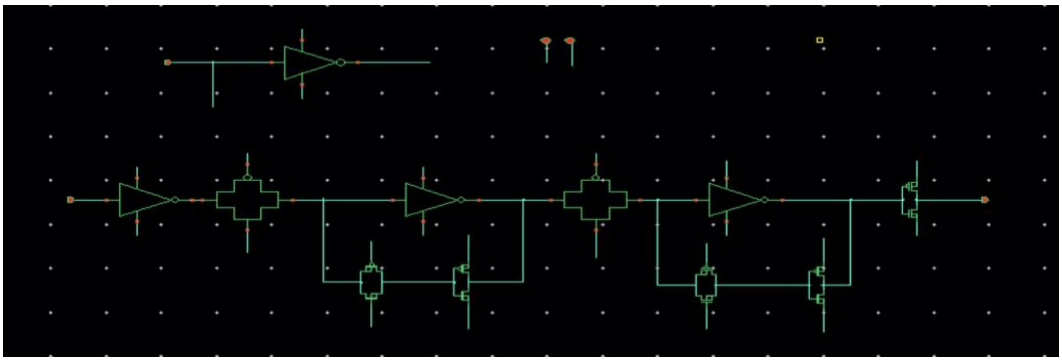


Figure 2: Flipflop designed in Cadence

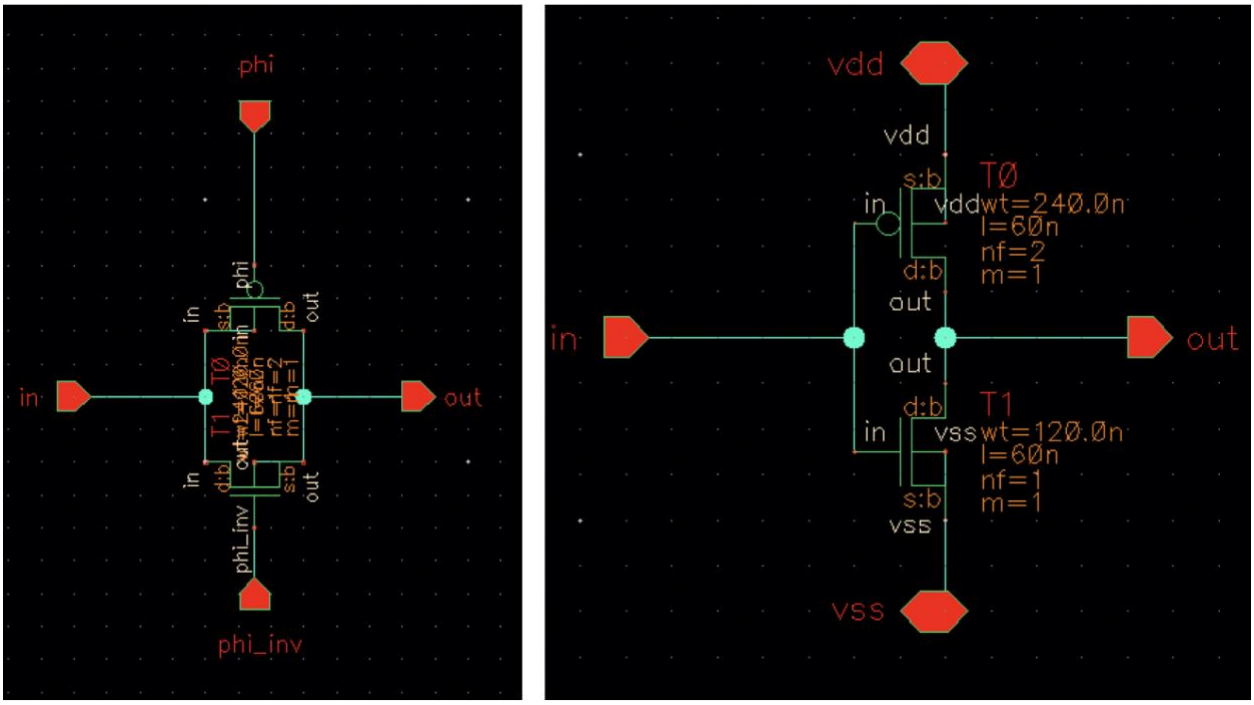


Figure 3:Pass gate and inverter used in the flipflop

After the circuitry was built the setup time and the propagation delay of the flipflop were evaluated. The input was swept from 4.9n seconds to 5n seconds with 0.005n seconds step. The period of the clock was set to 5n. The output can be shown in Fig. 4. The setup time was chosen based on the clock to output delay. At 4.9n seconds the c-q delay was around 102p seconds and at 4.95n seconds it increased by 10% shown in Fig. 5. **Thus, the setup time was evaluated to be $5n - 4.95n = 0.05n$ *seconds***. The propagation delay was also found from that as shown in Fig. 4. **The c-q propagation delay of the flipflop is 112.264204p seconds.**

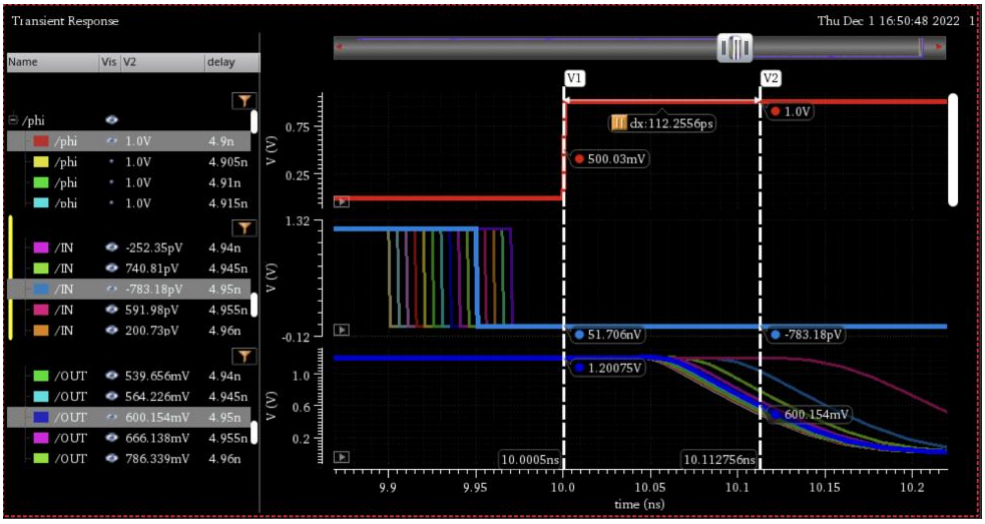


Figure 4:Setup time Evaluation

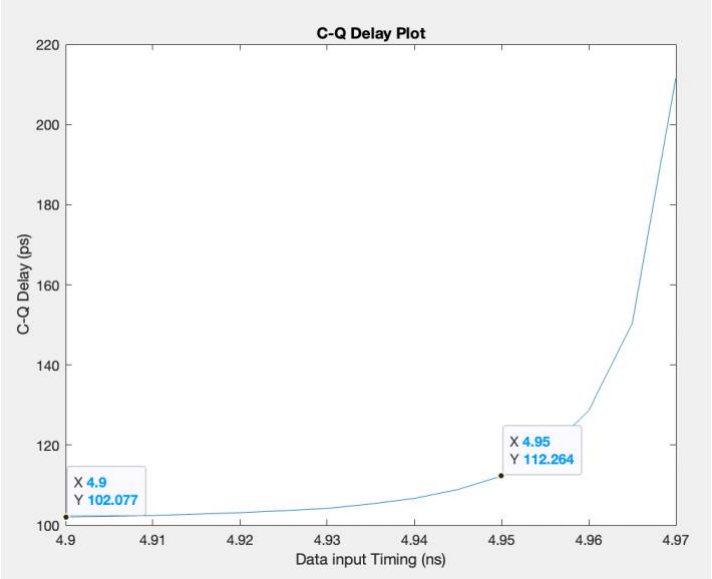


Figure 5:Clock to Q delay plot

ii) The Adder

a) One bit adder

a	b	Cin	Sum	Cout
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

$$\begin{aligned} Sum &= \sim a \sim b\, Cin + \sim a\, b \sim Cin + a \sim b \sim Cin + a\, b\, Cin \\ &= Cin\,(\sim a \sim b + a\, b) + \sim Cin\,(\sim a\, b + a \sim b) \\ &= Cin \oplus (a \oplus b) \end{aligned}$$

$$\begin{aligned} Cout &= \sim a\, b\, Cin + a \sim b\, Cin + a\, b \sim Cin + a\, b\, Cin \\ &= Cin\,(\sim a\, b + a \sim b) + a\, b\,(Cin + \sim Cin) \\ &= Cin\,(a \oplus b) + a\, b \end{aligned}$$

One bit adder can be implemented as follows based on the algebraic representation of Sum and Cout evaluated form the truth table.

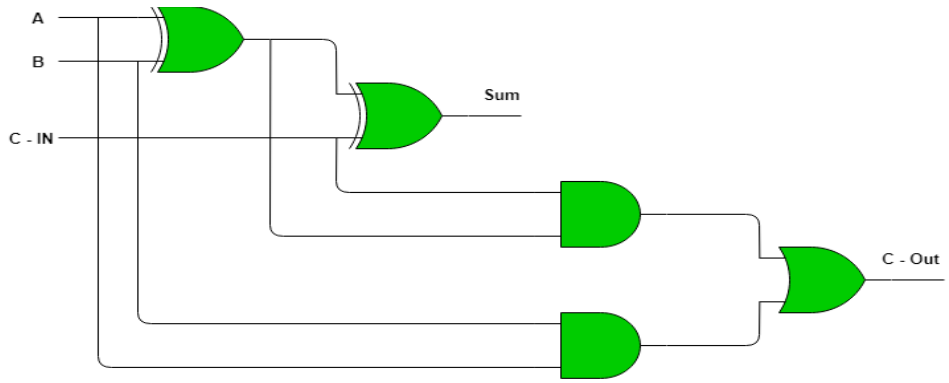


Figure 6:One-bit adder using non-universal gates [1]

In the digital IC design industry, the universal gates are used to implement every other gate because they are economical and easier to fabricate. Thus, the above design is optimized to be implemented in universal gate[1]s as shown in Fig. 8. The implementation can be achieved using 9

[1][1] source: https://www.geeksforgeeks.org/full-adder-in-digital-logic/?ref=rp

NAND gates or using 9 NOR gates. For this project NAND gates are chosen because NAND gate has low logic effort which makes it smaller and faster than NOR gate.[2]

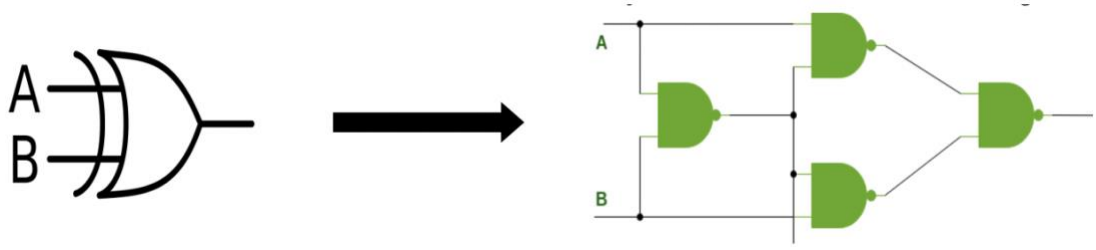


Figure 7: Equivalent XOR circuit Using NAND gates[1]

$$Cout = a\,b + Cin\,(a \oplus b)$$

Using DeMorgan's Law

$$Cout = \sim[\sim(a\,b)\sim((a \oplus b)Cin)]$$

For cout, the red representation shown is used during the implementation in order to reduce components. The resulting circuitry after replacing the gates in Fig. 6 by NAND gates, the following one-bit adder circuit was obtained.

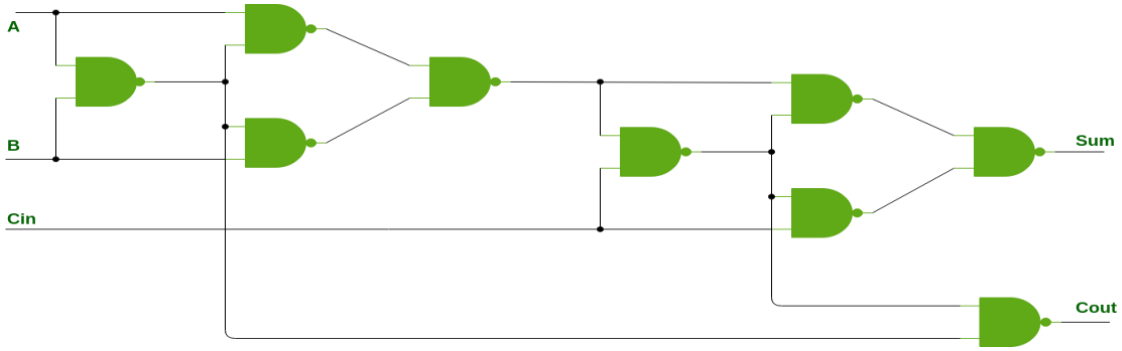


Figure 8: Single bit adder using NAND gates only[1]

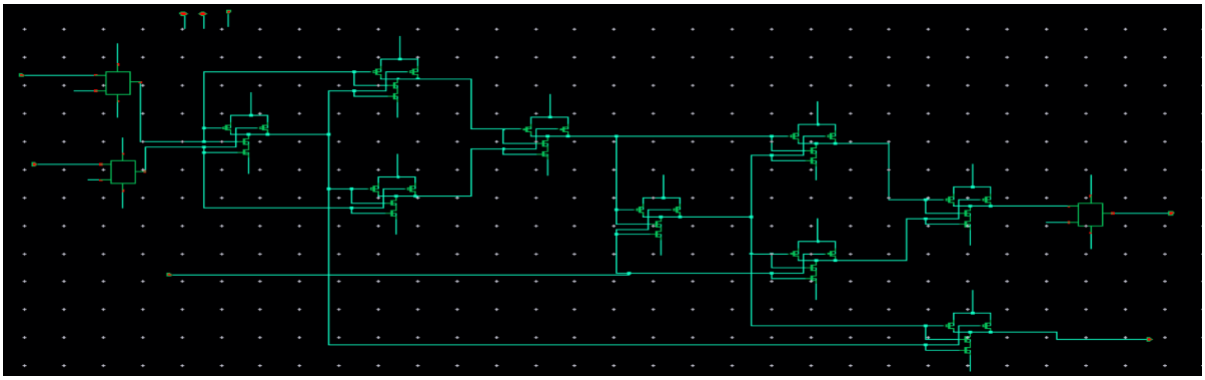
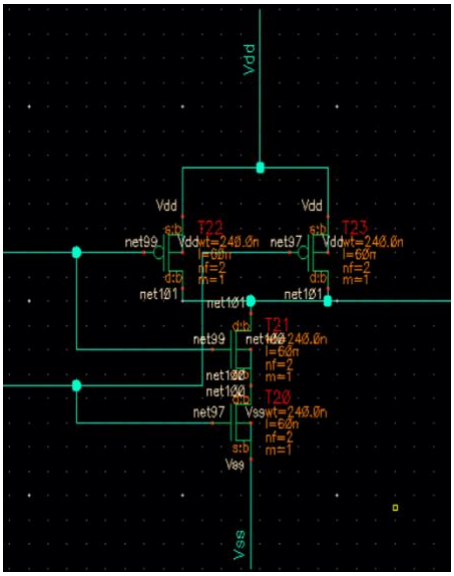


Figure 9:one-bit adder cadence design using NAND gates only

[2] [1] source: https://www.geeksforgeeks.org/full-adder-in-digital-logic/?ref=rp



The NAND gate that was used in the project is shown on the left-hand side. **The size was normalized to a unit inverter of size 2:1. Since the pull up transistors are parallel; they have the size of 2 and the pull-down serial transistors have size of 2 as well.**

Figure 10: NAND gate used for the project

In the final design, the inputs to the one-bit adder were flopped. The sum output was also flopped but since the carryout is required for the computation in the next one-bit adder it was not flopped. And it is necessary to make sure it arrives at the next stage before the next clock edge. Sample output is shown in Fig. 11. The first bit is given 0101011 sequenced while the second bit is given 1011001. The sum and carry out obtained were 1110010 and 0001001 respectively. N.B. neither the inputs nor the outputs shown below are flopped.

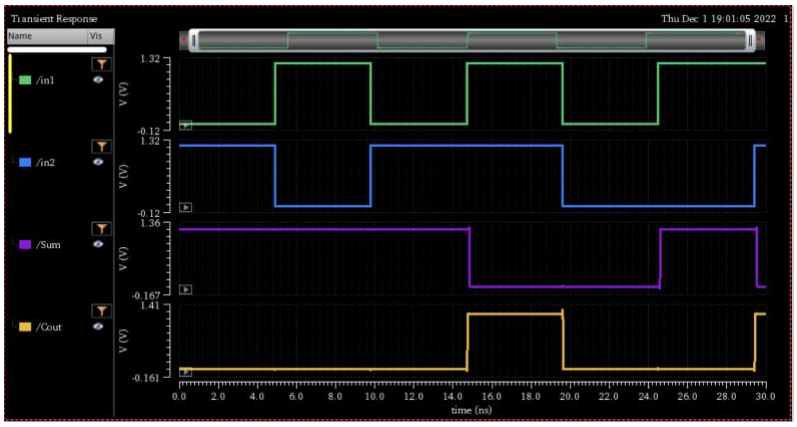


Figure 11:1-bit adder simulation result

The timing properties of the adder is shown in Fig. 12. For the sum to get into 50% of its initial value, it takes 93.081p seconds. For the carry out to get into 50% of its initial value it took 35.076p seconds. **Thus, the propagation delay of the adder is taken to be a 93.081p seconds.**

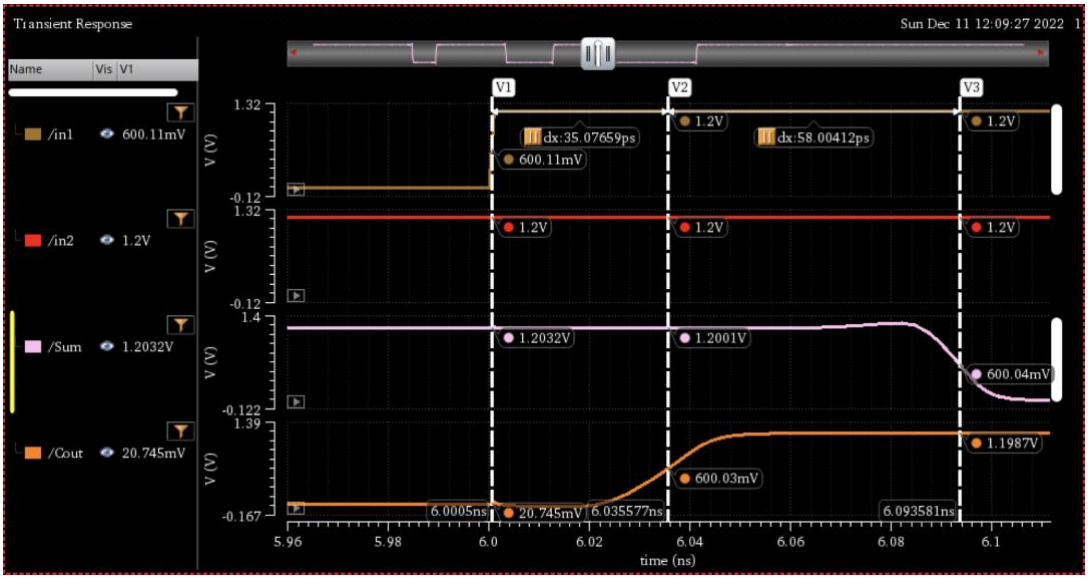


Figure 12: Timing property of 1-bit adder

b) 4-bit Adder

The parallel carry ahead 4-bit adder was designed using 4 single bit adders connected in parallel. This is advantageous for a high-speed adder but takes large areas. The c-out of every stage is fed as c-in to the next stage. The last c-out is then flopped. From this, we can perceive that the path from the inputs of the initial stage to the flopped c-out of the last stage is the longest path and will determine the maximum frequency of the whole circuit. The 4-bit adder is shown below:

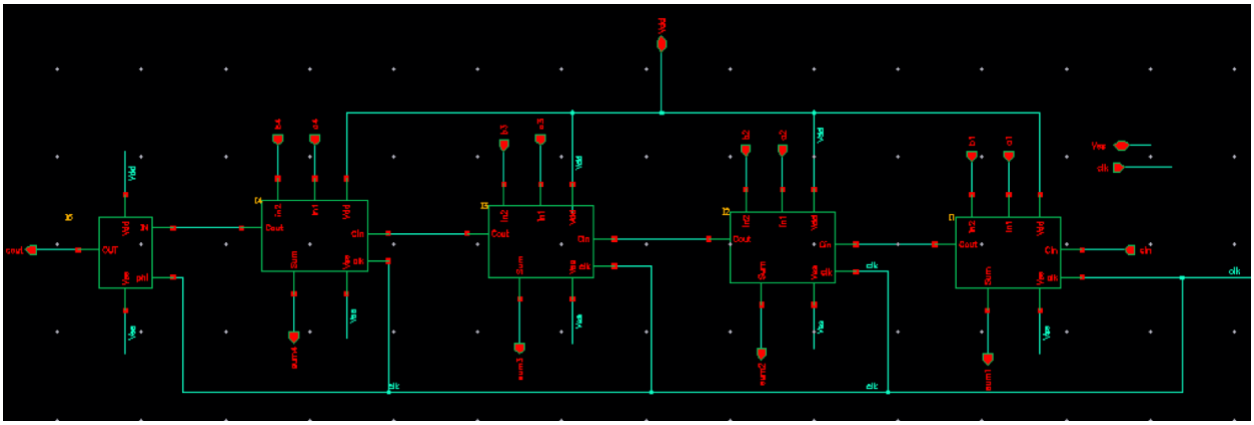
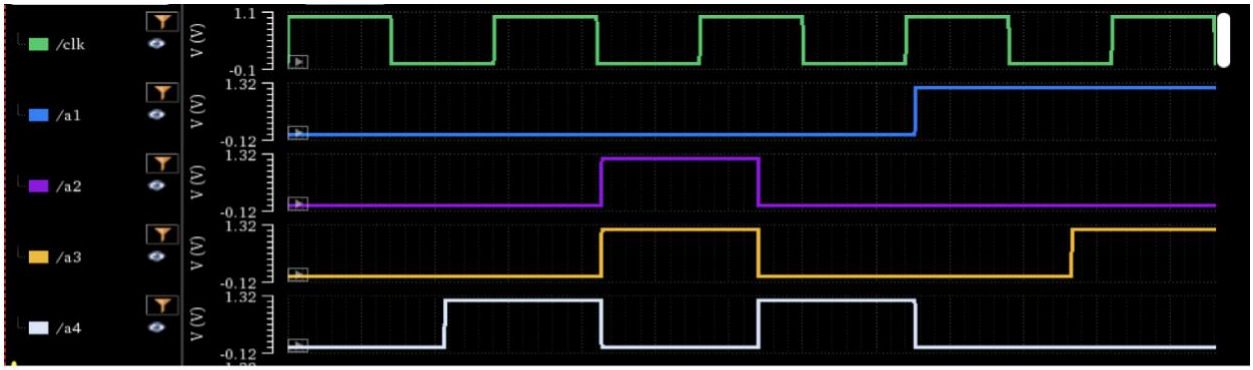
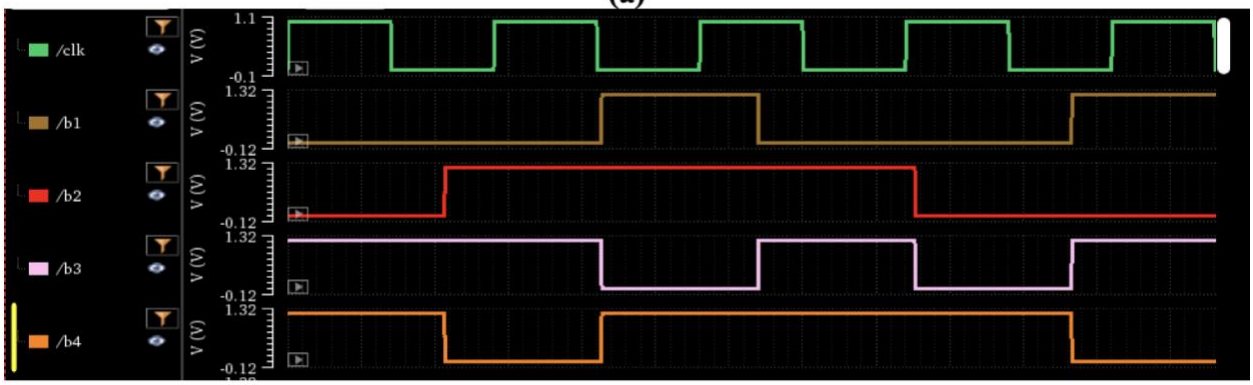


Figure 13:4-bit carry look ahead full adder

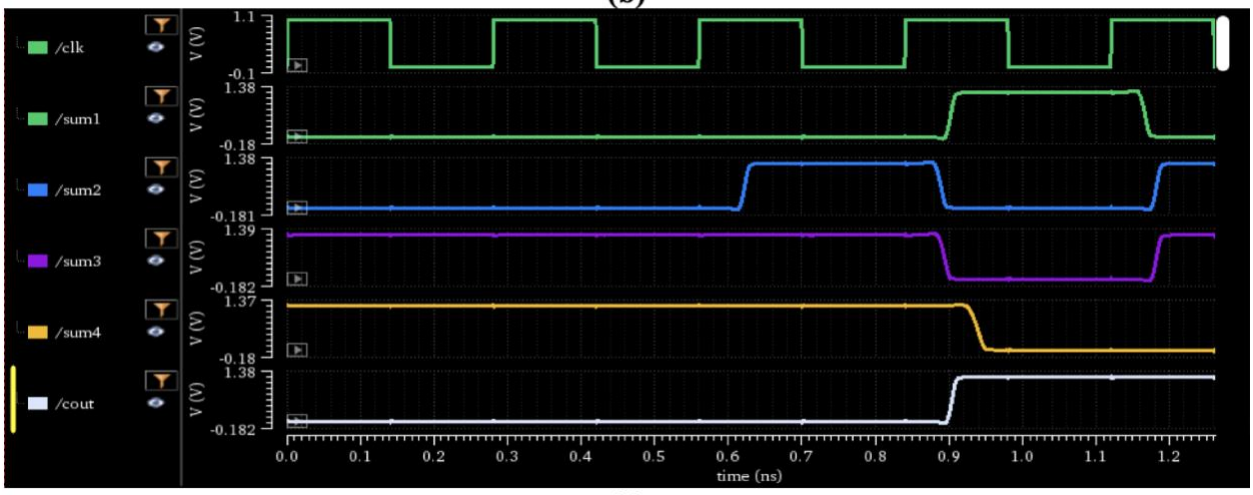
Sample output of the 4-bit adder is given below: first and second 4-bit numbers are displayed respectively, and the output of every stage is also displayed after.



(a)



(b)



(c)

Figure 14: (a) first input (b) second input (c) The output

1) The max. Frequency of the Circuit

As mentioned before, the carry path determines the maximum frequency of the circuitry. The max. frequency can be calculated using the formula:

$$Fmax \leq \frac{1}{t_{pcq} + t_{pd} + t_{setup}}$$

The clock-to-q propagation delay and setup time of the flipflop are 112.27p seconds and 50p seconds respectively. The propagation delay of the adder logic is 94p seconds. The rough guess using the above given formula, the maximum delay from flop to flop of a single stage is 256.27p seconds. Whereas for the longest path, the logic delay to the c-out of a single stage is around 35p seconds and for 4 stages is assumed to be 140p seconds. **Therefore, using the above formula for rough maximum delay considering the longest path is (112.27+140+50) p seconds = 302.27p seconds. This results in 3.308 GHz.**

The maximum frequency was also evaluated practically from the design. This is achieved by continuously decreasing the clock period to observe the final c-out or the sum to be captured in the next+1 clock edge instead of the next clock cycle. This signifies that the clock period chosen is smaller than the required one for the output to be captured in the next cycle. Once this property is observed the clock period is set to a correct width. And this is continuously experimented upon different input patterns and whenever the output is captured in the wrong clock edge, the clock period is increased. Following this method, **the clock period was of the found to be 300p seconds which means 3 GHz frequency.**

2) The Area of the Design

The area of the design is found by adding the width of all transistors used.

i) Each Flipflop →[4*(240+120) + 2*(240+120) + (8*120) + (480+240)] nm→3.84 um
ii) One bit adder (Logic sect) → 9*NAND = 9*(4*240n) → 8.64 um.
iii) One bit adder with FF → 3 ff + one bit adder (logic sect) →3*3.84 + 8.64 = 20.16 um
iv) The 4-bit adder design → 4 * (iii) + 1 ff = 84.48 um.

Area of the total design is width*length = 84.48um * 60nm =5.07 pm^2. = 5.07 x 10^{-6} um^2

The area can further be decreased by using a single inverter at the top module to generate the clock_inverse. My design has an inverter dedicated for clock_inverse generation in each flipflop **(realized this just now).**

3) Active leakage power with 50% input bits switching

The current when the 50% of the input bits are changing is plotted in Fig. 15. To calculate the active power a single period was chosen and the area under the signal was integrated as shown in Fig. 16:

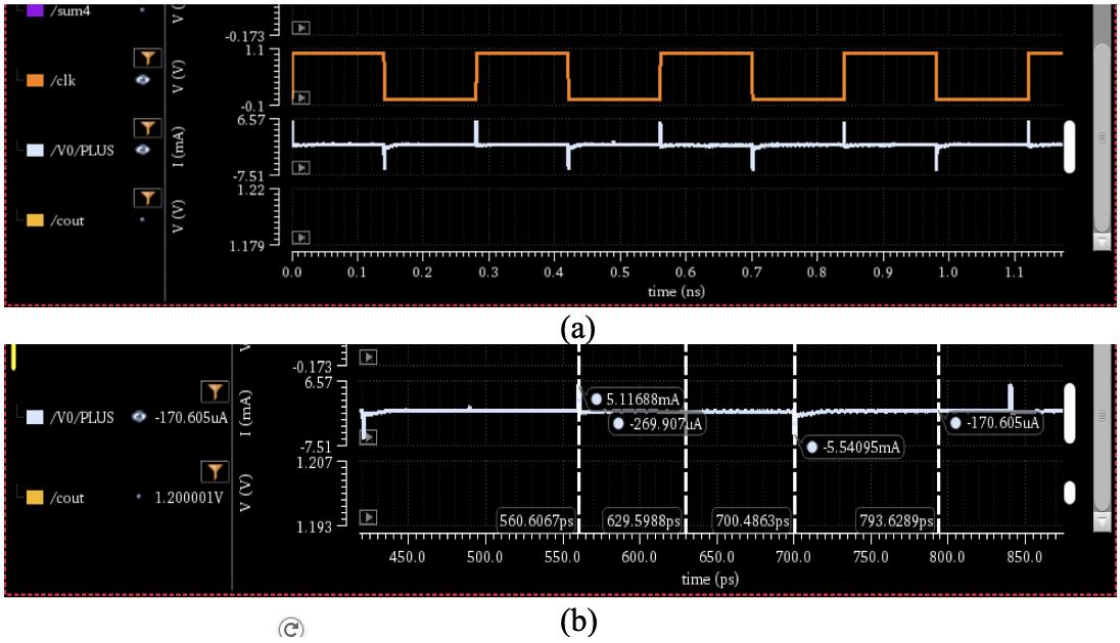


Figure 15: (a) Active current plot (b) values of the current

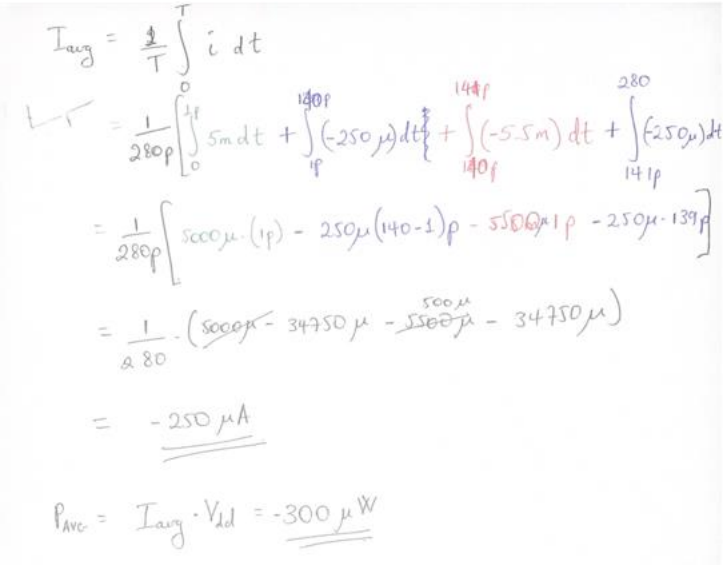


Figure 16: Power Calculation

The average current in a period was calculated to be around -250 microamps. Multiplying average current with Vdd gave the active power of -300 uW.

The average current was then calculated using the built-in calculator in Cadence and was found to be -180 microamps. Multiplying it with 1.2V resulted into active power of -216 uW.

4) Leakage current

All the inputs are set not to switch. The clock is also turned off. Fig. 17 shows the result of the simulation with everything off.

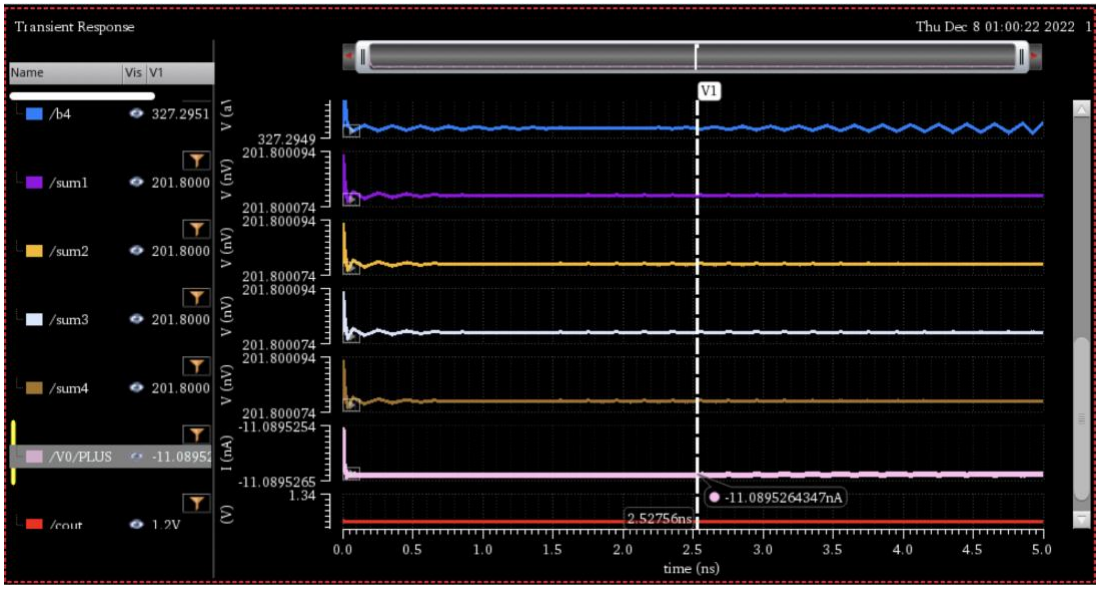


Figure 17: Leakage Current

The leakage current is found to be a constant of value 11.1n A. The leakage power is then calculated using

$$P_{leak} = I_{avg_{leak}} * V_{dd} = 11.1n * 1.2 = 13.32n\ W$$

5) Clock Skew and Clock Generator

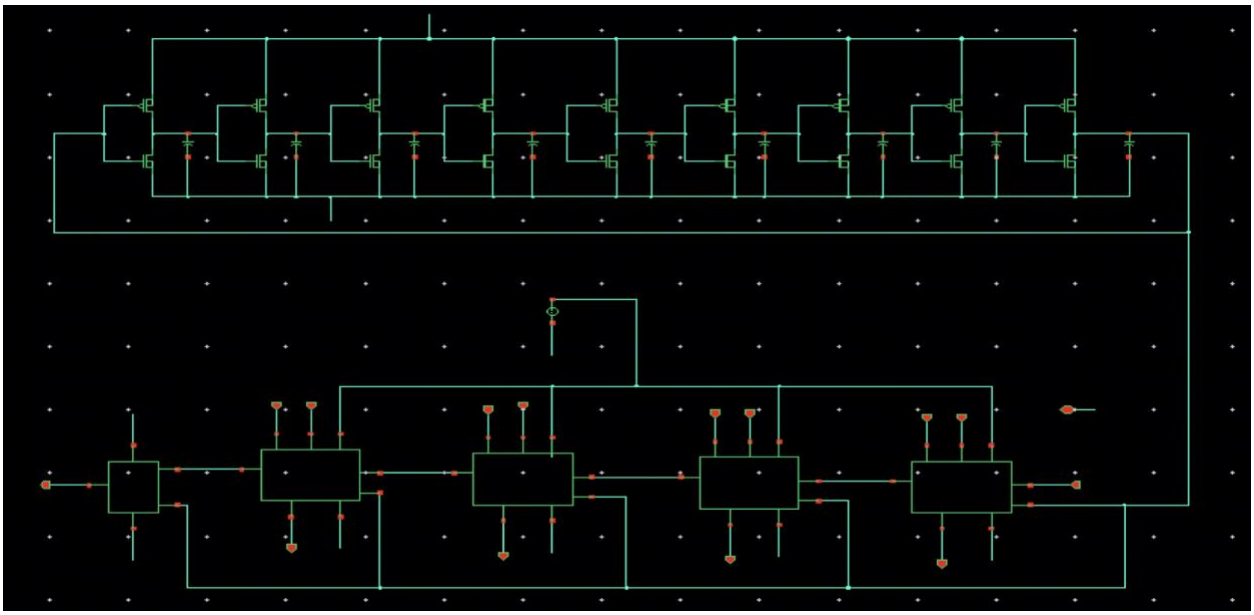


Figure 18: 4-bit adder with clock generated by an oscillator

A ring oscillator made by 9-inverters was designed as shown in Fig. 18. Capacitors of different values were connected at the outputs of each inverter to generate the desired clock period. The output of the oscillator was directed as input to the 4-bit adder designed. The adder worked fine. For the same input, the same output was evaluated. **And for a small circuitry such as the one designed here, using the same clock does not result in notable clock skew**. **However, there is more jitter as the rise time of the clock increased significantly** i.e., from 1p second to 671p seconds. The capacitors take time to be charged and discharged thus the clock rising and falling edges become sloppy. The clock of 5n seconds period can be shown in Fig. 19.

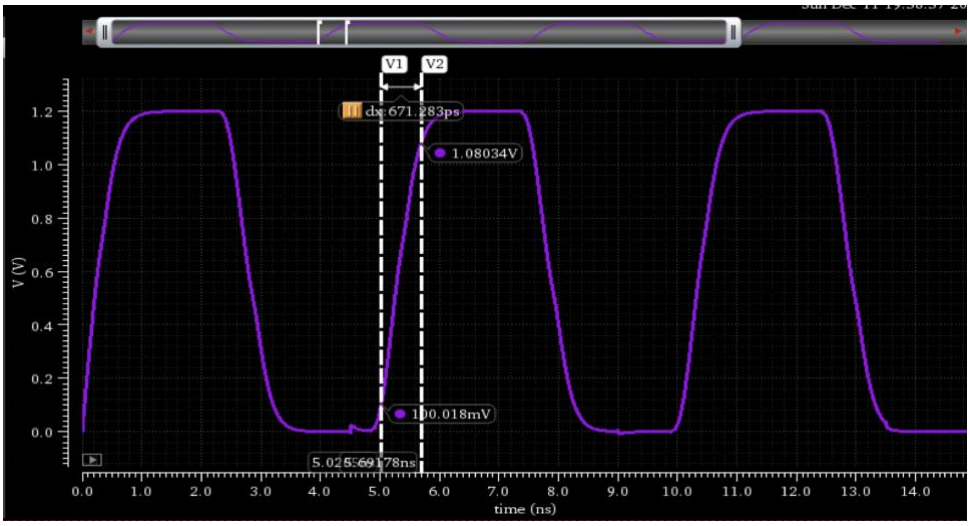


Figure 19: clock generated by the oscillator

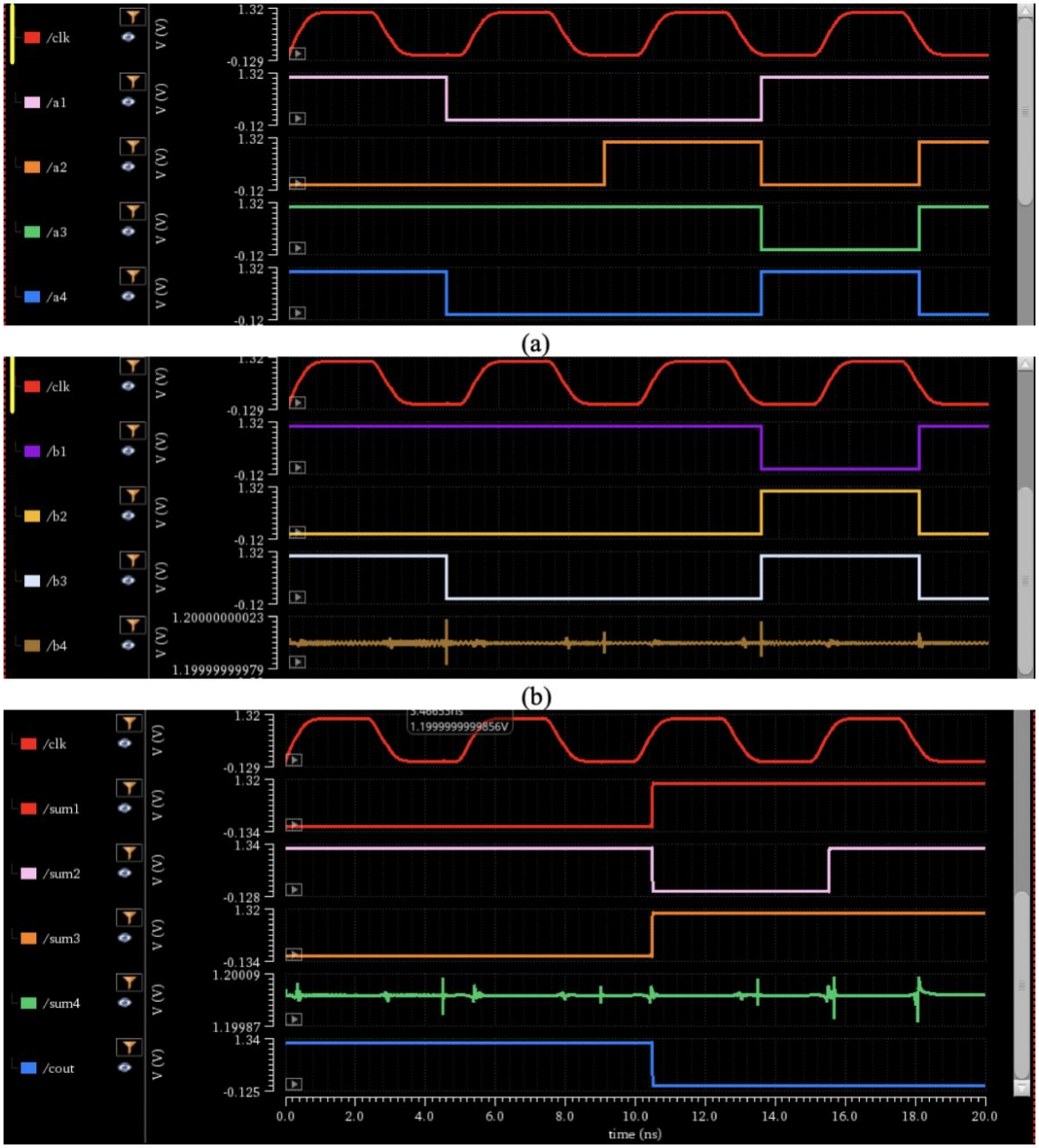


Figure 20:$-bit adder with oscillator simulation result