

2023 NYCU EE VLSI Lab Report

Lab01 A CMOS Full Adder: Hspice Simulation

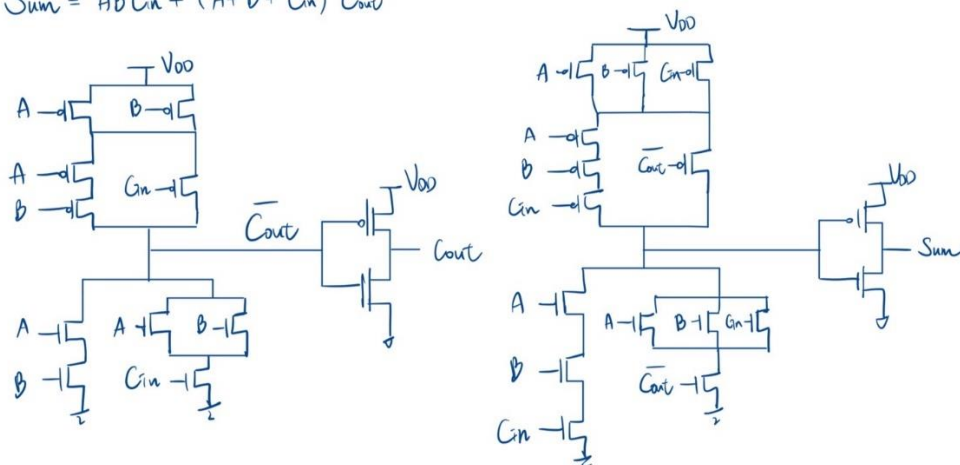
Student ID: 110511254 Name: 徐煜絨 Date: 2023/10/11

I. Summary of your Structure

1. Picture

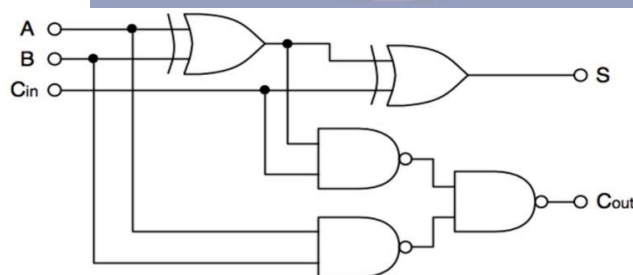
$$C_{out} = AB + (A+B) C_{in}$$

$$Sum = ABC_{in} + (A+B+C_{in}) \overline{C_{out}}$$



2. Design concept

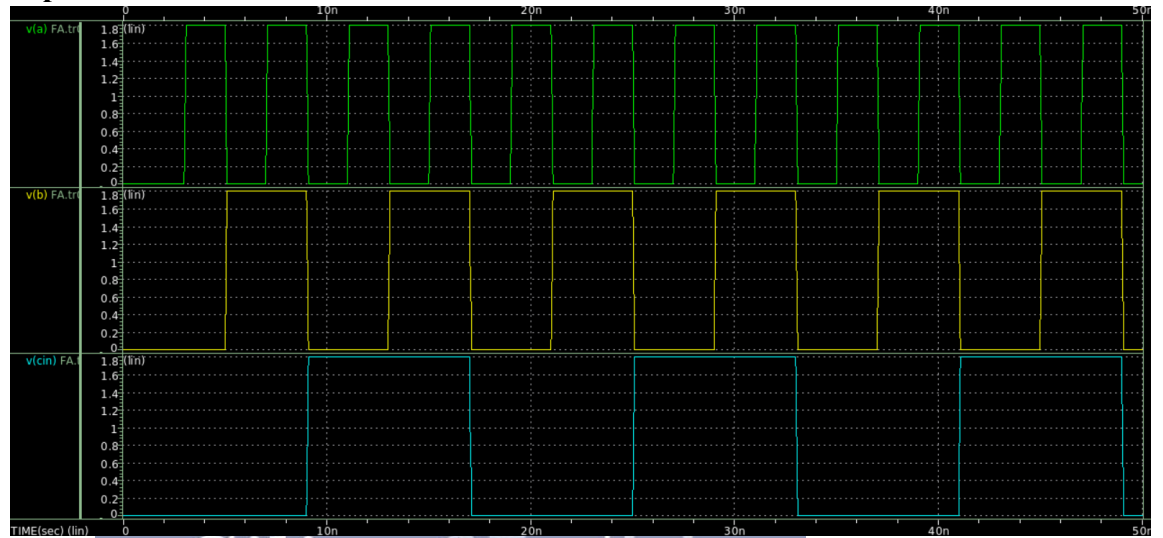
原先設計是使用兩個 xor gate 和三個 nand gate (如下圖)，但在執行時發現有肉眼可見且大小約 0.1V 的 glitch，猜測是因為輸入和輸出間有太多層 gate，導致後面訊號需要等待前面訊號而造成過多 delay。



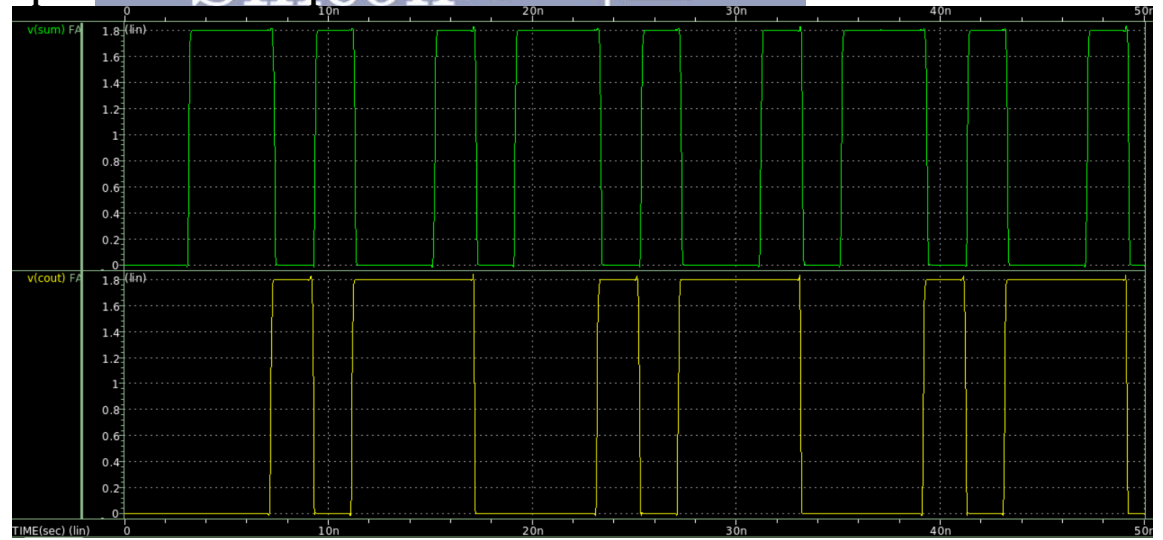
之後到網路上尋找和 full adder 有關的設計，有人將 SUM 視為 A, B, CIN, COUT 的組合 (如最上圖)，而這個設計相較原本可以讓輸出訊號較為平整，同時 propagation delay 也從 1.669×10^{-10} 秒降到 1.648×10^{-10} 秒 (當 Cload=10fF、pmos 和 cmos 的 width=1.8um，計算時間從輸入 A 第四次 rise 到輸出 SUM 第三次 rise。若計算是由 A 第三次 rise 到 SUM 第二次 rise，則能夠下降更多，從 2.839×10^{-10} 秒減少到 1.136×10^{-10} 秒)。

II. Output waveform (Unit: wp and wn -> μm ; output load -> fF)

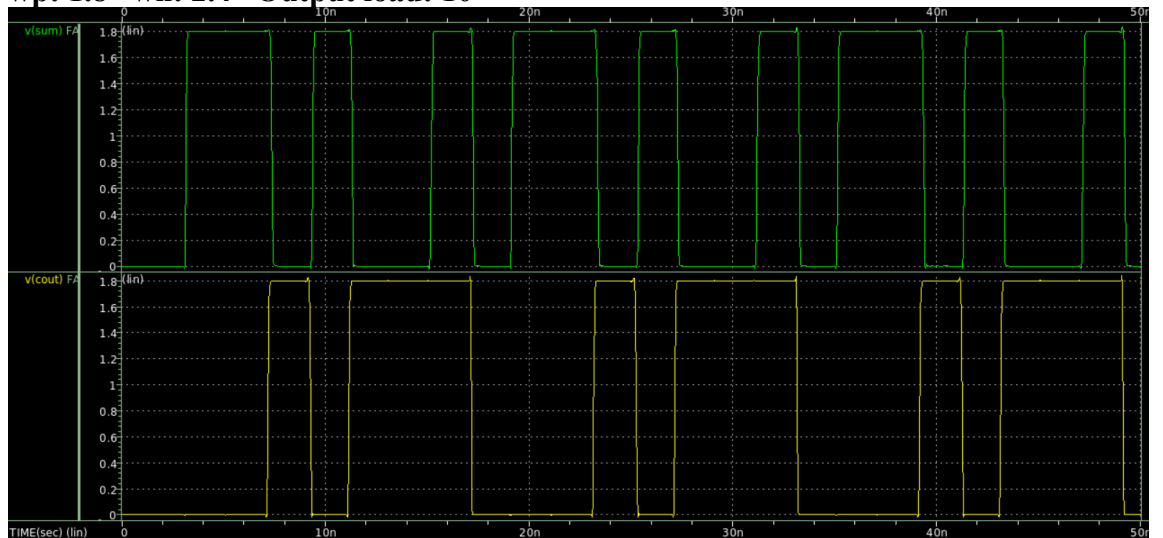
Input waveform:



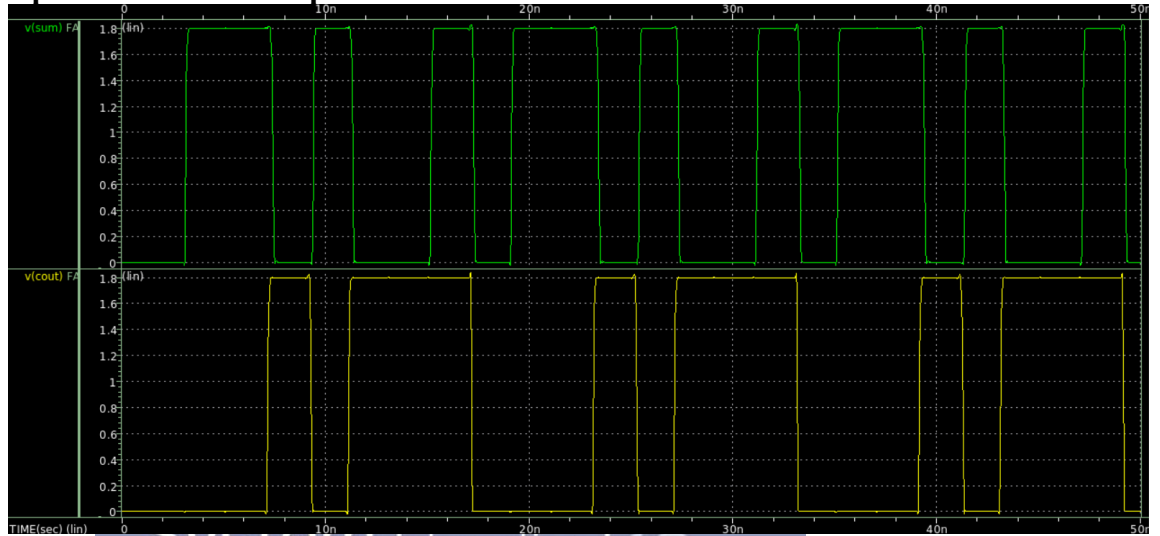
1. wp: 1.8 wn: 1.8 Output load: 10



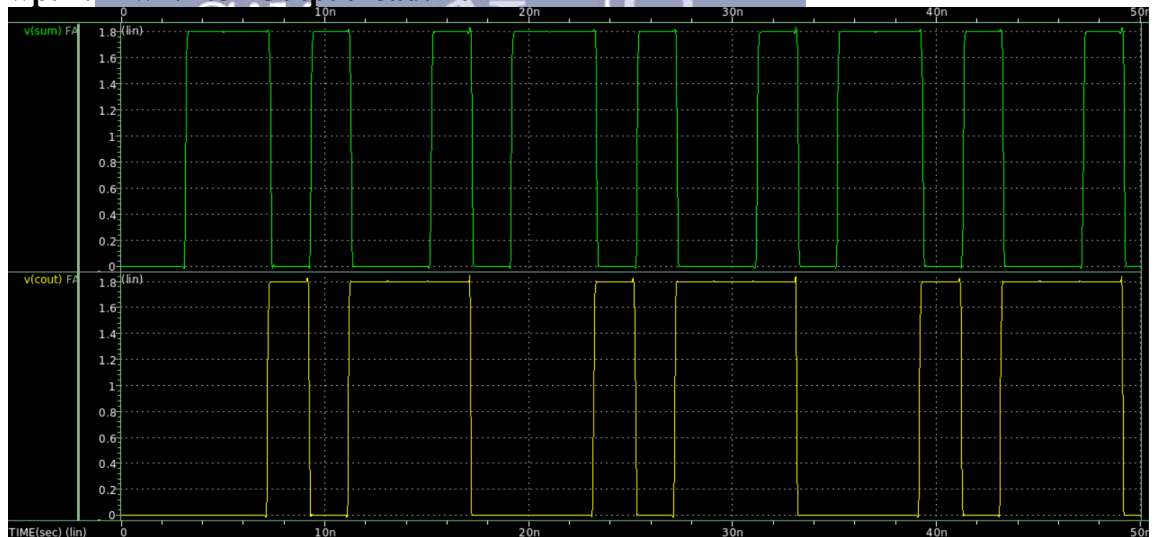
2. wp: 1.8 wn: 2.4 Output load: 10



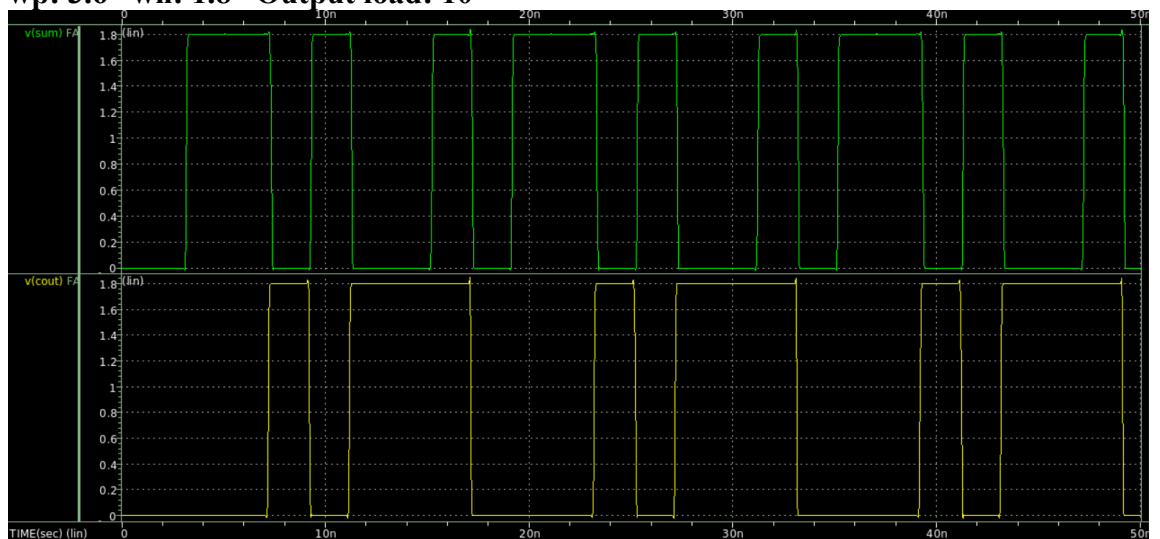
3. **wp: 1.8 wn: 3.6 Output load: 10**



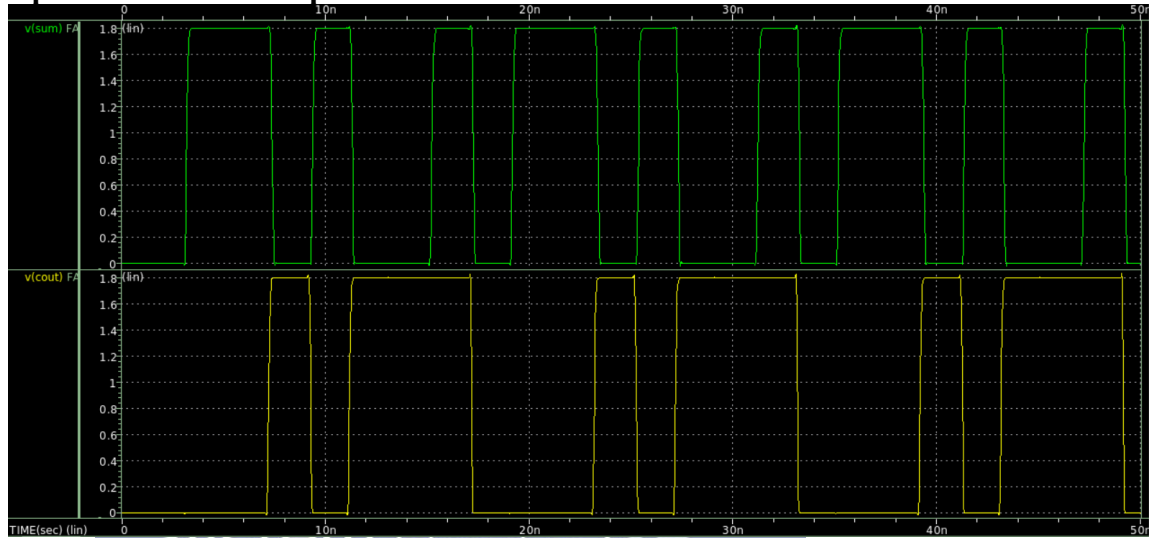
4. **wp: 2.4 wn: 1.8 Output load: 10**



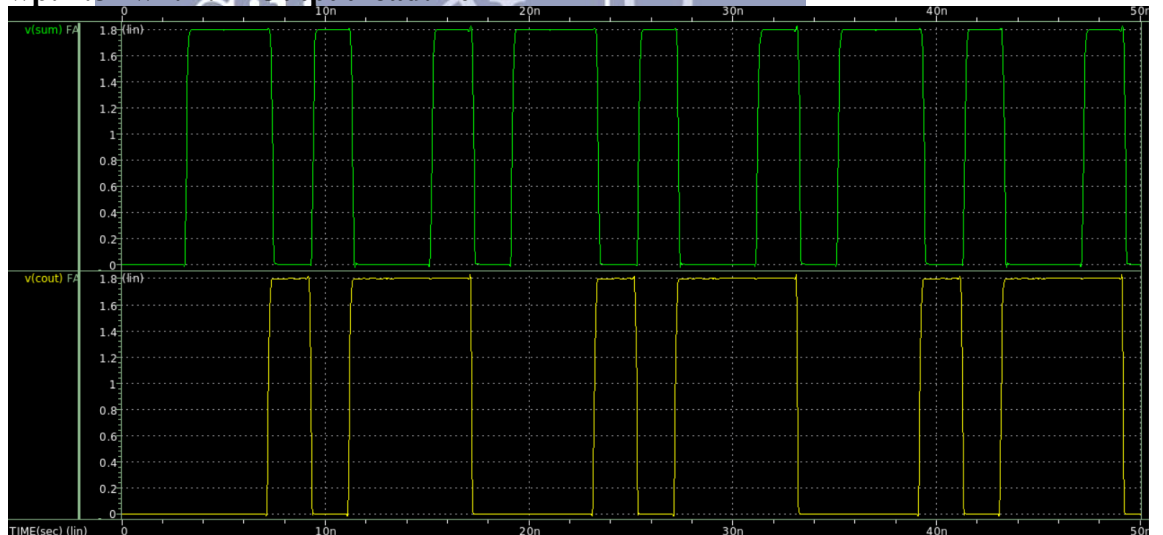
5. **wp: 3.6 wn: 1.8 Output load: 10**



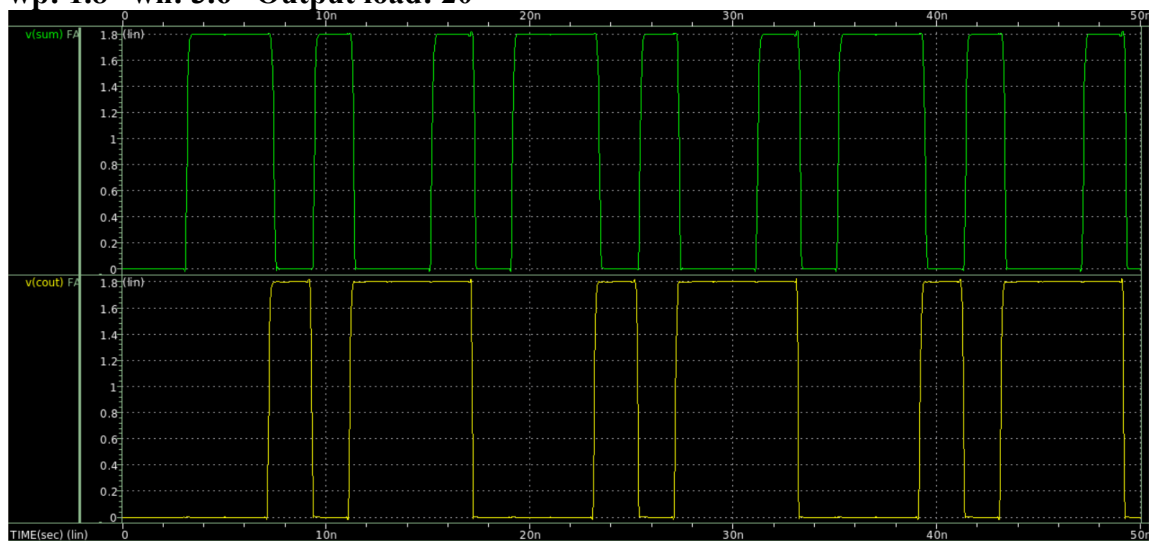
6. **wp: 1.8 wn: 1.8 Output load: 20**



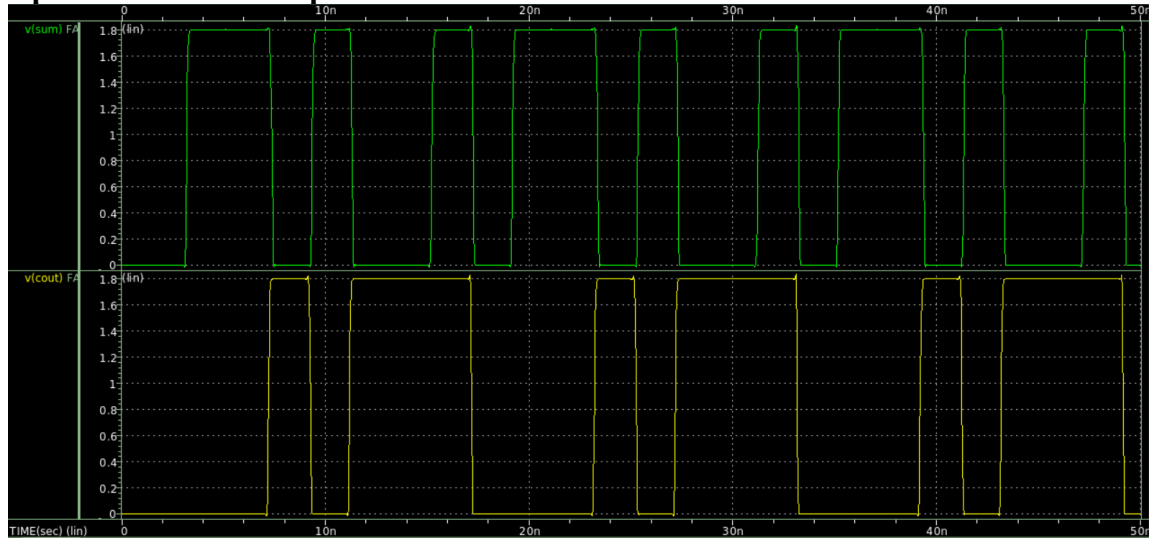
7. **wp: 1.8 wn: 2.4 Output load: 20**



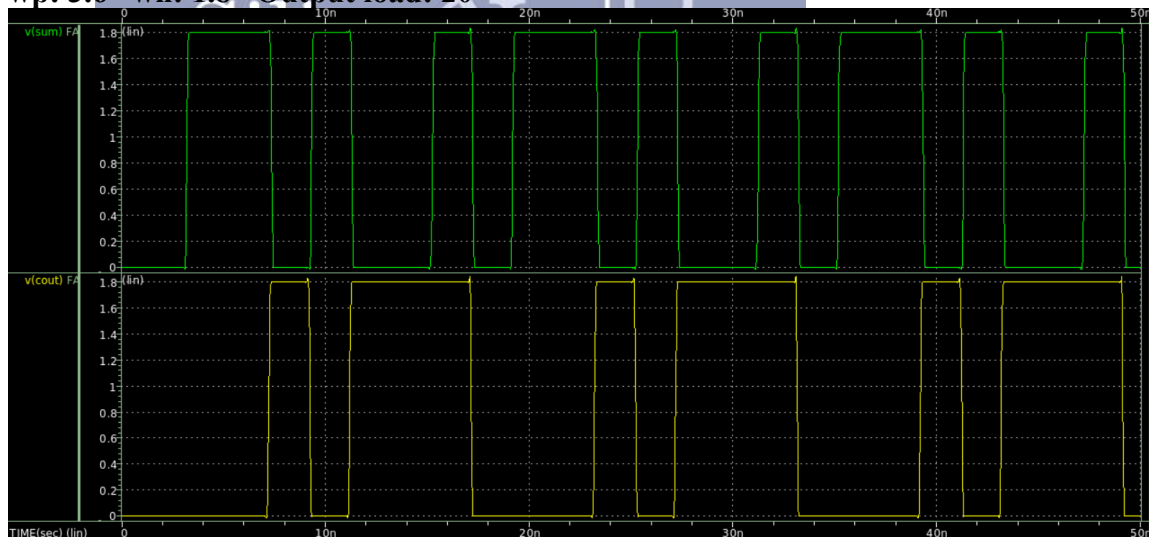
8. **wp: 1.8 wn: 3.6 Output load: 20**



9. wp: 2.4 wn: 1.8 Output load: 20



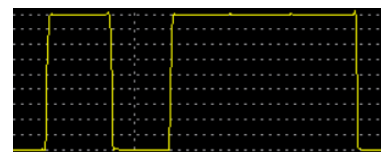
10. wp: 3.6 wn: 1.8 Output load: 20



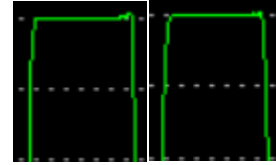
11. Observations

首先觀察第一到第三張輸出波形，這三張改變的是 wn ，比較前兩個可以發現當 wn 是 $1.8\mu m$ 時，波形上升和下降都比較垂直，第二個 ($wn=2.4\mu m$) 要上升前有些微的 delay，且上升斜率比較小。當觀察到第三個輸出波形時，可以發現上升和下降的線都不如前兩張輸出波形垂直，也有可能因為 delay 更大，導致訊號要變化時的 glitch 變得較不尖銳，不像前兩張有太突然的凸起（凸起時間拉長）。

接著是第一、四、五張比較。其中令我比較訝異的是第四張的 COUT 波形在偶數次輸出為 $1.8V$ 時（右側較寬的為第偶數次升到 $1.8V$ 的波），會出現兩次 glitch，此時 wp 是 $2.4\mu m$ ，但當 wp 是 1.8 和 $3.6\mu m$ 都不會有這樣的結果。至於 SUM 要下降的瞬間，隨著 wp 增加，glitch 變小，左側的量尺也因此每 $0.2V$ 的高度放大。



再來是第一和第六張比較，這次改變的是 Cload 的電容值，從 10fF 增加到 20fF。結果和預期相似，隨著 Cload 上升，輸出波形會變得更加平滑，當 SUM 和 COUT 即將貼近 1.8V，若 Cload=10fF，線是直接碰到 1.8V 再轉向變平，但若 Cload=20fF，在達到 1.8V 前會明顯出現曲線，這是加上電容會有的結果；而當 SUM 和 COUT 要從 1.8V 回到 0V，Cload=10fF 時的（右上左圖）glitch 比 Cload=20fF 時的嚴重（右上右圖）。



對 Cload=20fF 的最後五張，除了輸出在即將抵達 1.8V 和要從 1.8V 下降時相較於 Cload=10fF 的前五張會有較圓滑曲線外，不同 wp、wn 間的比較結果大致和前兩段描述相同，即隨著 wn 增加，輸出的上升和下降的斜率絕對值都會減少且 glitch 變嚴重，但特別的是 wp=1.8 μ m、輸入訊號同時從 1.8V 回到 0V 時，COUT 在 wn=2.4 μ m 的 glitch 比 wn=3.6 μ m 明顯。


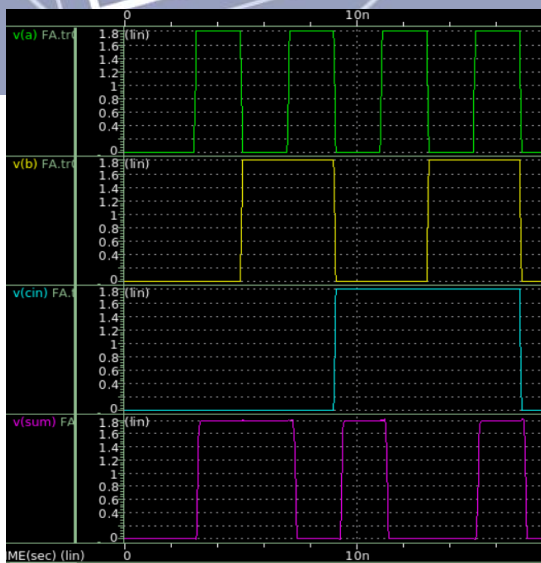
III. Measurements

1. Table

Cload (fF)	W _{nmos} (μ m)	W _{pmos} (μ m)	AVG_PW (W)	TPD (s)	TRISE (s)	TFALL (s)
10	1.8	1.8	5.931e-05	1.648e-10	1.022e-10	8.870e-11
10	2.4	1.8	6.815e-05	1.557e-10	1.078e-10	9.629e-11
10	3.6	1.8	8.620e-05	1.467e-10	1.064e-10	1.090e-10
10	1.8	2.4	6.647e-05	1.714e-10	9.263e-11	7.883e-11
10	1.8	3.6	8.062e-05	1.913e-10	8.826e-11	7.059e-11
20	1.8	1.8	6.902e-05	1.927e-10	1.517e-10	1.133e-10
20	2.4	1.8	7.782e-05	1.810e-10	1.462e-10	1.199e-10
20	3.6	1.8	9.544e-05	1.721e-10	1.475e-10	1.273e-10
20	1.8	2.4	7.598e-05	1.931e-10	1.309e-10	1.053e-10
20	1.8	3.6	9.012e-05	2.089e-10	1.148e-10	9.055e-11

2. Code (please describe)

Code	.meas TRAN AVG_PW AVG power
Description	我的 simulation 設定在 0.02ns~50ns，上面的 code 可以得到系統在這段時間消耗的平均功率。

Code	<pre>.meas TRAN TPD TRIG V(A) VAL='supply*0.5' rise=4 + TARG V(SUM) VAL='supply*0.5' rise=3</pre>
Description	<p>TPD 是這個量測的名字，量的是輸出和輸入間的 propagation delay，上面的方法是參考助教提供的 INV.sp 所決定，A 變化到 0.9V (supply=1.8V) 後，SUM 要過多久才會到 0.9V，如下圖，但下圖和上面的 code 並不符合，是為了能夠清楚表達 TPD 的概念，才將 A 和 SUM 的上升、下降方向畫相反。</p> 
Code	<pre>.meas TRAN TRISE TRIG V(SUM) VAL='supply*0.1' rise=3 + TARG V(SUM) VAL='supply*0.9' rise=3</pre>
Description	<p>因為相較於 COUT，SUM 較穩定（討論區有提及，我的波形確實如此），所以 TRISE 和 TFALL 都選擇測量 SUM 的波。量測時間是抓第三次的 rise，因為此時 A 和 B 兩個輸入波都已經穩定，CIN 雖然還在第一個週期，但 SUM 第三次上升時 CIN 並沒有變化（如下圖）。另外，0.1、0.9 倍 supply 一樣是參考助教提供的 INV.sp 而決定。</p> 
Code	<pre>.meas TRAN TFALL TRIG V(SUM) VAL='supply*0.9' fall=2 + TARG V(SUM) VAL='supply*0.1' fall=2</pre>

Description	<p>會取第二次 fall 和前面計算 rise 時間的原因類似，是為了能取到較穩定狀態的數據，在 SUM 第二次 fall 的時候，A 已經要進入第三個週期，B 走完半個週期回到 0V，CIN 則待在 1.8V 處沒有變化（如上圖），又推測此時 SUM 的改變只會和 A 的變化有關，所以選擇在 fall=2 時測量，如此也不會像 fall=3 時，A、B、CIN 都改變，使影響 TFALL 的原因變得複雜。</p>
-------------	--

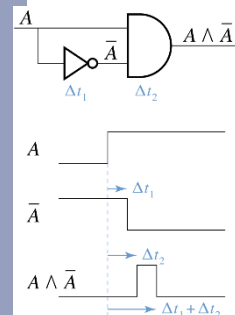
IV. Questions

1. Explain why there are glitches in FA combinational circuit sometimes, and how to fix it. Is it harmful for your overall design?

FA combinational circuit 中會有 glitch 有以下幾個原因：

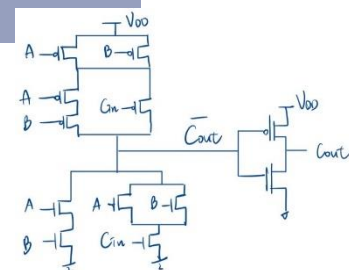
- Race conditions

若輸入訊號會受時間影響，race conditions 就有可能發生。當輸入同時改變或是按特定順序改變，因為各路徑有不同 delay，會使輸出端產生 glitch。如右邊示意圖，其表示因為 \bar{A} 訊號進入 and gate 前會先經過 inverter 造成 delay，使得實際輸出和期待輸出會有所出入，出現長達 Δt_1 的凸起。



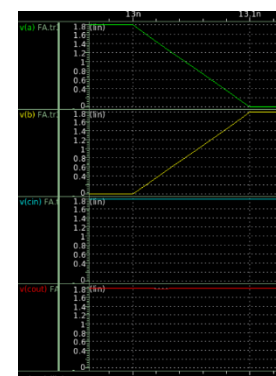
- Signal skew

輸入訊號的上升和下降邊緣不對齊，或是 trise, tfall, delay 時間不同，這些因素都可能造成訊號偏移，進而影響電路的運算過程和結果，使輸出出現 glitch。



- Delay discrepancies

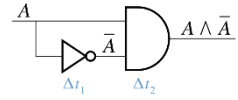
以我的設計為例，就算輸入端的 A, B, CIN 同時改變，他們在電路中會經過不同路徑長，並可能經過不同數量的電晶體，這些訊號就不容易同時匯入 \overline{COUT} ，造成輸出訊號和預想的不同，並在某些特定條件下出現 glitch（例如 CIN 不變、A 訊號從 1 降到 0、B 訊號從 0 升到 1，但因為 A、B 兩訊號影響到 \overline{COUT} 的時間不同，造成輸出訊號有些微的 glitch）。



● Others

其他因素還有如輸入訊號帶有雜訊、訊號傳遞途中有干擾，這兩項因為目前是用軟體模擬，猜測不會對輸出結果有太大影響；另外還有這次作業沒有出現的 feedback loop，在這種電路中，元件的狀態變化可能會導致短暫的不正常輸出。

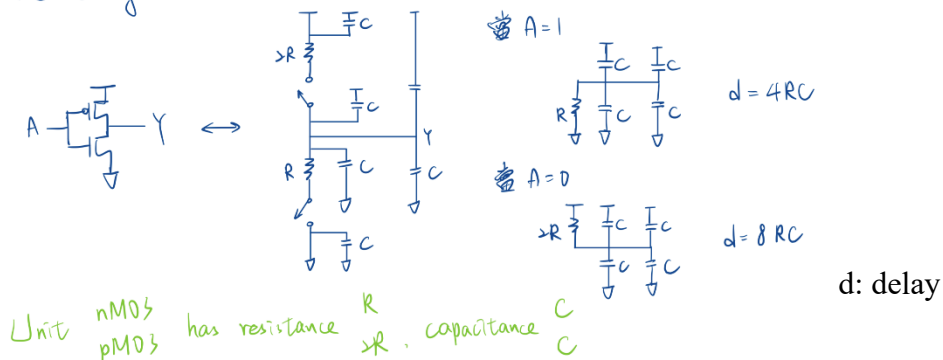
對於解決方法，若 race conditions 是造成 glitch 的主要原因，可以在輸入的地方使用 flip-flop 決定何時要把輸入訊號往後傳送到電路裡，例如在 and-gate 前增加一個 flip-flop，讓 A 和 \bar{A} 能同時進到 and-gate 裡。Signal skew 和 delay discrepancies 則能夠透過電路的佈局改善，像是對於比較早到的訊號給予較長的路徑，或在電路上設置能夠使傳播延遲的元件。



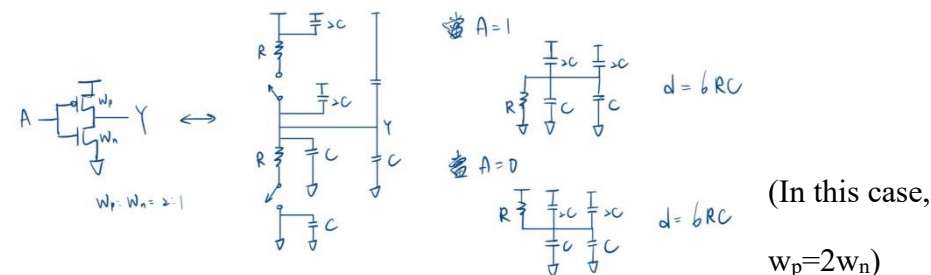
2. Explain how you decide MOSFETs' width with fixed channel length in your circuit (Hint: explain with mobility of PMOS NMOS, prove it with HSPICE)?

已知 nmos 在 saturation region 中 $I_{ds} = \frac{1}{2} \mu C_{ox} \frac{W}{L} (V_{gs} - V_t)^2$ ，且根據 VLSI 導論的課堂講義，相較於被電子所決定的 μ_n ，被電洞所決定的 μ_p 通常會是 μ_n 的 $\frac{1}{3}$ 到 $\frac{1}{2}$ 倍，再參考上面的方程式，假設今天有一組 inverter，如果上方的 pmos 和下方的 nmos，他們的 width 相同，就會造成兩者在導通時產生的電流不同，pmos 的電流將會比 nmos 的小，所以 output 會比較難從 GND 拉升到 VDD，卻容易從 VDD 轉到 GND。

RC delay model



在 length 固定時， I_{ds} 會正比於 μW ，若這時將 pmos 的 width 變成原先的兩倍，則會變成：

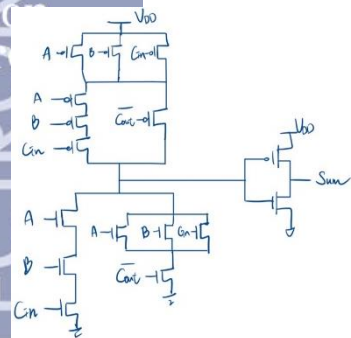


此時，不論 A 是從 0 變 1 或是 1 變 0，耗時並不會有太大的落差。

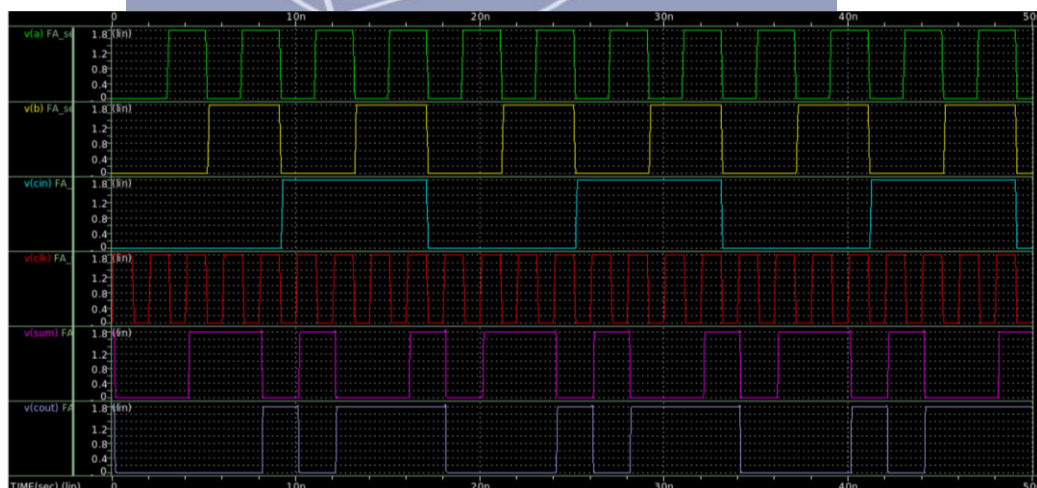
Cloud (fF)	W _{nmos} (μm)	W _{pmos} (μm)	AVG_PW (W)	FALL 比 RISE 快幾倍	TRISE (s)	TFALL (s)
10	1.8	1.8	5.931e-05	1.15	1.022e-10	8.870e-11
10	1.8	2.4	6.647e-05	1.18	9.263e-11	7.883e-11
10	1.8	3.6	8.062e-05	1.25	8.826e-11	7.059e-11
20	1.8	1.8	6.902e-05	1.34	1.517e-10	1.133e-10
20	1.8	2.4	7.598e-05	1.24	1.309e-10	1.053e-10
20	1.8	3.6	9.012e-05	1.27	1.148e-10	9.055e-11

這是擷取自第三部分的表格，從中可以發現當 W_{pmos} 增加，TRISE 和 TFALL 的表現會變好，變得更短。在這裡我另外計算 RISE 和 FALL 耗時的倍數關係，可以發現隨著 w_p 增加，TFALL 減少的速度會比 TRISE 更快（但當 $\text{Cloud}=20\text{fF}$ 、 $w_n=w_p=1.8\mu\text{m}$ 時，出現了意外高的 1.34，推測是模擬的誤差或物理特性造成）。

前述隨著 w_p 增加，RISE 所需時間可能也會增加（例如 inverter 就從 4RC 的 delay 變成 6RC），但表格中發現 TRISE 有改善，可能是因為 SUM 會用到 $\overline{\text{COUT}}$ 訊號，且右圖電路中還有許多 pmos，使整體在時間上的表現變好。



V. Bonus



上圖是 bonus 的波形圖，由上到下分別是 A, B, CIN, CLK, SUM, and COUT。題目規定輸入訊號只能在 CLK 的 negative edge 改變，且 DFF 要是 positive edge triggered，所以其他輸入訊號的週期要是 CLK 週期的偶數倍，才能確保符合題目限制（週期：A: 4ns、B: 8ns、CIN: 16ns、CLK: 2ns）。

Bonus 的 full adder 我使用 2 個 xor 和 3 個 nand 組成，並在輸出端加上 DFF，這裡使用這些邏輯閘是因為如果輸出要 positive edge 才觸發，就不需要像 combinational circuit 一樣，擔心輸出訊號會一直隨著輸入訊號變化並產生 glitch，加上 DFF 後，輸出只會要在要從 1 降回 0 時有些微的 glitch。

下表是各種組合所測量的結果，因為上面討論到 pmos 的 width 會影響 glitch，所以這次我只改變 wp，想看在 DFF 加入後會不會改善 wp 小時的情況。另外，TPD 原本是取 V(A) VAL='supply*0.5' rise=4 到 V(SUM) VAL='supply*0.5' rise=3，改成取 V(CLK) VAL='supply*0.5' rise=3 到 V(SUM) VAL='supply*0.5' rise=1 的耗時，因為 CLK 改變，輸出才會改變。表格中的紅字是比較後的最佳數據。

W _{nmos} (μm)	W _{pmos} (μm)	AVG_PW (W)	TPD (s)	TRISE (s)	TFALL (s)
1.8	0.54	1.590e-04	1.574e-10	5.379e-11	3.524e-11
1.8	0.72	1.747e-04	1.484e-10	4.563e-11	3.050e-11
1.8	1.2	2.136e-04	1.415e-10	4.244e-11	2.655e-11
1.8	1.8	2.600e-04	1.439e-10	3.550e-11	2.660e-11
1.8	3.6	3.793e-04	1.568e-10	3.451e-11	2.720e-11
1.8	7.2	6.127e-04	1.885e-10	4.257e-11	3.514e-11
1.8	12	9.332e-04	2.311e-10	4.917e-11	4.238e-11

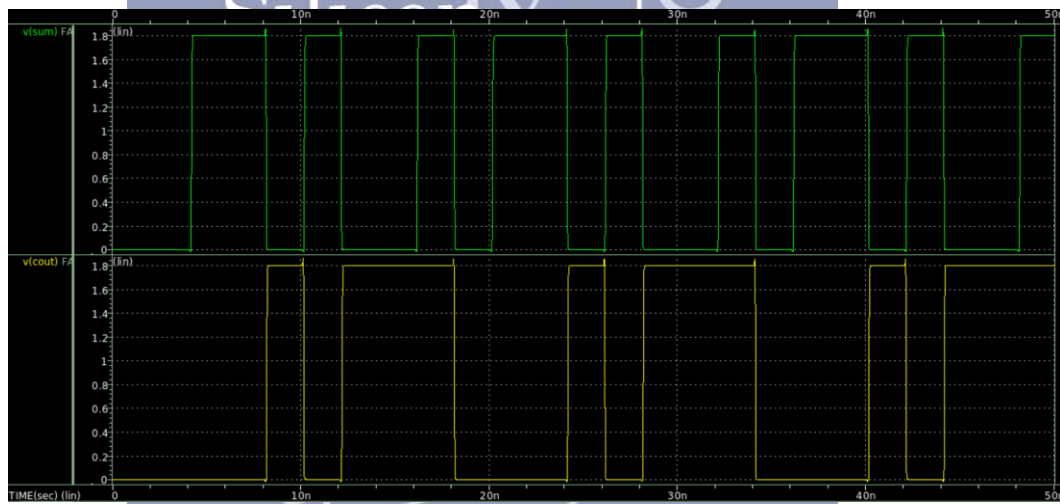
可以發現 pmos 在 width 最小時，會有最低的功耗，但 delay, rising, falling time 並非和 pmos 大小呈相關。最終要選用哪種 pmos 應該和客戶的需求有最大的關係。

在 waveform 中可以發現 $w_p \geq 1.8\mu\text{m}$ ，在輸出波形最一開始會有從 1 掉到 0 的波，應該可以視為還沒到 CLK posedge 前的暫態行為，之後我有試過 CLK 加上 2n 的 delay，但發現會讓輸出長時間待在 1.8V 的位置，所以 CLK 必須在系統一啟動就開始執行，並在最短時間內讓輸出穩定下來。另外比較特別的是隨著 wp 增加，SUM 要從 1 降到 0 時會有較小的 glitch，下面是各種組合的波形圖。

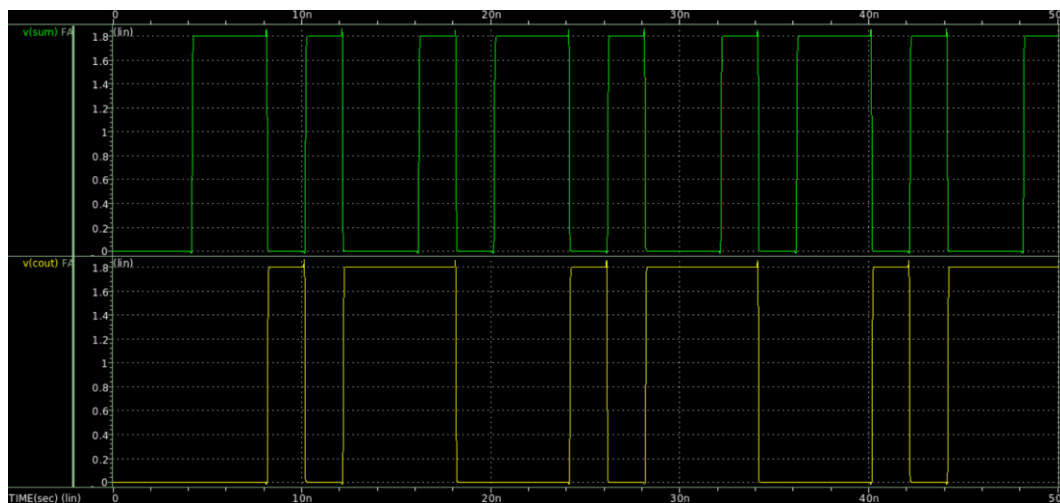
Input:



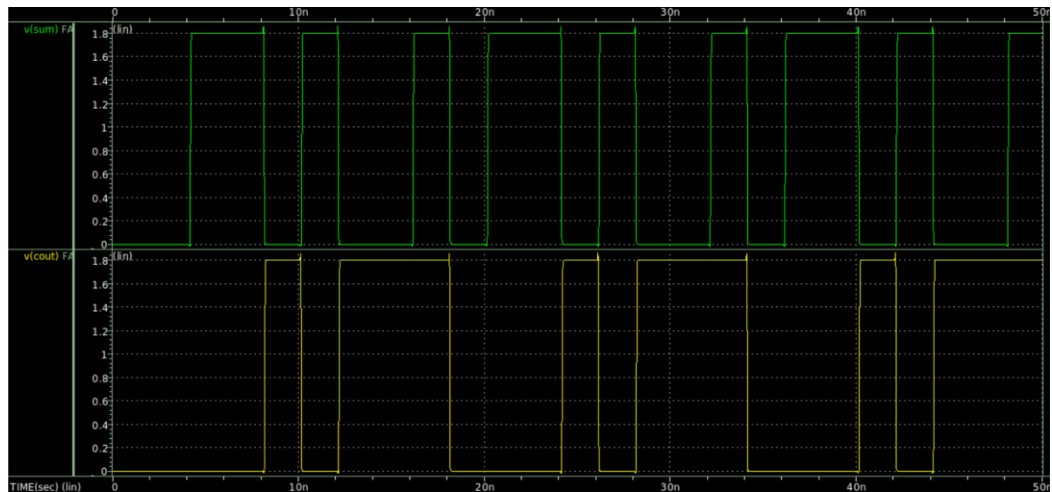
$W_p=0.54\mu m$



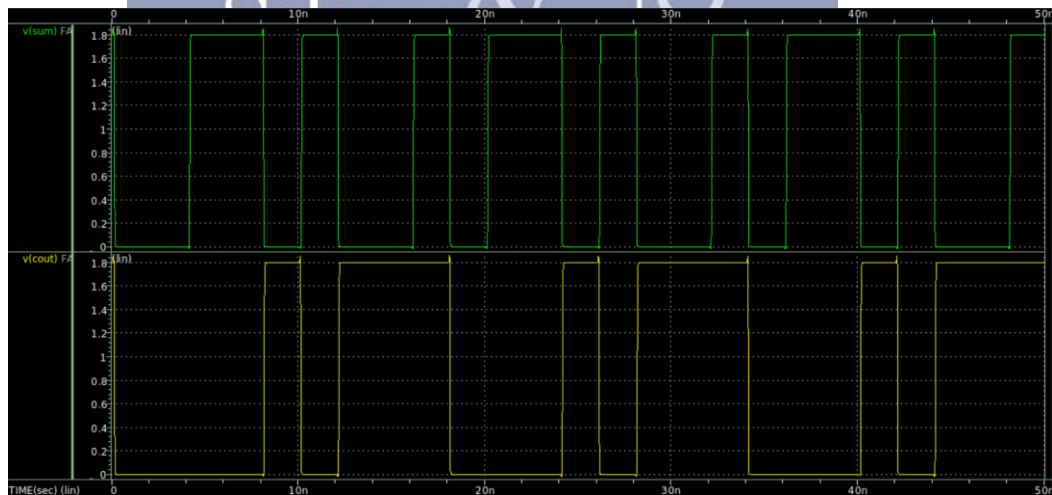
$W_p=0.72\mu m$



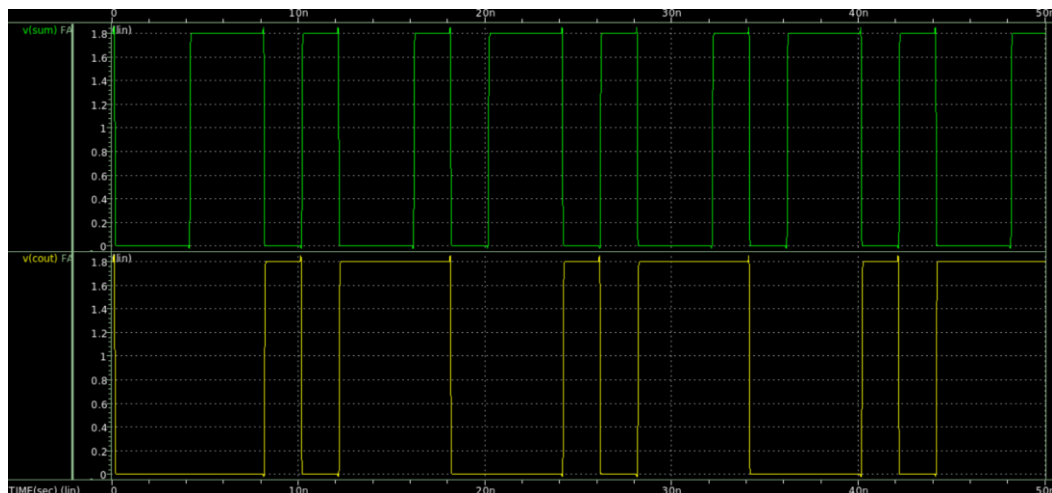
$W_p=1.2\mu m$



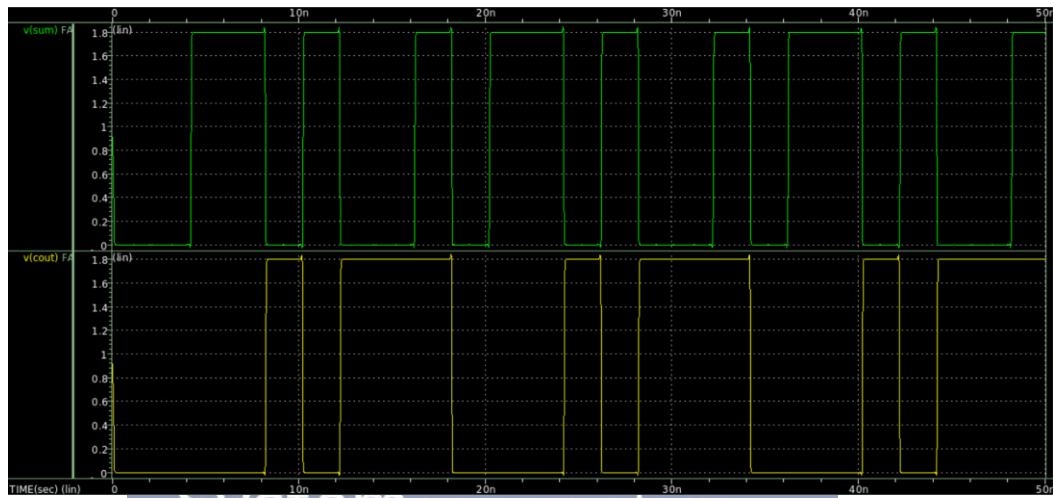
$W_p=1.8\mu m$



$W_p=3.6\mu m$



$W_p=7.2\mu m$



$W_p=12\mu m$

