

VIETNAM NATIONAL UNIVERSITY HO CHI MINH CITY
HO CHI MINH CITY UNIVERSITY OF TECHNOLOGY
FACULTY OF ELECTRICAL AND ELECTRONICS ENGINEERING



COURSE: INTRODUCTION TO IC DESIGN (EE3201)

LABORATORY REPORT

LAB 2: Synthesis and Standard Cell

Class: L02 – Semester: 251 – Group: 19
Supervising Instructor: M.Sc. Nguyen Trung Hieu

NO.	STUDENT'S FULL NAME	ID	INDIVIDUAL TASKS	STATUS
1	Huỳnh Trung Kiên	2311729	EXPERIMENT 3	100%
2	Nguyễn Đình Trọng Khôi	2311682	EXPERIMENT 1: NOT, OR, NAND, NOR EXPERIMENT 4	100%
3	Nguyễn Trọng Lộc	2311959	EXPERIMENT 1: MUX, FA EXPERIMENT 2	50%

Ho Chi Minh City, November 22, 2025

MỤC LỤC

MỤC TIÊU	4
CHUẨN BỊ CHO LAB 2	4
I. THÍ NGHIỆM 1	5
1.1 Cell NOT	6
1.1.1 Diện tích (area) và công suất (power).....	6
1.1.2 Độ trễ (delay)	6
1.2 Cell OR2	10
1.2.1 Diện tích (area) và công suất (power).....	10
1.2.2 Độ trễ (delay)	10
1.3 Cell NAND2	14
1.3.1 Diện tích (area) và công suất (power).....	14
1.3.2 Độ trễ (delay)	14
1.4 Cell NOR2	18
1.4.1 Diện tích (area) và công suất (power).....	18
1.4.2 Độ trễ (delay)	18
1.5 Cell MUX	22
1.5.1 Diện tích (area) và công suất (power).....	22
1.5.2 Độ trễ (delay)	22
1.6 Cell FULL ADDER	32
1.6.1 Diện tích (area) và công suất (power).....	32
1.6.2 Độ trễ (delay)	32
II. THÍ NGHIỆM 2	41
2.1 Sơ đồ mạch (schematic) và file netlist	41
2.1.1 Sơ đồ mạch	41
2.1.2. File Netlist.....	41
2.1.3 SDF FILE.....	41
2.2 Dạng sóng (waveform)	47
2.2.1 Trường hợp 1: T=20	47
2.2.2 Trường hợp 1: T=100	48

2.2.3 Trường hợp 1: T=200	49
2.2.4 Mô phỏng không có delay	50
III. THÍ NGHIỆM 3	52
3.1 HDL modules:.....	52
3.2 Measurement results:.....	56
3.3 Verification result (waveform):	57
IV. THÍ NGHIỆM 4	74
4.1 Sơ đồ mạch (schematic) và file netlist	75
4.1.1 Quy trình tổng hợp.....	75
4.1.2 Netlist sau tổng hợp và mô tả sơ đồ mạch.....	76
4.2 Dạng sóng (waveform)	78
4.2.1 File trễ thời gian (SDF) và các thông số chính	78
4.2.2 Trường hợp khi không có trễ (mô phỏng RTL – kết quả chuẩn)	84
4.2.3 Trường hợp 1: T=50 (mô phỏng gate-level có SDF).....	86
4.2.4 Trường hợp 1: T=100 (mô phỏng gate-level có SDF).....	90
4.2.5 Trường hợp 1: T=200 (mô phỏng gate-level có SDF).....	94
V. KẾT LUẬN.....	100

MỤC TIÊU

- Mục tiêu của bài lab này là để sinh viên thực hiện đầy đủ quy trình thiết kế số từ thiết kế RTL → tổng hợp (synthesis) → mô phỏng mức cổng (gate-level simulation), sử dụng Genus và Xcelium để tiến hành tổng hợp và kiểm chứng một mạch số.
- Sử dụng Genus, thiết kế RTL viết bằng HDL được tổng hợp thành netlist mức cổng dựa trên thư viện standard cell, đồng thời sinh ra thông tin timing tương ứng dưới dạng file Standard Delay Format (SDF) và các tham số khác như công suất (power), diện tích (area) và số lượng cell sử dụng (cell usage).
- Sử dụng Xcelium, sinh viên thực hiện mô phỏng mức cổng của netlist đã được tổng hợp kèm chú thích SDF (SDF annotation), cho phép kiểm chứng cả tính đúng đắn về chức năng (functional correctness) và hành vi thời gian (timing behavior) của mạch.

CHUẨN BỊ CHO LAB 2

- Hiểu khái niệm và luồng (flow) của quá trình tổng hợp trong thiết kế mạch số.
- Đọc hướng dẫn truy cập server và tài liệu hướng dẫn sử dụng các công cụ Xcelium, SimVision và Genus.
- Hiểu cách viết và áp dụng các script TCL (Tool Command Language) để tự động hóa quá trình tổng hợp trong Genus.

I. THÍ NGHIỆM 1

Mục tiêu: Khảo sát các đặc tính của các standard cell dưới những process corner cụ thể trong công nghệ 45nm.

Mô tả: Mỗi thư viện standard cell được đặc tả tại một corner nhất định, biểu diễn một tập các điều kiện môi trường – Process, Voltage và Temperature (PVT) – mà tại đó các cell được mô phỏng và đo đạc. Dựa vào tên file thư viện, ta có thể suy ra các điều kiện corner tương ứng.

Ví dụ: corner của thư viện fast_vdd1v0_basicCells_hvt.lib là:

- Process = Fast
- Voltage = 1.0 V
- Nhiệt độ $\approx 25^{\circ}\text{C}$
- Loại transistor: High-Vt (HVT).

Trong thí nghiệm này, chúng ta sẽ khảo sát các standard cell (NOT, OR2, NAND2, NOR2, MUX, và FULL ADDER) dưới 8 process corner khác nhau trong công nghệ 45nm. Các điều kiện cho tám corner này được cung cấp trong Bảng 1.

Bảng 1. Điều kiện vận hành của 8 corner được phân tích

Tên thư viện	Corner (Quá trình)	VDD(V)	Nhiệt độ ($^{\circ}\text{C}$)	Loại ngưỡng (Vt)
fast_vdd1v0_basicCells_hvt.lib	Nhanh(FF)	1.0	25 (typ.)	HVT (ngưỡng cao)
fast_vdd1v2_basicCells_hvt.lib	Nhanh (FF)	1.2	25 (typ.)	HVT (ngưỡng cao)
slow_vdd1v0_basicCells_hvt.lib	Chậm (SS)	1.0	125 (xấu nhất)	HVT (ngưỡng cao)
slow_vdd1v2_basicCells_hvt.lib	Chậm (SS)	1.2	125 (xấu nhất)	HVT (ngưỡng cao)
fast_vdd1v0_basicCells_lvt.lib	Nhanh (FF)	1.0	25 (typ.)	LVT (ngưỡng thấp)
fast_vdd1v2_basicCells_lvt.lib	Nhanh (FF)	1.2	25 (typ.)	LVT (ngưỡng thấp)
slow_vdd1v0_basicCells_lvt.lib	Chậm (SS)	1.0	125 (xấu nhất)	LVT (ngưỡng thấp)
slow_vdd1v2_basicCells_lvt.lib	Chậm (SS)	1.2	125 (xấu nhất)	LVT (ngưỡng thấp)

1.1 Cell NOT

1.1.1 Diện tích (area) và công suất (power)

Tham số diện tích (area) và diện tích (power) của cell NOT đo được trên 8 corner được thể hiện trong bảng dưới đây:

Library (.lib)	Area (μm^2)	Total Power (nW)
fast_vdd1v0_basicCells_hvt.lib	0.684	5.92740
fast_vdd1v0_basicCells_lvt.lib	0.684	7.80289
fast_vdd1v2_basicCells_hvt.lib	0.684	8.61296
fast_vdd1v2_basicCells_lvt.lib	0.684	10.7250
slow_vdd1v0_basicCells_hvt.lib	0.684	3.99470
slow_vdd1v0_basicCells_lvt.lib	0.684	4.28894
slow_vdd1v2_basicCells_hvt.lib	0.684	5.82108
slow_vdd1v2_basicCells_lvt.lib	0.684	6.20997

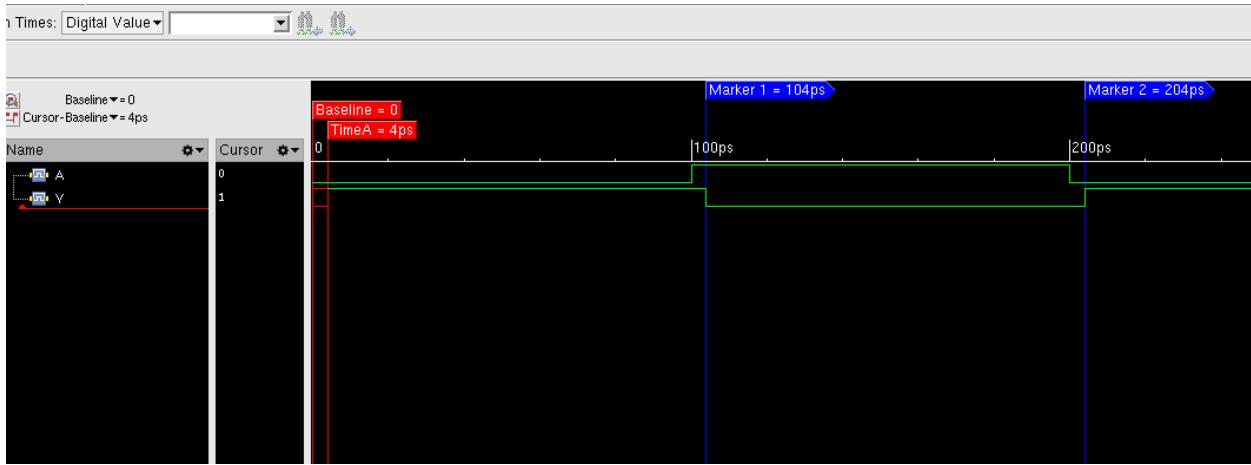
1.1.2 Độ trễ (delay)

Độ trễ lan truyền của cell NOT được đo bằng cách áp mẫu kích vào ngõ A, kết hợp với xem xét thời điểm chuyển mức tại ngõ ra Y. Các giá trị delay tương ứng cho từng corner được thể hiện trong bảng bên dưới.

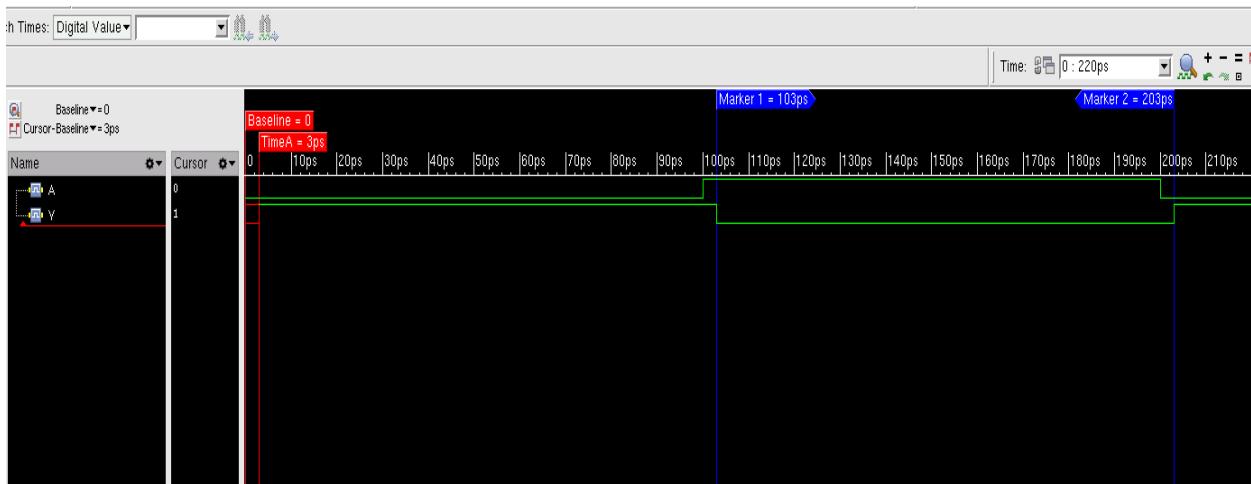
Corner	A -> Y rise (ns)	A -> Y fall (ns)
fast_vdd1v0_basicCells_hvt.lib	0.004	0.004
fast_vdd1v0_basicCells_lvt.lib	0.003	0.003
fast_vdd1v2_basicCells_hvt.lib	0.003	0.003
fast_vdd1v2_basicCells_lvt.lib	0.003	0.003
slow_vdd1v0_basicCells_hvt.lib	0.009	0.015
slow_vdd1v0_basicCells_lvt.lib	0.006	0.007
slow_vdd1v2_basicCells_hvt.lib	0.006	0.008
slow_vdd1v2_basicCells_lvt.lib	0.004	0.005

Các dạng waveform tương ứng với các thư viện được thể hiện qua các hình ảnh dưới đây:

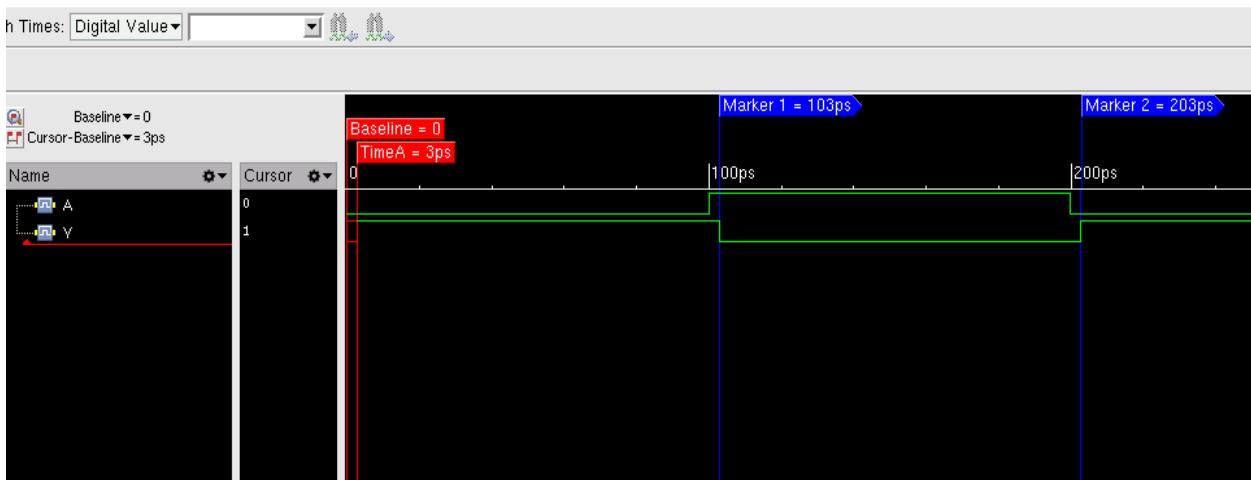
a) fast_vdd1v0_basicCells_hvt.lib



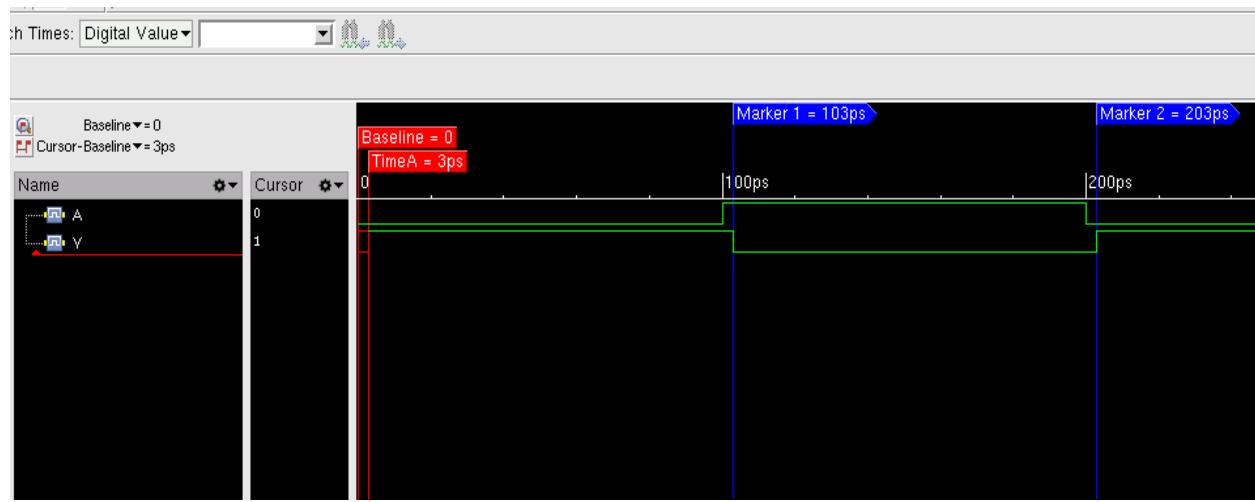
b) fast_vdd1v0_basicCells_lvt.lib



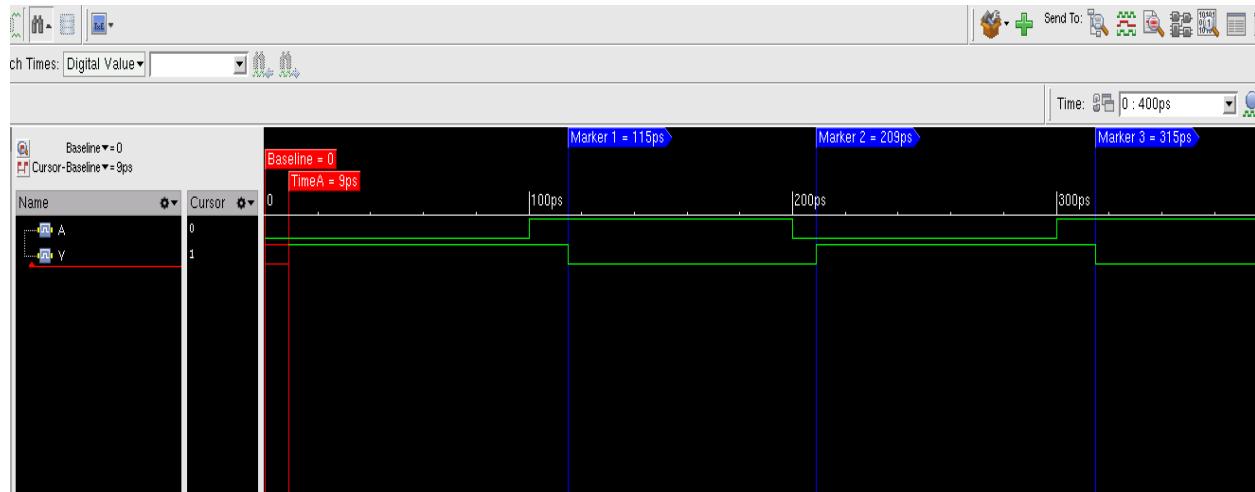
c) fast_vdd1v2_basicCells_hvt.lib



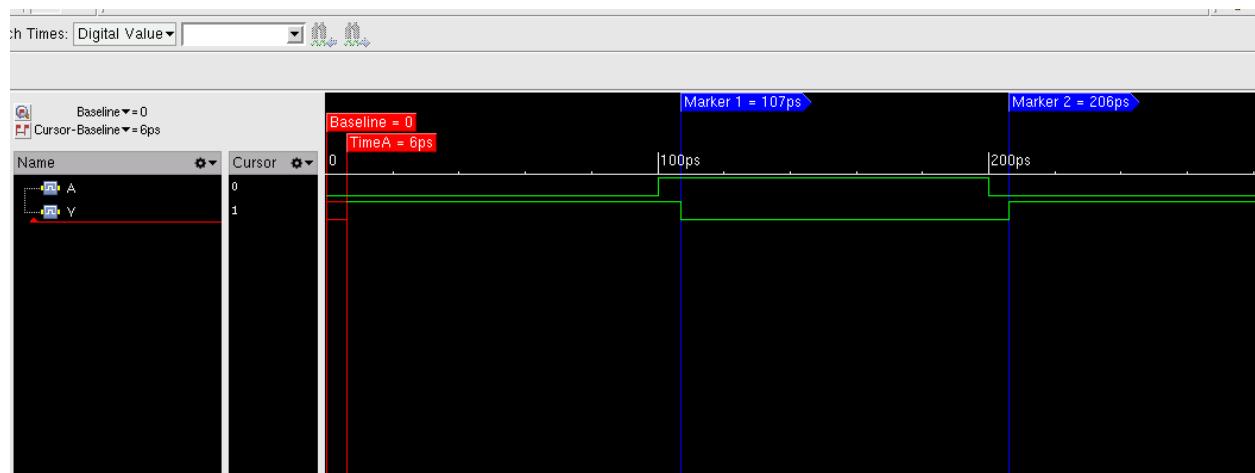
d) fast_vdd1v2_basicCells_lvt.lib



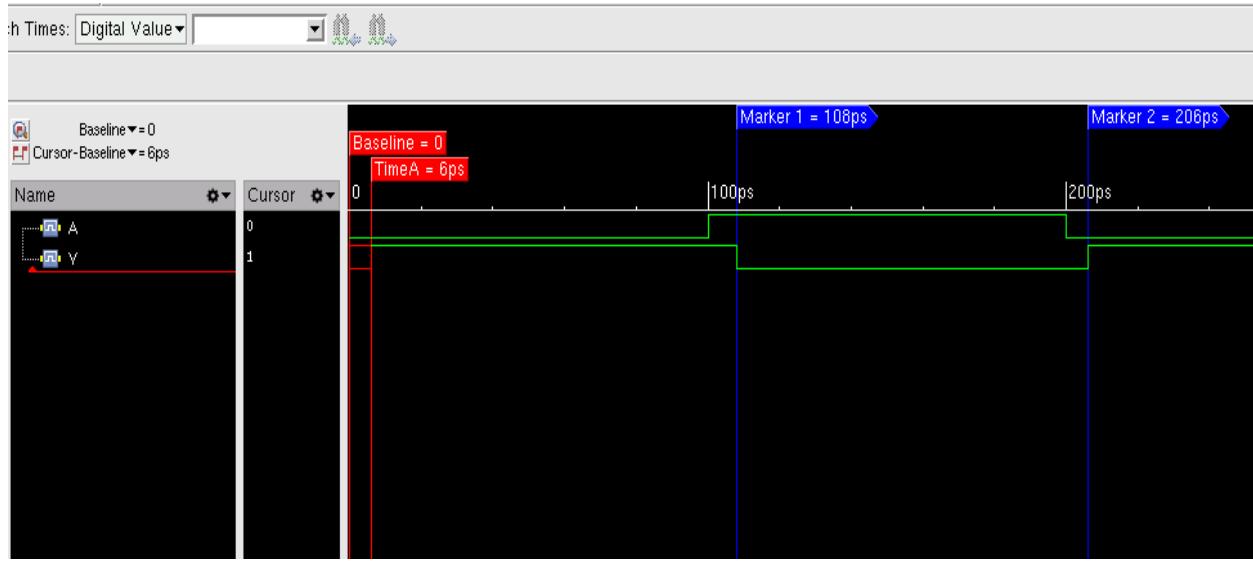
e) slow_vdd1v0_basicCells_hvt.lib



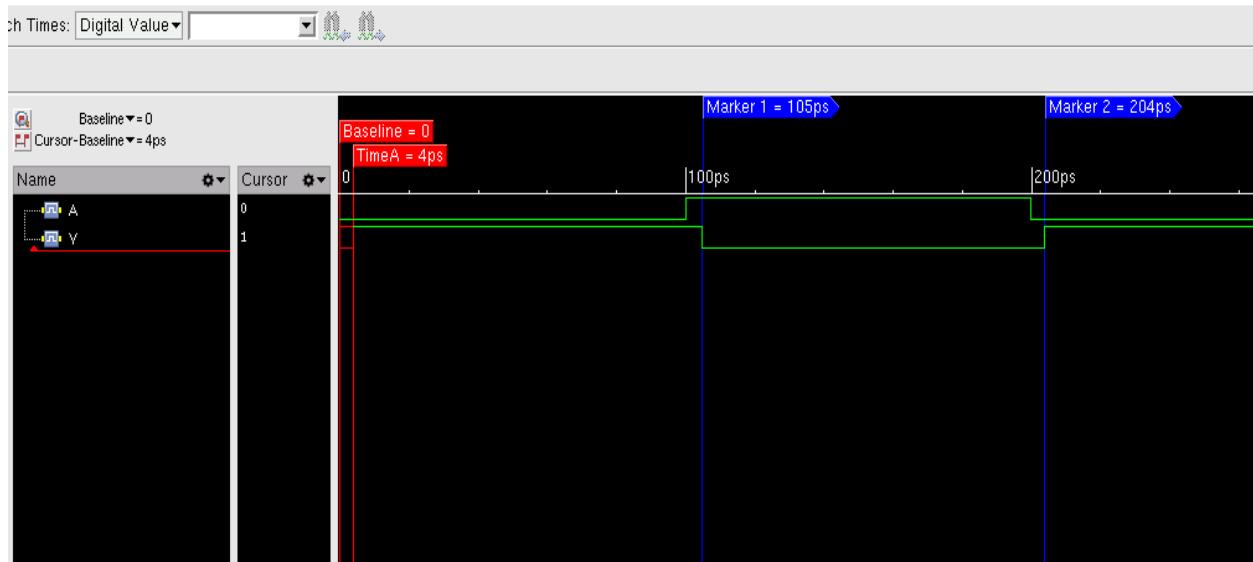
f) slow_vdd1v0_basicCells_lvt.lib



g) slow_vdd1v2_basicCells_hvt.lib



h) slow_vdd1v2_basicCells_lvt.lib



Kết luận: Tất cả giá trị trễ đo từ waveform đều trùng khớp với thông số trong file SDF của từng corner \Rightarrow xác nhận mô phỏng GLS và gán SDF là **chính xác**.

1.2 Cell OR2

1.2.1 Diện tích (area) và công suất (power)

Tham số diện tích (area) và diện tích (power) của cell OR2 đo được trên 8 corner được thể hiện trong bảng dưới đây:

Library (.lib)	Area (μm^2)	Total Power (nW)
fast_vdd1v0_basicCells_hvt.lib	1.368	17.3733
fast_vdd1v0_basicCells_lvt.lib	1.368	22.3841
fast_vdd1v2_basicCells_hvt.lib	1.368	25.3194
fast_vdd1v2_basicCells_lvt.lib	1.368	33.4466
slow_vdd1v0_basicCells_hvt.lib	1.368	11.4746
slow_vdd1v0_basicCells_lvt.lib	1.368	12.64
slow_vdd1v2_basicCells_hvt.lib	1.368	16.8838
slow_vdd1v2_basicCells_lvt.lib	1.368	18.4242

1.2.2 Độ trễ (delay)

Độ trễ lan truyền của cell OR2 được đo bằng cách áp mẫu kích vào ngõ A và B, kết hợp với xem xét thời điểm chuyển mức tại ngõ ra Y. Các giá trị delay tương ứng cho từng corner được thể hiện trong bảng bên dưới.

Corner	A -> Y rise (ns)	A -> Y fall (ns)	B -> Y rise (ns)	B -> Y fall (ns)
fast_vdd1v0_basicCells_hvt.lib	0.017	0.029	0.015	0.026
fast_vdd1v0_basicCells_lvt.lib	0.011	0.017	0.009	0.015
fast_vdd1v2_basicCells_hvt.lib	0.013	0.021	0.011	0.019
fast_vdd1v2_basicCells_lvt.lib	0.009	0.014	0.008	0.012
slow_vdd1v0_basicCells_hvt.lib	0.077	0.099	0.065	0.088
slow_vdd1v0_basicCells_lvt.lib	0.033	0.052	0.027	0.045
slow_vdd1v2_basicCells_hvt.lib	0.037	0.050	0.031	0.044

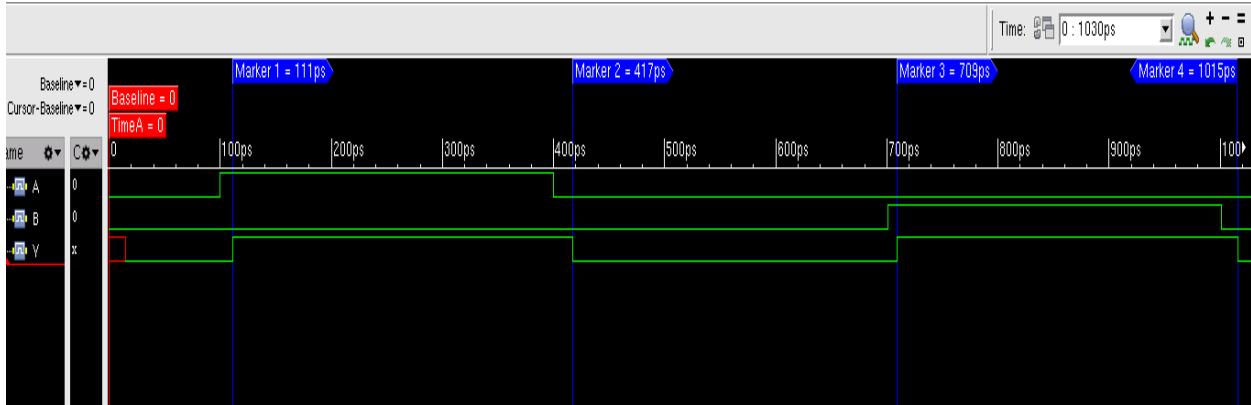
Corner	A -> Y rise (ns)	A -> Y fall (ns)	B -> Y rise (ns)	B -> Y fall (ns)
slow_vdd1v2_basicCells_lvt.lib	0.022	0.034	0.018	0.030

Các dạng waveform tương ứng với các thư viện được thể hiện qua các hình ảnh dưới đây:

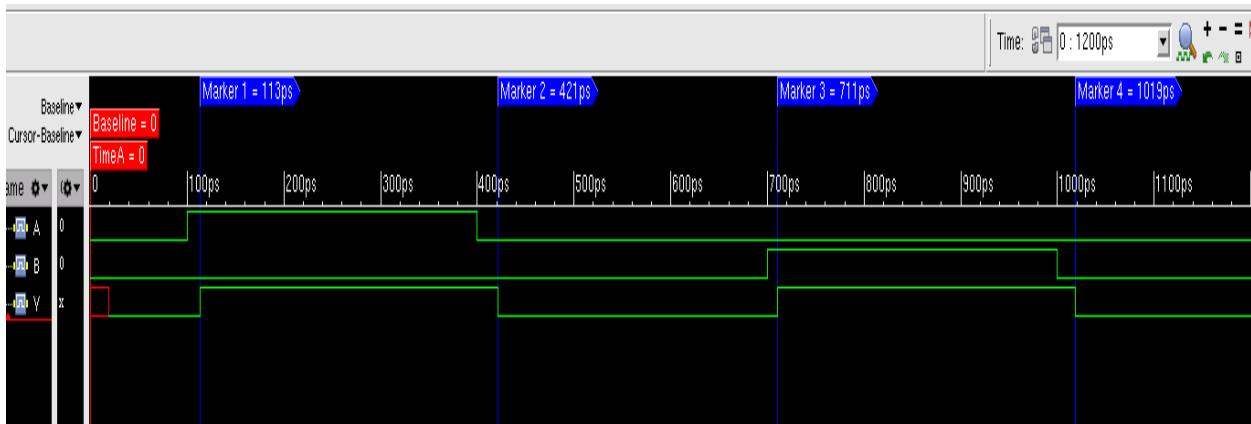
a) fast_vdd1v0_basicCells_hvt.lib



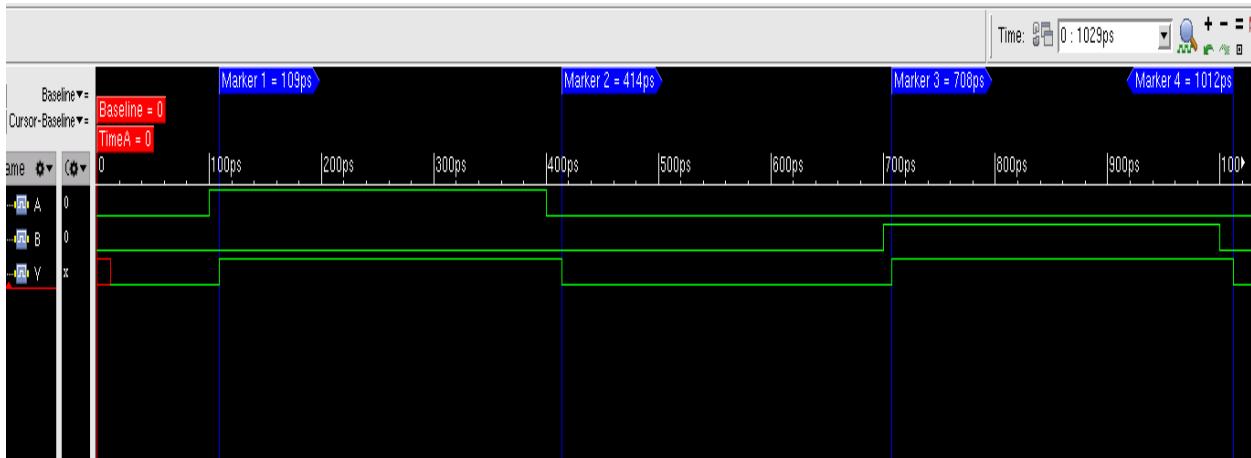
b) fast_vdd1v0_basicCells_lvt.lib



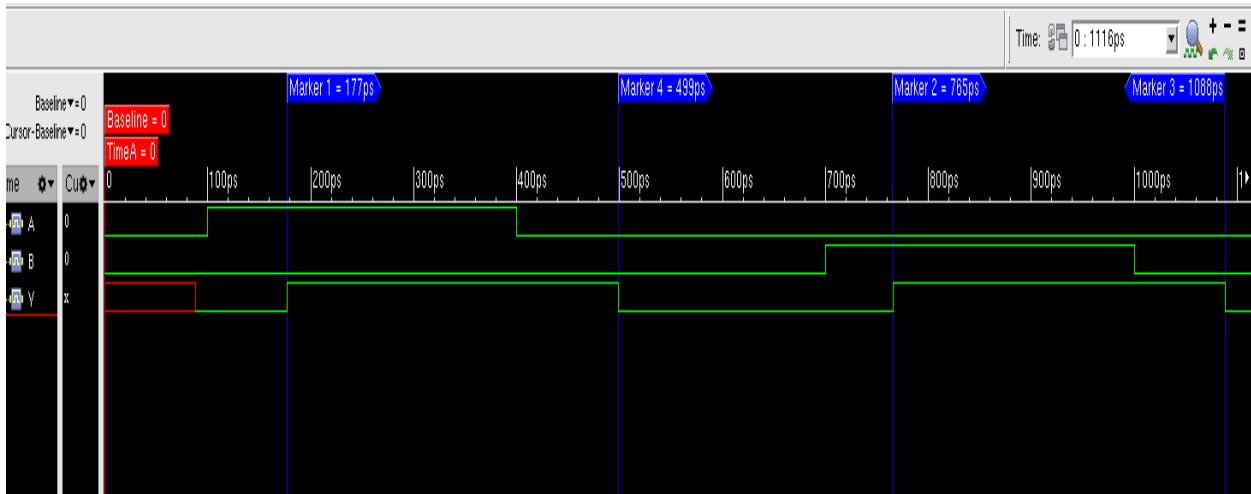
c) fast_vdd1v2_basicCells_hvt.lib



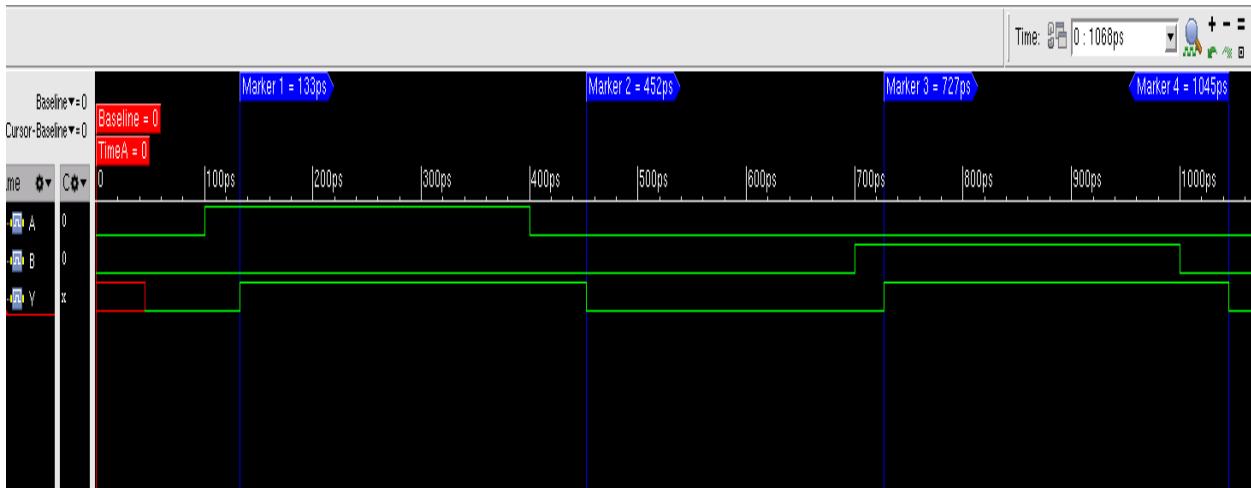
d) fast_vdd1v2_basicCells_lvt.lib



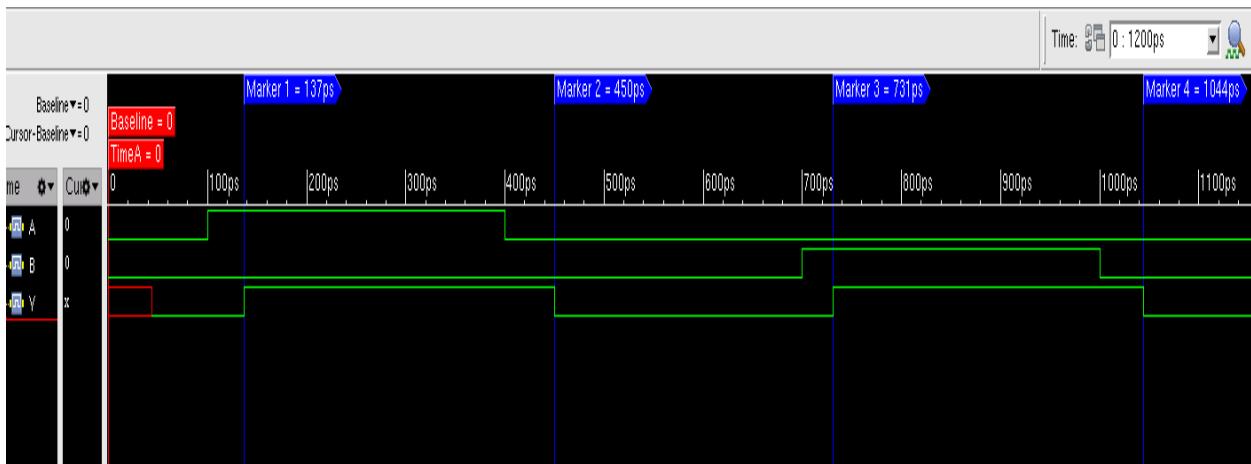
e) slow_vdd1v0_basicCells_hvt.lib



f) slow_vdd1v0_basicCells_lvt.lib



g) slow_vdd1v2_basicCells_hvt.lib



h) slow_vdd1v2_basicCells_lvt.lib



Kết luận: Tất cả giá trị trễ đo từ waveform đều trùng khớp với thông số trong file SDF của từng corner \Rightarrow xác nhận mô phỏng GLS và gán SDF là **chính xác**.

1.3 Cell NAND2

1.3.1 Diện tích (area) và công suất (power)

Tham số diện tích (area) và diện tích (power) của cell NAND2 đo được trên 8 corner được thể hiện trong bảng dưới đây:

Library (.lib)	Area (μm^2)	Total Power (nW)
fast_vdd1v0_basicCells_hvt.lib	1.026	10.5962
fast_vdd1v0_basicCells_lvt.lib	1.026	14.2490
fast_vdd1v2_basicCells_hvt.lib	1.026	15.4143
fast_vdd1v2_basicCells_lvt.lib	1.026	20.0986
slow_vdd1v0_basicCells_hvt.lib	1.026	7.20719
slow_vdd1v0_basicCells_lvt.lib	1.026	7.73734
slow_vdd1v2_basicCells_hvt.lib	1.026	10.4670
slow_vdd1v2_basicCells_lvt.lib	1.026	11.1834

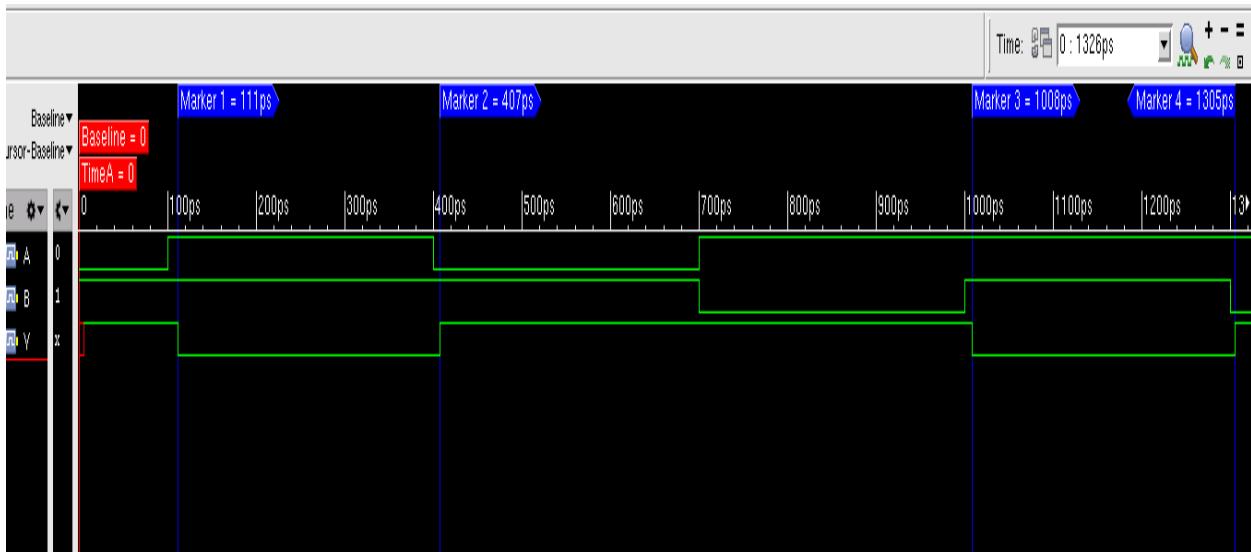
1.3.2 Độ trễ (delay)

Độ trễ lan truyền của cell NAND2 được đo bằng cách áp mẫu kích vào ngõ A và B, kết hợp với xem xét thời điểm chuyển mức tại ngõ ra Y. Các giá trị delay tương ứng cho từng corner được thể hiện trong bảng bên dưới.

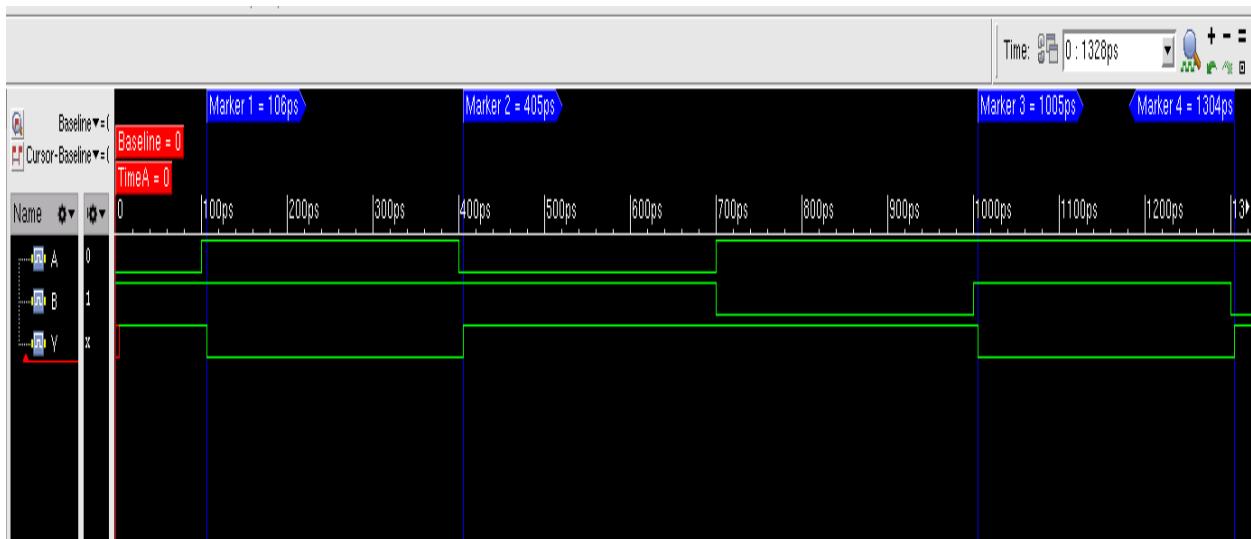
Corner	A -> Y rise (ns)	A -> Y fall (ns)	B -> Y rise (ns)	B -> Y fall (ns)
fast_vdd1v0_basicCells_hvt.lib	0.007	0.011	0.005	0.008
fast_vdd1v0_basicCells_lvt.lib	0.005	0.006	0.004	0.005
fast_vdd1v2_basicCells_hvt.lib	0.006	0.008	0.004	0.006
fast_vdd1v2_basicCells_lvt.lib	0.004	0.005	0.003	0.004
slow_vdd1v0_basicCells_hvt.lib	0.017	0.066	0.012	0.050
slow_vdd1v0_basicCells_lvt.lib	0.010	0.024	0.007	0.018
slow_vdd1v2_basicCells_hvt.lib	0.010	0.029	0.007	0.021

Corner	A -> Y rise (ns)	A -> Y fall (ns)	B -> Y rise (ns)	B -> Y fall (ns)
slow_vdd1v2_basicCells_lvt.lib	0.008	0.015	0.006	0.012

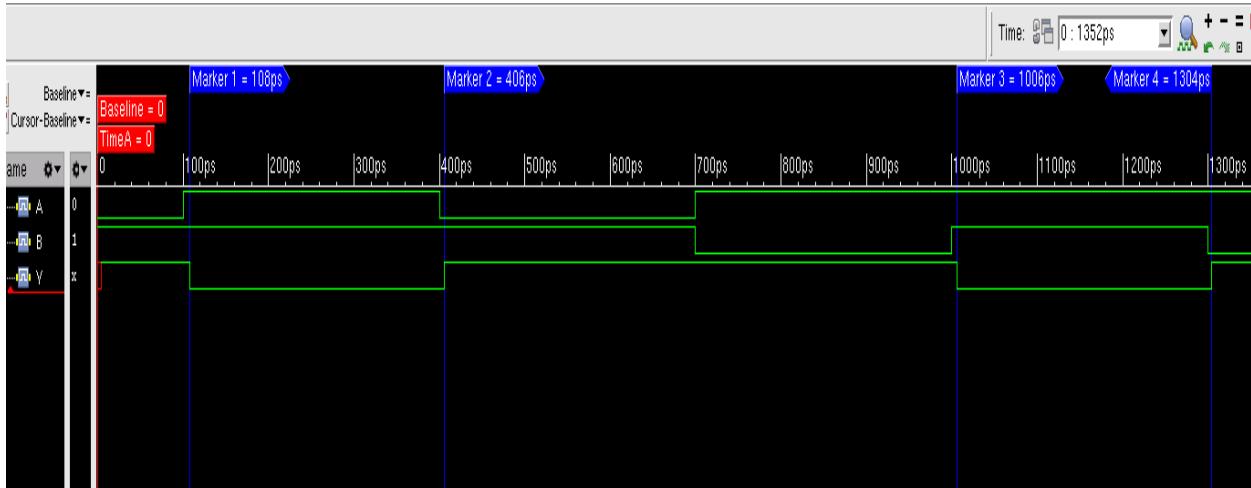
Các dạng waveform tương ứng với các thư viện được thể hiện qua các hình ảnh dưới đây:
a) *fast_vdd1v0_basicCells_hvt.lib*



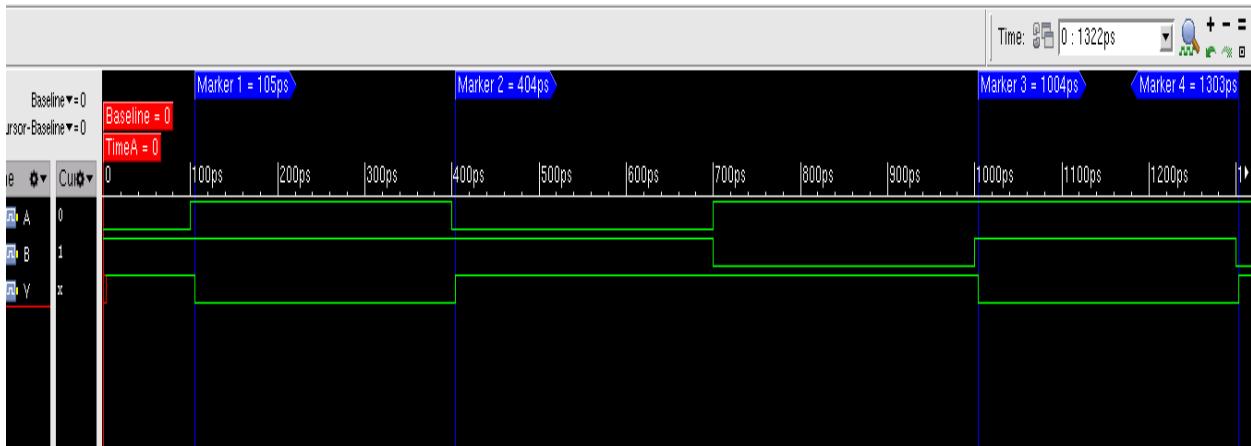
b) *fast_vdd1v0_basicCells_lvt.lib*



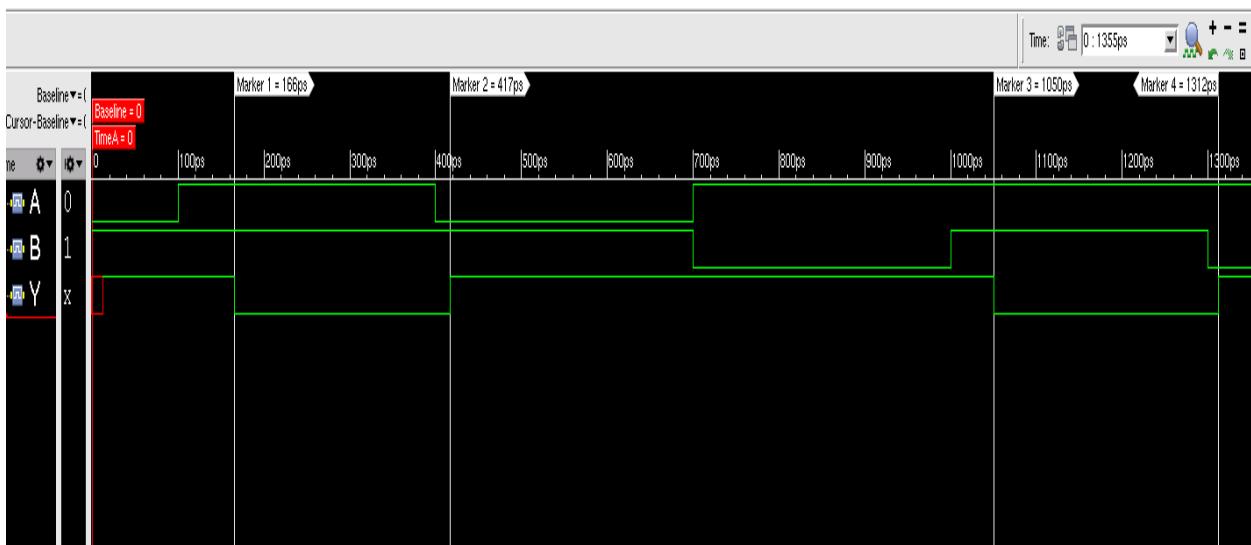
c) fast_vdd1v2_basicCells_hvt.lib



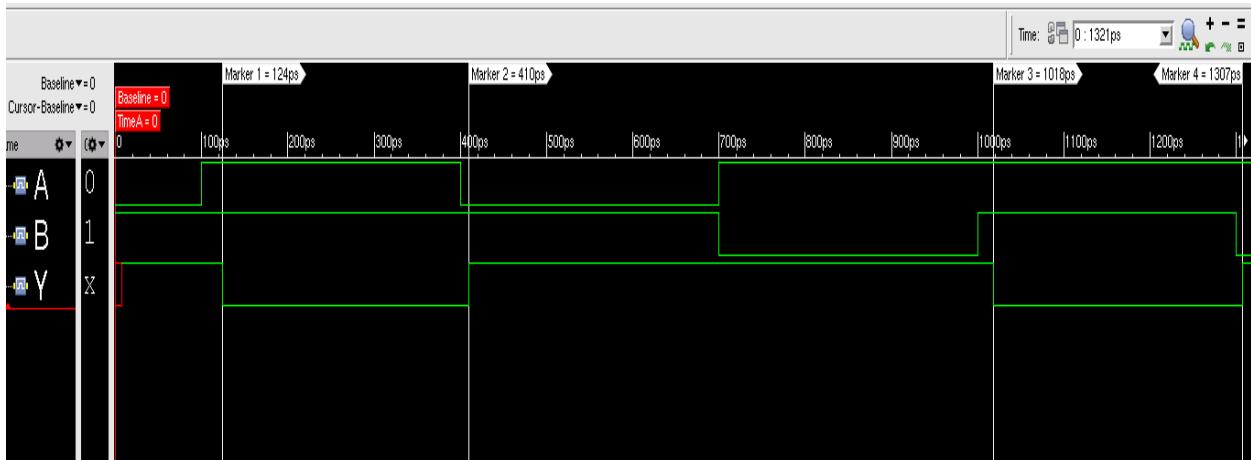
d) fast_vdd1v2_basicCells_lvt.lib



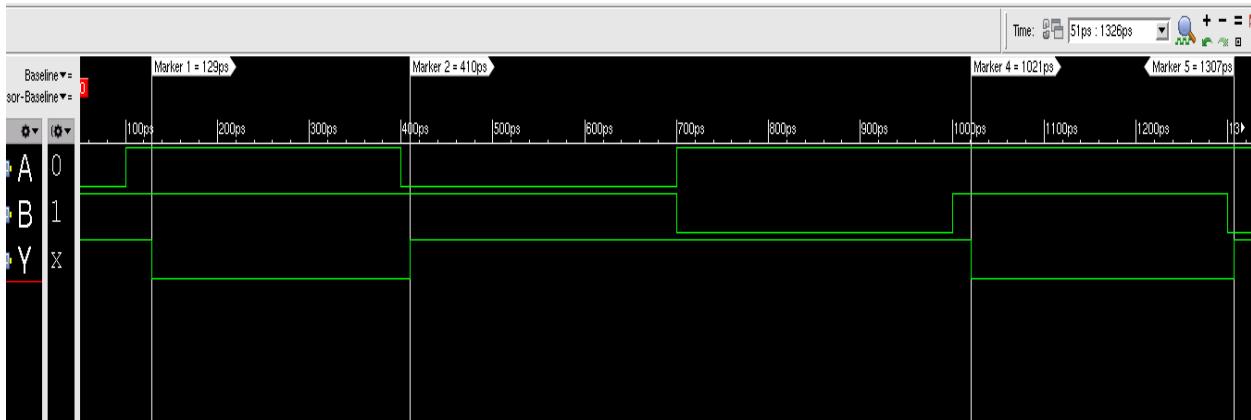
e) slow_vdd1v0_basicCells_hvt.lib



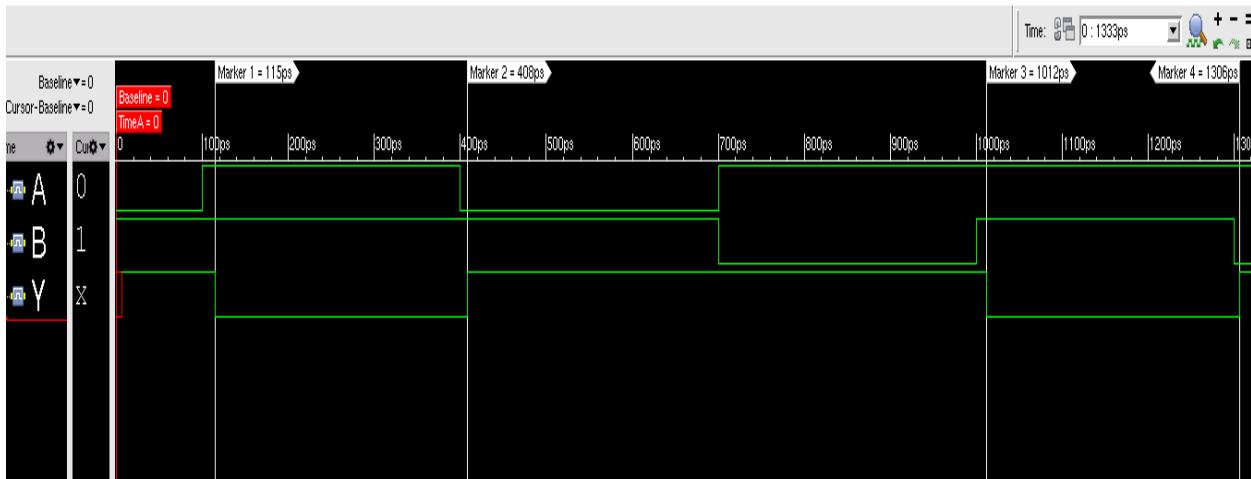
f) slow_vdd1v0_basicCells_lvt.lib



g) slow_vdd1v2_basicCells_hvt.lib



h) slow_vdd1v2_basicCells_lvt.lib



Kết luận: Tất cả giá trị trễ đo từ waveform đều trùng khớp với thông số trong file SDF của từng corner \Rightarrow xác nhận mô phỏng GLS và gán SDF là **chính xác**.

1.4 Cell NOR2

1.4.1 Diện tích (area) và công suất (power)

Tham số diện tích (area) và diện tích (power) của cell NOR2 đo được trên 8 corner được thể hiện trong bảng dưới đây:

Library (.lib)	Area (μm^2)	Total Power (nW)
fast_vdd1v0_basicCells_hvt.lib	1.026	10.7544
fast_vdd1v0_basicCells_lvt.lib	1.026	14.4019
fast_vdd1v2_basicCells_hvt.lib	1.026	15.6633
fast_vdd1v2_basicCells_lvt.lib	1.026	20.3360
slow_vdd1v0_basicCells_hvt.lib	1.026	7.30888
slow_vdd1v0_basicCells_lvt.lib	1.026	7.82275
slow_vdd1v2_basicCells_hvt.lib	1.026	10.6519
slow_vdd1v2_basicCells_lvt.lib	1.026	11.3492

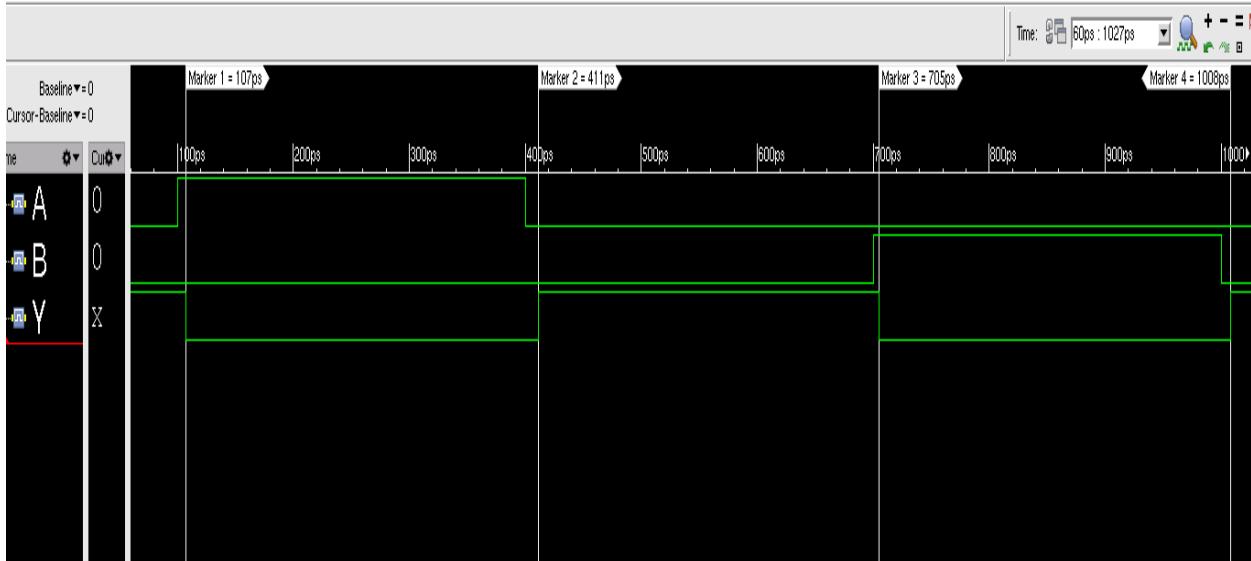
1.4.2 Độ trễ (delay)

Độ trễ lan truyền của cell NOR2 được đo bằng cách áp mẫu kích vào ngõ A và B, kết hợp với xem xét thời điểm chuyển mức tại ngõ ra Y. Các giá trị delay tương ứng cho từng corner được thể hiện trong bảng bên dưới.

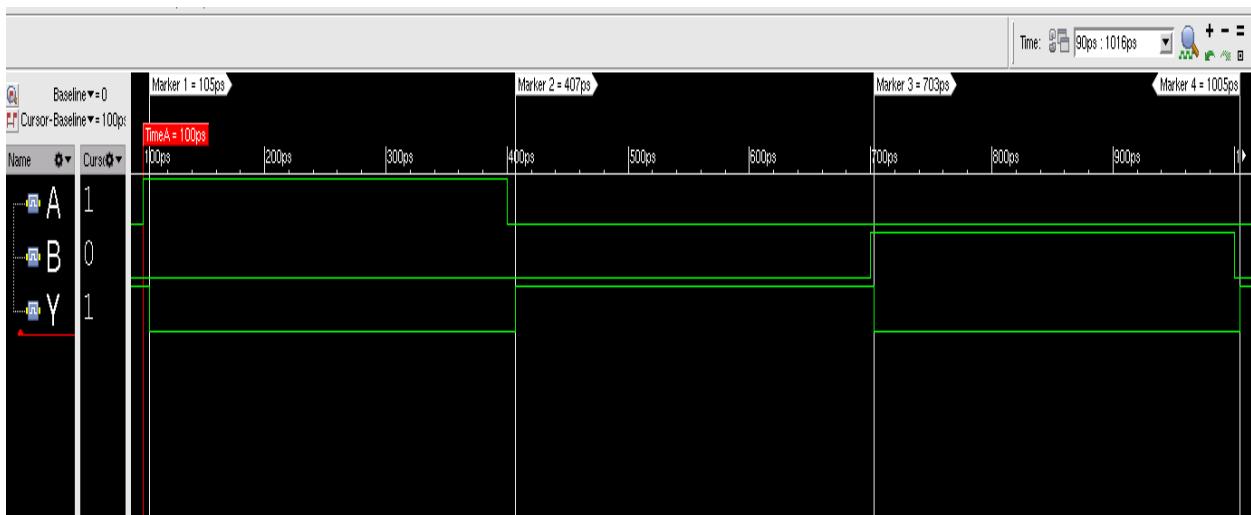
Corner	A -> Y rise (ns)	A -> Y fall (ns)	B -> Y rise (ns)	B -> Y fall (ns)
fast_vdd1v0_basicCells_hvt.lib	0.011	0.007	0.008	0.005
fast_vdd1v0_basicCells_lvt.lib	0.007	0.005	0.005	0.003
fast_vdd1v2_basicCells_hvt.lib	0.009	0.006	0.006	0.004
fast_vdd1v2_basicCells_lvt.lib	0.006	0.004	0.004	0.003
slow_vdd1v0_basicCells_hvt.lib	0.036	0.030	0.025	0.020
slow_vdd1v0_basicCells_lvt.lib	0.020	0.014	0.013	0.008
slow_vdd1v2_basicCells_hvt.lib	0.019	0.016	0.013	0.010

Corner	A -> Y rise (ns)	A -> Y fall (ns)	B -> Y rise (ns)	B -> Y fall (ns)
slow_vdd1v2_basicCells_lvt.lib	0.013	0.010	0.008	0.006

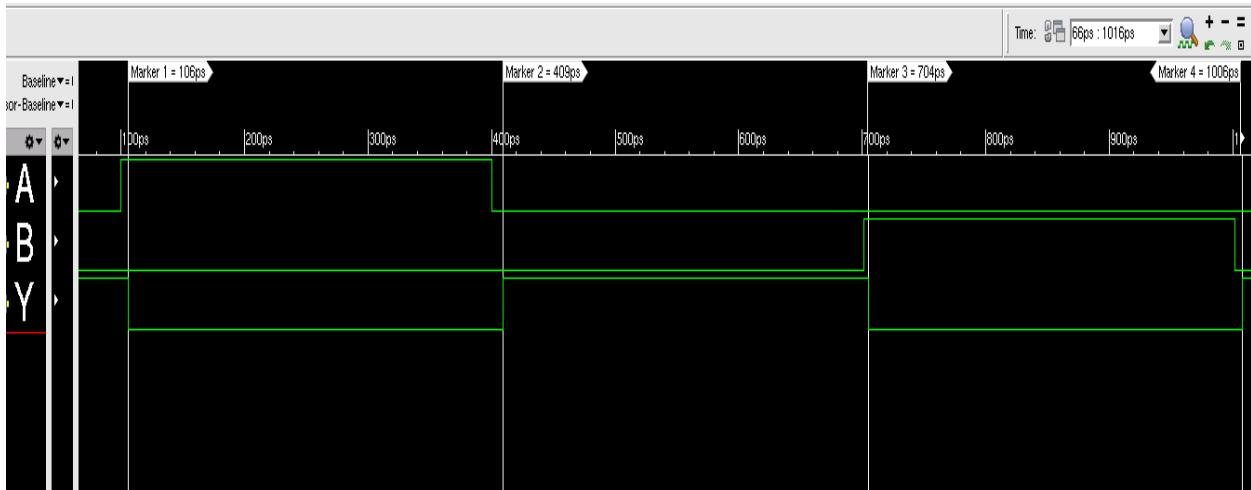
Các dạng waveform tương ứng với các thư viện được thể hiện qua các hình ảnh dưới đây:
 a) fast_vdd1v0_basicCells_hvt.lib



b) fast_vdd1v0_basicCells_lvt.lib



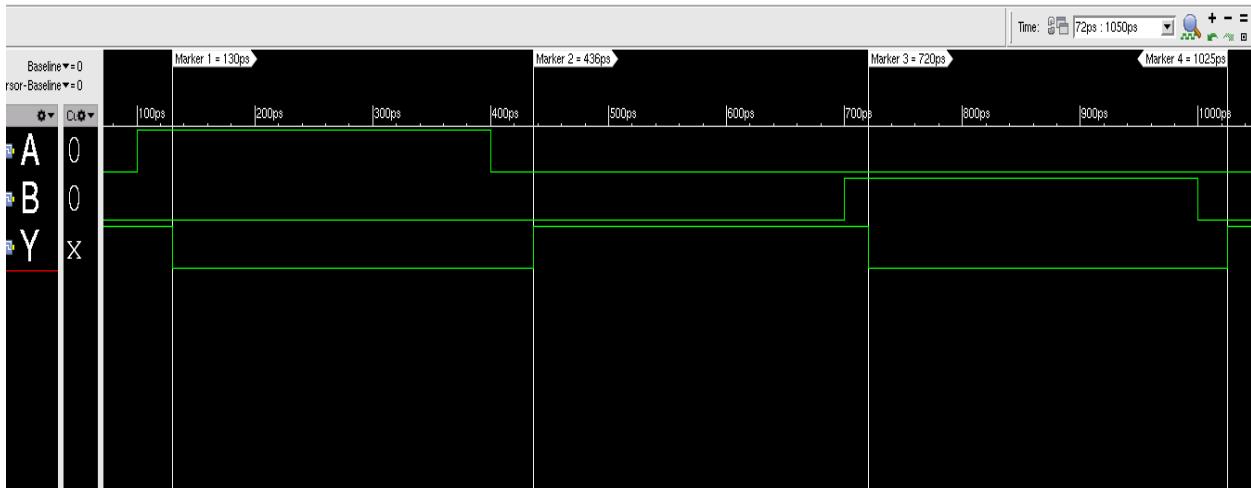
c) fast_vdd1v2_basicCells_hvt.lib



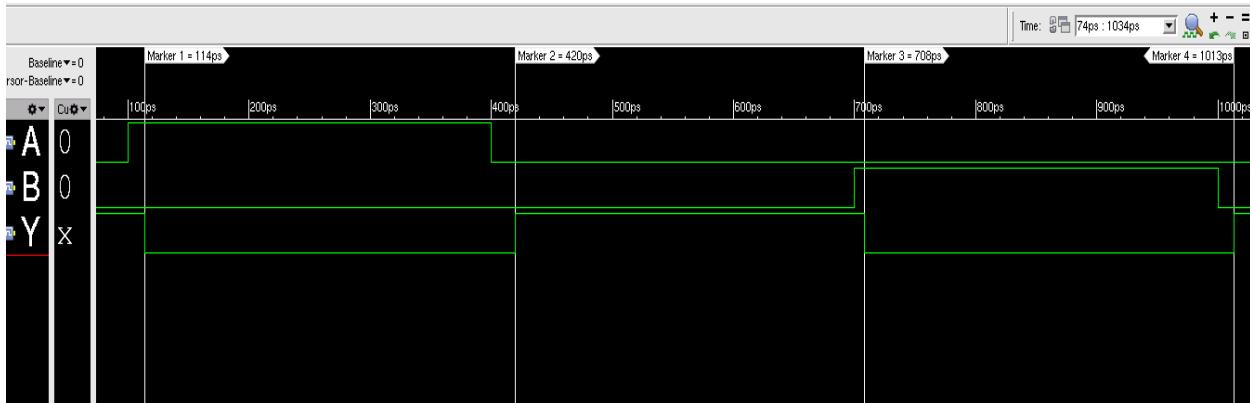
d) fast_vdd1v2_basicCells_lvt.lib



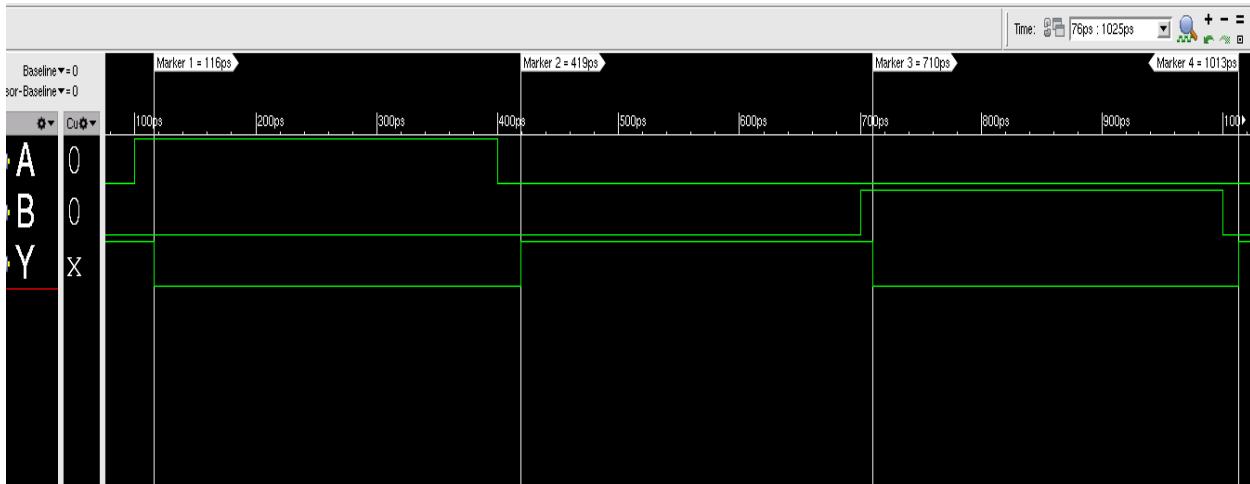
e) slow_vdd1v0_basicCells_hvt.lib



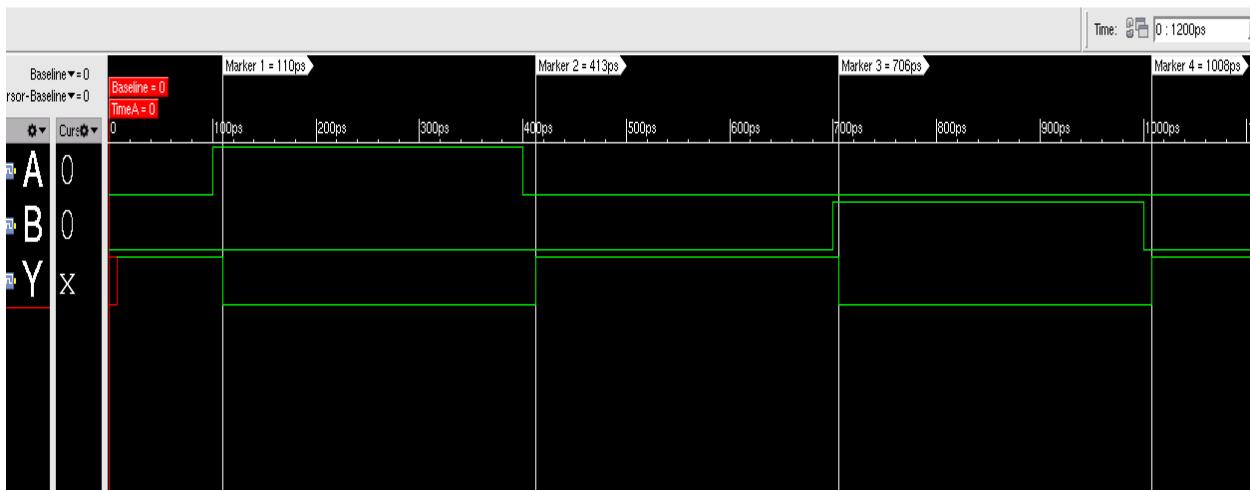
f) slow_vdd1v0_basicCells_lvt.lib



g) slow_vdd1v2_basicCells_hvt.lib



h) slow_vdd1v2_basicCells_lvt.lib



Kết luận: Tất cả giá trị trễ đo từ waveform đều trùng khớp với thông số trong file SDF của từng corner \Rightarrow xác nhận mô phỏng GLS và gán SDF là **chính xác**.

1.5 Cell MUX

1.5.1 Diện tích (area) và công suất (power)

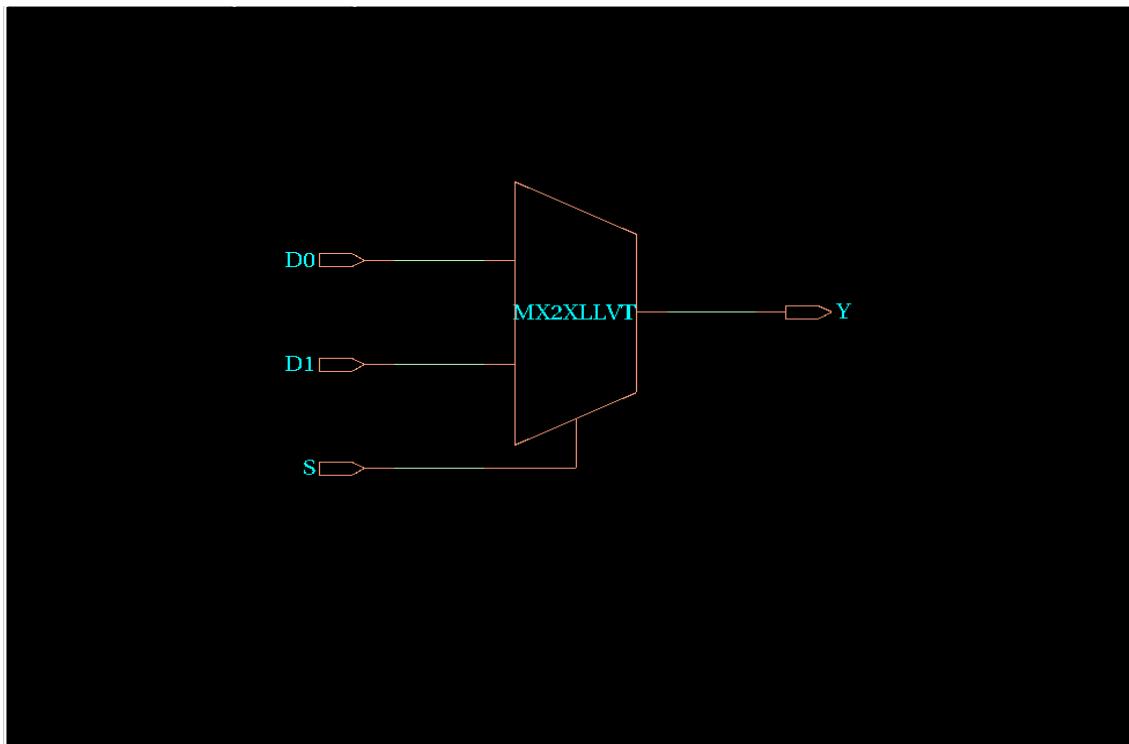
Corner	Library name	Area(μm^2)	Total Power (μW)
1	fast_vdd1v0_basicCells_hvt.lib	2.394	0.0366
2	fast_vdd1v0_basicCells_lvt.lib	2.394	0.0461
3	fast_vdd1v2_basicCells_hvt.lib	2.394	0.0547
4	fast_vdd1v2_basicCells_lvt.lib	2.394	0.0649
5	slow_vdd1v0_basicCells_hvt.lib	2.394	0.0239
6	slow_vdd1v0_basicCells_lvt.lib	2.394	0.0272
7	slow_vdd1v2_basicCells_hvt.lib	2.394	0.0362
8	slow_vdd1v2_basicCells_lvt.lib	2.394	0.0406

1.5.2 Độ trễ (delay)

STT	Corner	Đường đo	t_pd_rise(ns)	t_pd_fall(ns)
1	fast_vdd1v0_basicCells_hvt.lib	D0-Y	0.034	0.036
		D1-Y	0.034	0.037
		S-y	0.038	0.040
2	fast_vdd1v0_basicCells_lvt.lib	D0-Y	0.020	0.022
		D1-Y	0.019	0.022
		S-Y	0.022	0.023
3	fast_vdd1v2_basicCells_hvt.lib	D0-Y	0.024	0.027
		D1-Y	0.024	0.027
		S-Y	0.027	0.029
4	fast_vdd1v2_basicCells_lvt.lib	D0-Y	0.015	0.018
		D1-Y	0.015	0.018

		S-Y	0.017	0.019
5	slow_vdd1v0_basicCells_hvt.lib	D0-Y	0.203	0.126
		D1-Y	0.202	0.130
		S-Y	0.199	0.152
6	slow_vdd1v0_basicCells_lvt.lib	D0-Y	0.077	0.066
		D1-Y	0.076	0.068
		S-Y	0.080	0.073
7	slow_vdd1v2_basicCells_hvt.lib	D0-Y	0.088	0.089
		D1-Y	0.087	0.089
		S-Y	0.089	0.074
8	slow_vdd1v2_basicCells_lvt.lib	D0-Y	0.048	0.044
		D1-Y	0.048	0.045
		S-Y	0.051	0.048

Netlist sinh ra:

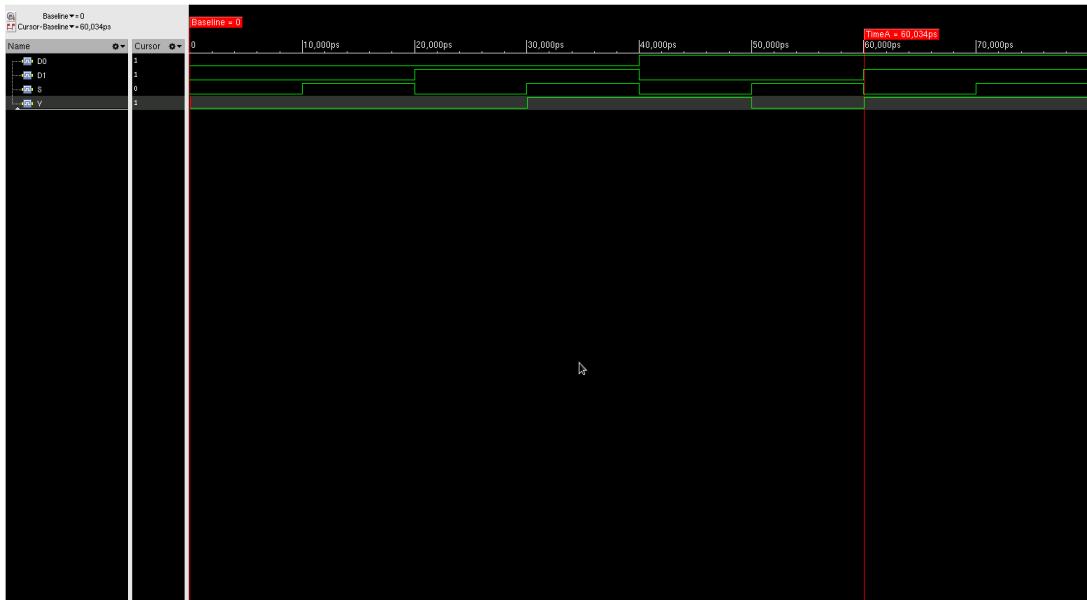


Wave form các trường hợp :

TH1: fast_vdd1v0_basicCells_hvt.lib:



=> ta nhìn thấy rõ ra Y bị delay 1 khoảng 36ps(t_{fall})



Nhìn vào waveform, ta thấy rõ ra Y bị delay 1 khoảng 34ps(t_{rise})

TH2: fast_vdd1v2_basicCells_hvt.lib:



- Ngõ ra Y bị delay 1 khoảng 27ps(t_rise)



- Ngõ ra Y bị delay 1 khoảng 24ps(t_rise)

TH3: slow_vdd1v0_basicCells_hvt.lib:



Quan sát Dựa vào hình dạng waveform, ta thấy ngõ ra Y bị delay 1 khoảng 199ps(t_{rise})

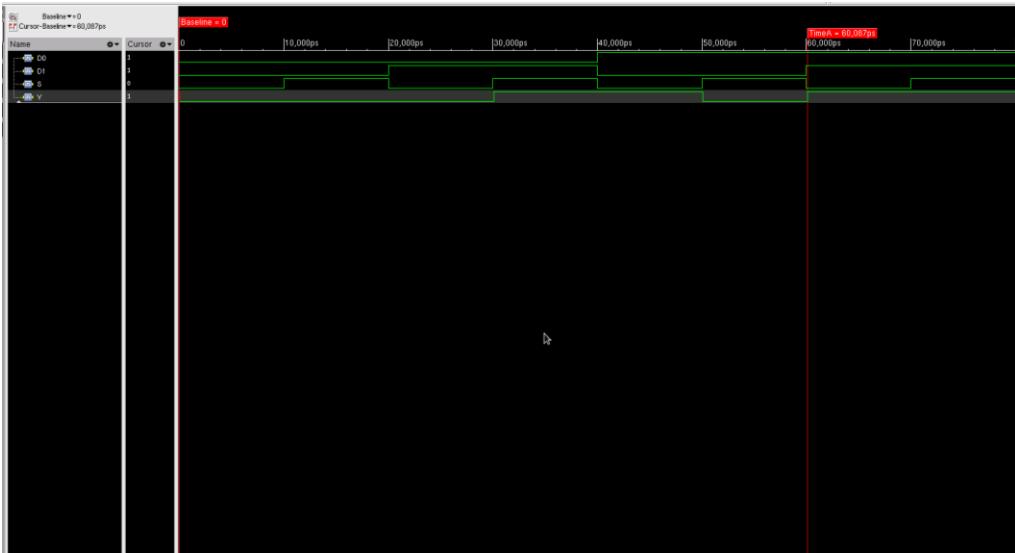


Dựa trên hình dạng waveform, ta thấy rằng ngõ ra Y bị delay 1 khoảng 199ps(t_{rise})

TH4: slow_vdd1v2_basicCells_hvt.lib:



- ngõ ra Y bị delay 1 khoảng 89ps(t_{rise})



- ta nhận thấy rằng ngõ ra Y bị delay 1 khoảng 87ps(t_{rise})

-TH5: fast_vdd1v0_basicCells_lvt.lib:

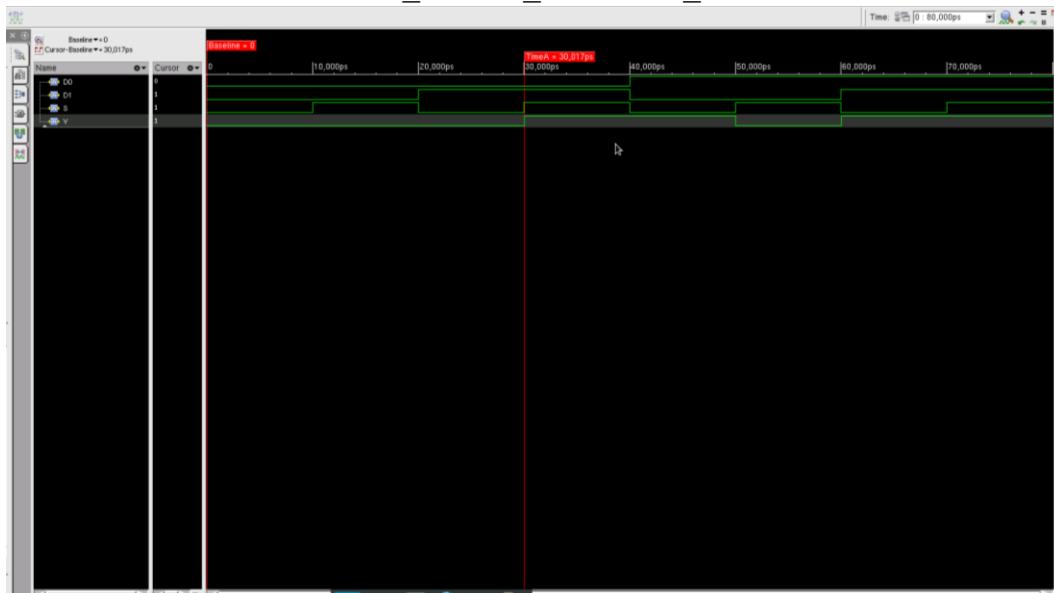


- Quan sát hình dạng waveform, ta nhận thấy ngõ ra Y bị delay 1 khoảng 22ps(t_{rise})

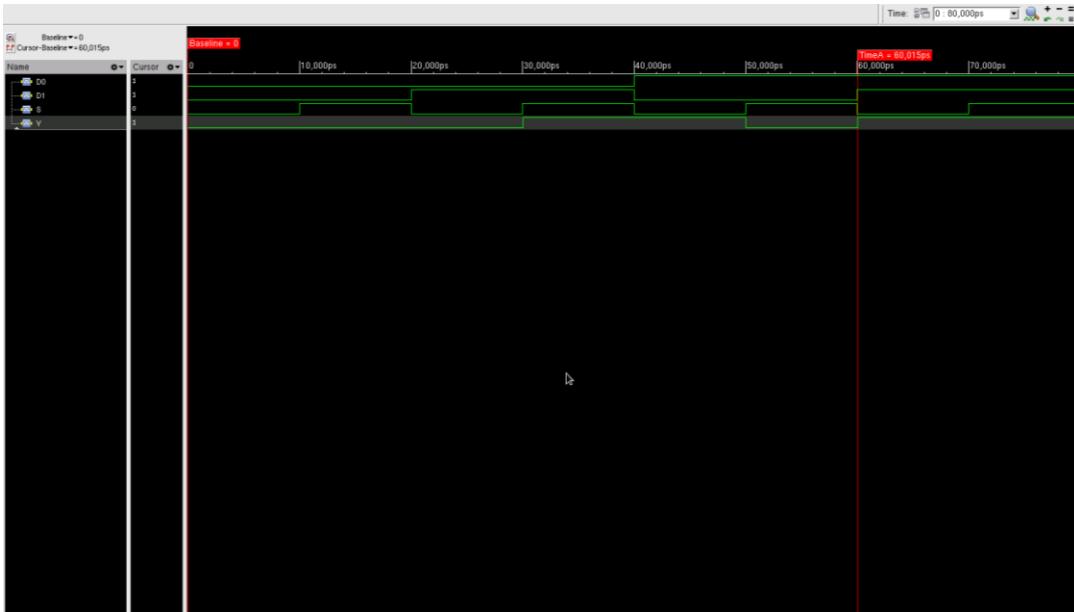


Quan sát hình dạng waveform, ta nhận thấy ngõ ra Y bị delay 1 khoảng 19ps(t_{rise})

TH6:fast_vdd1v2_basicCells_lvt.lib:

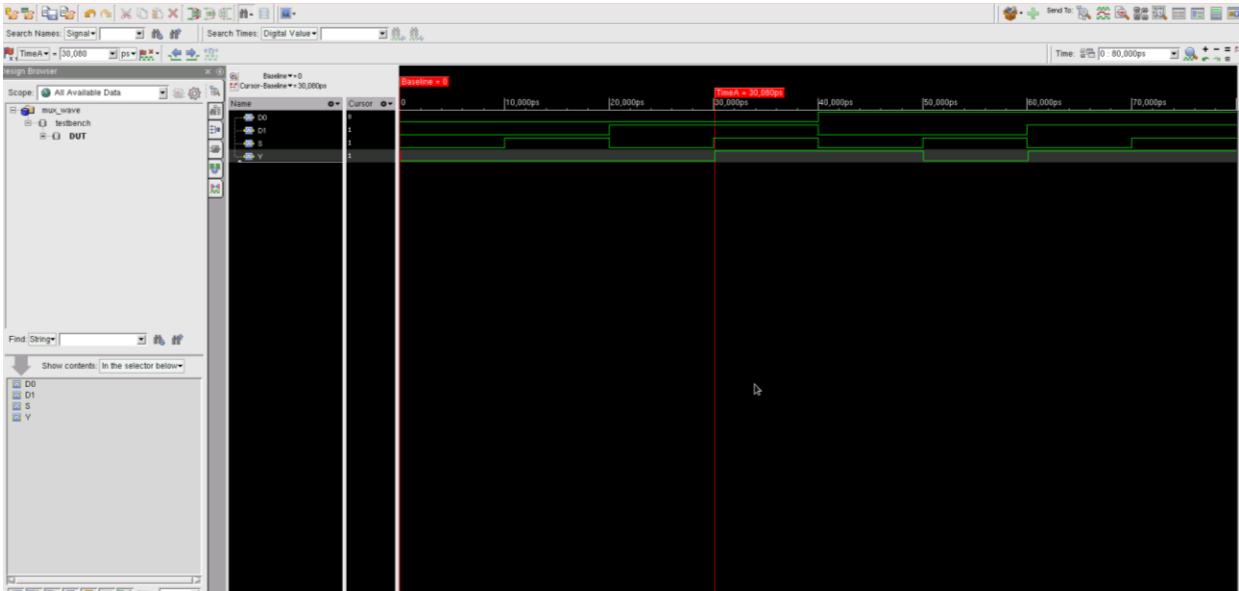


Quan sát hình dạng waveform, ta nhận thấy ngõ ra Y bị delay 1 khoảng 17ps(t_{rise})

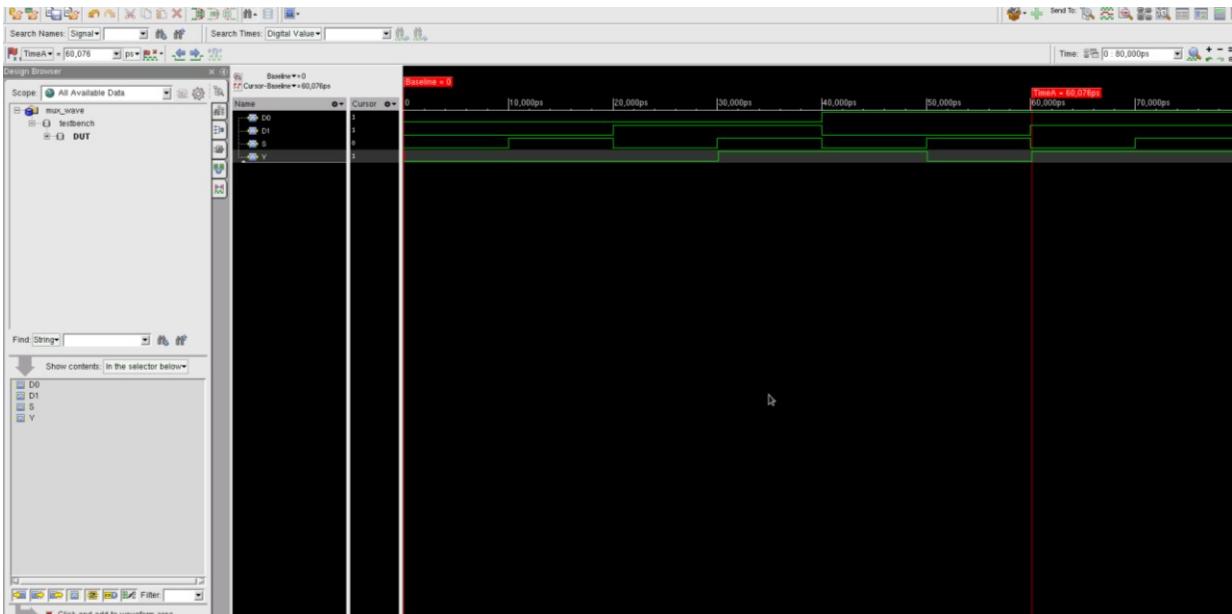


Quan sát hình dạng waveform, ta nhận thấy ngõ ra Y bị delay 1 khoảng 15ps(t_{rise})

TH7: slow_vdd1v0_basicCells_1vt.lib:

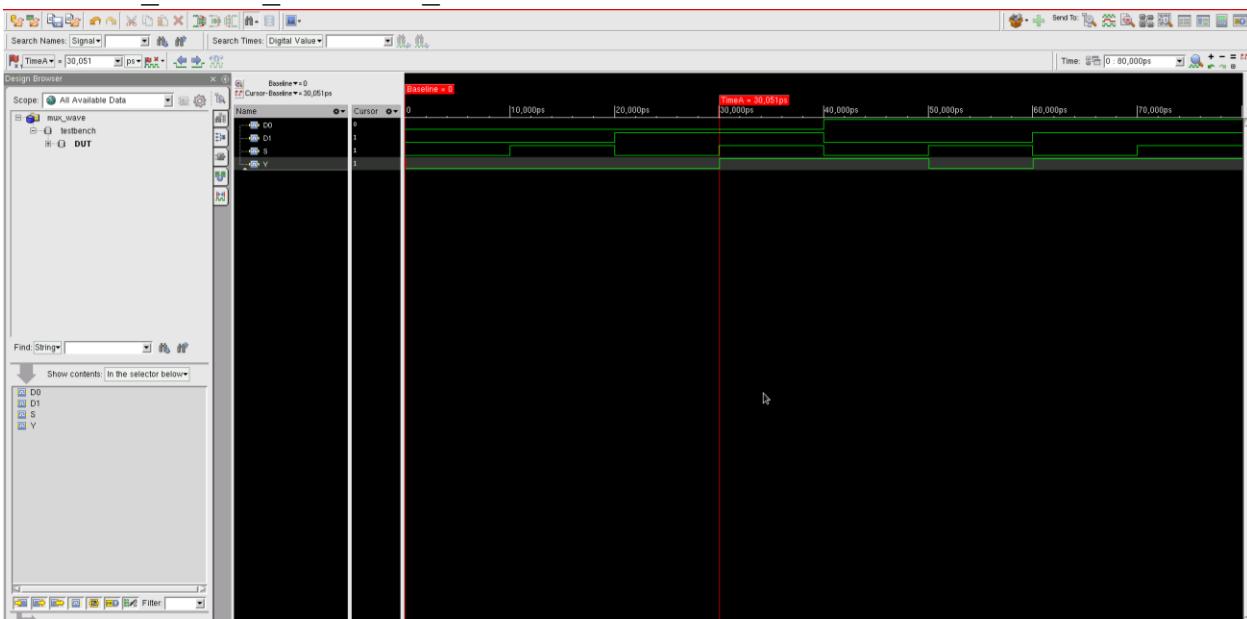


- Dựa vào hình dạng waveform, ta thấy ngõ ra Y bị delay 1 khoảng 80ps(t_{rise})

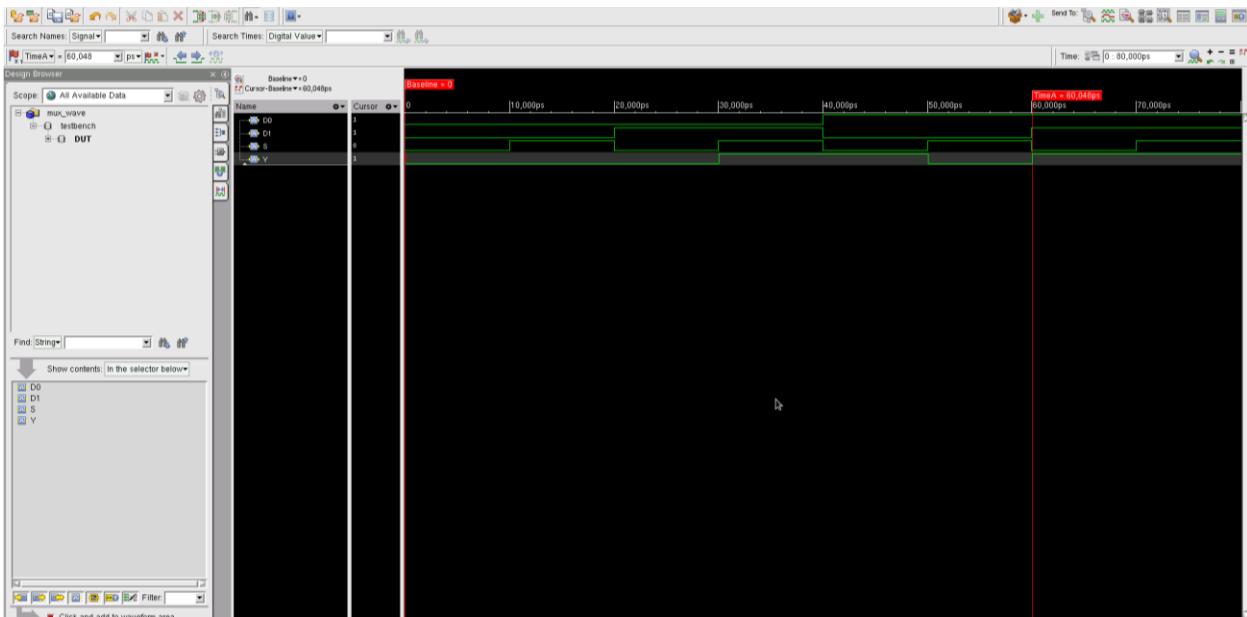


- Dựa trên dạng sóng waveform, ta nhận thấy ngõ ra Y bị delay 1 khoảng 76ps(t_{rise})

TH8:slow_vdd1v0_basicCells_lvt.lib:



Dựa trên dạng sóng waveform, ta nhận thấy ngõ ra Y bị delay 1 khoảng 51ps(t_{rise})



Dựa trên dạng sóng waveform, ta nhận thấy ngõ ra Y bị delay 1 khoảng 48ps(t_{fall})

1.6 Cell FULL ADDER

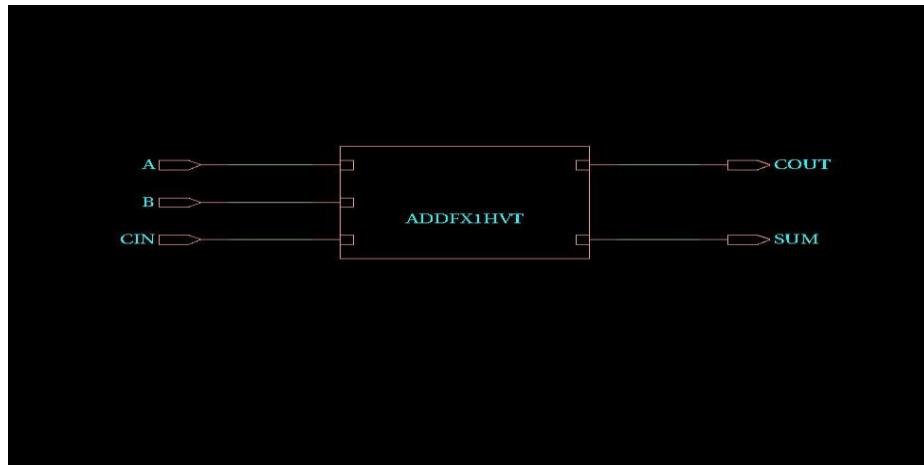
1.6.1 Diện tích (area) và công suất (power)

Corner	Library name	Area(μm^2)	Total Power (μW)
1	fast_vdd1v0_basicCells_hvt.lib	5.13	0.0872
2	fast_vdd1v0_basicCells_lvt.lib	5.13	0.1131
3	fast_vdd1v2_basicCells_hvt.lib	5.13	0.1354
4	fast_vdd1v2_basicCells_lvt.lib	5.13	0.1644
5	slow_vdd1v0_basicCells_hvt.lib	5.13	0.0537
6	slow_vdd1v0_basicCells_lvt.lib	5.13	0.0644
7	slow_vdd1v2_basicCells_hvt.lib	5.13	0.0841
8	slow_vdd1v2_basicCells_lvt.lib	5.13	0.0990

1.6.2 Độ trễ (delay)

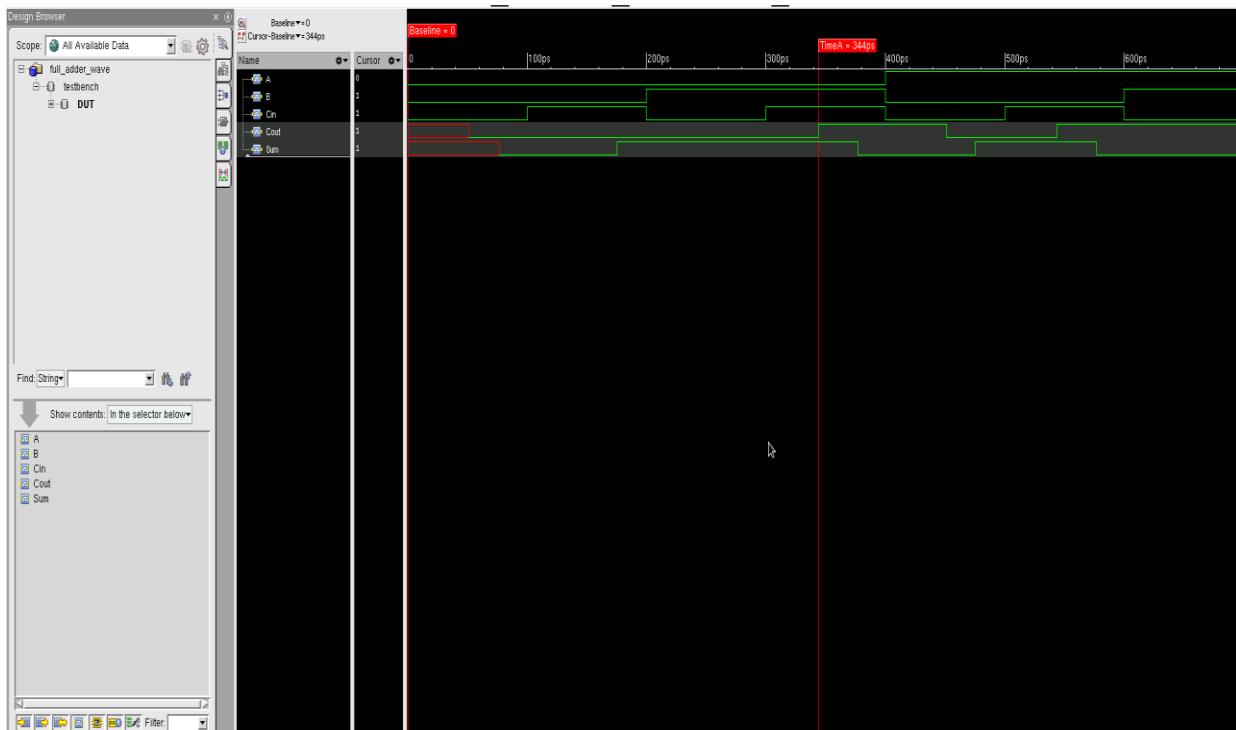
STT	Corner	CI→CO O	A→CO	B→C O	CI→ S	A→S	B→S
1	fast_vdd1v0_basicCells_hvt.lib	0.051	0.053	0.054	0.077	0.081	0.083
2	fast_vdd1v0_basicCells_lvt.lib	0.030	0.032	0.031	0.047	0.047	0.048
3	fast_vdd1v2_basicCells_hvt.lib	0.036	0.039	0.038	0.056	0.061	0.062
4	fast_vdd1v2_basicCells_lvt.lib	0.023	0.025	0.025	0.037	0.037	0.038
5	slow_vdd1v0_basicCells_hvt.lib	0.133	0.144	0.149	0.220	0.230	0.227
6	slow_vdd1v0_basicCells_lvt.lib	0.091	0.104	0.108	0.166	0.171	0.170
7	slow_vdd1v2_basicCells_hvt.lib	0.112	0.124	0.121	0.163	0.181	0.176
8	slow_vdd1v2_basicCells_lvt.lib	0.062	0.068	0.068	0.106	0.107	0.108

Netlist sinh ra:

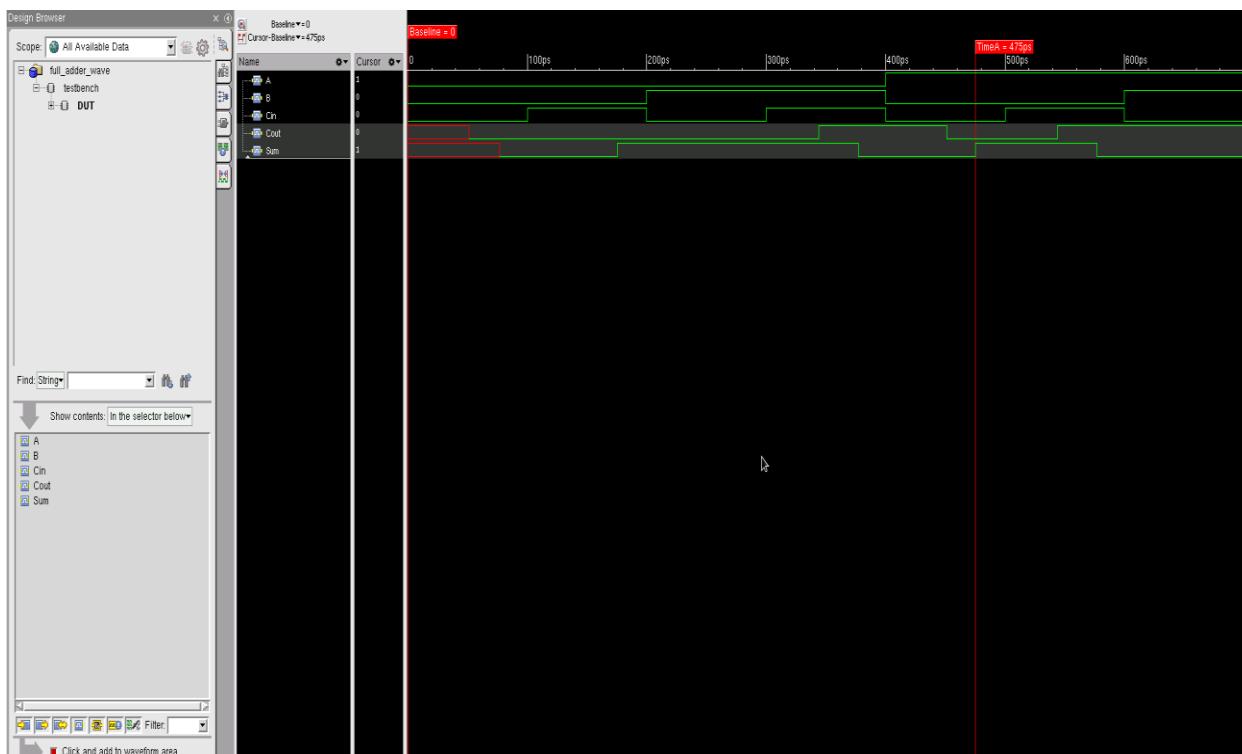


Wave form các trường hợp :

TH1: fast_vdd1v0_basicCells_hvt.lib:

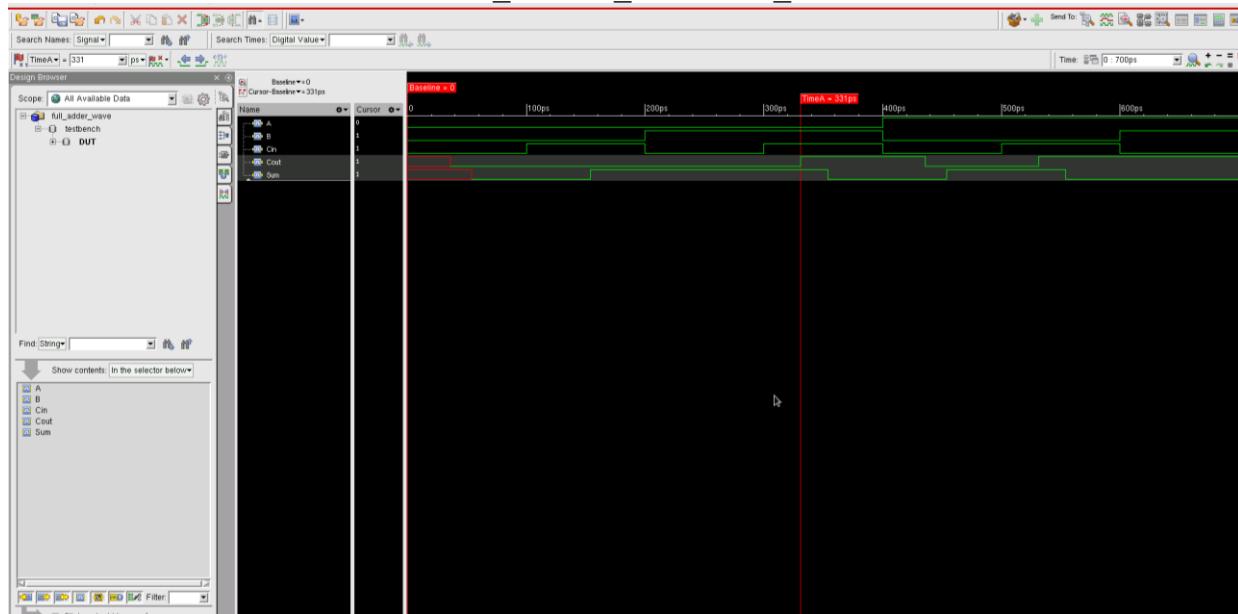


Dựa vào hình dạng waveform, ta thấy ngõ ra Cout bị delay 1 khoảng 44ps(t_{rise})

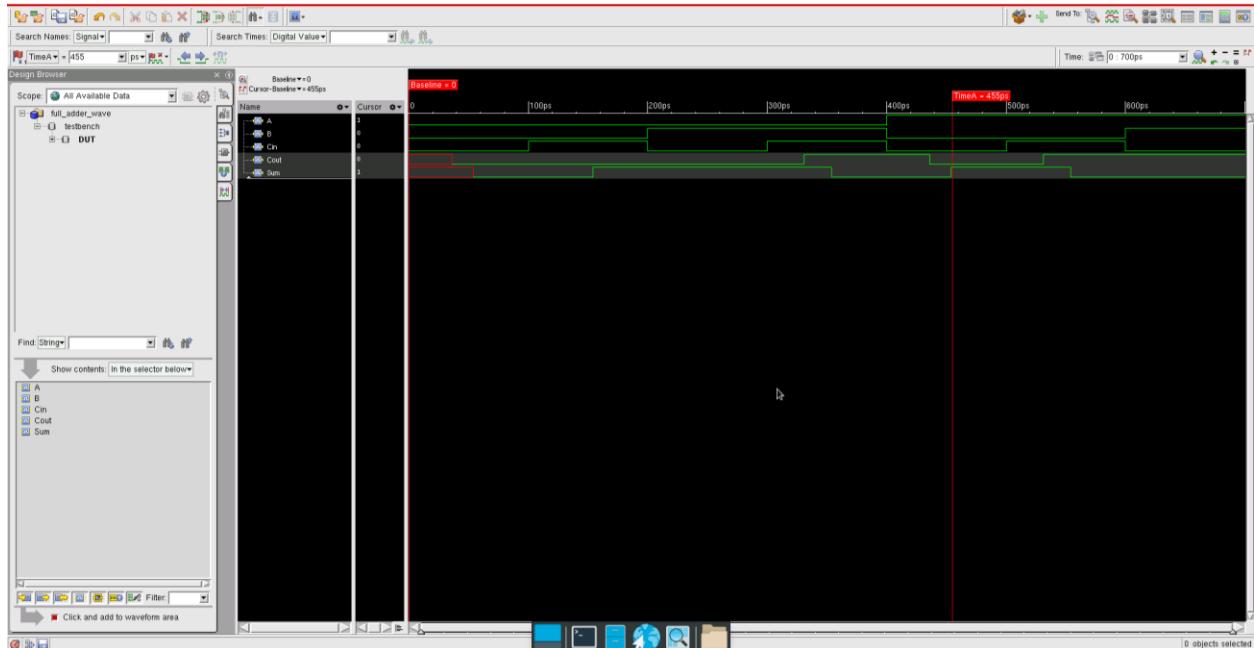


- Dựa vào hình dạng waveform, ta thấy ngõ ra S bị delay 1 khoảng 75ps(t_{rise})

TH2: fast_vdd1v2_basicCells_hvt.lib:

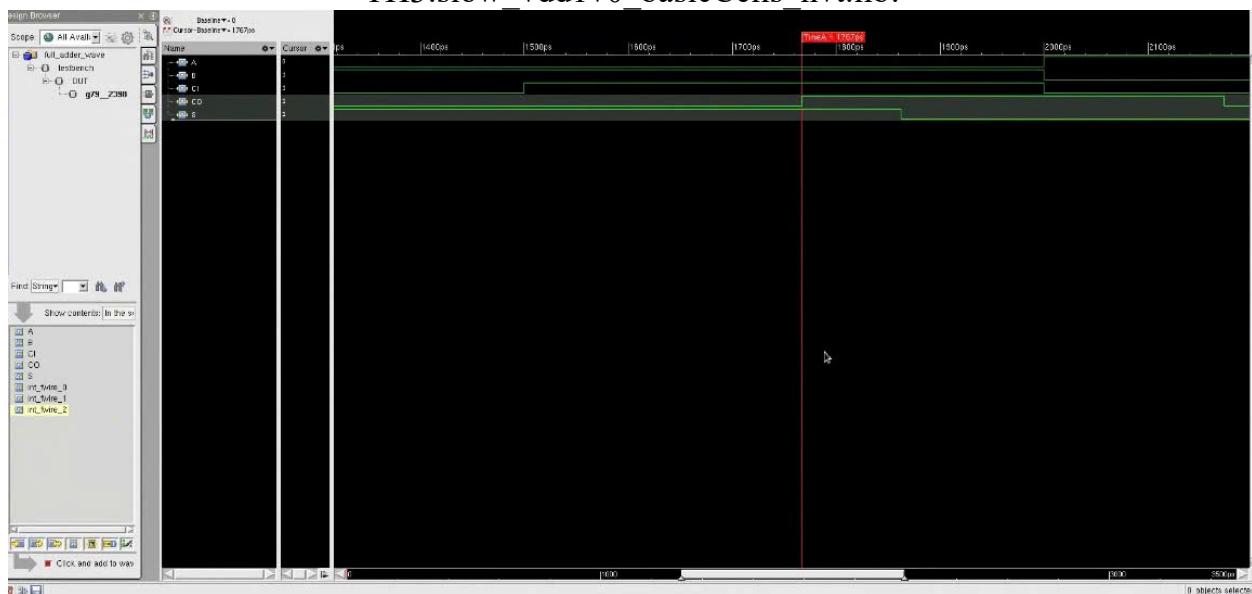


- Dựa vào hình dạng waveform, ta thấy ngõ ra Cout bị delay 1 khoảng 31ps(t_{rise})

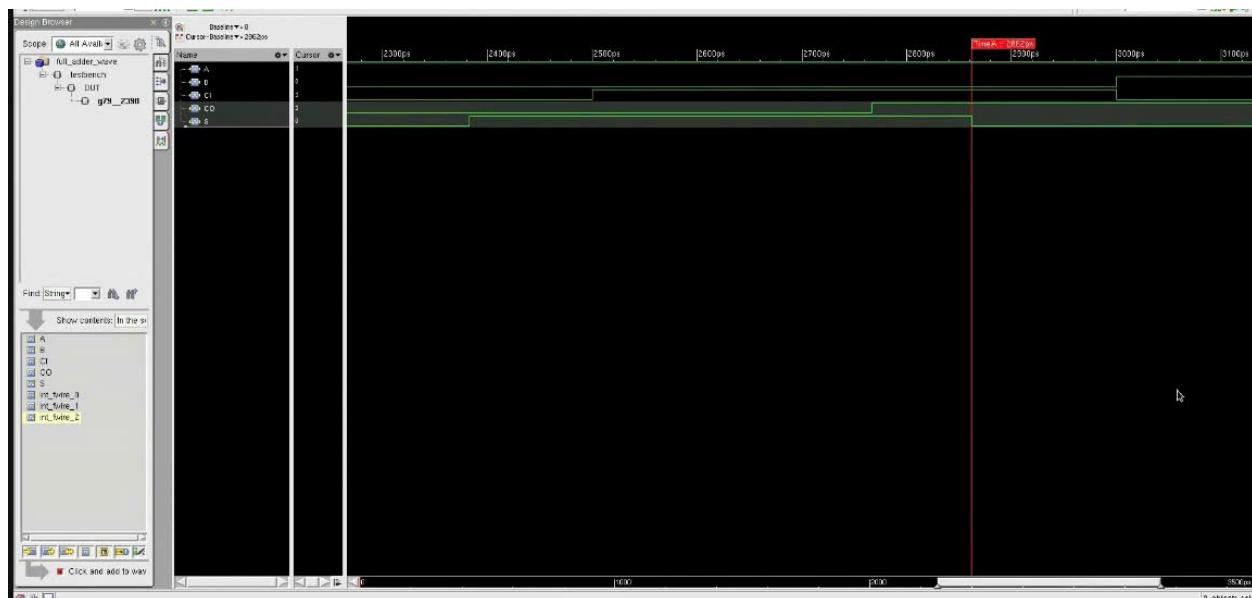


- Dựa vào hình dạng waveform, ta thấy ngõ ra S bị delay 1 khoảng 54ps(t_{rise})

TH3:slow vdd1v0 basicCells hvt.lib:

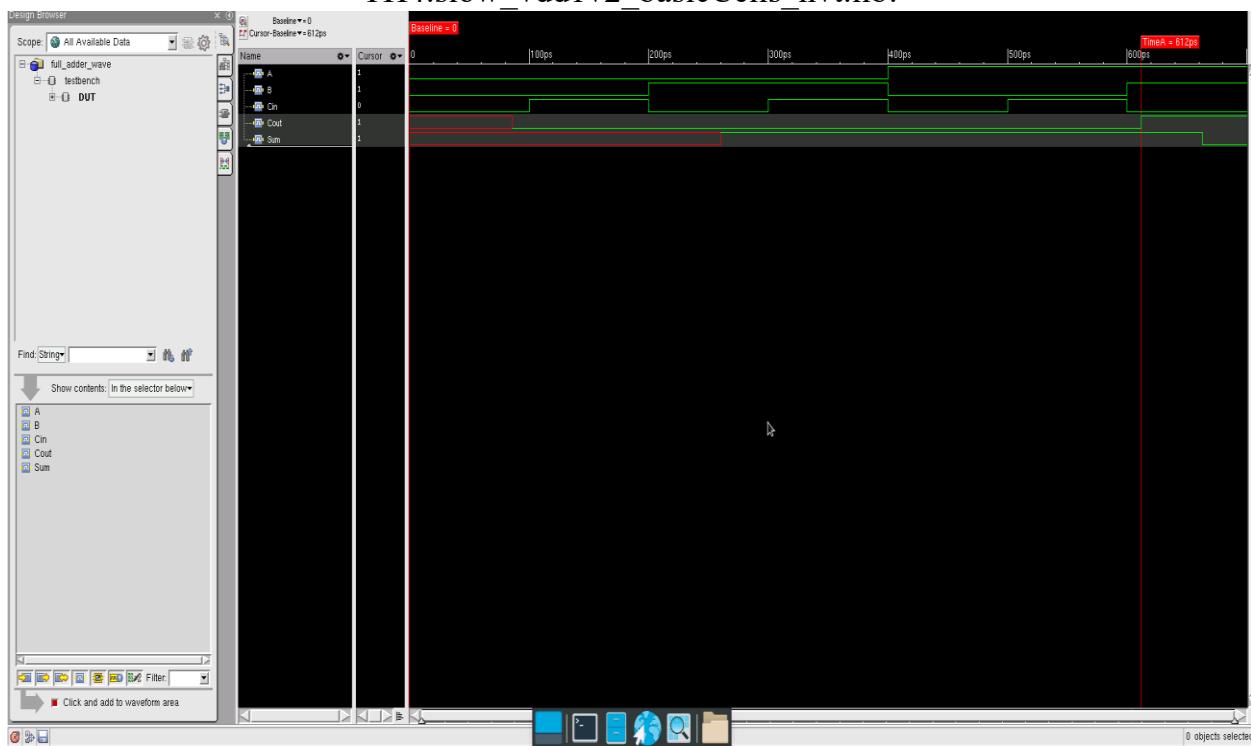


- Dựa vào hình dạng waveform, ta thấy ngõ ra Cout bị delay 1 khoảng 267ps(t_{rise})

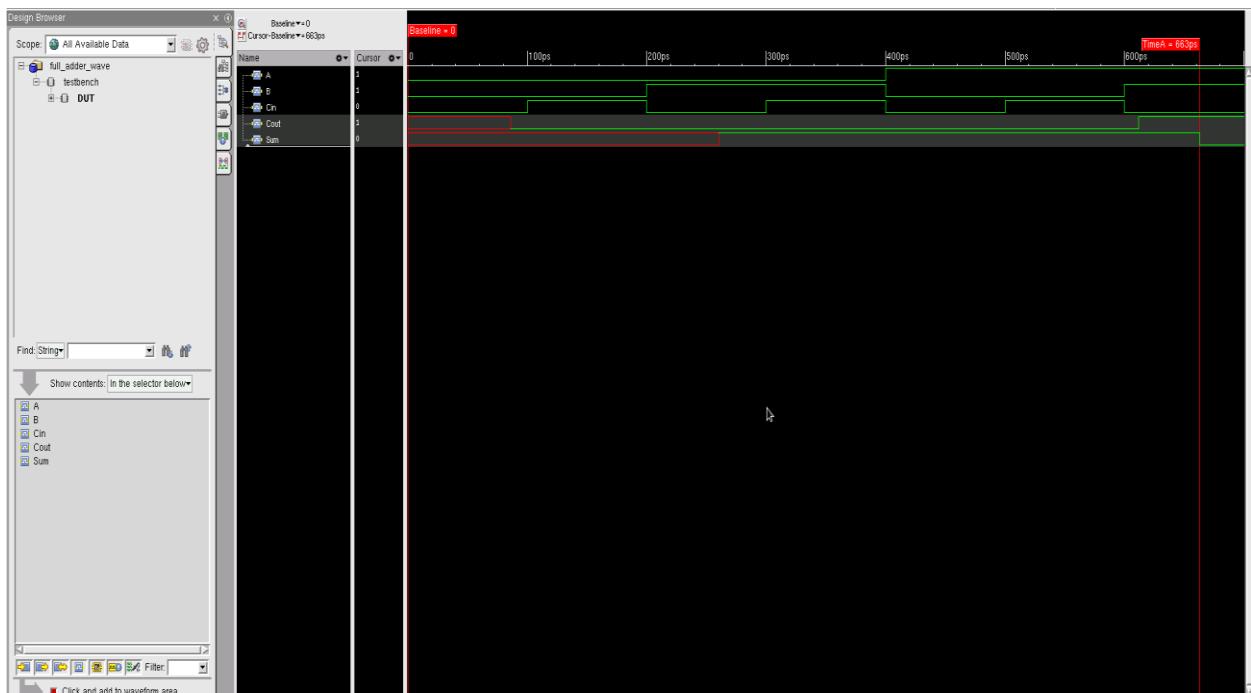


- Dựa vào hình dạng waveform, ta thấy ngõ ra S bị delay 1 khoảng 362ps(t_{fall})

TH4:slow_vdd1v2_basicCells_hvt.lib:

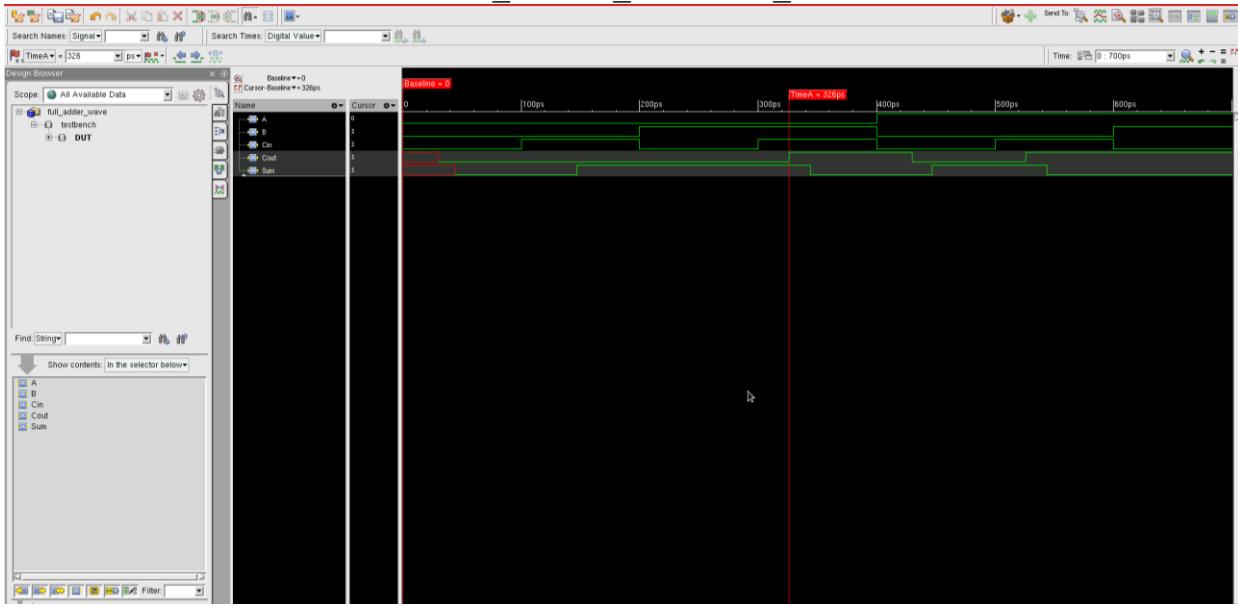


- Dựa vào hình dạng waveform, ta thấy ngõ ra Cout bị delay 1 khoảng 112ps(t_{rise})



- Dựa vào hình dạng waveform, ta thấy ngõ ra S bị delay 1 khoảng 163ps(t_{fall})

TH5:fast_vdd1v0_basicCells_lvt.lib:

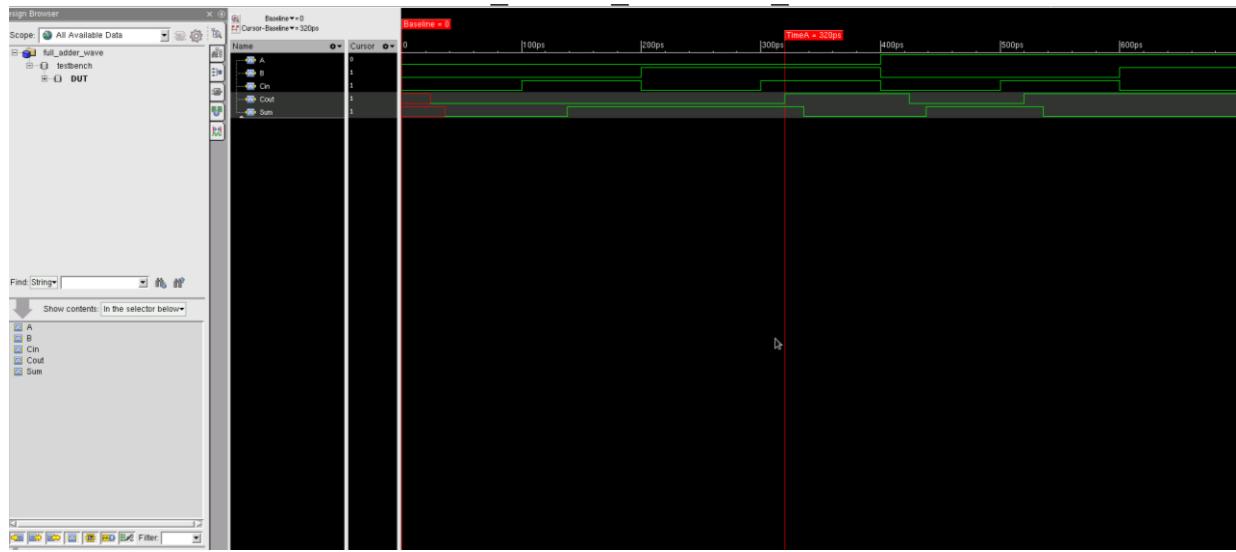


- Dựa vào hình dạng waveform, ta thấy ngõ ra Cout bị delay 1 khoảng 26ps(t_{rise})

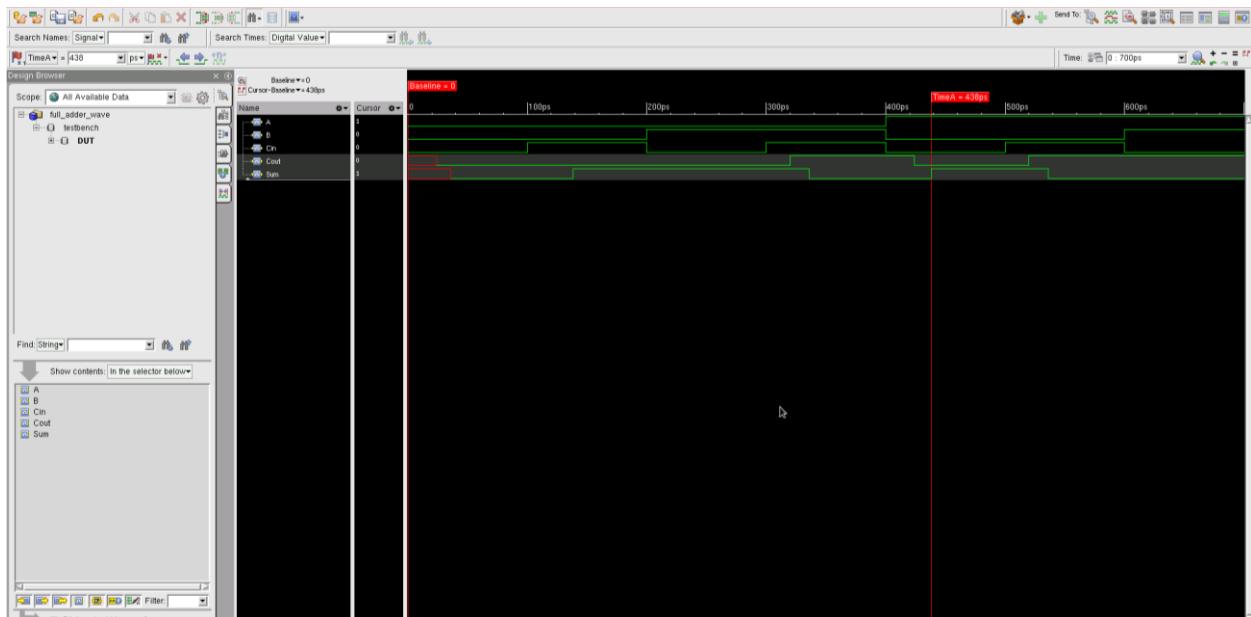


- Dựa vào hình dạng waveform, ta thấy ngõ ra S bị delay 1 khoảng 47ps(t_{fall})

TH6:fast_vdd1v2_basicCells_lvt.lib:

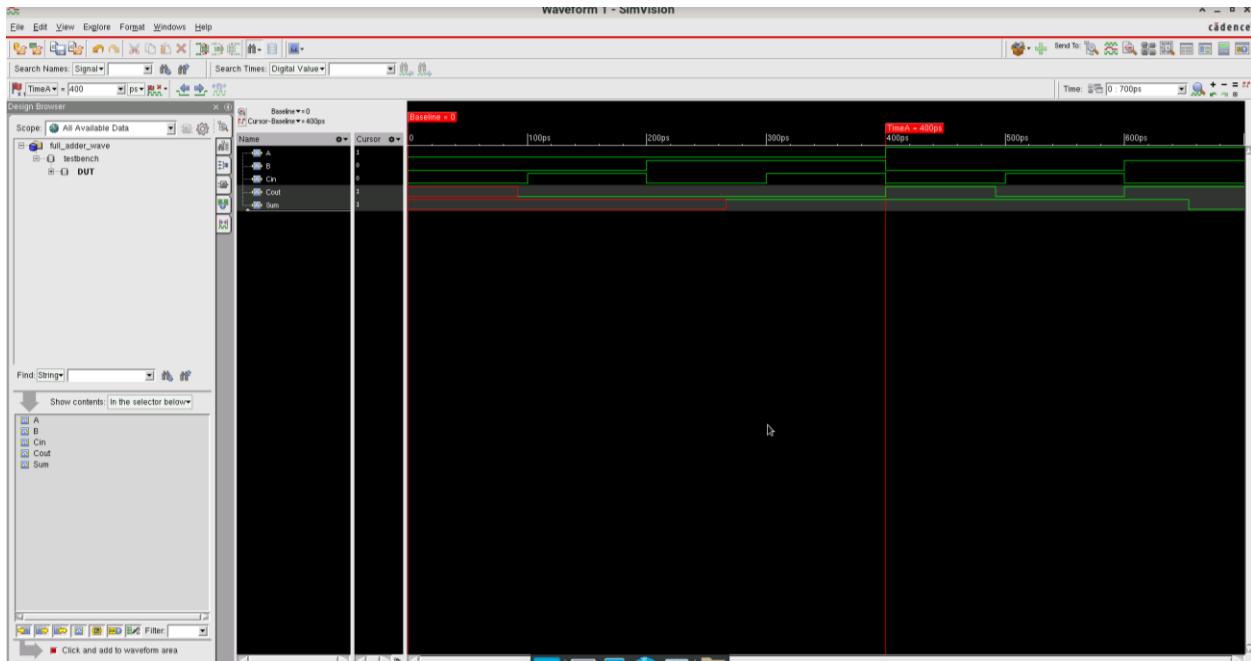


- Dựa vào hình dạng waveform, ta thấy ngõ ra Cout bị delay 1 khoảng 20ps(t_{rise})

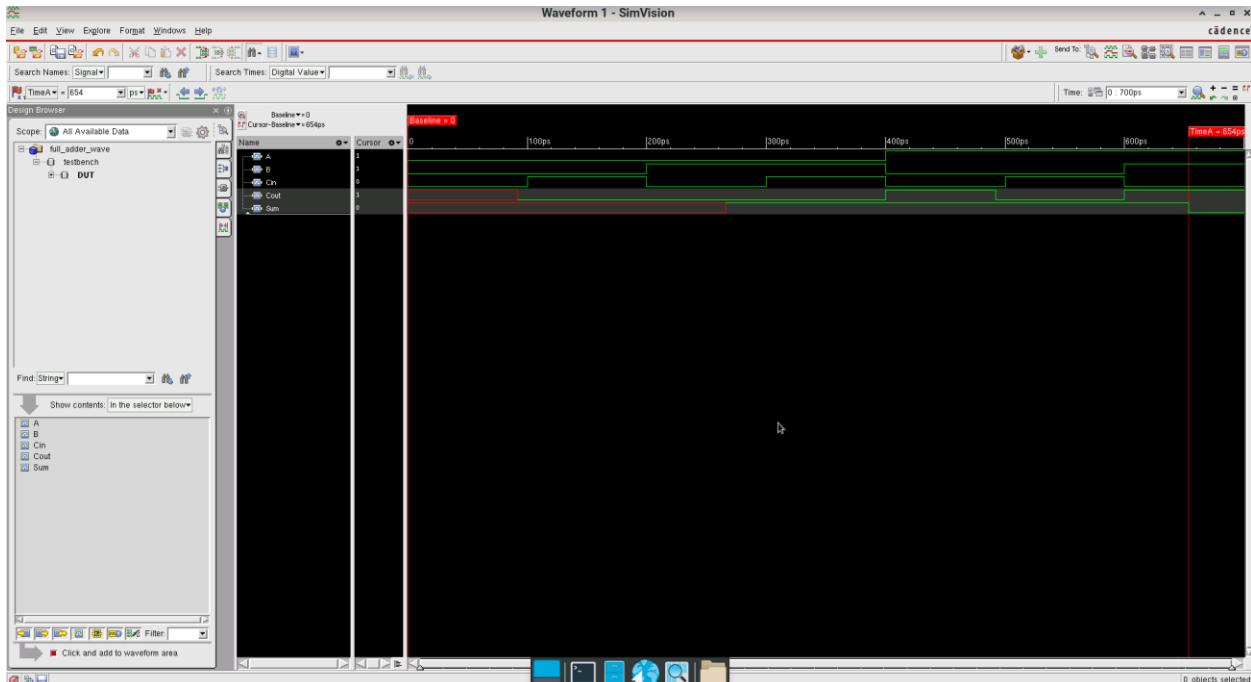


- Dựa vào hình dạng waveform, ta thấy ngõ ra S bị delay 1 khoảng 38ps(t_{rise})

TH7:slow_vdd1v0_basicCells_lvt.lib:

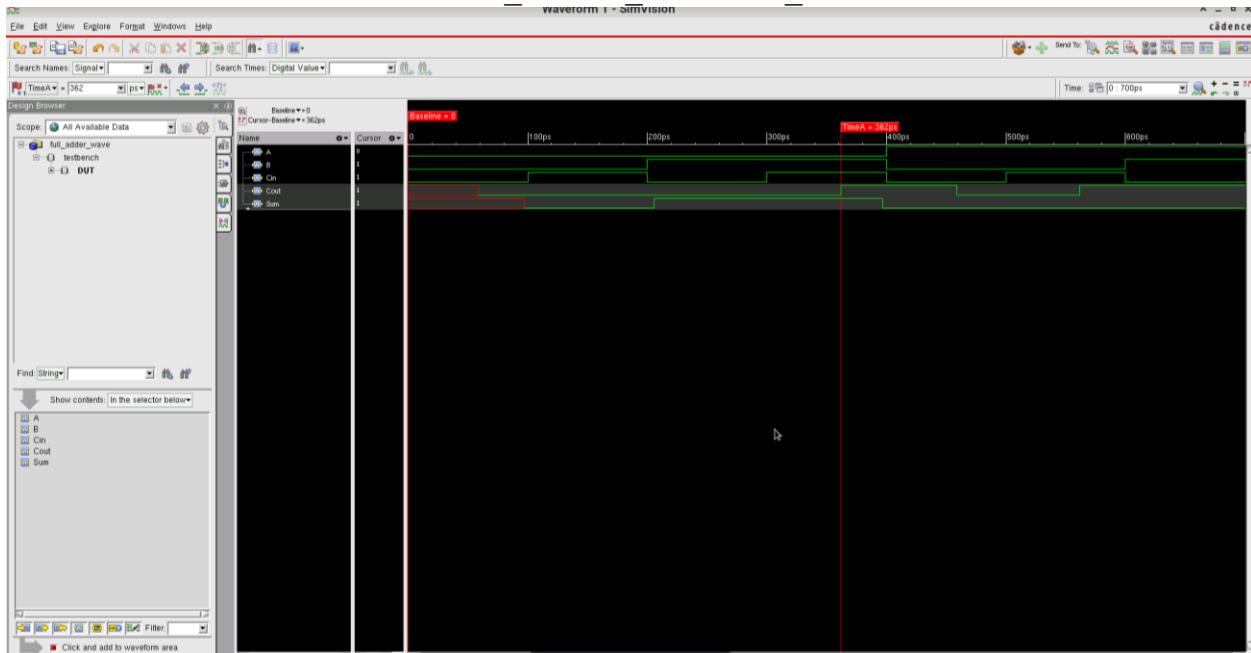


- Dựa vào hình dạng waveform, ta thấy ngõ ra Cout bị delay 1 khoảng 100ps(t_{rise})

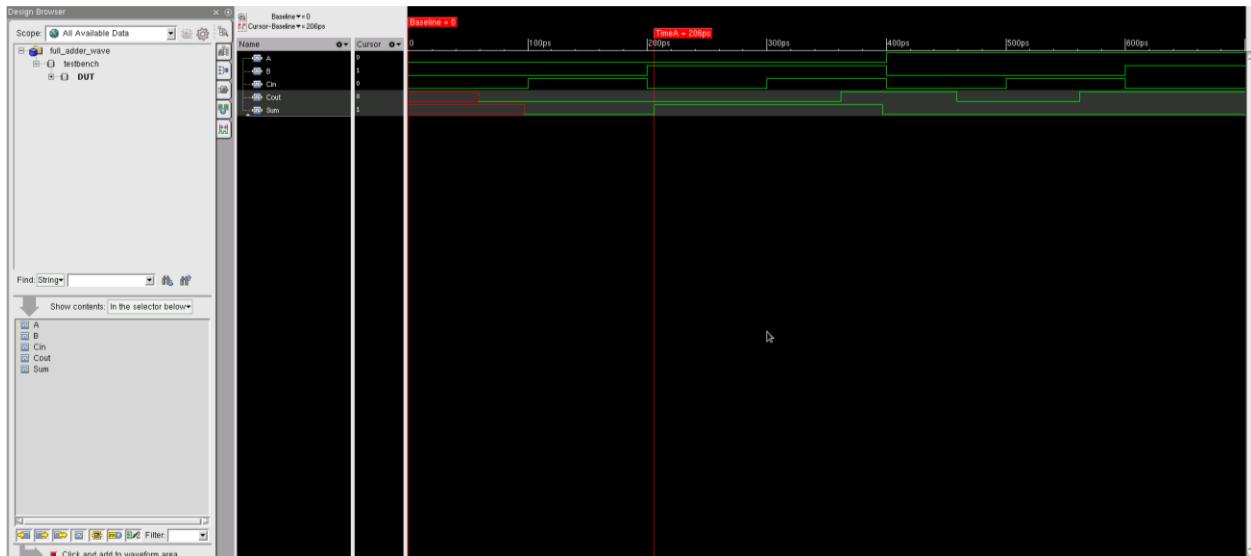


- Dựa vào hình dạng waveform, ta thấy ngõ ra S bị delay 1 khoảng 154ps(t_{fall})

TH8:slow_vdd1v2_basicCells_lvt.lib:



- Dựa vào hình dạng waveform, ta thấy ngõ ra Cout bị delay 1 khoảng 62ps(t_{rise})

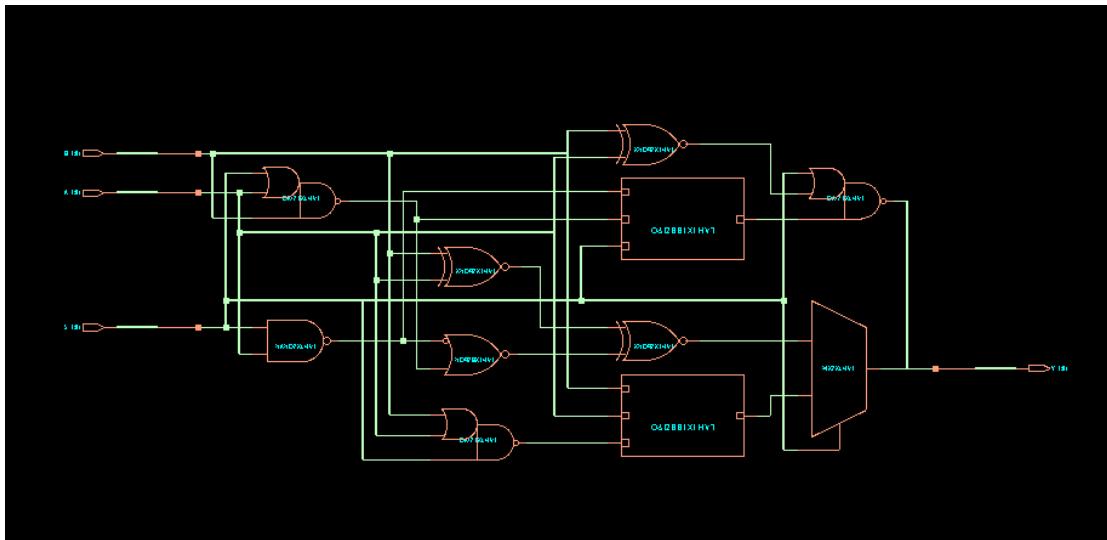


- Dựa vào hình dạng waveform, ta thấy ngõ ra Y bị delay 1 khoảng 106ps(t_{rise})

II. THÍ NGHIỆM 2

2.1 Sơ đồ mạch (schematic) và file netlist

2.1.1 Sơ đồ mạch



2.1.2. File Netlist

```
// Generated by Cadence Genus(TM) Synthesis Solution 21.19-s055_1
// Generated on: Nov 21 2025 17:15:48 +07 (Nov 21 2025 10:15:48 UTC)

// Verification Directory fv/alu

module alu(A, B, S, Y);
    input [1:0] A, B, S;
    output [1:0] Y;
    wire [1:0] A, B, S;
    wire [1:0] Y;
    wire n_0, n_1, n_2, n_3, n_4, n_5, n_6, n_7;
    wire n_9;
    MX2XLHVT g364__2398(.A (n_9), .B (n_3), .S0 (S[1]), .Y (Y[1]));
    XNOR2X1HVT g367__5107(.A (n_0), .B (n_6), .Y (n_9));
    OAI21XLHVT g365__6260(.A0 (S[1]), .A1 (n_2), .B0 (n_7), .Y (Y[0]));
    OAI2BB1X1HVT g368__4319(.A0N (n_5), .A1N (n_4), .B0 (S[1]), .Y (n_7));
    NOR2BX1HVT g370__8428(.AN (n_5), .B (n_4), .Y (n_6));
    OAI2BB1X1HVT g366__5526(.A0N (B[1]), .A1N (A[1]), .B0 (n_1), .Y
        (n_3));
    XNOR2X1HVT g372__6783(.A (B[0]), .B (A[0]), .Y (n_2));
    OAI21XLHVT g369__3680(.A0 (B[1]), .A1 (A[1]), .B0 (S[0]), .Y (n_1));
    XNOR2X1HVT g373__1617(.A (B[1]), .B (A[1]), .Y (n_0));
    OAI21XLHVT g371__2802(.A0 (S[0]), .A1 (A[0]), .B0 (B[0]), .Y (n_4));
    NAND2XLHVT g374__1705(.A (S[0]), .B (A[0]), .Y (n_5));
endmodule
```

2.1.3 SDF FILE

```
(DELAYFILE
(SDFVERSION "OVI 3.0")
(DESIGN    "alu")
(DATE      "Fri Nov 21 17:45:37 +07 2025")
```

```

(VENDOR      "Cadence, Inc.")
(PROGRAM    "Genus(TM) Synthesis Solution")
(VERSION    "21.19-s055_1")
(DIVIDER   .)
(VOLTAGE   ::1.08)
(PROCESS   "::1.0")
(TEMPERATURE ::125.0)
(TIMESCALE  1ns)
(CELL
  (CELLTYPE "MX2XLHVT")
  (INSTANCE g364_2398)
  (DELAY
    (ABSOLUTE
      (PORT A (::0.000))
      (PORT B (::0.000))
      (PORT S0 (::0.000))
      (IOPATH A Y (::0.095) (::0.070))
      (IOPATH B Y (::0.097) (::0.080))
      (IOPATH S0 Y (::0.089) (::0.074)))
    )
  )
)
(CELL
  (CELLTYPE "XNOR2X1HVT")
  (INSTANCE g367_5107)
  (DELAY
    (ABSOLUTE
      (PORT A (::0.000)))

```

```
(PORT B (":0.000))
(IOPATH A Y (":0.0131) (":0.0110))
(IOPATH B Y (":0.0112) (":0.0087))
)
)
)
(CELL
(CELLTYPE "OAI21XLHVT")
(INSTANCE g365_6260)
(DELAY
(ABSOLUTE
(PORT A0 (":0.000))
(PORT A1 (":0.000))
(PORT B0 (":0.000))
(IOPATH A0 Y (":0.028) (":0.045))
(IOPATH B0 Y (":0.026) (":0.044))
(IOPATH A1 Y (":0.030) (":0.041))
)
)
)
(CELL
(CELLTYPE "OAI2BB1X1HVT")
(INSTANCE g368_4319)
(DELAY
(ABSOLUTE
(PORT A0N (":0.000))
(PORT A1N (":0.000))
(PORT B0 (":0.000))
```

```
(IOPATH A1N Y (>::0.098) (>::0.077))
(IOPATH B0 Y (>::0.012) (>::0.035))
(IOPATH A0N Y (>::0.092) (>::0.074))
)
)
)
(CELL
  (CELLTYPE "NOR2BX1HVT")
  (INSTANCE g370_8428)
  (DELAY
    (ABSOLUTE
      (PORT AN (>::0.000))
      (PORT B (>::0.000))
      (IOPATH B Y (>::0.057) (>::0.045))
      (IOPATH AN Y (>::0.063) (>::0.062))
    )
  )
)
(CELL
  (CELLTYPE "OAI2BB1X1HVT")
  (INSTANCE g366_5526)
  (DELAY
    (ABSOLUTE
      (PORT A0N (>::0.000))
      (PORT A1N (>::0.000))
      (PORT B0 (>::0.000))
      (IOPATH A1N Y (>::0.075) (>::0.049))
      (IOPATH B0 Y (>::0.042) (>::0.057))
    )
  )
)
```

```
(IOPATH A0N Y (::0.082) (::0.052))
)
)
)

(CELL
  (CELLTYPE "XNOR2X1HVT")
  (INSTANCE g372_6783)
  (DELAY
    (ABSOLUTE
      (PORT A (::0.000))
      (PORT B (::0.000))
      (IOPATH A Y (::0.125) (::0.104))
      (IOPATH B Y (::0.103) (::0.072))
    )
  )
)

(CELL
  (CELLTYPE "OAI21XLHVT")
  (INSTANCE g369_3680)
  (DELAY
    (ABSOLUTE
      (PORT A0 (::0.000))
      (PORT A1 (::0.000))
      (PORT B0 (::0.000))
      (IOPATH A0 Y (::0.040) (::0.058))
      (IOPATH B0 Y (::0.014) (::0.048))
      (IOPATH A1 Y (::0.033) (::0.046))
    )
  )
)
```

```
)  
)  
(CELL  
  (CELLTYPE "XNOR2X1HVT")  
  (INSTANCE g373_1617)  
  (DELAY  
    (ABSOLUTE  
      (PORT A (::0.000))  
      (PORT B (::0.000))  
      (IOPATH A Y (::0.125) (::0.104))  
      (IOPATH B Y (::0.103) (::0.072))  
    )  
  )  
)  
(CELL  
  (CELLTYPE "OAI21XLHVT")  
  (INSTANCE g371_2802)  
  (DELAY  
    (ABSOLUTE  
      (PORT A0 (::0.000))  
      (PORT A1 (::0.000))  
      (PORT B0 (::0.000))  
      (IOPATH A0 Y (::0.046) (::0.067))  
      (IOPATH B0 Y (::0.017) (::0.057))  
      (IOPATH A1 Y (::0.039) (::0.055))  
    )  
  )  
)
```

```

(CELL
  (CELLTYPE "NAND2XLHVT")
  (INSTANCE g374_1705)
  (DELAY
    (ABSOLUTE
      (PORT A (::0.000))
      (PORT B (::0.000))
      (IOPATH A Y (::0.016) (::0.047))
      (IOPATH B Y (::0.014) (::0.039))
    )
  )
)
)

```

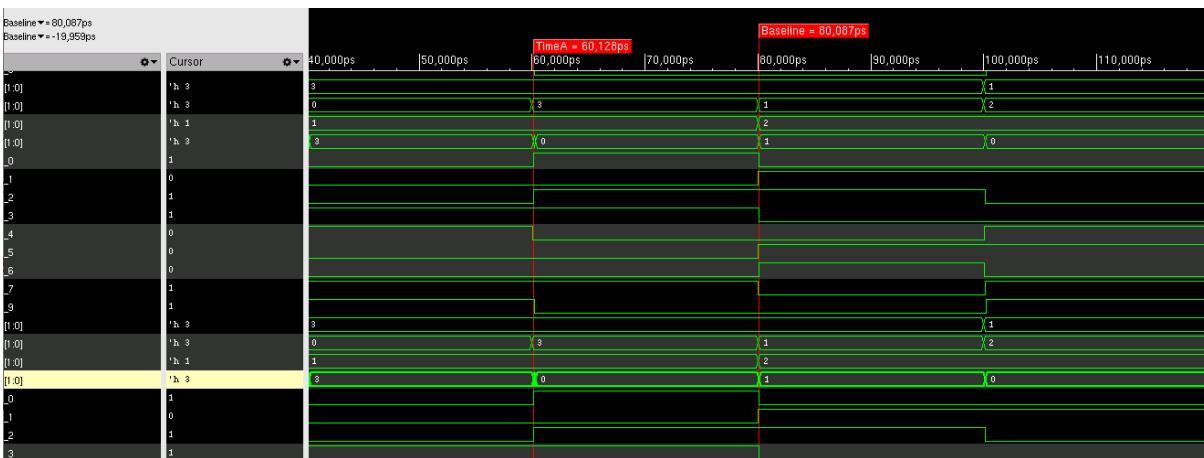
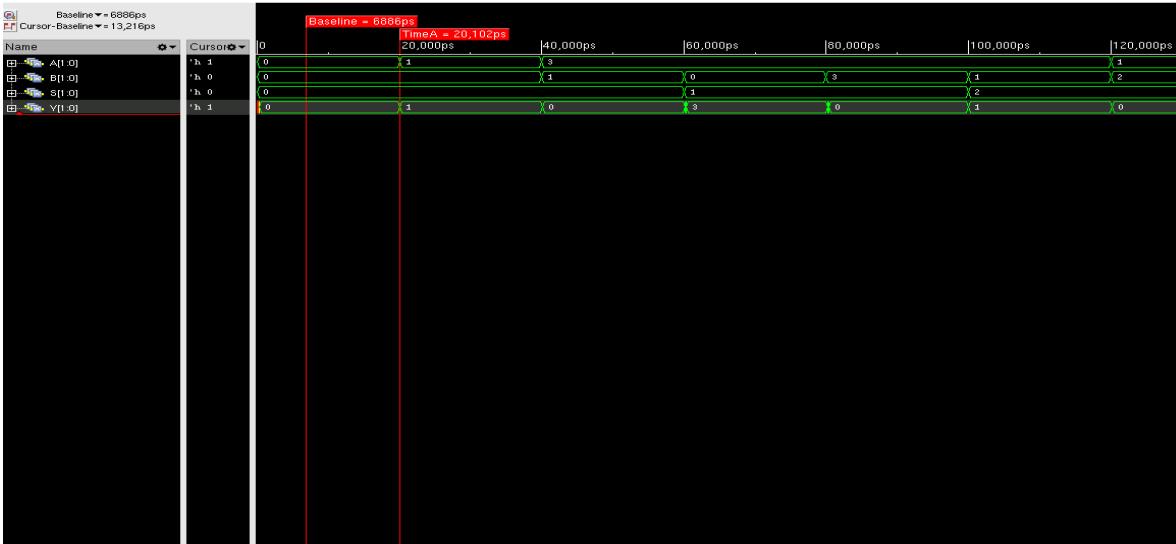
2.2 Dạng sóng (waveform)

2.2.1 Trường hợp 1: $T=20$

Ta có thể quan sát thấy khi thay đổi ngõ vào A thì ngõ ra Y sẽ có delay mới thay đổi theo. Làm tương tự khi ngõ vào B và S thay đổi thì cũng như vậy. Khi có càng nhiều ngõ vào thay đổi thì sự delay ở ngõ ra Y càng lớn.

Lý do là sau khi tổng hợp, những phép toán trong ALU không còn là “phép tính lý tưởng” như ở RTL. Chúng được thay bằng một mạng gồm rất nhiều cổng logic thật trong thư viện chuẩn. Mỗi cổng đều có thời gian lan truyền riêng, và tín hiệu từ A, B, S phải đi qua nhiều tầng cổng như XOR, AND, OR, MUX... trước khi ra được Y.

Vì vậy khi mô phỏng thực tế: A thay đổi dẫn đến tín hiệu cần thời gian đi qua các cổng tương ứng làm Y đổi trễ. Tương tự B hoặc S thay đổi làm đường truyền khác của ALU cũng có trễ tương tự. Còn Nếu nhiều ngõ vào thay đổi cùng lúc, số lượng tín hiệu chuyển mạch trong mạch tăng lên → sự kết hợp các đường logic làm độ trễ quan sát được ở Y lớn hơn.



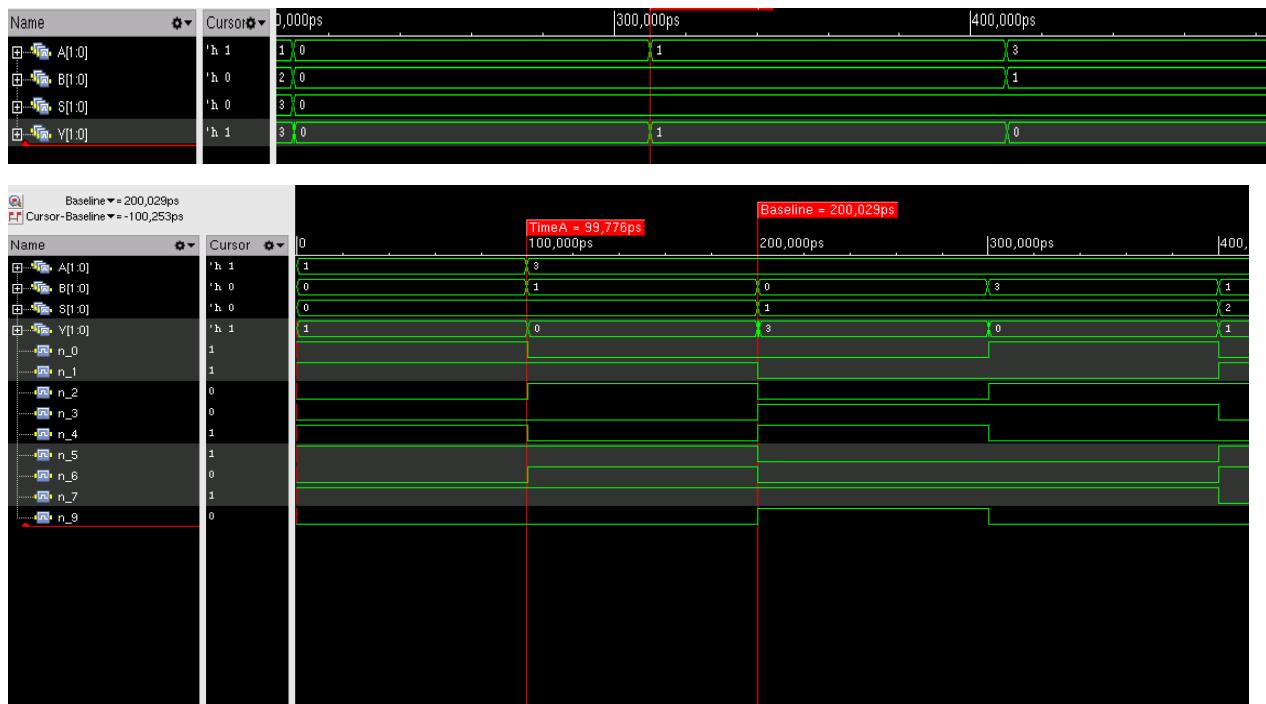
Phóng to chỗ nhiễu để thấy rõ delay.

2.2.2 Trường hợp 1: $T=100$

Ta quan sát thấy độ trễ khi xuất kết quả ra nhưng mà không rõ ràng như trường hợp

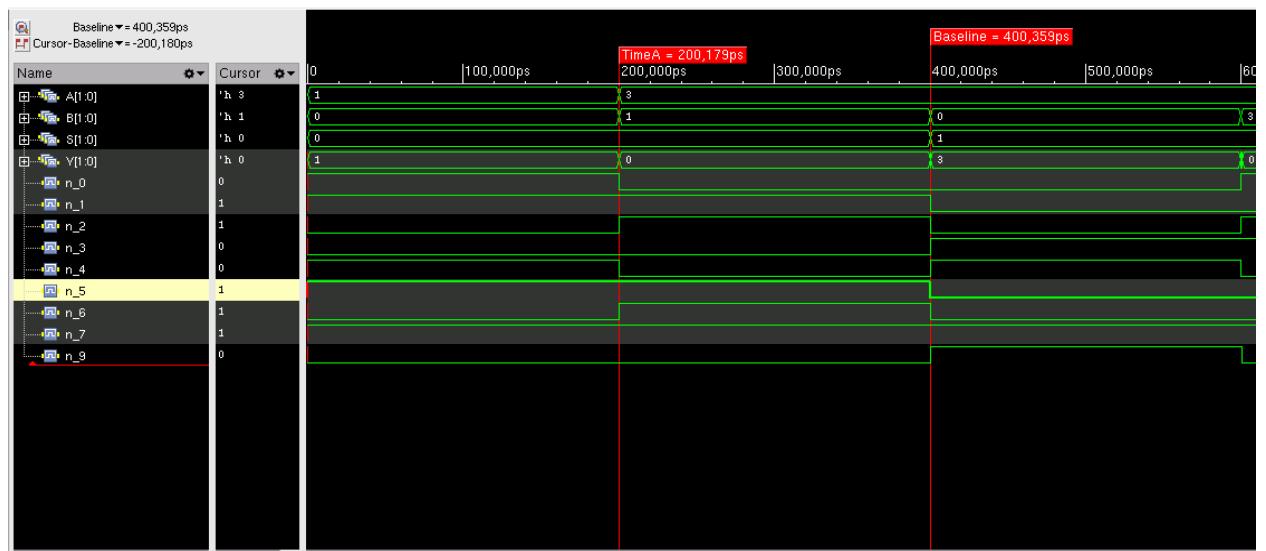
Nguyên nhân là vì khoảng thời gian giữa hai vector test (100 ns) lớn hơn rất nhiều so với hầu hết thời gian delay bên trong các đường logic của ALU (chỉ vài chục đến vài trăm picô-giây). Khi A, B hoặc S thay đổi, Y vẫn bị trễ một chút do tín hiệu phải đi qua nhiều cổng logic trong netlist. Tuy nhiên độ trễ này rất nhỏ so với 100 ns và các chuyển mức tạm thời (glitches) nếu có xuất hiện cũng biến mất rất nhanh.

Do đó, ngay sau khi input thay đổi một thời gian rất ngắn, output Y đã ổn định hoàn toàn, và ổn định suốt phần lớn khoảng T. Nhìn vào waveform, ta chỉ thấy một đoạn chuyển mức nhỏ ở đầu mỗi chu kỳ, sau đó là một đoạn dài mà Y giữ nguyên giá trị đúng.



Phóng to điểm nhiễu thấy chẽ delay và chu kì đúng T=100

2.2.3 Trường hợp 1: T=200



Phóng to điểm nhiễu thấy thời gian delay chi tiết và T=20

Ở trường hợp $T = 200$ ns, các tín hiệu đầu vào thay đổi rất chậm so với thời gian delay thực tế của ALU. Mỗi vector được giữ trong 200 ns, trong khi độ trễ lan truyền lớn nhất của mạch chỉ vào khoảng 0.1–0.13 ns. Điều này có nghĩa là ALU luôn có rất nhiều thời gian để ổn định sau mỗi lần A, B hoặc S thay đổi.

Do đó, dù delay vẫn tồn tại bên trong mạch (do các cổng logic HVT của corner slow_vdd1v2), nó không còn gây ảnh hưởng đáng kể lên waveform. Phần trễ chỉ chiếm một đoạn cực nhỏ ngay sau thời điểm input đổi, và nhanh chóng biến mất. Phần thời gian còn lại gần như toàn bộ 200 ns, ngõ ra Y ổn định hoàn toàn.

Kết quả là waveform thu được ở $T = 200$ ns gần như trùng khớp 100% với mô phỏng RTL không delay. Sự khác biệt duy nhất chỉ nằm ở cạnh bị dời sang phải một khoảng rất nhỏ (vài chục đến vài trăm picô-giây), nhưng điều này gần như không thể thấy bằng mắt khi so với khoảng T lớn như 200 ns.

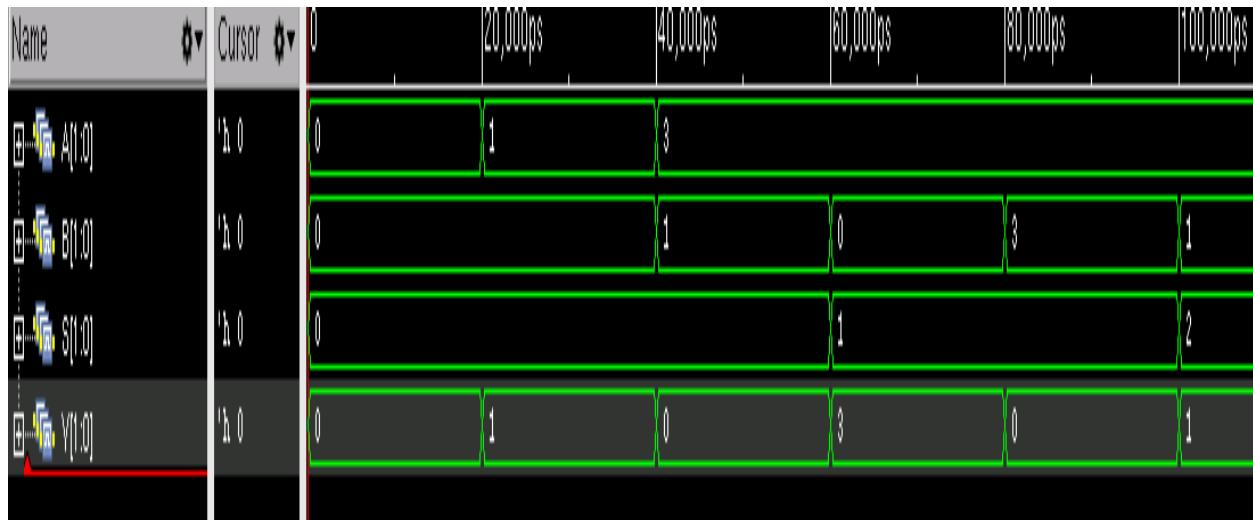
2.2.4 Mô phỏng không có delay

Khi loại bỏ file SDF trong mô phỏng, ALU sẽ chạy ở dạng RTL lý tưởng, có nghĩa là tất cả các phép toán được thực hiện ngay lập tức và không có bất kỳ propagation delay nào từ các cổng logic. Vì vậy, dạng sóng mô phỏng thu được phản ánh đúng 100% mô hình toán học của ALU.

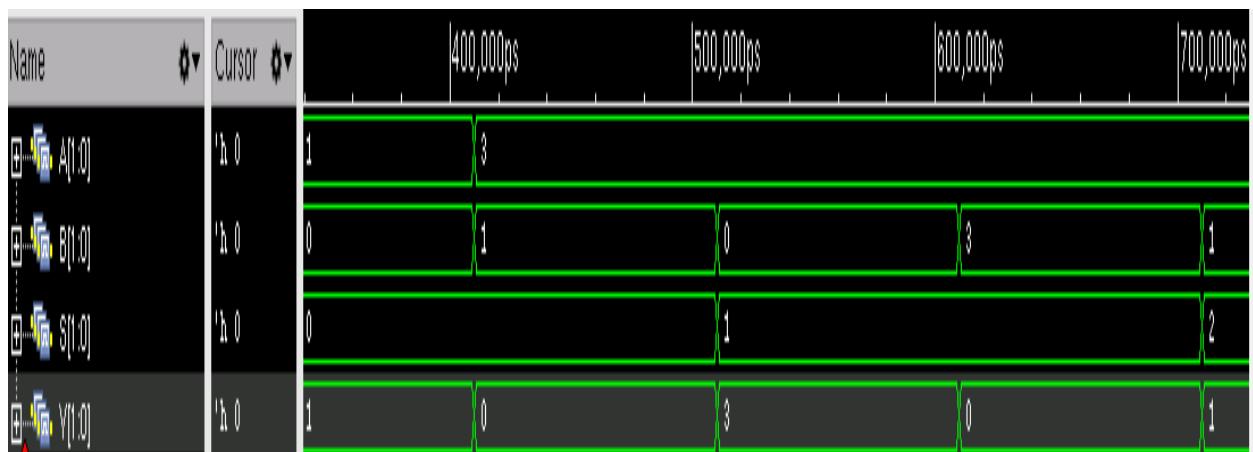
Quan sát ba trường hợp $T = 20$ ns, 100 ns và 200 ns, ta nhận thấy rằng khi T càng lớn thì dạng sóng của mô phỏng có SDF càng giống với dạng sóng lý tưởng không delay. Với T nhỏ (ví dụ 20 ns), có thể nhìn thấy rõ các cạnh bị dời sang phải và một vài chuyển mức tạm thời ngắn trước khi ổn định. Nhưng khi T tăng lên 100 ns và đặc biệt là 200 ns, phần thời gian mà output chưa ổn định trở nên quá nhỏ so với toàn khoảng T , khiến waveform thực tế gần như trùng với waveform lý thuyết của RTL.

Đến $T = 200$ ns, độ trễ của ALU (chỉ khoảng 0.1 ns) trở nên quá nhỏ so với thời gian giữ từng vector, nên dạng sóng gần như đồng nhất 100% với mô phỏng không có delay và điểm khác biệt duy nhất chỉ là cạnh ngõ ra bị dời một lượng rất nhỏ mà gần như không thể nhìn thấy khi quan sát theo khoảng thời gian lớn.

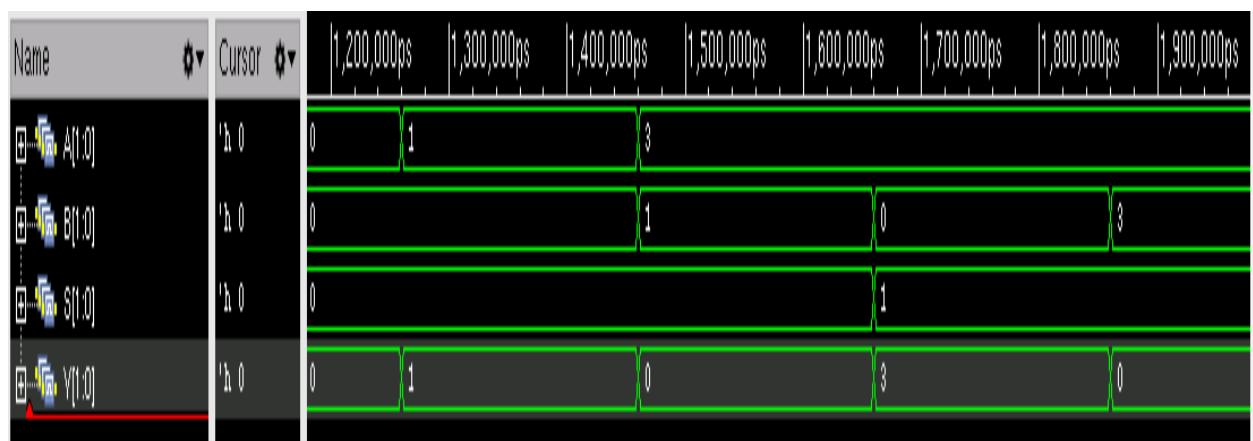
a) Trường hợp 1: T=20



b) Trường hợp 1: T=100



c) Trường hợp 1: T=200



III. THÍ NGHIỆM 3

3.1 HDL modules:

- **dff.sv:**

```
module dff(
    input logic i_d,
    input logic i_clk,
    output logic o_q
);
    always_ff @(posedge i_clk)
        o_q <= i_d;
endmodule
```

- **dff_testbench.sv:**

```
`timescale 1ns/1ps
```

```
`ifdef GATE_LEVEL
`include
"/home/yellow/ee3201_19/Desktop/LAB2/ex3/03_outputs/slow_vdd1v2_lvt/netlist.sv"          //kiem tra khi doi lib
`include
"/home/yellow/ee3201_19/Desktop/LAB2/PDK45nm/gpdk045_verilog/slow_vdd1v2_basicCells_lvt.v"      //kiem tra khi doi lib
`else
`include "/home/yellow/ee3201_19/Desktop/LAB2/ex3/00_src/dff.sv"
`endif
```

```
module tb_dff;
```

```
// tín hiệu
logic i_clk;
logic i_d;
logic o_q;
```

```
dff DUT (
    .i_d(i_d),
    .i_clk(i_clk),
    .o_q(o_q)
);
```

```

`ifdef ANNOTATION
initial begin
    // back-annotate SDF

$ddf_annotate("/home/yellow/ee3201_19/Desktop/LAB2/ex3/03_outputs/slow_vdd
lv2_lvt/delay.sdf", DUT, , "annotate.log", "MAXIMUM");      //kiem tra khi doi
lib
end
`endif

// VCD waveform
initial begin
    $dumpfile("dff_wave.vcd");

$dumpfile("/home/yellow/ee3201_19/Desktop/LAB2/ex3/03_outputs/slow_vddlv2_
lvt/dff_wave.vcd");                      //kiem tra khi doi lib
    $dumpvars(0, tb_dff.DUT);
end

// clock generation: period = 10 ns (100 MHz)
initial i_clk = 1'b0;
always #5 i_clk = ~i_clk;

// STIMULUS
initial begin
    // --- ensure all inputs/outputs are defined before first case
    i_d = 1'b1;      // initialize input -> ensures o_q will be defined after a few
clocks

    // wait several positive edges so Q is driven and not X anymore
repeat (5) @(posedge i_clk);

    // ===== Case 1: reset behavior (D=0 -> Q should be 0 after
posedge) =====
    #2
    i_d = 1'b0;
    // wait two clock cycles to observe stable reset behavior
    @(posedge i_clk);
    @(posedge i_clk);

```

```

// ===== Case 2: set behavior (D=1 -> Q should be 1 after
posedge) =====
#2
i_d = 1'b1;
// wait two clock cycles to observe set behavior
@(posedge i_clk);
@(posedge i_clk);
#2

// ===== Case 3: setup tests =====
// 3a: SETUP (posedge D before posedge CK) -- 0 -> 1 shortly BEFORE clock
edge
i_d = 1'b0;      // prepare for 0->1 transition
@(negedge i_clk); // reference: posedge in ~5 ns
#4.995;          // ~10 ps before next posedge (timescale 1ns/1ps)
i_d = 1'b1;      // 0->1 near posedge (tests SETUP (posedge D) ...)
@(posedge i_clk); // let the posedge pass
@(posedge i_clk); // small settle
#2

// 3b: SETUP (negedge D before posedge CK) -- 1 -> 0 shortly BEFORE clock
edge
i_d = 1'b1;      // prepare for 1->0 transition
@(negedge i_clk);
#4.995;          // ~10 ps before posedge
i_d = 1'b0;      // 1->0 near posedge (tests SETUP (negedge D) ...)
@(posedge i_clk);
@(posedge i_clk);
#2

// ===== Case 4: hold tests =====
// 4a: HOLD (posedge D after posedge CK) -- 0 -> 1 shortly AFTER clock edge
i_d = 1'b0;
@(posedge i_clk);
#0.001;          // 1 ps after posedge
i_d = 1'b1;      // change immediately after posedge (tests HOLD (posedge
D) ...)
@(posedge i_clk);
@(posedge i_clk);

```

#2

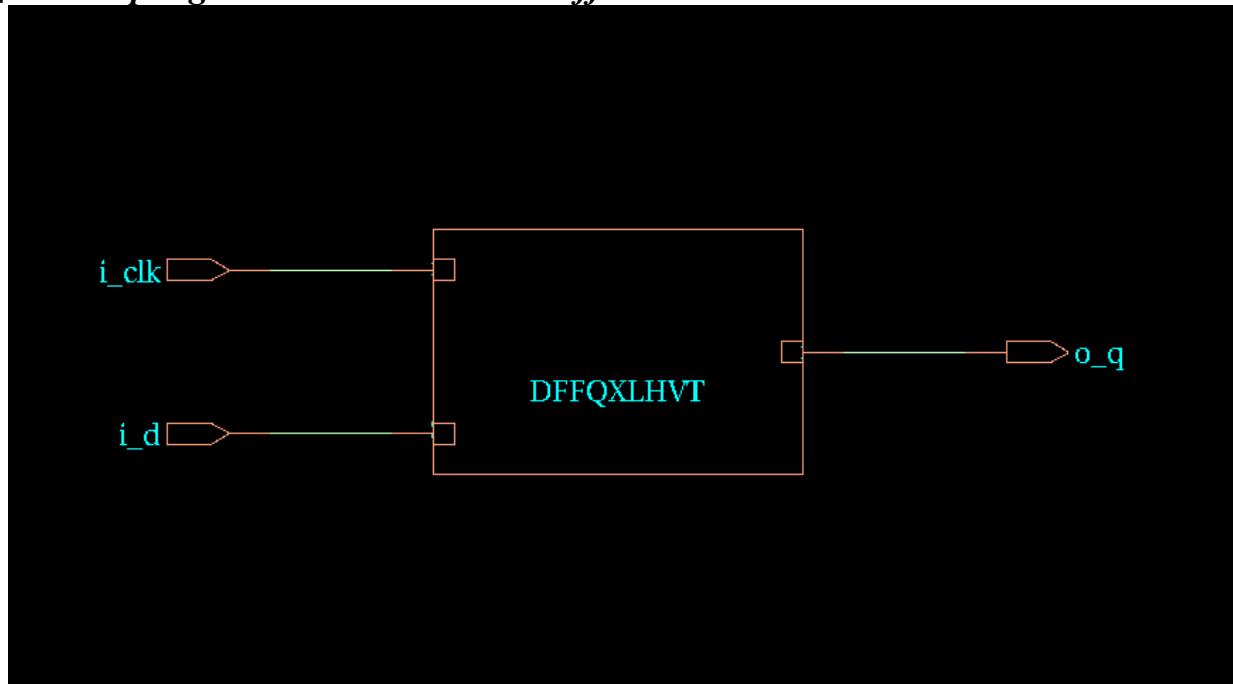
```
// 4b: HOLD (negedge D after posedge CK) -- 1 -> 0 shortly AFTER clock edge
i_d = 1'b1;
@(posedge i_clk);
#0.001;           // 1 ps after posedge
i_d = 1'b0;       // change immediately after posedge (tests HOLD (negedge
D) ...)
@(posedge i_clk);
@(posedge i_clk);
#2

// extra cycles to observe final behaviour
i_d = 1'b1;
@(posedge i_clk);
@(posedge i_clk);

$finish;
end

endmodule
```

- **Kết quả generate netlist module dff.sv:**



3.2 Measurement results:

a) area và total power:

Corner	Library name	Total area (μm^2)	Total Power (W)
1	fast_vdd1v0_basicCells_hvt.lib	5.472	2.67793e-07
2	fast_vdd1v2_basicCells_hvt.lib	5.472	3.91969e-07
3	slow_vdd1v0_basicCells_hvt.lib	5.472	2.11036e-07
4	slow_vdd1v2_basicCells_hvt.lib	5.472	3.16817e-07
5	fast_vdd1v0_basicCells_lvt.lib	5.472	3.25211e-07
6	fast_vdd1v2_basicCells_lvt.lib	5.472	4.67600e-07
7	slow_vdd1v0_basicCells_lvt.lib	5.472	2.42533e-07
8	slow_vdd1v2_basicCells_lvt.lib	5.472	3.58466e-07

b) delay:

STT	Corner	Đường đo	t_rise(ns)	t_fall(ns)
1	fast_vdd1v0_basicCells_hvt.lib	CK-Q	0.051	0.058
2	fast_vdd1v2_basicCells_hvt.lib	CK-Q	0.037	0.042
3	slow_vdd1v0_basicCells_hvt.lib	CK-Q	0.243	0.292
4	slow_vdd1v2_basicCells_hvt.lib	CK-Q	0.115	0.135
5	fast_vdd1v0_basicCells_lvt.lib	CK-Q	0.029	0.033
6	fast_vdd1v2_basicCells_lvt.lib	CK-Q	0.024	0.027
7	slow_vdd1v0_basicCells_lvt.lib	CK-Q	0.106	0.121
8	slow_vdd1v2_basicCells_lvt.lib	CK-Q	0.068	0.078

3.3 Verification result (waveform):

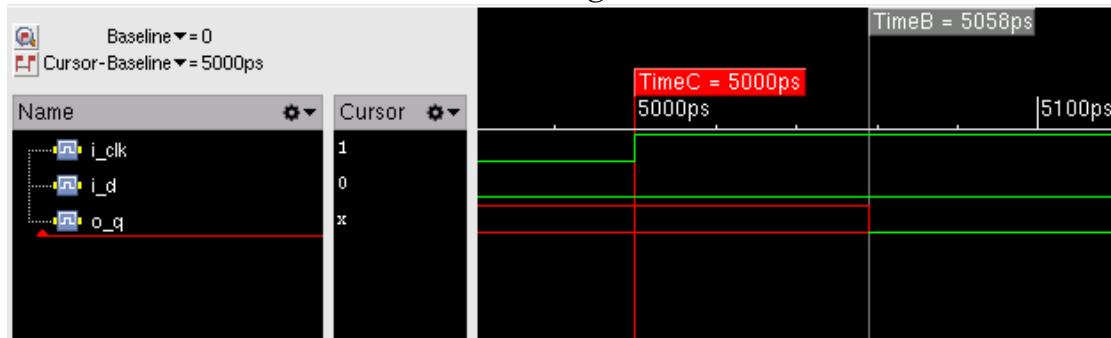
a) *fast_vdd1v0_basicCells_hvt.lib*:

```
(DELAYFILE
  (SDFVERSION "OVI 3.0")
  (DESIGN      "dff")
  (DATE        "Sat Nov 22 09:33:53 +07 2025")
  (VENDOR      "Cadence, Inc.")
  (PROGRAM     "Genus(TM) Synthesis Solution")
  (VERSION     "21.19-s055_1")
  (DIVIDER    .)
  (VOLTAGE    ::1.1)
  (PROCESS     "::1.0")
  (TEMPERATURE ::0.0)
  (TIMESCALE   1ns)
  (CELL
    (CELLTYPE "DFFQXLHVT")
    (INSTANCE o_q_reg)
    (DELAY
      (ABSOLUTE
        (PORT CK (::0.000))
        (PORT D (::0.000))
        (IOPATH CK Q (::0.051) (::0.058)))
      )
    )
    (TIMINGCHECK
      (HOLD (negedge D) (posedge CK) (::0.000))
      (HOLD (posedge D) (posedge CK) (::0.000))
      (SETUP (negedge D) (posedge CK) (::0.019))
      (SETUP (posedge D) (posedge CK) (::0.025)))
    )
  )
)
```

*- Delay:
Rise edge:*

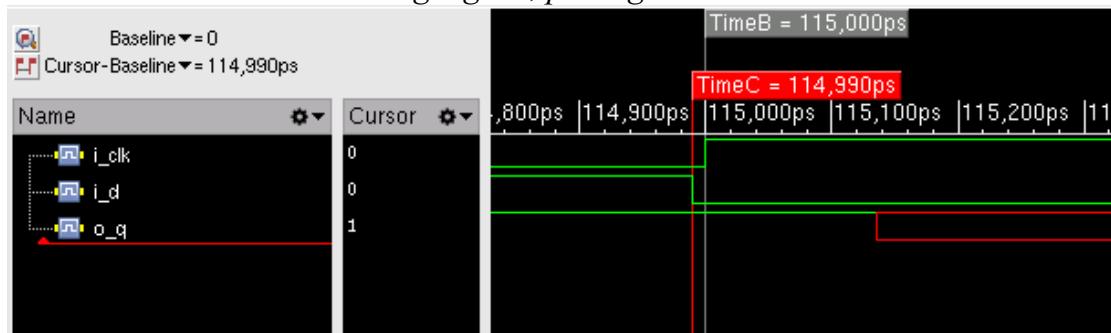


Fall edge:



Setup:

Negedge D, posedge CLK:

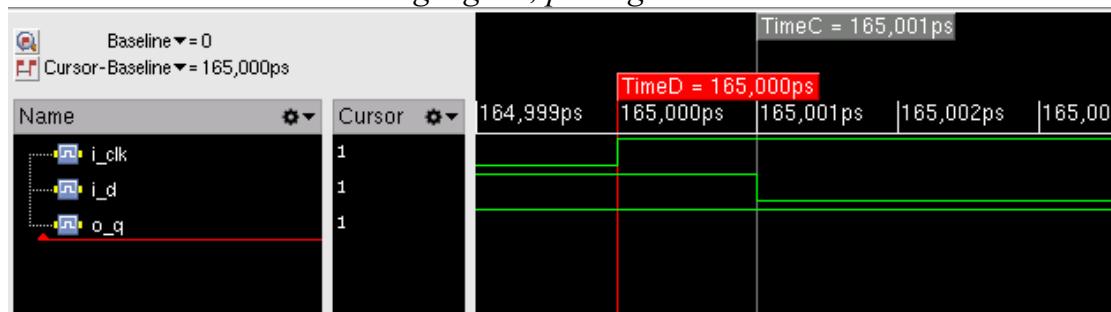


Posedge D, posedge CLK:

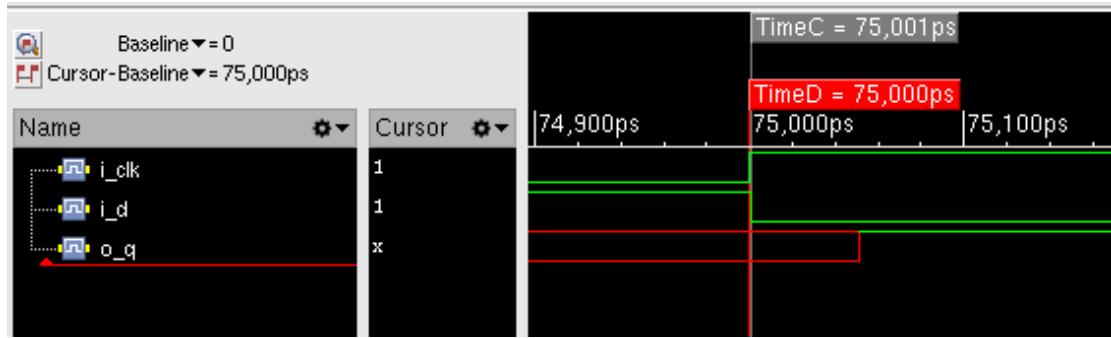


Hold:

Negedge D, posedge CLK:



Posedge D, posedge CLK:



b) fast_vdd1v2_basicCells_hvt.lib:

```

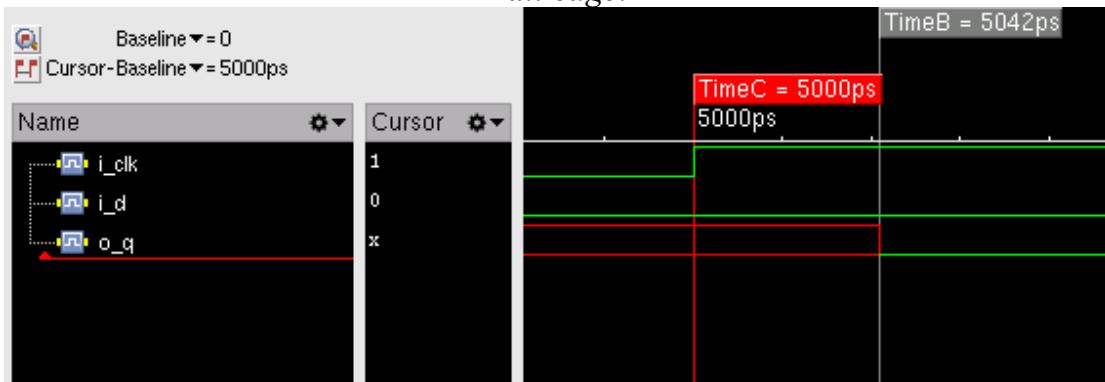
(DELAYFILE
  (SDFVERSION "OVI 3.0")
  (DESIGN "dff")
  (DATE "Sat Nov 22 14:15:15 +07 2025")
  (VENDOR "Cadence, Inc.")
  (PROGRAM "Genus(TM) Synthesis Solution")
  (VERSION "21.19-s055_1")
  (DIVIDER .)
  (VOLTAGE ::1.32)
  (PROCESS "::1.0")
  (TEMPERATURE ::0.0)
  (TIMESCALE 1ns)
  (CELL
    (CELLTYPE "DFFQXLHVT")
    (INSTANCE o_q_reg)
    (DELAY
      (ABSOLUTE
        (PORT CK (::0.000))
        (PORT D (::0.000))
        (IOPATH CK Q (::0.037) (::0.042)))
      )
    )
    (TIMINGCHECK
      (HOLD (negedge D) (posedge CK) (::0.000))
      (HOLD (posedge D) (posedge CK) (::0.000))
      (SETUP (negedge D) (posedge CK) (::0.012))
      (SETUP (posedge D) (posedge CK) (::0.016)))
    )
  )
)

```

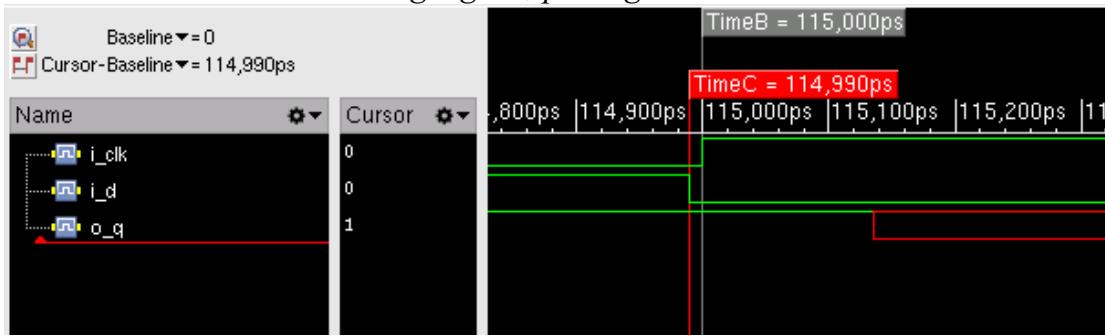
*- Delay:
Rise edge:*



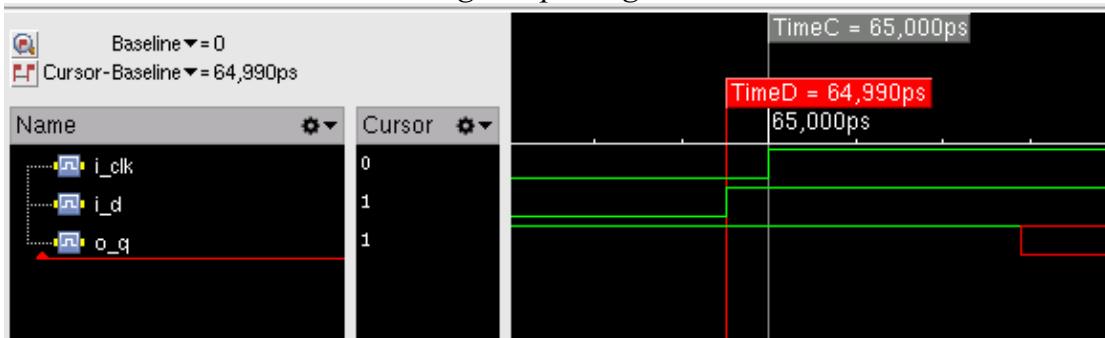
Fall edge:



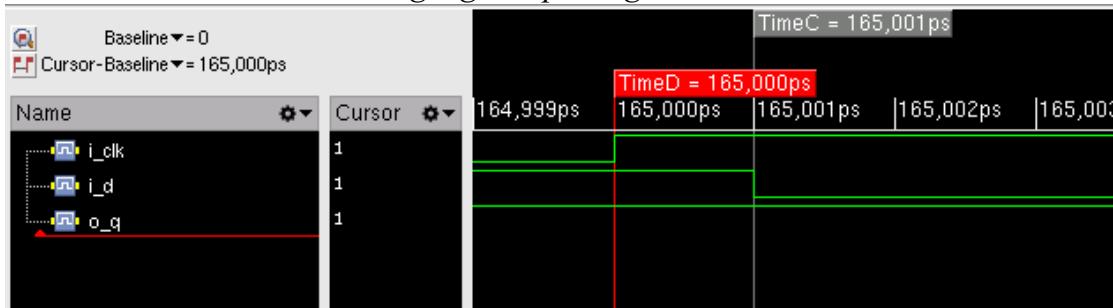
*- Setup:
Negedge D, posedge CLK:*



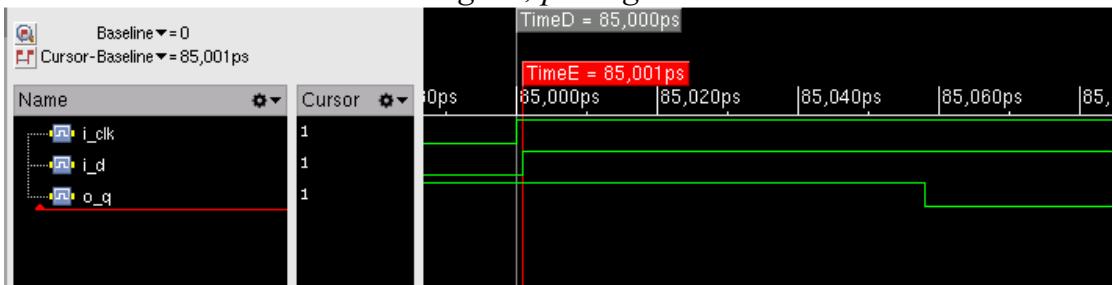
Posedge D, posedge CLK:



Hold:
- *Negedge D, posedge CLK:*



Posedge D, posedge CLK:



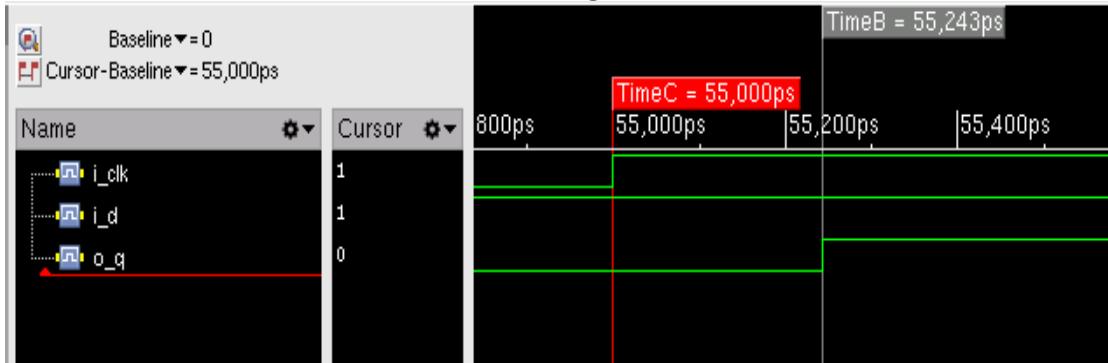
c) slow vdd1v0 basicCells hvt.lib:

```

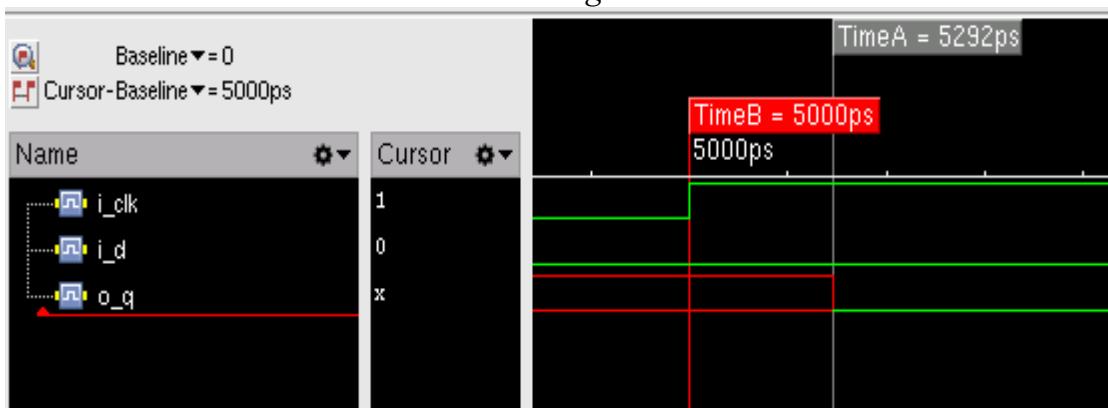
(DELAYFILE
  (SDFVERSION    "OVI 3.0")
  (DESIGN        "dff")
  (DATE          "Sat Nov 22 15:42:38 +07 2025")
  (VENDOR        "Cadence, Inc.")
  (PROGRAM       "Genus(TM) Synthesis Solution")
  (VERSION       "21.19-s055_1")
  (DIVIDER      .)
  (VOLTAGE       ::0.9)
  (PROCESS       "::1.0")
  (TEMPERATURE   ::125.0)
  (TIMESCALE     1ns)
  (CELL
    (CELLTYPE    "DFFHQX1HVT")
    (INSTANCE    o_q_reg)
    (DELAY
      (ABSOLUTE
        (PORT CK (::0.000))
        (PORT D (::0.000))
        (IOPATH CK Q (::0.243) (::0.292))
      )
    )
    (TIMINGCHECK
      (HOLD (negedge D) (posedge CK) (::0.051))
      (HOLD (posedge D) (posedge CK) (::0.000))
      (SETUP (negedge D) (posedge CK) (::0.058))
      (SETUP (posedge D) (posedge CK) (::0.177))
    )
  )
)
)

```

Delay:
Rise edge:

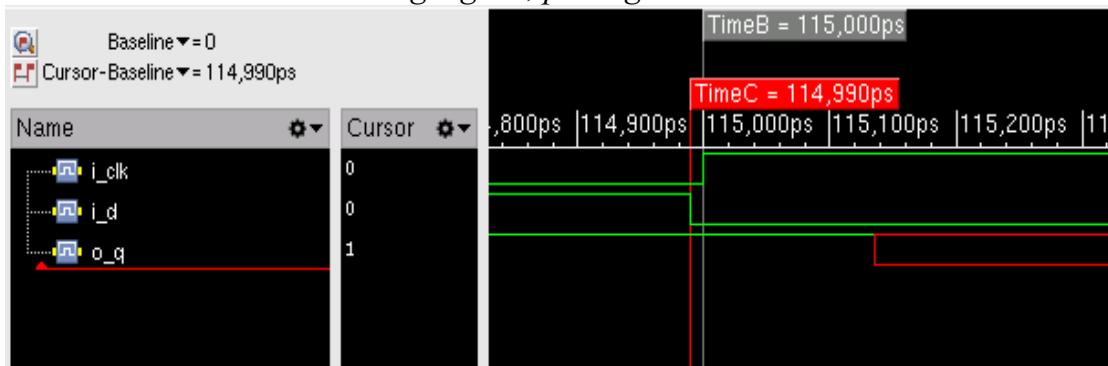


Fall edge:

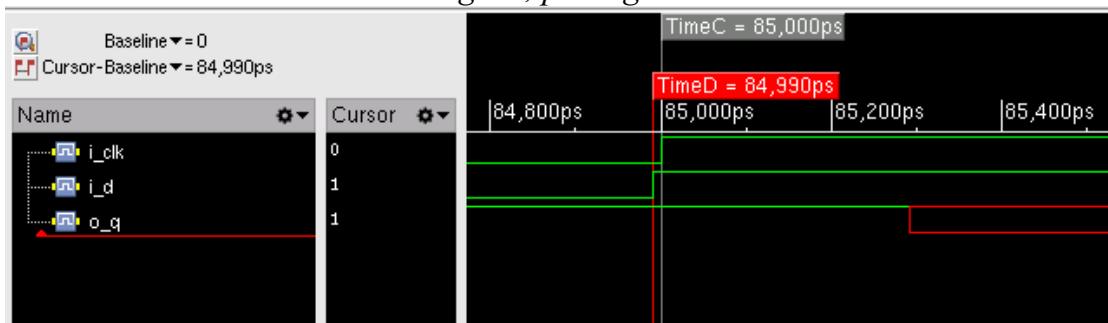


Setup:

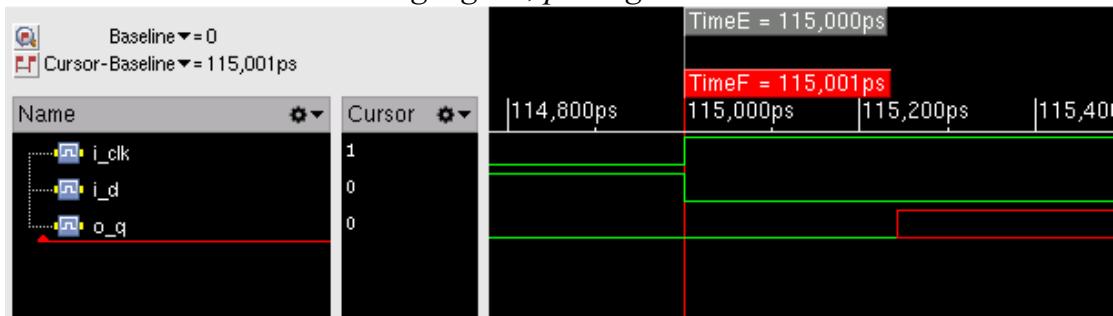
Negedge D, posedge CLK:



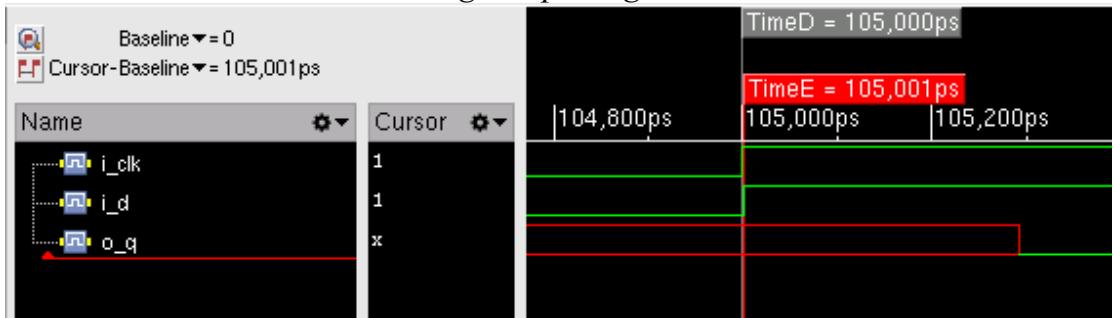
Posedge D, posedge CLK:



Hold:
Negedge D, posedge CLK:



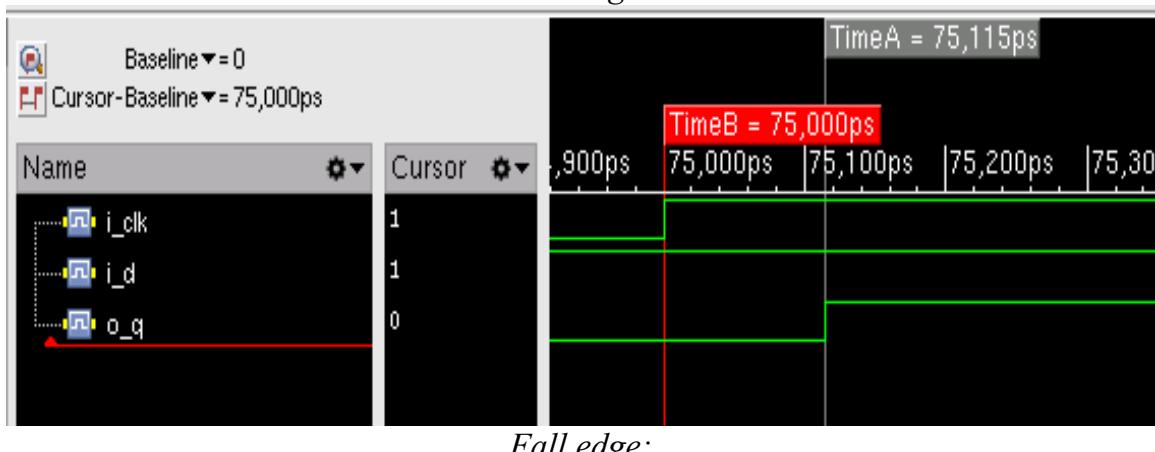
Posedge D, posedge CLK:



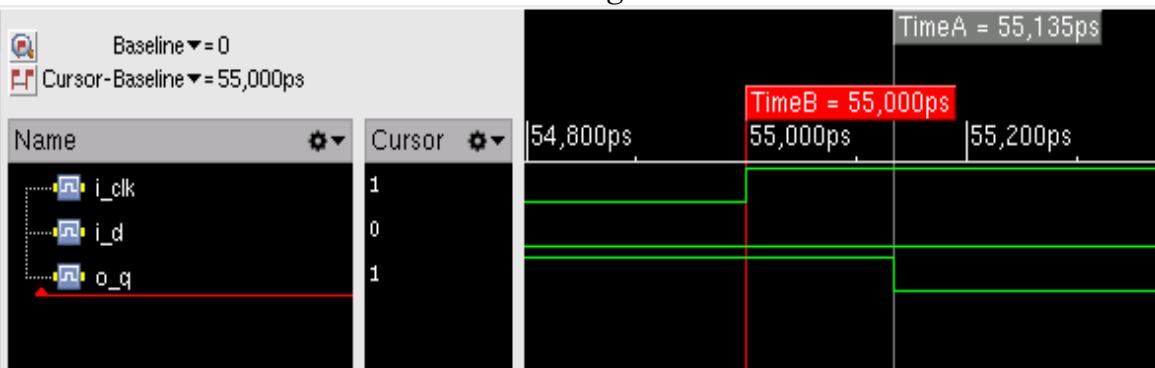
d) *slow_vdd1v2_basicCells_hvt.lib:*

```
(DELAYFILE
  (SDFVERSION "OVI 3.0")
  (DESIGN      "dff")
  (DATE        "Sat Nov 22 17:40:15 +07 2025")
  (VENDOR      "Cadence, Inc.")
  (PROGRAM     "Genus(TM) Synthesis Solution")
  (VERSION     "21.19-s055_1")
  (DIVIDER    .)
  (VOLTAGE    ::1.08)
  (PROCESS     ::1.0")
  (TEMPERATURE ::125.0)
  (TIMESCALE   1ns)
  (CELL
    (CELLTYPE "DFFHQX1HVT")
    (INSTANCE  o_q_reg)
    (DELAY
      (ABSOLUTE
        (PORT CK (::0.000))
        (PORT D  (::0.000))
        (IOPATH CK Q (::0.115) (::0.135)))
      )
    )
    (TIMINGCHECK
      (HOLD (negedge D) (posedge CK) (::0.018)|)
      (HOLD (posedge D) (posedge CK) (::0.000))
      (SETUP (negedge D) (posedge CK) (::0.026))
      (SETUP (posedge D) (posedge CK) (::0.072))
    )
  )
)
```

Delay:
Rise edge:

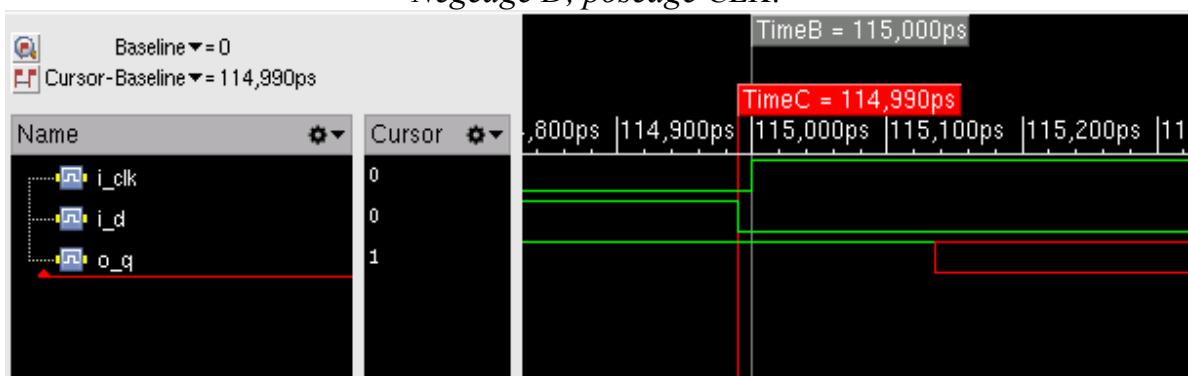


Fall edge:

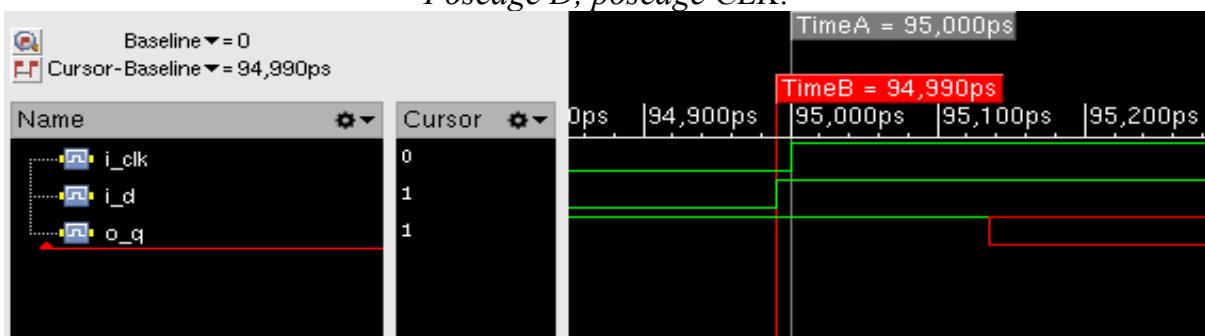


Setup:

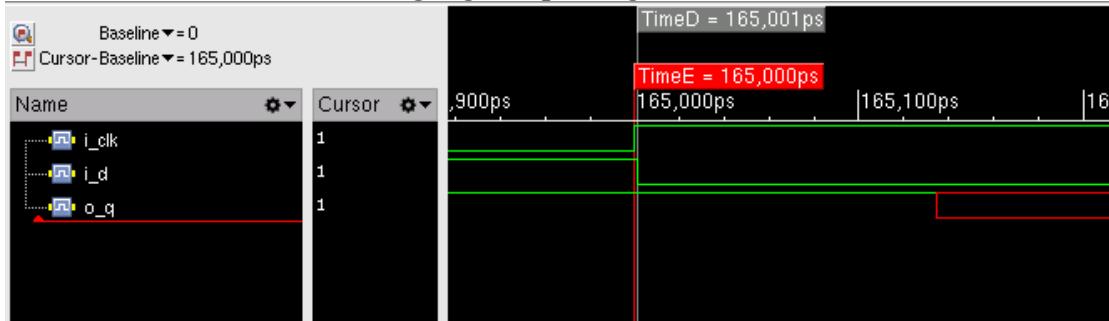
Negedge D, posedge CLK:



Posedge D, posedge CLK:



- Hold:
Negedge D, posedge CLK:



Posedge D, posedge CLK:

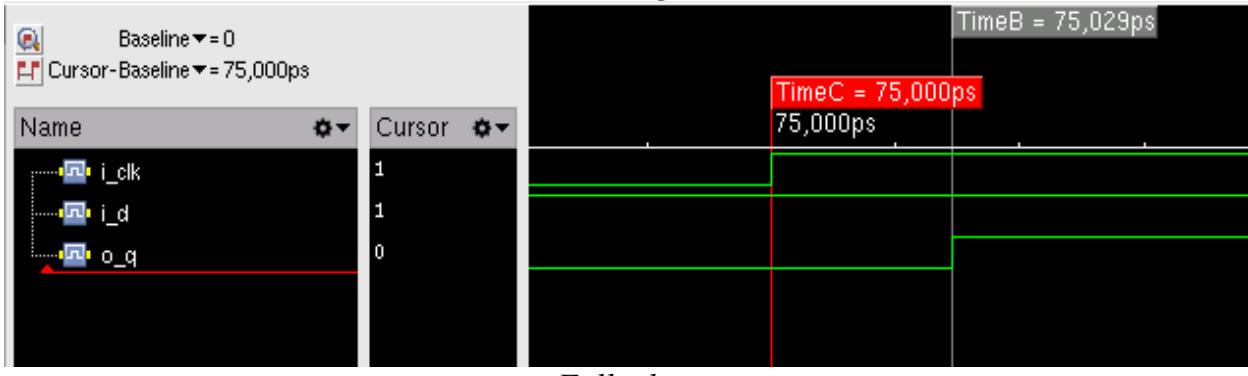


e) *fast_vdd1v0_basicCells_lvt.lib:*

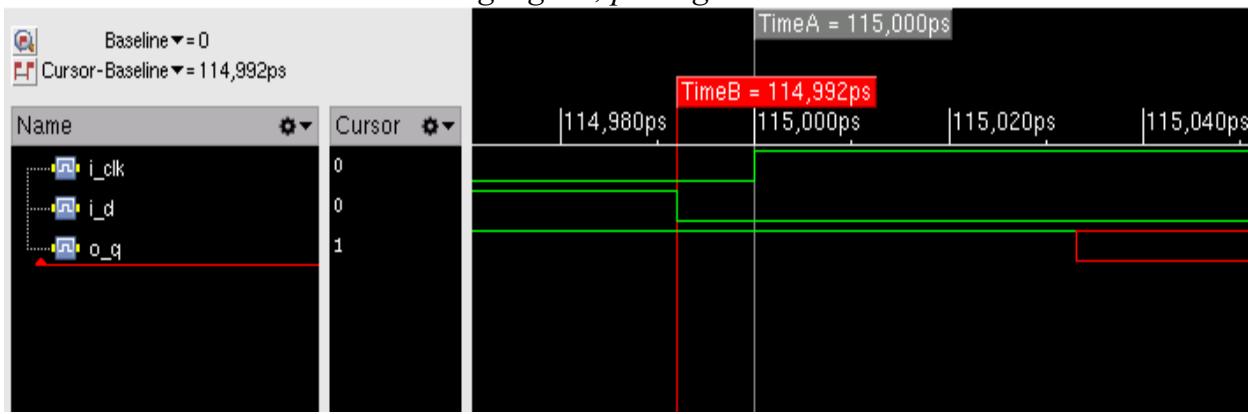
```

|(DELAYFILE
  (SDFVERSION "OVI 3.0")
  (DESIGN "dff")
  (DATE "Sat Nov 22 18:29:34 +07 2025")
  (VENDOR "Cadence, Inc.")
  (PROGRAM "Genus(TM) Synthesis Solution")
  (VERSION "21.19-s055_1")
  (DIVIDER .)
  (VOLTAGE ::1.1)
  (PROCESS ::1.0")
  (TEMPERATURE ::0.0)
  (TIMESCALE 1ns)
  (CELL
    (CELLTYPE "DFFQXLLVT")
    (INSTANCE o_q_reg)
    (DELAY
      (ABSOLUTE
        (PORT CK (::0.000))
        (PORT D (::0.000))
        (IOPATH CK Q (::0.029) (::0.033)))
      )
    )
  (TIMINGCHECK
    (HOLD (negedge D) (posedge CK) (::0.000))
    (HOLD (posedge D) (posedge CK) (::0.000))
    (SETUP (negedge D) (posedge CK) (::0.010))
    (SETUP (posedge D) (posedge CK) (::0.013))
  )
)
)
```

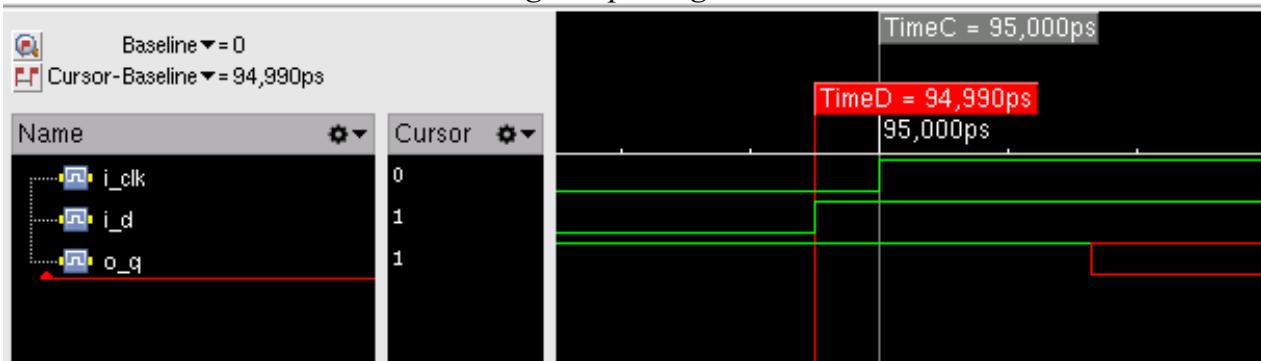
Delay:
Rise edge:



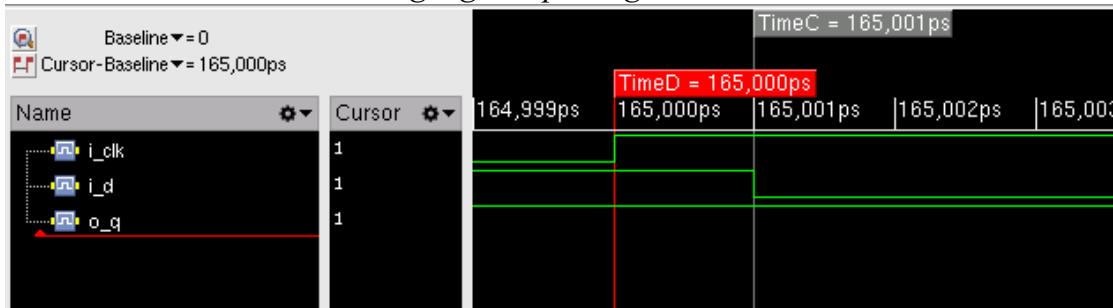
Setup:
Negedge D, posedge CLK:



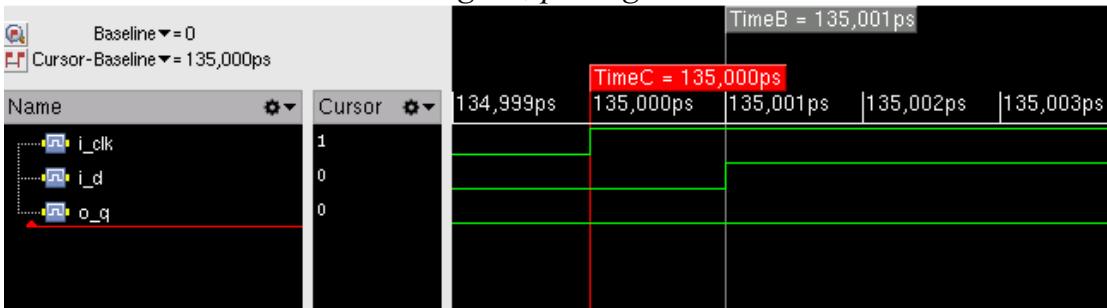
Posedge D, posedge CLK:



- Hold:
Negedge D, posedge CLK:



Posedge D, posedge CLK:



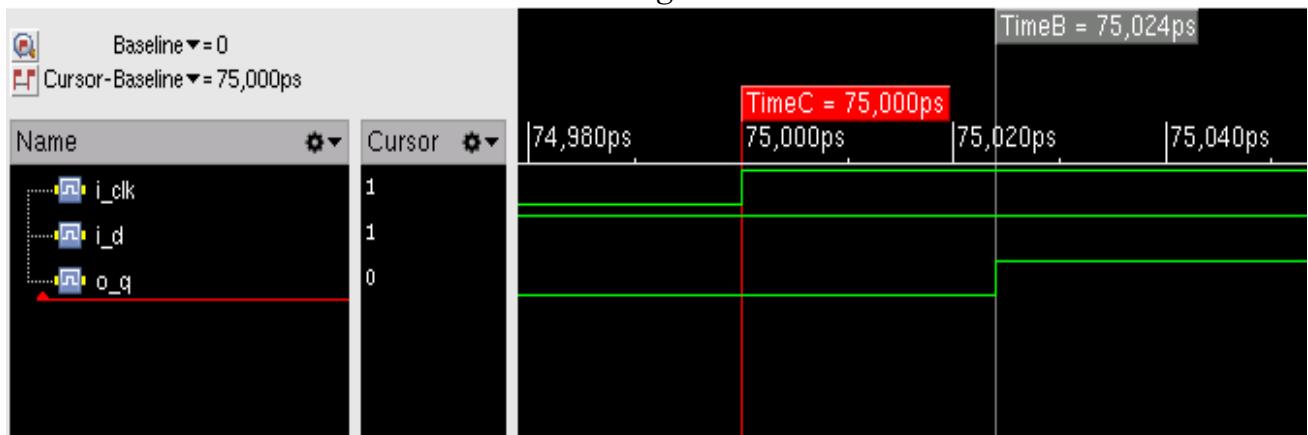
f) fast_vdd1v2_basicCells_lvt.lib:

```

(DELAYFILE
  (SDFVERSION "OVI 3.0")
  (DESIGN      "dff")
  (DATE        "Sat Nov 22 19:13:36 +07 2025")
  (VENDOR     "Cadence, Inc.")
  (PROGRAM    "Genus(TM) Synthesis Solution")
  (VERSION    "21.19-s055_1")
  (DIVIDER   .)
  (VOLTAGE   ::1.32)
  (PROCESS    "::1.0")
  (TEMPERATURE ::0.0)
  (TIMESCALE  1ns)
  (CELL
    (CELLTYPE "DFFQXLLVT")
    (INSTANCE o_q_reg)
    (DELAY
      (ABSOLUTE
        (PORT CK (::0.000))
        (PORT D (::0.000))
        (IOPATH CK Q (::0.024) (::0.027)))
      )
    )
    (TIMINGCHECK
      (HOLD (negedge D) (posedge CK) (::0.000))
      (HOLD (posedge D) (posedge CK) (::0.000))
      (SETUP (negedge D) (posedge CK) (::0.007))
      (SETUP (posedge D) (posedge CK) (::0.010)))
    )
  )
)

```

Delay:
Rise edge:



Fall edge:

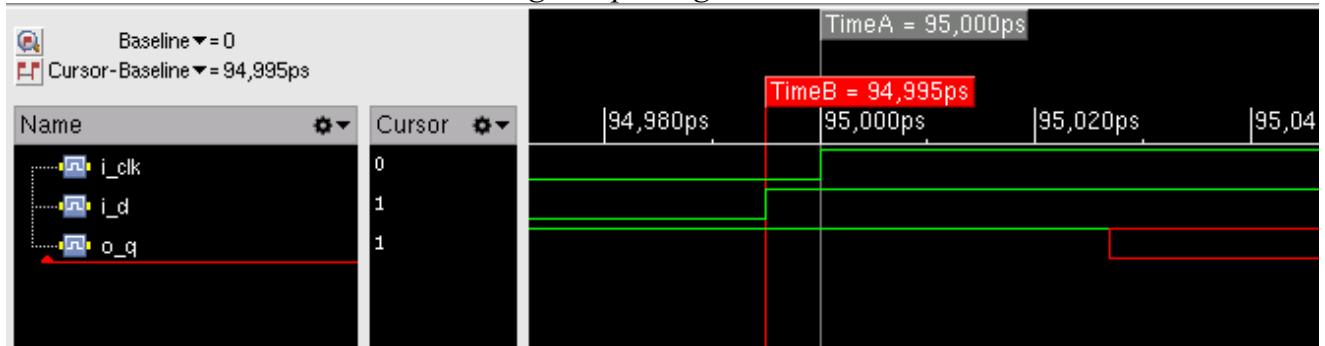


Setup:

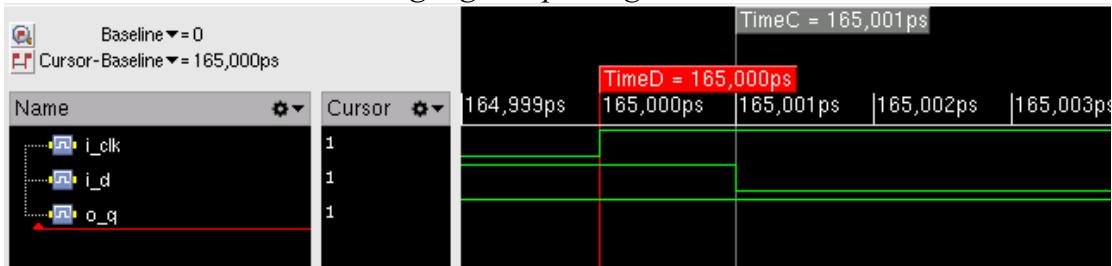
Negedge D, posedge CLK:



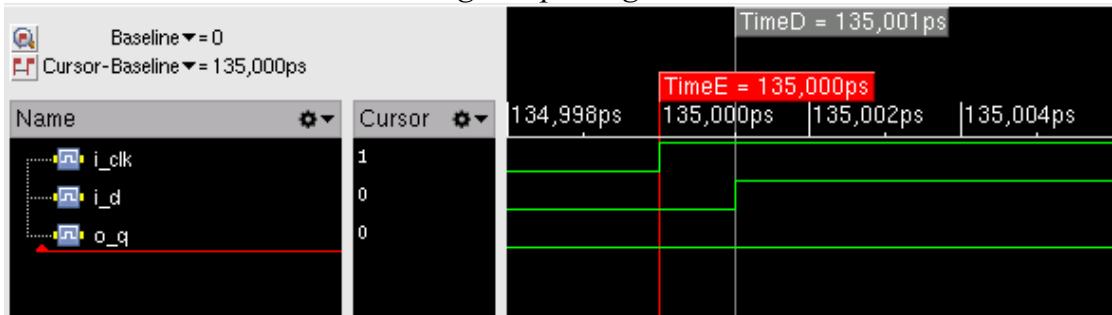
Posedge D, posedge CLK:



- Hold:
Negedge D, posedge CLK:



Posedge D, posedge CLK:



g) *slow_vdd1v0_basicCells_lvt.lib:*

```

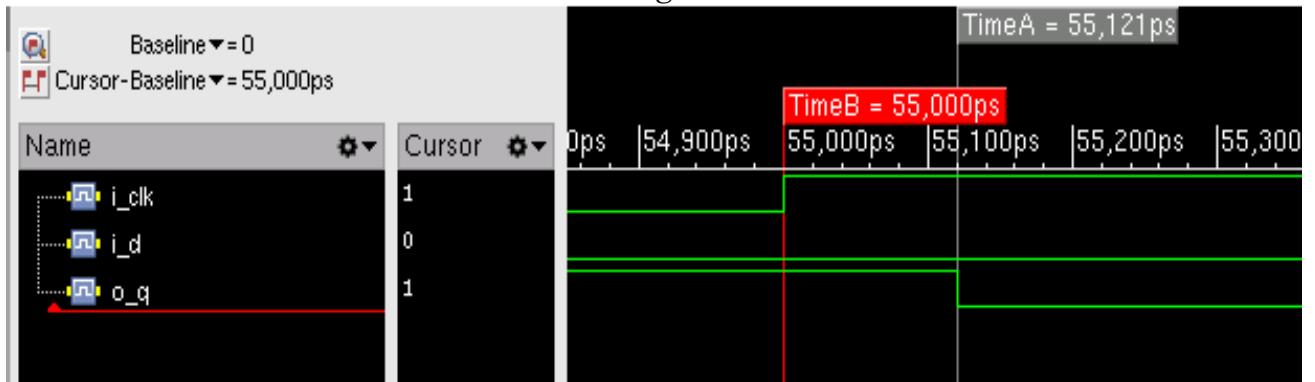
|(DELAYFILE
  (SDFVERSION "OVI 3.0")
  (DESIGN "dff")
  (DATE "Sat Nov 22 19:58:00 +07 2025")
  (VENDOR "Cadence, Inc.")
  (PROGRAM "Genus(TM) Synthesis Solution")
  (VERSION "21.19-s055_1")
  (DIVIDER .)
  (VOLTAGE ::0.9)
  (PROCESS "::1.0")
  (TEMPERATURE ::125.0)
  (TIMESCALE 1ns)
  (CELL
    (CELLTYPE "DFFHQX1LVT")
    (INSTANCE o_q_reg)
    (DELAY
      (ABSOLUTE
        (PORT CK (::0.000))
        (PORT D (::0.000))
        (IOPATH CK Q (::0.106) (::0.121)))
      )
    )
    (TIMINGCHECK
      (HOLD (negedge D) (posedge CK) (::0.014))
      (HOLD (posedge D) (posedge CK) (::0.000))
      (SETUP (negedge D) (posedge CK) (::0.029))
      (SETUP (posedge D) (posedge CK) (::0.061)))
    )
  )
)

```

Delay:
Rise edge:



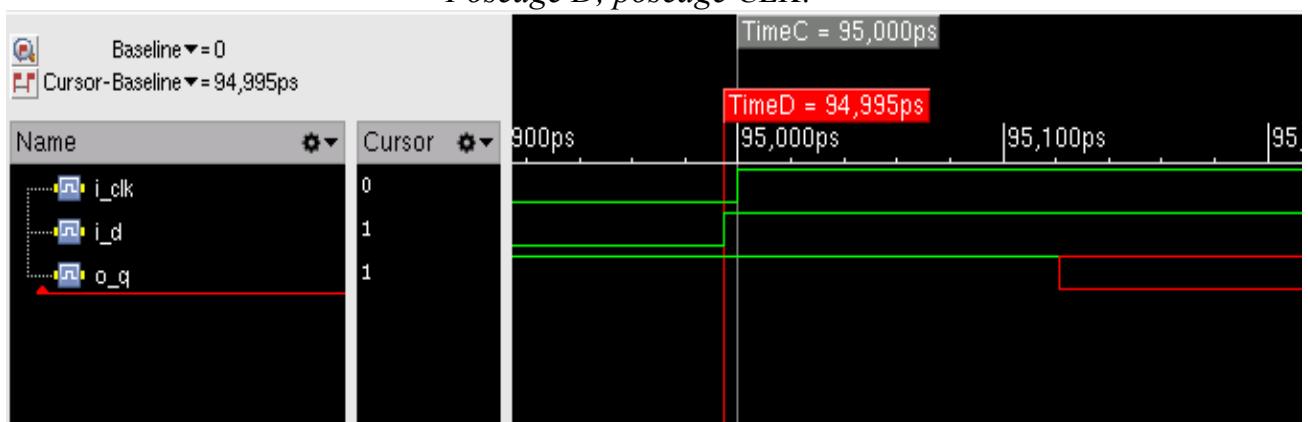
Fall edge:



Setup:
Negedge D, posedge CLK:



Posedge D, posedge CLK:



Hold:
Negedge D, posedge CLK:



Posedge D, posedge CLK:



h) slow_vdd1v2_basicCells_lvt.lib:

```

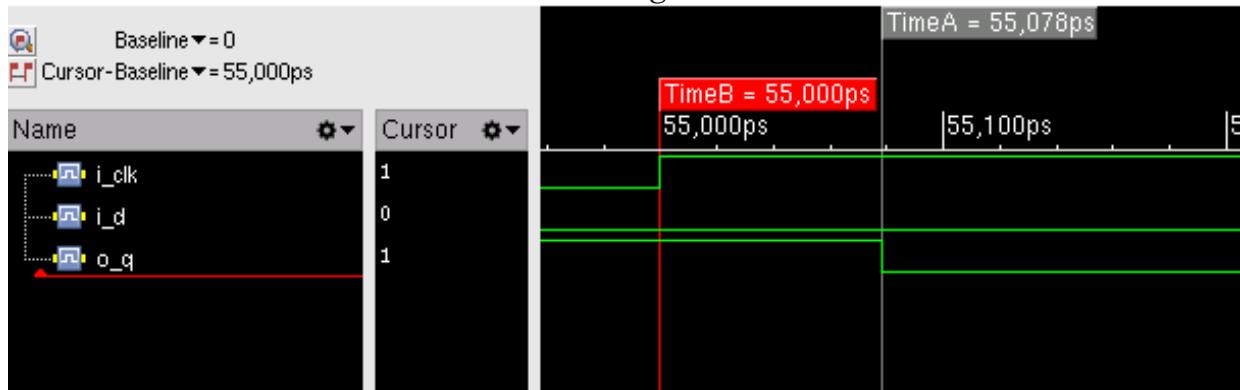
(DELAYFILE
  (SDFVERSION "OVI 3.0")
  (DESIGN "dff")
  (DATE "Sat Nov 22 21:50:08 +07 2025")
  (VENDOR "Cadence, Inc.")
  (PROGRAM "Genus(TM) Synthesis Solution")
  (VERSION "21.19-s055_1")
  (DIVIDER .)
  (VOLTAGE ::1.08)
  (PROCESS "::1.0")
  (TEMPERATURE ::125.0)
  (TIMESCALE 1ns)
  (CELL
    (CELLTYPE "DFFHQX1LVT")
    (INSTANCE o_q_reg)
    (DELAY
      (ABSOLUTE
        (PORT CK (::0.000))
        (PORT D (::0.000))
        (IOPATH CK Q (::0.068) (::0.078)))
      )
    )
    (TIMINGCHECK
      (HOLD (negedge D) (posedge CK) (::0.005))
      (HOLD (posedge D) (posedge CK) (::0.000))
      (SETUP (negedge D) (posedge CK) (::0.016))
      (SETUP (posedge D) (posedge CK) (::0.036)))
    )
  )
)

```

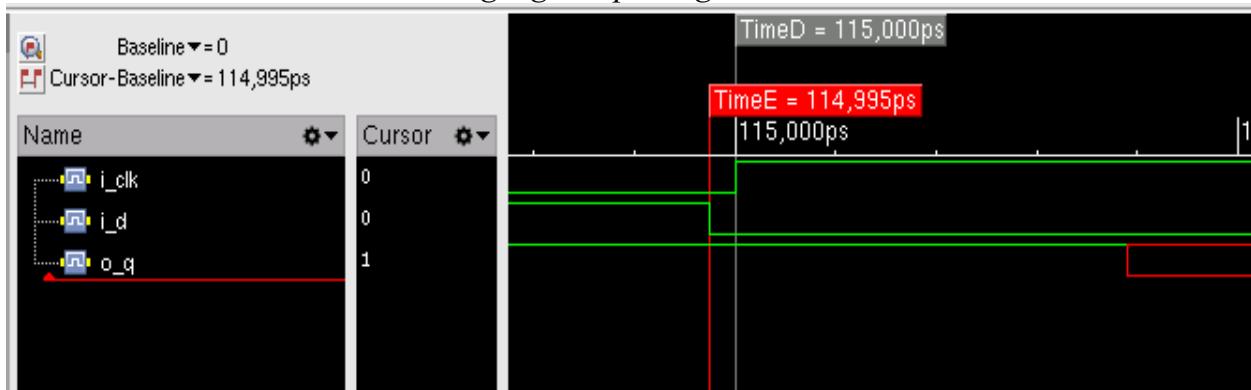
Delay:
Rise edge:



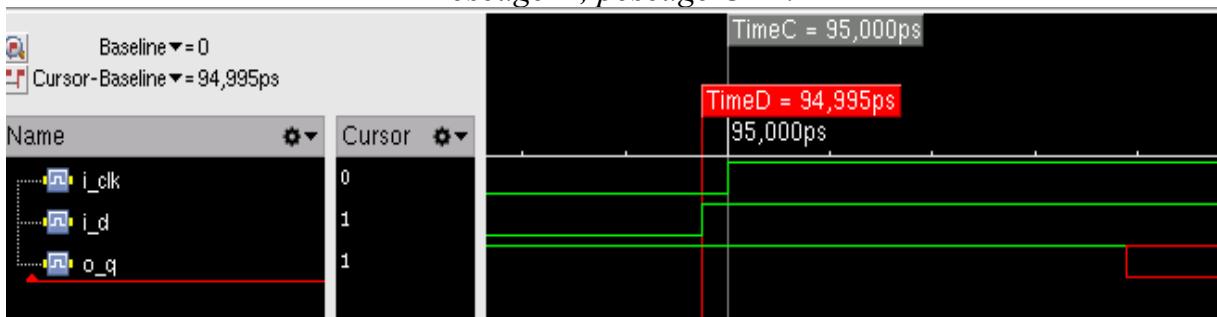
Fall edge:



Setup:
Negedge D, posedge CLK:



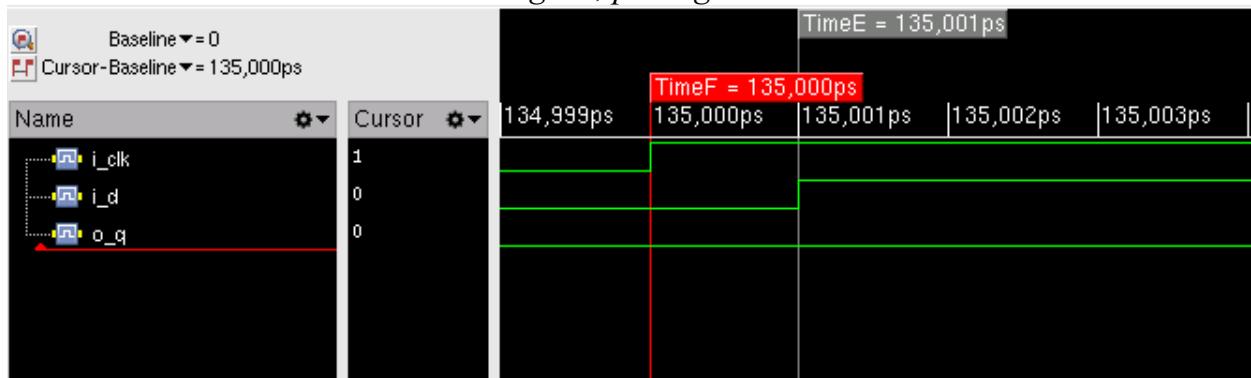
Posedge D, posedge CLK:



Hold:
Negedge D, posedge CLK:



Posedge D, posedge CLK:



IV. THÍ NGHIỆM 4

Trong thí nghiệm 4, đối tượng khảo sát là một mạch tuần tự đơn giản – bộ đếm tăng (up-counter) 2 bit được mô tả ở mức RTL như sau:

```
//////////
```

```
module ex4 (
    input logic clk,
    input logic rst,
    output logic [1:0] Q
);
    always_ff @ (posedge clk) begin
        if (rst) Q <= 2'b00;
        else Q <= Q + 2'b01;
    end
endmodule
```

```
//////////
```

Chức năng của mạch khảo sát: Mạch ex4 là một bộ đếm tăng (up-counter) 2 bit, hoạt động dựa trên xung clock và tín hiệu reset đồng bộ. Chức năng mong muốn của mạch được mô tả như sau:

- Khi tín hiệu $rst = 1$ tại cạnh lên của clk , ngõ ra Q được đưa về giá trị $2'b00$.
- Khi $rst = 0$, cứ mỗi cạnh lên của clk thì giá trị Q tăng thêm 1 theo modulo 4, theo chuỗi tuần hoàn: $00 \rightarrow 01 \rightarrow 10 \rightarrow 11 \rightarrow 00 \rightarrow \dots$

Mục tiêu của thí nghiệm

Thí nghiệm 4 được thực hiện với các mục tiêu chính sau:

+ Tổng hợp (synthesize) mạch RTL sang netlist gate-level

- Sử dụng thư viện công nghệ slow_vdd1v2_basicCells_hvt.lib.
- Thực hiện tổng hợp không sử dụng bất kỳ ràng buộc thời gian nào (không dùng file .sdc), tức là công cụ tự lựa chọn cấu trúc mạch mà không bị ép bởi yêu cầu timing cụ thể.

+ Quan sát và so sánh dạng sóng (waveform) ở hai mức mô phỏng

- Mô phỏng ở mức RTL (không có trễ) để thu được kết quả chuẩn (golden) về mặt chức năng của mạch đếm.
- Mô phỏng ở mức gate-level có back-annotate SDF với ba giá trị chu kỳ clock: $T = 50$ ns; $T = 100$ ns và $T = 200$ ns. Từ đó, so sánh dạng sóng thu được với trường hợp RTL không trễ.

- + Phân tích ảnh hưởng của trễ lan truyền (delay) đến hành vi mạch
 - o Dựa trên các waveform ở mức gate-level, đánh giá mối quan hệ giữa chu kỳ clock T và tổng trễ đường dữ liệu.
 - o Xem xét với mỗi giá trị T = 50 ns, 100 ns, 200 ns thì mạch còn hoạt động đúng hay không, cụ thể là Q có còn đếm đúng theo chuỗi chuẩn hay bắt đầu xuất hiện sai lệch.

Thông qua thí nghiệm này, ta có thể:

- + Thấy rõ sự khác biệt giữa mô phỏng RTL và mô phỏng gate-level có trễ:
 - o RTL cho kết quả lý tưởng, không có trễ.
 - o Gate-level cho thấy ảnh hưởng của delay trong FF và công logic (theo thư viện công nghệ và file SDF).
- + Quan sát ảnh hưởng của corner “slow” trong thư viện slow_vdd1v2_basicCells_hvt.lib đến tốc độ hoạt động của mạch (do cell chậm hơn, trễ lớn hơn).
- + Hiểu rõ lý do tại sao khi thiết kế mạch số, ta cần so sánh chu kỳ clock T với tổng trễ đường dữ liệu (timing) để đảm bảo mạch vẫn hoạt động đúng chức năng, không vi phạm các ràng buộc setup/hold.

4.1 Sơ đồ mạch (schematic) và file netlist

4.1.1 Quy trình tổng hợp

Trong thí nghiệm này, mã RTL của mạch up-counter 2 bit (module ex4) được tổng hợp sang dạng gate-level netlist bằng công cụ Cadence Genus. Thư viện công nghệ sử dụng là slow_vdd1v2_basicCells_hvt.lib, tương ứng corner “slow”, điện áp nguồn 1.2 V và transistor loại HVT (High-Vt), mô tả trường hợp mạch hoạt động chậm nhất.

Quy trình tổng hợp được thực hiện theo các bước chính:

- Thiết lập đường dẫn đến thư viện công nghệ và thư mục chứa mã RTL.
- Đọc thư viện:
read_libs slow_vdd1v2_basicCells_hvt.lib
- Đọc mã RTL:
read_hdl -sv ex4.sv
- Thực hiện elaborate thiết kế ex4 để tạo mô hình logic trung gian.
- Chạy các bước tổng hợp: syn_generic và syn_map (có thể thêm syn_opt nếu cần tối ưu thêm).
- Sau khi tổng hợp xong:
 - o Ghi netlist gate-level ra file ex4_netlist_s1v2h.sv.
 - o Ghi file SDF chứa thông tin trễ của các cell ra file ex4_delay_s1v2h.sdf.

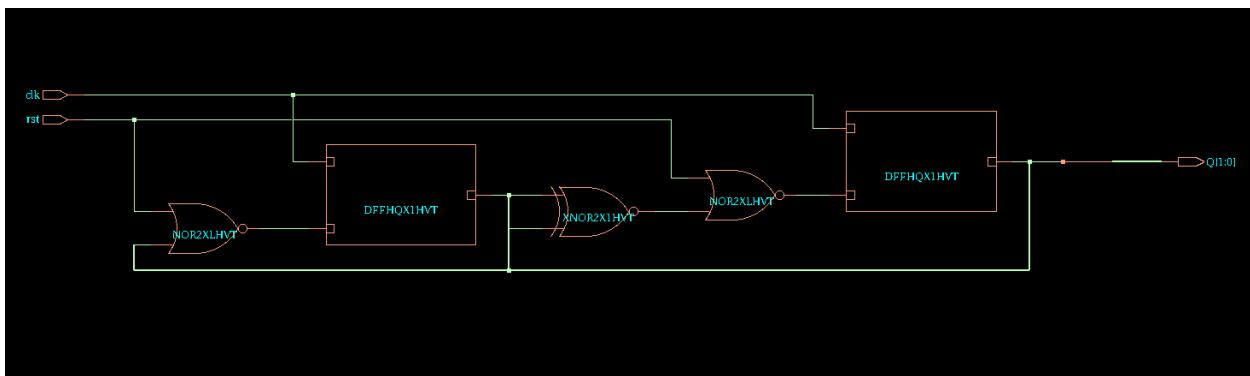
- Sinh thêm các báo cáo về diện tích, công suất và timing (area, power, timing report) nếu cần cho phân tích.

Trong toàn bộ quá trình này, không sử dụng bất kỳ ràng buộc thời gian nào (không dùng file .sdc), đúng với yêu cầu của đề bài. Điều này có nghĩa là công cụ không bị ép bởi các mục tiêu timing cụ thể, mà chỉ tổng hợp dựa trên cấu trúc logic và thư viện công nghệ.

4.1.2 Netlist sau tổng hợp và mô tả sơ đồ mạch

Sơ đồ mạch sau tổng hợp được công cụ Genus sinh ra ở mức gate-level, sử dụng các cell trong thư viện slow_vdd1v2_basicCells_hvt.lib.

Hình 4.1.2 trình bày sơ đồ mạch (schematic) của module ex4 sau khi tổng hợp.



Hình 4.1.2 Sơ đồ mạch gate-level của ex4

Từ sơ đồ schematic trên và file netlist, ta có thể mô tả cấu trúc mạch như sau:

- Mạch gồm hai phần tử chốt D (D flip-flop) loại DFFHQX1HVT:
 - \Q_reg[0]: lưu trữ bit thấp Q[0].
 - \Q_reg[1]: lưu trữ bit cao Q[1].
 Cả hai FF đều sử dụng cùng một tín hiệu clock clk.
- Phần logic tổ hợp bao gồm:
 - Một cổng XNOR hai ngõ vào XNOR2X1HVT (g22_5107) với ngõ vào là Q[0] và Q[1], ngõ ra là n_1. Cổng này đóng vai trò tạo tín hiệu trung gian phục vụ việc xác định trạng thái kế tiếp của bit cao Q[1] trong phép cộng 1.
 - Hai cổng NOR hai ngõ vào NOR2XLHVT:
 - g23_6260: nhận ngõ vào là rst và Q[0], ngõ ra là n_0. Tín hiệu n_0 được đưa tới ngõ D của FF \Q_reg[0]. Cổng NOR này vừa xử lý reset, vừa xác định bit thấp của trạng thái kế tiếp.
 - g20_2398: nhận ngõ vào là rst và n_1, ngõ ra là n_2. Tín hiệu n_2 được đưa tới ngõ D của FF \Q_reg[1]. Cổng NOR này đảm nhiệm việc reset bit cao và lựa chọn trạng thái tiếp theo cho Q[1].

File netlist tương ứng của mạch sau tổng hợp được liệt kê trong đoạn mã dưới đây:

```
///////////
// Generated by Cadence Genus(TM) Synthesis Solution 21.19-s055_1
// Generated on: Nov 20 2025 23:28:59 +07 (Nov 20 2025 16:28:59 UTC)
// Verification Directory fv/ex4

module ex4 (clk, rst, Q);

    input clk, rst;
    output [1:0] Q;

    wire clk, rst;
    wire [1:0] Q;
    wire n_0, n_1, n_2;

    DFFHQX1HVT \Q_reg[1] (.CK(clk), .D(n_2), .Q(Q[1]));
    NOR2XLHVT g20_2398 (.A(rst), .B(n_1), .Y(n_2));
    DFFHQX1HVT \Q_reg[0] (.CK(clk), .D(n_0), .Q(Q[0]));
    XNOR2X1HVT g22_5107 (.A(Q[0]), .B(Q[1]), .Y(n_1));
    NOR2XLHVT g23_6260 (.A(rst), .B(Q[0]), .Y(n_0));

Endmodule
/////////
```

Dựa trên schematic và netlist:

- Khi $rst = 1$, hai công NOR nhận $rst = 1$ nên các tín hiệu n_0 và n_2 đều bằng 0. Tại cạnh lên clock tiếp theo, hai FF chốt giá trị $D = 0$, do đó $Q[1:0] = 00$, bộ đếm được reset về 0.
- Khi $rst = 0$, tổ hợp XNOR + hai công NOR tạo ra bộ giá trị n_0, n_2 sao cho tại mỗi cạnh lên clock, hai FF cập nhật giá trị Q tương ứng với Q hiện tại cộng thêm 1 (modulo 4).

Như vậy, hình 4.2.1 cùng với netlist cho thấy rõ: mạch đếm 2 bit ban đầu đã được hiện thực hóa bằng hai FF và mạng logic gồm một công XNOR và hai công NOR, đúng với chức năng mô tả ở mức RTL.

4.2 Dạng sóng (waveform)

4.2.1 File trễ thời gian (SDF) và các thông số chính

Sau khi tổng hợp mạch ex4, công cụ Genus sinh ra file trễ thời gian ở dạng SDF với tên ex4_delay_s1v2h.sdf. Đây là file dùng để mô tả trễ lan truyền và ràng buộc thời gian của các cell trong netlist, phục vụ cho mô phỏng gate-level có trễ (back-annotate SDF).

Nội dung file SDF sinh ra như sau:

```
//////////
```

```
(DELAYFILE
```

```
  (SDFVERSION "OVI 3.0")
```

```
  (DESIGN    "ex4")
```

```
  (DATE      "Thu Nov 20 23:28:59 +07 2025")
```

```
  (VENDOR    "Cadence, Inc.")
```

```
  (PROGRAM   "Genus(TM) Synthesis Solution")
```

```
  (VERSION   "21.19-s055_1")
```

```
  (DIVIDER   .)
```

```
  (VOLTAGE   ::1.08)
```

```
  (PROCESS   "::1.0")
```

```
  (TEMPERATURE ::125.0)
```

```
  (TIMESCALE  1ns)
```

```
  (CELL
```

```
    (CELLTYPE "DFFHQX1HVT")
```

```
    (INSTANCE Q_reg\[1\])
```

```
    (DELAY
```

```
      (ABSOLUTE
```

```
        (PORT CK (::0.000))
```

```
        (PORT D (::0.000))
```

```
        (IOPATH CK Q (::0.119) (::0.140))
```

```
)  
)  
(TIMINGCHECK  
  (HOLD (negedge D) (posedge CK) (":0.009))  
  (HOLD (posedge D) (posedge CK) (":0.000))  
  (SETUP (negedge D) (posedge CK) (":0.037))  
  (SETUP (posedge D) (posedge CK) (":0.087))  
)  
)
```

```
(CELL  
  (CELLTYPE "NOR2XLHVT")  
  (INSTANCE g20_2398)  
  (DELAY  
    (ABSOLUTE  
      (PORT A (":0.000))  
      (PORT B (":0.000))  
      (IOPATH A Y (":0.028) (":0.022))  
      (IOPATH B Y (":0.030) (":0.025))  
    )  
  )  
)
```

```
(CELL  
  (CELLTYPE "DFFHQX1HVT")  
  (INSTANCE Q_reg\[0\])  
  (DELAY
```

```
(ABSOLUTE
  (PORT CK (":0.000))
  (PORT D (":0.000))
  (IOPATH CK Q (":0.119) (":0.140))
)
)

(TIMINGCHECK
  (HOLD (negedge D) (posedge CK) (":0.009))
  (HOLD (posedge D) (posedge CK) (":0.000))
  (SETUP (negedge D) (posedge CK) (":0.037))
  (SETUP (posedge D) (posedge CK) (":0.087))
)
)

(CELL
  (CELLTYPE "XNOR2X1HVT")
  (INSTANCE g22_5107)
  (DELAY
    (ABSOLUTE
      (PORT A (":0.000))
      (PORT B (":0.000))
      (IOPATH A Y (":0.131) (":0.110))
      (IOPATH B Y (":0.109) (":0.077))
    )
  )
)
```

```

(CELL
  (CELLTYPE "NOR2XLHVT")
  (INSTANCE g23_6260)
  (DELAY
    (ABSOLUTE
      (PORT A (":0.000))
      (PORT B (":0.000))
      (IOPATH A Y (":0.028) (":0.022))
      (IOPATH B Y (":0.031) (":0.024))
    )
  )
)
)
/////////////////

```

Giải thích các thông số

a) Cell DFFHQX1HVT – các FF Q_reg[1] và Q_reg[0]

Ví dụ khôi:

```
/////////////////
```

```

(CELL
  (CELLTYPE "DFFHQX1HVT")
  (INSTANCE Q_reg\[1\])
  (DELAY
    (ABSOLUTE
      (PORT CK (":0.000))
      (PORT D (":0.000))
      (IOPATH CK Q (":0.119) (":0.140))
    )
  )
)

```

```

)
(TIMINGCHECK
  (HOLD (negedge D) (posedge CK) (":0.009))
  (HOLD (posedge D) (posedge CK) (":0.000))
  (SETUP (negedge D) (posedge CK) (":0.037))
  (SETUP (posedge D) (posedge CK) (":0.087))
)
)
///////////

```

- CELLCODE "DFFHQX1HVT": tên cell của flip-flop D trong thư viện.
- INSTANCE Q_reg[1]: tên instance cụ thể trong netlist, đây là FF lưu bit Q[1].

Trong khối DELAY:

- IOPATH CK Q (":0.119) (":0.140):
 - Trễ từ CK → Q khi Q lên (rise) là 0,119 ns.
 - Trễ từ CK → Q khi Q xuống (fall) là 0,140 ns.
 Đây chính là clock-to-Q delay của FF ở corner slow này.

Trong khối TIMINGCHECK:

- SETUP (posedge D) (posedge CK) (":0.087): thời gian setup, D phải ổn định ít nhất 0,087 ns trước cạnh lên CK.
- SETUP (negedge D) (posedge CK) (":0.037): setup cho cạnh xuống D.
- HOLD (negedge D) (posedge CK) (":0.009): thời gian hold tối thiểu sau cạnh lên CK cho cạnh xuống D là 0,009 ns.
- HOLD (posedge D) (posedge CK) (":0.000): hold cho cạnh lên D là 0 ns.

FF Q_reg[0] có khối SDF tương tự, chỉ khác INSTANCE là Q_reg[0].

b) Các cổng NOR2XLHVT – g20_2398 và g23_6260

```

///////////

```

```

(CELL
  (CELLCODE "NOR2XLHVT")

```

```

(INSTANCE g20_2398)
(DELAY
  (ABSOLUTE
    (PORT A (::0.000))
    (PORT B (::0.000))
    (IOPATH A Y (::0.028) (::0.022))
    (IOPATH B Y (::0.030) (::0.025))
  )
)
)
/////////////////

```

- CELLCODE "NOR2XLHVT": cỗng NOR 2 ngõ vào, loại HVT, kích thước nhỏ.
- INSTANCE g20_2398: tên instance NOR trong netlist.

Trong các dòng IOPATH:

- IOPATH A Y (::0.028) (::0.022):
 - A → Y: trễ khi Y lên $\approx 0,028$ ns, khi Y xuống $\approx 0,022$ ns.
- IOPATH B Y (::0.030) (::0.025):
 - B → Y: trễ khi Y lên $\approx 0,030$ ns, khi Y xuống $\approx 0,025$ ns.

Cell g23_6260 cũng là NOR2XLHVT với các giá trị rất gần (0,028–0,031 ns), dùng để sinh tín hiệu n_0.

c) Cỗng XNOR2X1HVT – g22_5107

```
/////////////////
```

```

(CELL
  (CELLTYPE "XNOR2X1HVT")
  (INSTANCE g22_5107)
  (DELAY
    (ABSOLUTE

```

```

(PORT A (::0.000))
(PORT B (::0.000))
(IOPATH A Y (::0.131) (::0.110))
(IOPATH B Y (::0.109) (::0.077))
)
)
)
///////////

```

- CELLCODE "XNOR2X1HVT": cổng XNOR hai ngõ vào.
- INSTANCE g22_5107: instance XNOR trong netlist, dùng để tạo tín hiệu n_1.

Các IOPATH:

- A → Y: trễ khoảng 0,131 ns (rise) và 0,110 ns (fall).
- B → Y: trễ khoảng 0,109 ns (rise) và 0,077 ns (fall).

Những giá trị này thể hiện rằng XNOR là cổng tương đối chậm hơn so với NOR (trễ cỡ 0,08–0,13 ns), do cấu trúc logic phức tạp hơn.

Kết luận: file ex4_delay_s1v2h.sdf mô tả đầy đủ điều kiện công nghệ (điện áp, nhiệt độ, process) và các thông số trễ quan trọng của các cell trong mạch ex4, bao gồm:

- clock-to-Q delay và các ràng buộc setup/hold của hai FF,
- trễ lan truyền qua các cổng NOR và XNOR.

Các thông số này sẽ được dùng ở các mục 4.2.x tiếp theo để phân tích chi tiết dạng sóng (waveform) khi mô phỏng gate-level có back-annotate SDF cho các giá trị T khác nhau.

4.2.2 Trường hợp khi không có trễ (mô phỏng RTL – kết quả chuẩn)

Trong trường hợp này, mạch ex4 được mô phỏng ở mức RTL. Testbench chỉ include file ex4.sv, không sử dụng netlist gate-level và không back-annotate SDF (không dùng các define GATE_LEVEL, ANNOTATION). Điều này tương ứng với mô hình lý tưởng: các phần tử không có trễ lan truyền, tín hiệu được cập nhật tức thời tại cạnh clock.

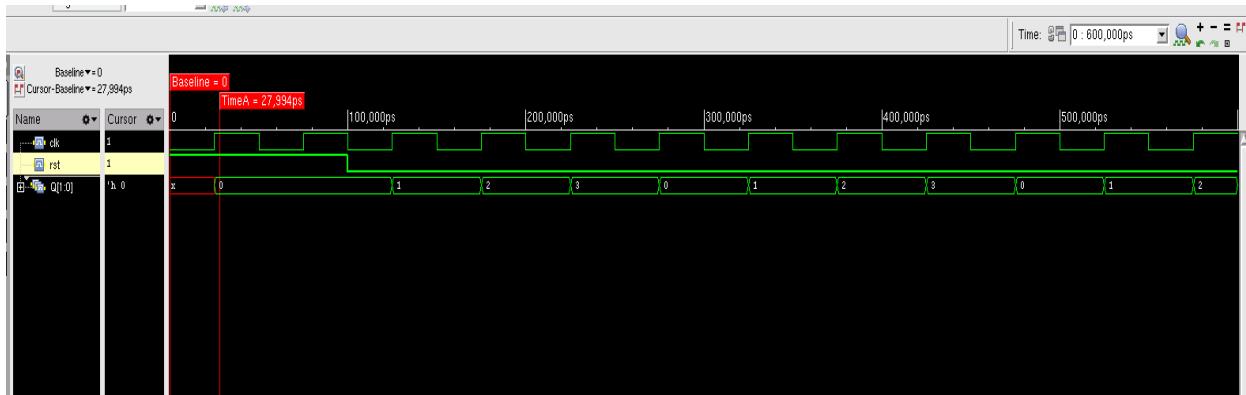
Về lý thuyết:

Ở mức RTL, khôi always_ff @ (posedge clk) sẽ cập nhật giá trị Q ngay tại thời điểm posedge clk, theo đúng mô tả:

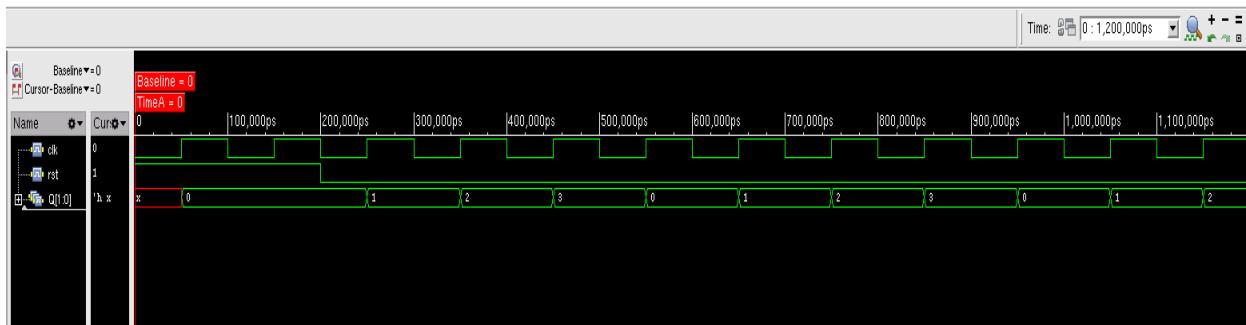
- Nếu $rst = 1$ tại cạnh lên clk , Q được gán $2'b00$.
- Nếu $rst = 0$ tại cạnh lên clk , Q được gán $Q + 1$.

Do không có trễ, trong waveform ta sẽ thấy biên clk và biên thay đổi của Q trùng nhau theo chiều dọc (không có khoảng lệch thời gian).

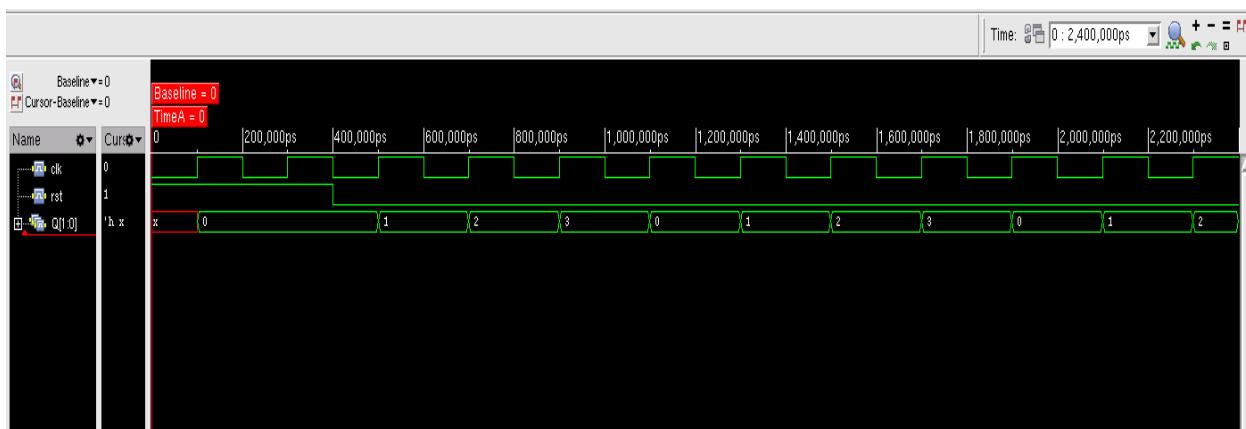
Các hình 4.2.1[a,b,c] dưới đây sẽ trình bày các dạng sóng mạch ex4 ở mức RTL, với các tín hiệu clk , rst và $Q[1:0]$ tương ứng với testbench $T=50$, $T=100$, $T=200$.



Hình 4.2.1.a Dạng sóng RTL (không có trễ) của mạch ex4 cho $T=50$



Hình 4.2.1.b Dạng sóng RTL (không có trễ) của mạch ex4 cho $T=100$



Hình 4.2.1.c Dạng sóng RTL (không có trễ) của mạch ex4 cho $T=200$

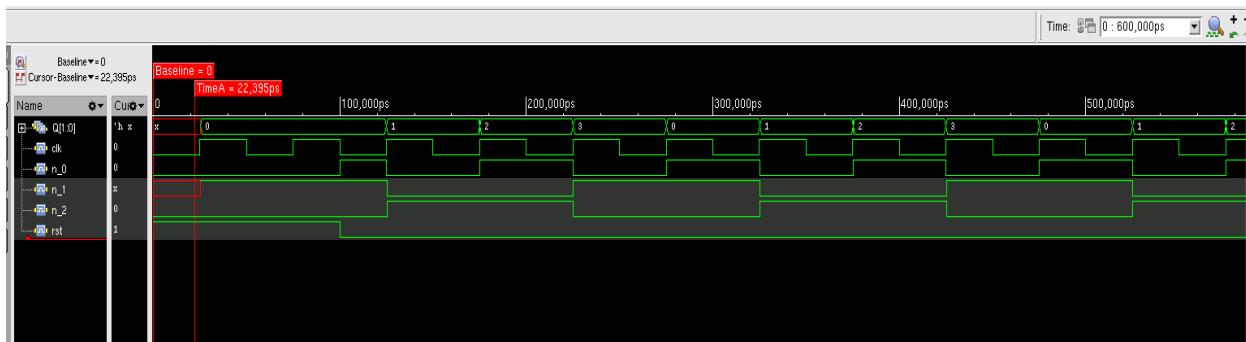
Phân tích dạng sóng:

- Trong khoảng $rst = 1$, ngõ ra Q luôn bằng $2'b00$. Điều này thể hiện chức năng reset của mạch: mỗi cạnh lên clk khi $rst = 1$ đều đưa Q về 0.
- Sau khi rst chuyển về 0, tại mỗi cạnh lên của clk , Q thay đổi ngay lập tức theo chuỗi: $00 \rightarrow 01 \rightarrow 10 \rightarrow 11 \rightarrow 00 \rightarrow 01 \rightarrow \dots$
- Không có độ trễ giữa cạnh lên clk và thời điểm Q thay đổi: biên thay đổi của Q nằm đúng tại posedge clk .

Kết luận: Dạng sóng ở mức RTL thể hiện đúng chức năng mong muốn của bộ đếm 2 bit và được coi là kết quả chuẩn (golden). Ba trường hợp $T = 50$ ns, 100 ns, 200 ns mô phỏng gate-level có trễ ở các mục sau sẽ được so sánh với kết quả này.

4.2.3 Trường hợp 1: $T=50$ (mô phỏng gate-level có SDF)

4.2.3.1 Dạng sóng tổng quát khi có trễ



Hình 4.2.3.1 Dạng sóng tổng quát của trường hợp $T=50$ khi có trễ

Hình trên mô tả dạng sóng mô phỏng gate-level của mạch ex4 với chu kỳ clock $T = 50$ ns. Netlist ex4_netlist_s1v2h.sv được nối với thư viện slow_vdd1v2_basicCells_hvt.lib và back-annotate file trễ ex4_delay_s1v2h.sdf lên instance DUT.

Quan sát ở thang thời gian $0 \rightarrow 600$ ns:

- Trong hai chu kỳ đầu, $rst = 1$ nên $Q[1:0]$ luôn bằng 00.
- Sau khi rst được hạ xuống 0, cứ mỗi cạnh lên của clk , Q tăng 1 theo chuỗi:

$$00 \rightarrow 01 \rightarrow 10 \rightarrow 11 \rightarrow 00 \rightarrow 01 \rightarrow 10 \rightarrow 11 \rightarrow \dots$$

- Chuỗi đếm hoàn toàn trùng với mô tả RTL (mục 4.2.1.a – không có trễ), chỉ khác là thời điểm Q đổi bị dời trễ so với cạnh clock.

- Sự trễ này là do trễ clock-to-Q của các flip-flop DFFHQX1HVT và trễ lan truyền qua các cổng NOR / XNOR trong SDF (IOPATH CK→Q khoảng 0,119–0,140 ns; IOPATH của các cổng logic khoảng 0,02–0,13 ns).

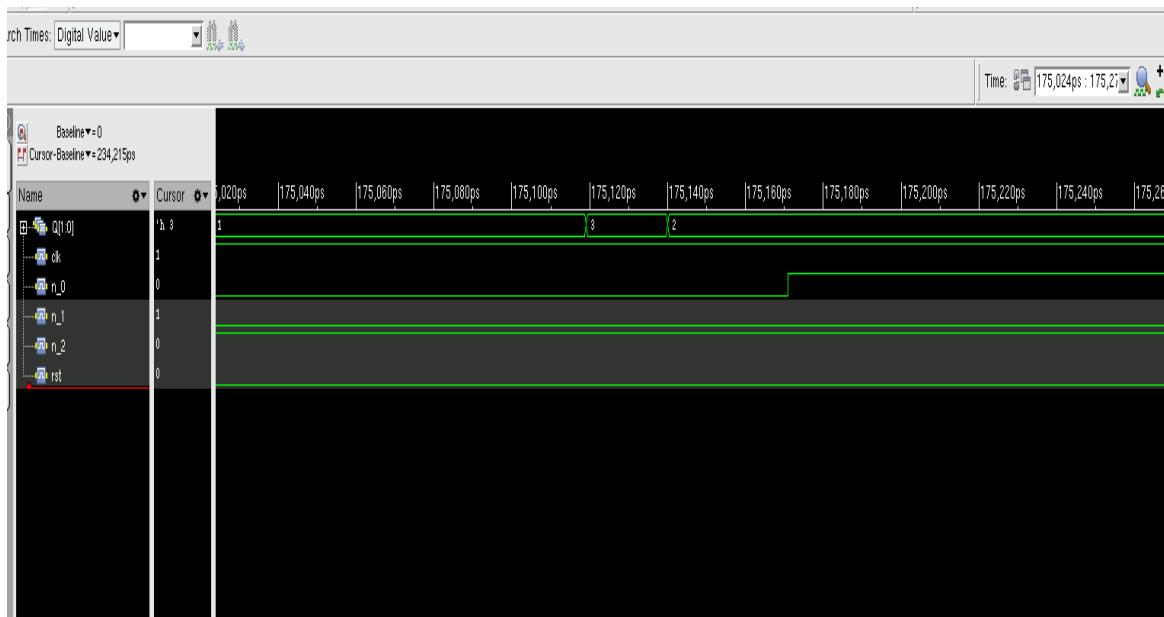
Vì $T = 50$ ns lớn hơn rất nhiều so với tổng trễ (~0,2–0,3 ns), nên trước mỗi cạnh clk tiếp theo, Q đã ổn định ở giá trị đúng. Đây là lý do testbench ex4_T50_tb.sv báo “TEST PASSED (no error)”.

4.2.3.2 Phân tích glitch khi có trễ

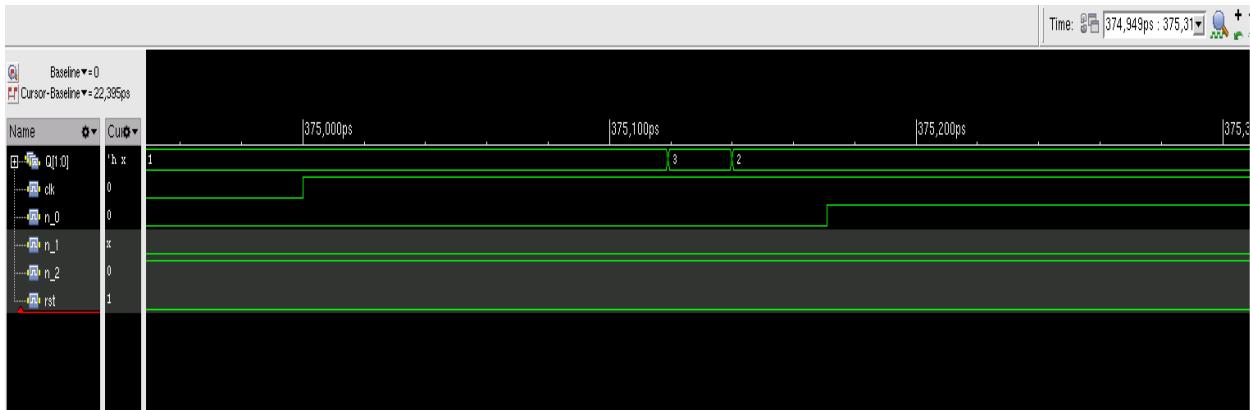
Khi mô phỏng RTL không có trễ, bộ đếm 2 bit cho chuỗi giá trị:

- $Q[1:0] = 00 \rightarrow 01 \rightarrow 10 \rightarrow 11 \rightarrow 00 \rightarrow \dots$

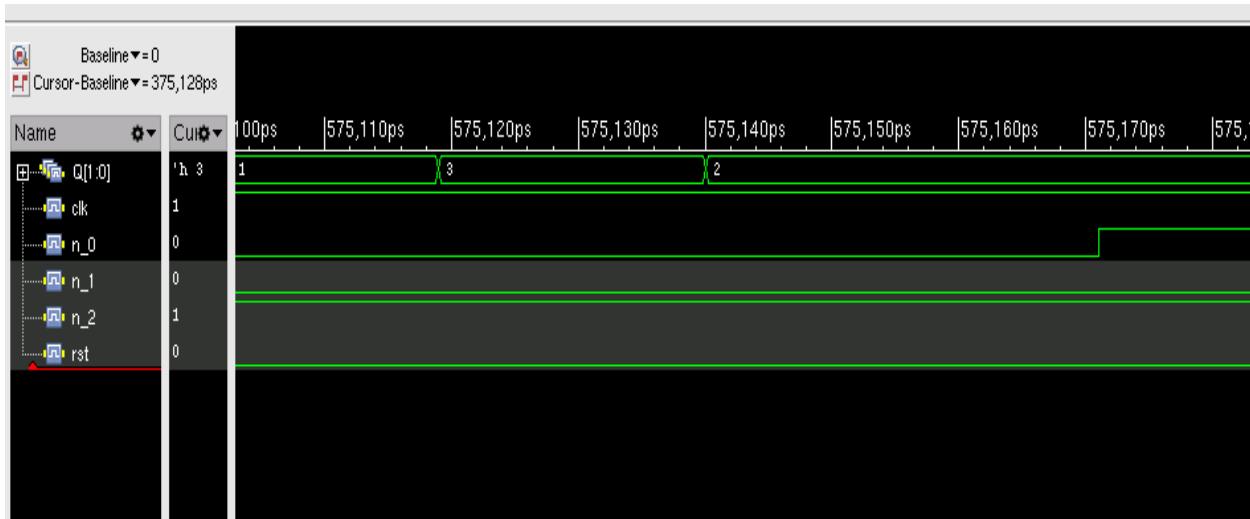
Mô phỏng gate-level có gắn SDF ở $T = 50$ ns cho thấy, nếu chỉ quan sát tại các cạnh lên của clk thì chuỗi Q vẫn giữ đúng thứ tự trên; testbench tự kiểm tra cũng báo “PASS”. Tuy nhiên, khi phóng to dạng sóng quanh một số cạnh clock, bus $Q[1:0]$ lại xuất hiện thêm một trạng thái trung gian 11 (thập phân là 3) trong quá trình chuyển từ 01 (1) lên 10 (2). Ba hình 4.2.3.2.a, 4.2.3.2.b và 4.2.3.2.c lần lượt dưới đây minh họa rõ hiện tượng này.



Hình 4.2.3.2.a



Hình 4.2.3.2.b



Hình 4.2.3.2.c

Hình 4.2.3.2.a thể hiện dạng sóng gate-level quanh một cạnh clock tại thời điểm xấp xỉ 175 ns. Trước cạnh lên của clk, Q[1:0] ổn định ở 01 (giá trị 1). Ngay sau cạnh clock, trong một khoảng thời gian rất ngắn, bus Q[1:0] nhảy sang 11 (3), rồi mới ổn định về 10 (2). Các tín hiệu nội bộ n_0, n_1, n_2 cũng thay đổi ngay sau đó, nhưng đều nằm trong vùng thời gian rất hẹp sau cạnh ck. Như vậy, tại cạnh này có thể thấy rõ chuỗi 1 → 3 → 2.

Hình 4.2.3.2.b chụp một cạnh clock khác ở khoảng 375 ns. Mặc dù là một chu kỳ khác, dạng chuyển vẫn giống hệt: Q đang ở 01, sau cạnh clk xuất hiện 11 trong một khoảng rất ngắn rồi ổn định về 10. Điều này cho thấy trạng thái trung gian “3” không phải là một lỗi ngẫu nhiên ở một chu kỳ riêng lẻ mà lặp lại ở nhiều chu kỳ có cùng điều kiện dữ liệu (trạng thái cũ là 1, trạng thái mới là 2).

Hình 4.2.3.2.c là một cạnh clock quanh thời điểm xấp xỉ 575 ns, được phóng chi tiết hơn để xem rõ trật tự thay đổi từng bit. Trước cạnh clk, Q[1:0] = 01. Sau cạnh lên, bit Q1 (MSB) tăng lên 1 trước, trong khi Q0 (LSB) vẫn chưa kịp đổi, khiến bus Q = 11 (3). Một khoảng

thời gian ngắn sau đó, Q0 mới giảm từ 1 xuống 0, bus Q trở thành 10 (2). Như vậy, trật tự chính xác của hai bit tại cạnh này là:

- $Q[1:0]: 01 (1) \rightarrow 11 (3) \rightarrow 10 (2)$

Hiện tượng này xuất phát trực tiếp từ trễ CK→Q không đối xứng của flip-flop trong SDF. Trong file ex4_delay_s1v2h.sdf, cell DFFHQX1HVT có thông số:

- CK→Q ($0 \rightarrow 1$) $\approx 0,119$ ns
- CK→Q ($1 \rightarrow 0$) $\approx 0,140$ ns

Trong bước đếm từ 1 lên 2:

- Q0 phải đổi từ $1 \rightarrow 0 \rightarrow$ dùng trễ $1 \rightarrow 0 = 0,140$ ns.
- Q1 phải đổi từ $0 \rightarrow 1 \rightarrow$ dùng trễ $0 \rightarrow 1 = 0,119$ ns.

Giả sử cạnh lên clk xuất hiện tại thời điểm t_0 :

- $\text{Ở } t \approx t_0 + 0,119$ ns, Q1 đã lên 1, Q0 vẫn còn $1 \rightarrow Q = 11 (3)$.
- $\text{Ở } t \approx t_0 + 0,140$ ns, Q0 mới rơi xuống $0 \rightarrow Q = 10 (2)$.

Khoảng thời gian $Q = 11$ tồn tại xấp xỉ:

$$\Delta t \approx 0,140 \text{ ns} - 0,119 \text{ ns} = 0,021 \text{ ns} = 21 \text{ ps}$$

So với chu kỳ $T = 50$ ns, 21 ps là một khoảng rất nhỏ, nhưng vẫn đủ để simulator hiển thị rõ một đoạn ngắn có giá trị 3 trên bus. Đây chính là đoạn “nhảy sai” $1 \rightarrow 3 \rightarrow 2$ quan sát được trong các hình.

Có thể đổi chiều thêm với ba bước đếm còn lại để thấy vì sao glitch chỉ nổi bật ở bước $1 \rightarrow 2$:

- Bước $0 \rightarrow 1$: 00 → 01, chỉ Q0 đổi từ $0 \rightarrow 1$, Q1 giữ 0 → bus đi thẳng $0 \rightarrow 1$, không có mã trung gian.
- Bước $2 \rightarrow 3$: 10 → 11, cũng chỉ Q0 đổi từ $0 \rightarrow 1$, Q1 giữ 1 → bus đi thẳng $2 \rightarrow 3$.
- Bước $3 \rightarrow 0$: 11 → 00, cả Q0 và Q1 đều đi từ $1 \rightarrow 0$. Cả hai bit đều dùng cùng loại trễ $1 \rightarrow 0 = 0,140$ ns, cell giống nhau, tải tương đương nên hai bit hạ xuống gần như đồng thời; nếu có lệch thì cũng rất nhỏ, simulator khó thể hiện thành trạng thái 01 hoặc 10 rõ ràng. Vì vậy trên waveform hầu như chỉ thấy $3 \rightarrow 0$ “đi thẳng”.

Ngược lại, ở bước $1 \rightarrow 2$:

- Hai bit cùng đổi và đổi ngược chiều (Q0 giảm, Q1 tăng).
- Mỗi bit lại dùng một giá trị trễ CK→Q khác nhau ($0 \rightarrow 1$ nhanh hơn $1 \rightarrow 0$).

Sự kết hợp này làm cho khoảng lêch thời gian giữa hai bit đầu lớn để bus Q[1:0] đi qua mã hoàn chỉnh 11 (3) trước khi ổn định ở 10 (2), tạo thành chuỗi 1 → 3 → 2 rất dễ nhìn thấy.

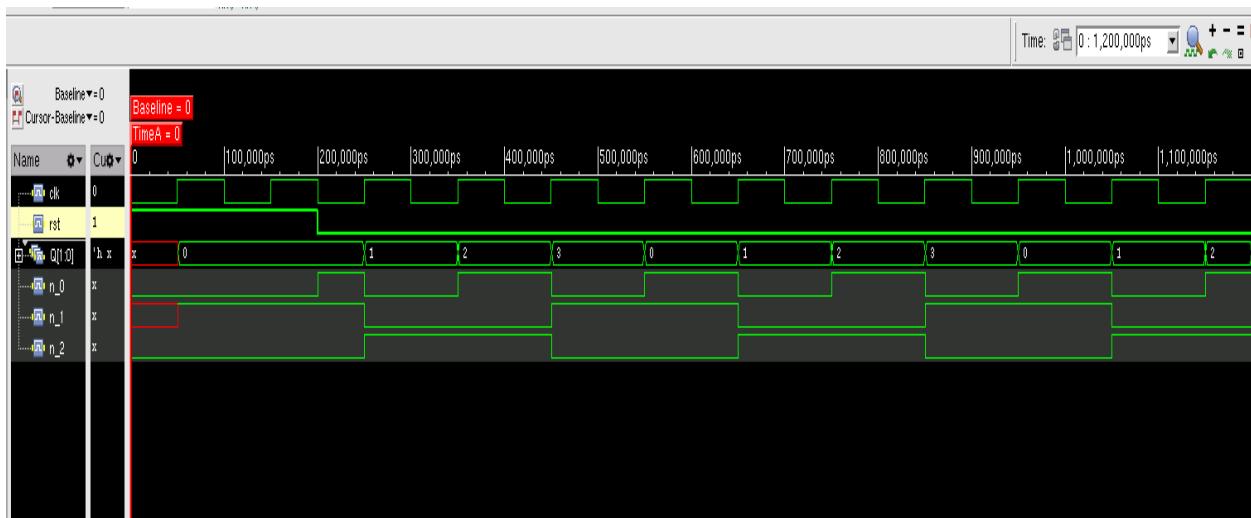
Kết luận cho T = 50 ns:

- Ở mức “lấy mẫu theo cạnh clock”, bộ đếm vẫn chính xác, vì tại mỗi posedge clk tiếp theo, Q đã ổn định về trạng thái đúng (2, 3, 0, ...) trước khi bị chốt lại.
- Dạng sóng chi tiết cho thấy rõ ảnh hưởng của trễ CK→Q và trễ qua các cổng logic trong thư viện slow_vdd1v2_basicCells_hvt.lib: hai bit của bus không bao giờ đổi đúng tuyệt đối đồng thời, nên trong một số trường hợp đặc biệt (như 01 → 10) sẽ xuất hiện các trạng thái trung gian không tồn tại trong mô phỏng RTL.

Vì vậy hiện tượng 1 → 3 → 2 ở T = 50 ns vì vậy không phải lỗi thiết kế, mà là minh họa cụ thể cho sự khác nhau giữa mô phỏng RTL không trễ và mô phỏng gate-level có SDF mà bài chúng ta đã phân tích ở trên.

4.2.4 Trường hợp 1: T=100 (mô phỏng gate-level có SDF)

4.2.4.1 Dạng sóng tổng quát khi có trễ



Hình 4.2.4.1 Dạng sóng tổng quát của trường hợp T = 100 khi có trễ

Hình trên thể hiện dạng sóng mô phỏng gate-level của mạch ex4 với chu kỳ clock T=100 ns. Netlist ex4_netlist_s1v2h.sv được nối với thư viện slow_vdd1v2_basicCells_hvt.lib và file trễ ex4_delay_s1v2h.sdf được back-annotate lên instance DUT.

Quan sát trong khoảng 0 → 1200 ns:

- Hai chu kỳ đầu: rst = 1, ngõ ra Q[1:0] bị reset về 00 và giữ nguyên.

- Sau khi rst hạ xuống 0, cứ mỗi cạnh lên của clk thì Q tăng 1 đơn vị:
 - $00 \rightarrow 01 \rightarrow 10 \rightarrow 11 \rightarrow 00 \rightarrow 01 \rightarrow 10 \rightarrow 11 \rightarrow \dots$
- Chuỗi giá trị hoàn toàn trùng với mô tả RTL (phần không có trễ), chỉ khác là thời điểm Q đổi bị trễ một chút so với cạnh clock.
- Độ trễ này phù hợp với số liệu trong SDF:
 - Trễ CK \rightarrow Q của DFFHQX1HVT khoảng 0,119–0,140 ns.
 - Trễ IOPATH qua các cổng NOR2 / XNOR2 khoảng 0,02–0,13 ns.

Vì chu kỳ T = 100 ns lớn hơn rất nhiều so với tổng trễ đường dữ liệu (~0,2–0,3 ns), nên trước mỗi cạnh clk tiếp theo, Q luôn ổn định ở giá trị đúng. Do đó khi testbench chờ một khoảng thời gian nhỏ sau cạnh clock rồi mới so sánh, kết quả đều PASS.

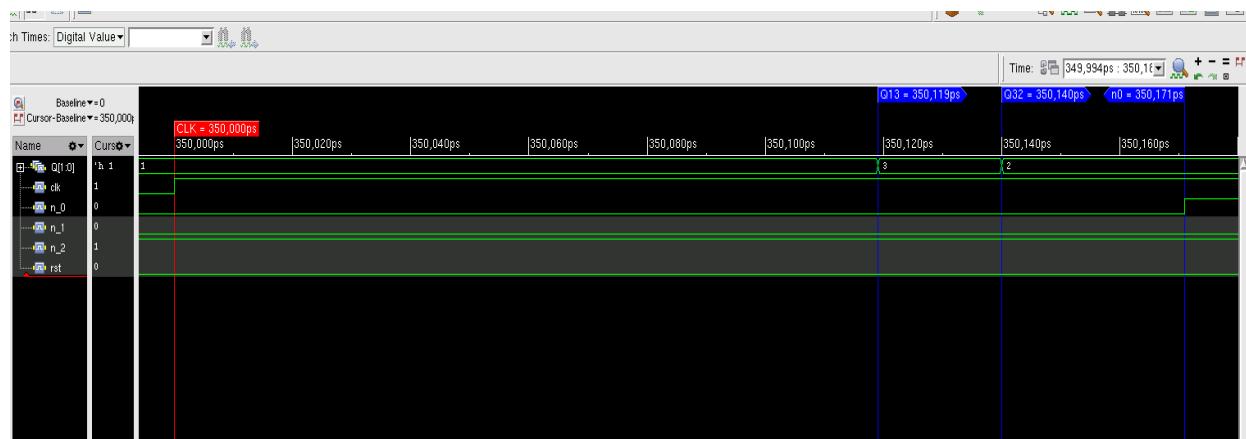
4.2.4.1 Phân tích glitch khi có trễ

Khi mô phỏng RTL (không có trễ), bộ đếm 2 bit đi qua các trạng thái:

- $Q[1:0] = 00 \rightarrow 01 \rightarrow 10 \rightarrow 11 \rightarrow 00 \rightarrow \dots$

Mô phỏng gate-level có SDF với T = 100 ns cho thấy, nếu chỉ nhìn tại các thời điểm posedge clk thì chuỗi Q vẫn đúng như trên. Tuy nhiên, khi phóng to dạng sóng quanh một số cạnh clock, bus Q[1:0] xuất hiện thêm trạng thái trung gian 11 (thập phân là 3) trong quá trình chuyển từ 01 (1) sang 10 (2), giống với hiện tượng ở trường hợp T = 50 ns.

Ba hình 4.2.4.2.a, 4.2.4.2.b và 4.2.4.2.c lần lượt là các cạnh clock điển hình (quanh 350 ns, 750 ns và 1150 ns) cho thấy rõ glitch này.

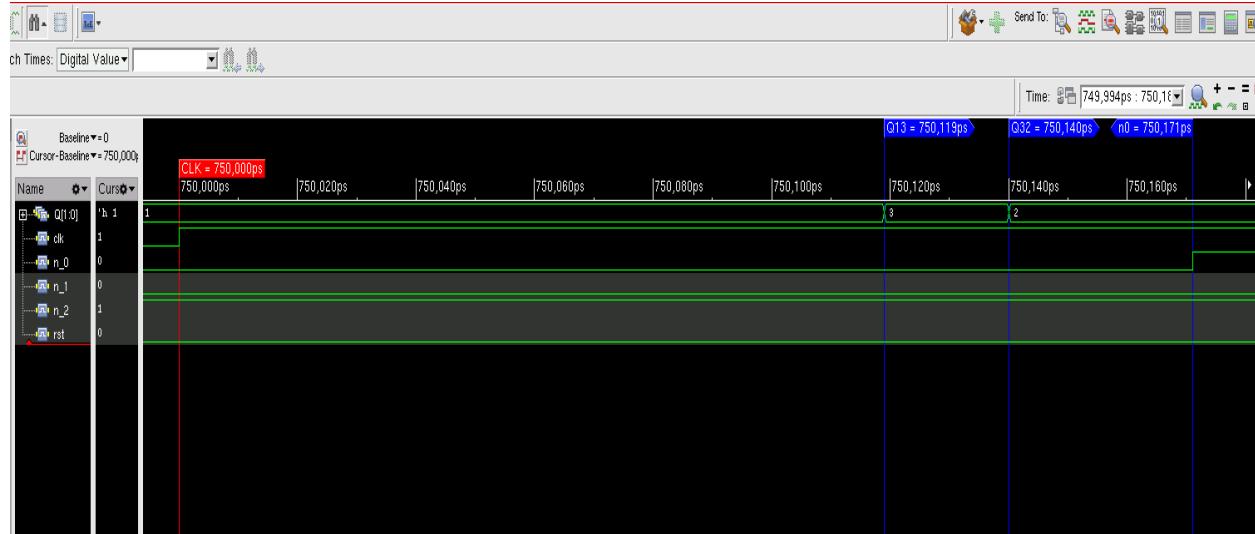


Hình 4.2.4.2.a – Cạnh clock khoảng 350 ns

Hình 4.2.4.2.a – Dạng sóng gate-level tại T = 100 ns quanh cạnh clock $t \approx 350$ ns.

- Trước cạnh lên của clk ($t \approx 350$ ns), Q[1:0] ổn định ở 01, tương ứng giá trị 1.
- Sau cạnh clock, waveform cho thấy Q không nhảy 1 → 2 mà đi theo chuỗi:
 - 01 (1) → 11 (3) → 10 (2).

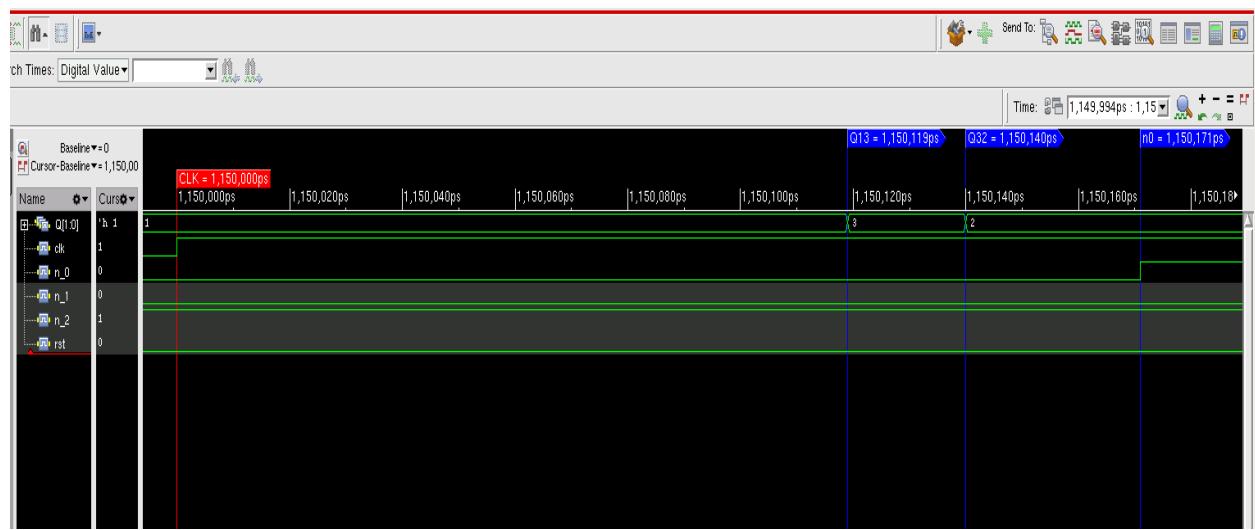
- Khoảng thời gian $Q = 11$ rất ngắn, nhưng đủ để waveform hiển thị rõ mức “3” nằm giữa “1” và “2”.
- Các nút nội bộ n_0, n_1, n_2 cũng bắt đầu thay đổi ngay sau đó, với độ trễ lớn hơn, đúng với việc chúng nằm sâu hơn trong mạng logic (qua NOR / XNOR).



Hình 4.2.4.2.b – Cảnh clock khoảng 750 ns

Hình 4.2.4.2.b – Dạng sóng gate-level tại $T = 100$ ns quanh cảnh clock $t \approx 750$ ns.

- Đây là một chu kỳ khác, nhưng trạng thái trước cảnh vẫn là $Q = 01$ (1).
- Sau cảnh lên của clk, bus Q một lần nữa đi theo chuỗi:
 - $01 \rightarrow 11 \rightarrow 10$.
- Sự lặp lại của chuỗi $1 \rightarrow 3 \rightarrow 2$ ở một thời điểm khác chứng tỏ đây là hành vi có hệ thống mỗi khi mạch chuyển từ 1 sang 2, chứ không phải một glitch ngẫu nhiên do noise hay lỗi mô phỏng.



Hình 4.2.4.2.c – Cảnh clock khoảng 1150 ns

Hình 4.2.4.2.c – Dạng sóng gate-level tại $T = 100$ ns quanh cạnh clock $t \approx 1150$ ns, zoom chi tiết thứ tự đổi bit.

Trên hình có thể đọc được thứ tự đổi bit tương tự $T = 50$ ns:

- Trước cạnh clk: $Q[1:0] = 01$ (1).
- Sau cạnh:
 - $Q1$ (MSB) tăng từ 0 lên 1 trước, $Q0$ vẫn đang 1 $\rightarrow Q = 11$ (3).
 - Sau một khoảng ngắn, $Q0$ mới rơi từ 1 xuống 0 $\rightarrow Q = 10$ (2).

Tức là:

- $Q[1:0]: 01 (1) \rightarrow 11 (3) \rightarrow 10 (2)$.

Điễn giải theo SDF:

- Trễ CK $\rightarrow Q$ trong ex4_delay_s1v2h.sdf:
 - CK $\rightarrow Q (0 \rightarrow 1) \approx 0,119$ ns
 - CK $\rightarrow Q (1 \rightarrow 0) \approx 0,140$ ns

Ở bước đếm 1 \rightarrow 2:

- $Q0$ đổi 1 \rightarrow 0 \rightarrow dùng trễ 1 \rightarrow 0 = 0,140 ns.
- $Q1$ đổi 0 \rightarrow 1 \rightarrow dùng trễ 0 \rightarrow 1 = 0,119 ns.

Nếu gọi thời điểm cạnh lên clk là t_0 :

- $t \approx t_0 + 0,119$ ns: $Q1$ đã là 1, $Q0$ vẫn 1 $\rightarrow Q = 11$ (3).
- $t \approx t_0 + 0,140$ ns: $Q0$ rơi xuống 0 $\rightarrow Q = 10$ (2).

Khoảng tồn tại của $Q = 11$ là:

$$\Delta t \approx 0,140 \text{ ns} - 0,119 \text{ ns} = 0,021 \text{ ns} = 21 \text{ ps}$$

So với chu kỳ $T = 100$ ns, 21 ps là cực nhỏ, nhưng simulator vẫn hiển thị được một đoạn rất ngắn có giá trị 3. Đó chính là đoạn “nhảy” 1 \rightarrow 3 \rightarrow 2 quan sát được trên ba hình.

So sánh với các bước đếm còn lại:

- 0 \rightarrow 1: 00 \rightarrow 01, chỉ $Q0 0 \rightarrow 1$, $Q1$ giữ 0 \rightarrow bus đi thẳng 0 \rightarrow 1, không có trạng thái trung gian.
- 2 \rightarrow 3: 10 \rightarrow 11, cũng chỉ $Q0 0 \rightarrow 1$, $Q1$ giữ 1 \rightarrow bus đi thẳng 2 \rightarrow 3.

- $3 \rightarrow 0$: $11 \rightarrow 00$, cả Q0 và Q1 đều $1 \rightarrow 0$, dùng cùng loại trễ $1 \rightarrow 0 = 0,140$ ns, cell giống nhau, tải gần tương đương \rightarrow hai bit rời gần như cùng lúc, waveform nhìn như nhảy thẳng $3 \rightarrow 0$.

Riêng bước $1 \rightarrow 2$:

- Cả hai bit cùng đổi, lại đổi ngược chiều (Q1 tăng, Q0 giảm).
- Hai hướng chuyển mức dùng hai giá trị trễ CK \rightarrow Q khác nhau ($0 \rightarrow 1$ nhanh hơn $1 \rightarrow 0$).

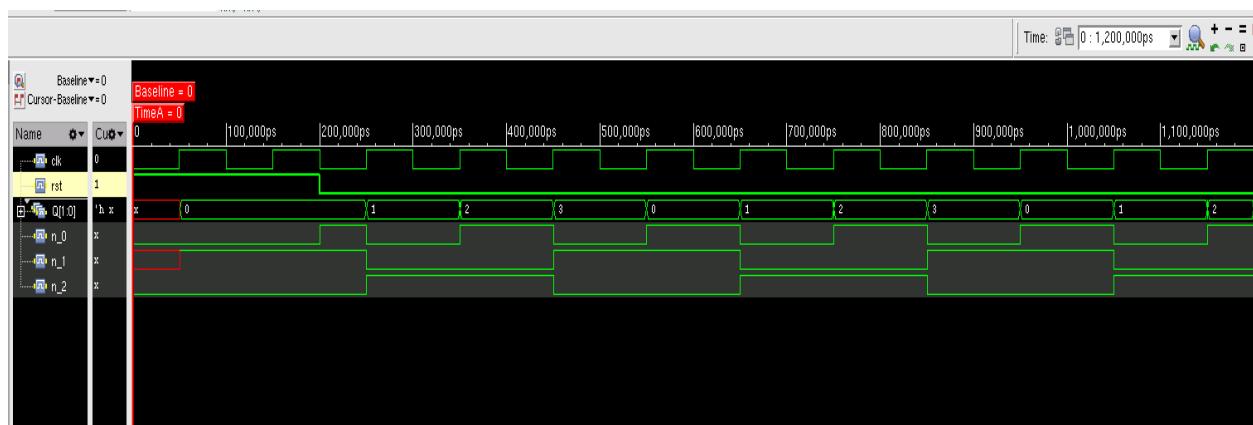
Nên bus Q[1:0] bị “kéo” qua mã 11 trong vài chục pico-giây rồi mới ổn định ở 10. Đó là lý do glitch $1 \rightarrow 3 \rightarrow 2$ nổi bật ở bước này, còn các bước khác nhìn “sạch”.

Kết luận cho $T = 100$ ns

- Về chức năng: tại mỗi cạnh posedge clk, Q vẫn đếm đúng giống RTL; không có bit nào bị bỏ sót hay lặp sai chu kỳ.
- Về chi tiết dạng sóng: $T = 100$ ns cho thấy lại đúng hiện tượng $1 \rightarrow 3 \rightarrow 2$ giống $T = 50$ ns, do trễ CK \rightarrow Q không đổi xứng của DFF và trễ logic trong thư viện slow_vdd1v2_basicCells_hvt.lib.
- Do T vẫn lớn hơn tổng trễ rất nhiều, glitch chỉ tồn tại ngắn và không gây lỗi setup/hold. Đây là ví dụ rõ ràng cho sự khác biệt giữa mô phỏng “no delay” (RTL) và mô phỏng gate-level có SDF.

4.2.5 Trường hợp 1: $T=200$ (mô phỏng gate-level có SDF)

4.2.5.1 Dạng sóng tổng quát khi có trễ



Hình 4.2.5.1 Dạng sóng tổng quát của trường hợp $T = 200$ khi có trễ

Hình 4.2.5.1 thể hiện dạng sóng mô phỏng gate-level của mạch ex4 với chu kỳ clock $T = 200$ ns. Netlist ex4_netlist_s1v2h.sv được nối với thư viện

slow_vdd1v2_basicCells_hvt.lib và file trễ ex4_delay_s1v2h.sdf được back-annotate lên instance DUT.

Quan sát trên khoảng thời gian $0 \rightarrow 2400$ ns:

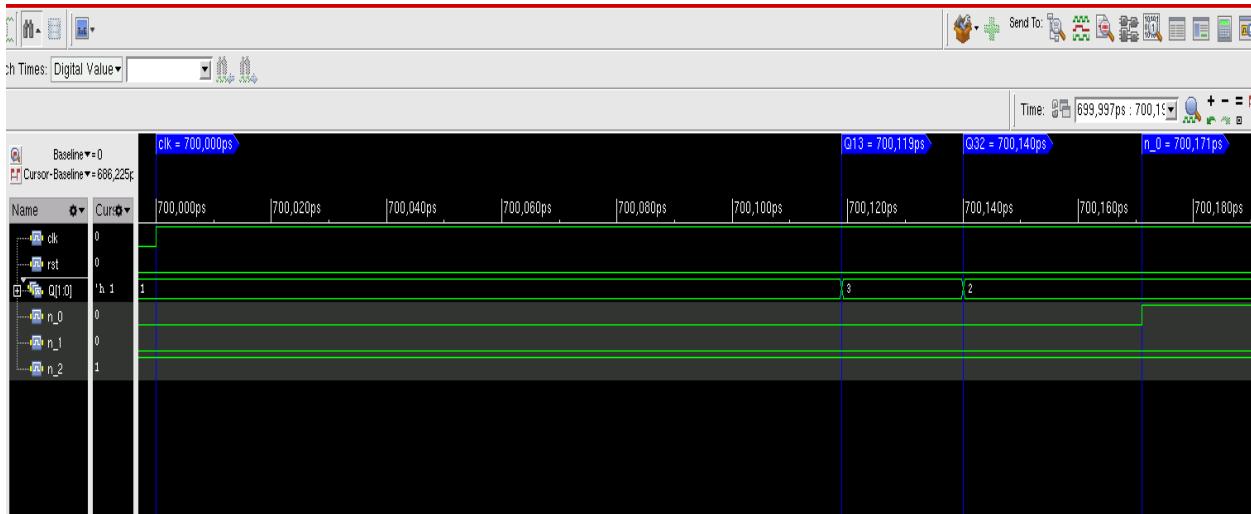
- Trong hai chu kỳ đầu, tín hiệu rst được giữ ở mức 1 nên Q[1:0] luôn bằng 00.
- Sau khi rst được đưa về 0, ở mỗi cạnh lên của clk bộ đếm tăng 1 theo chuỗi:
 - $00 \rightarrow 01 \rightarrow 10 \rightarrow 11 \rightarrow 00 \rightarrow 01 \rightarrow 10 \rightarrow 11 \rightarrow \dots$
- Chuỗi giá trị này trùng hoàn toàn với mô phỏng RTL không trễ: mỗi cạnh posedge clk tương ứng với một bước đếm mới, không có trạng thái bị bỏ qua hay bị lặp.
- Thời điểm Q đổi không trùng đúng với cạnh clk mà bị dịch lùi lại một khoảng nhỏ, phù hợp với trễ CK→Q ($\approx 0,119 - 0,140$ ns) và trễ qua các công logic NOR/XNOR ($\approx 0,02 - 0,13$ ns) trong SDF.

Vì chu kỳ clock $T = 200$ ns còn lớn hơn nhiều so với tổng trễ đường dữ liệu ($\sim 0,2 - 0,3$ ns), nên trước mỗi cạnh clk tiếp theo, Q luôn ổn định ở giá trị đúng. Do đó về mặt chức năng, counter vẫn hoạt động bình thường ở $T = 200$ ns.

4.2.5.2 Phân tích glitch khi có trễ

Giống như hai trường hợp $T = 50$ ns và $T = 100$ ns, mô phỏng gate-level có SDF với $T = 200$ ns vẫn cho kết quả đúng ở các thời điểm “lấy mẫu theo cạnh clock”, nhưng nếu phóng to dạng sóng quanh một số cạnh clk, bus Q[1:0] lại xuất hiện trạng thái trung gian 11 (3) trong quá trình chuyển từ 01 (1) sang 10 (2).

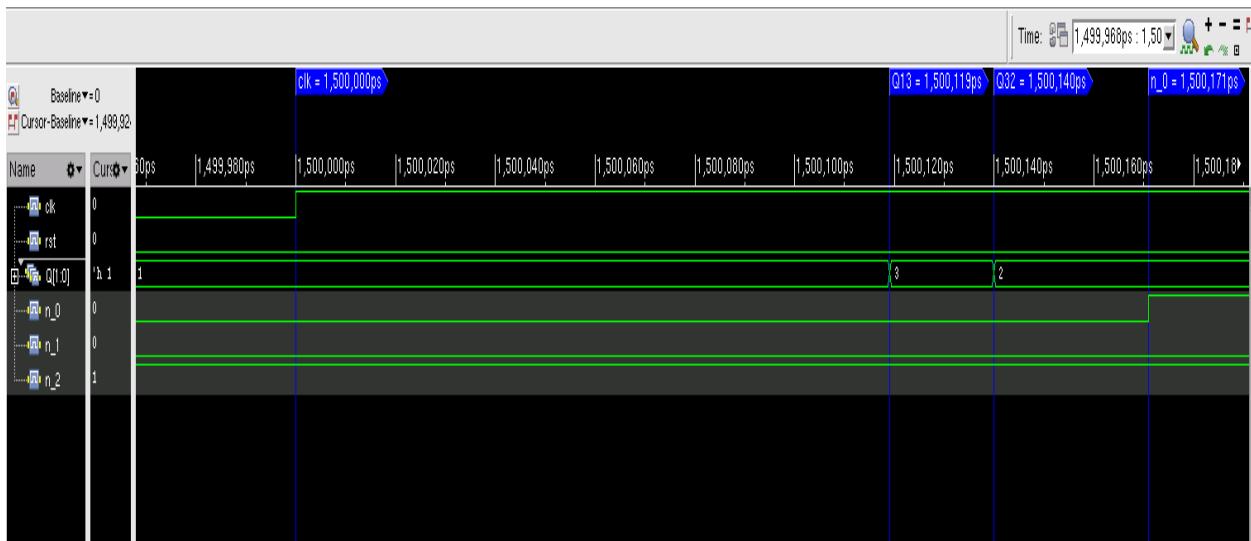
Ba hình 4.2.5.2.a, 4.2.5.2.b và 4.2.5.2.c minh họa các cạnh clock điển hình (lần lượt xấp xỉ 700 ns, 1500 ns và 2300 ns) nơi có thể quan sát rõ hiện tượng này.



Hình 4.2.5.2.a – Cảnh clock khoảng 700 ns

Hình 4.2.5.2.a – Dạng sóng gate-level tại $T = 200$ ns quanh cảnh clock $t \approx 700$ ns.

- Trước cảnh lén của clk, $Q[1:0]$ đang ổn định ở 01 (giá trị 1).
- Sau cảnh clock, waveform cho thấy Q không đi thẳng 1 → 2 mà trải qua chuỗi:
 - 01 (1) → 11 (3) → 10 (2).
- Đoạn $Q = 11$ (3) tồn tại trong một khoảng thời gian rất ngắn (vài chục pico-giây), sau đó Q ổn định về 10 trước khi đến cảnh clk tiếp theo.
- Các tín hiệu nội bộ n_0, n_1, n_2 cũng bắt đầu thay đổi sau clk, nhưng độ trễ lớn hơn vì chúng nằm sâu hơn trong mạng logic.

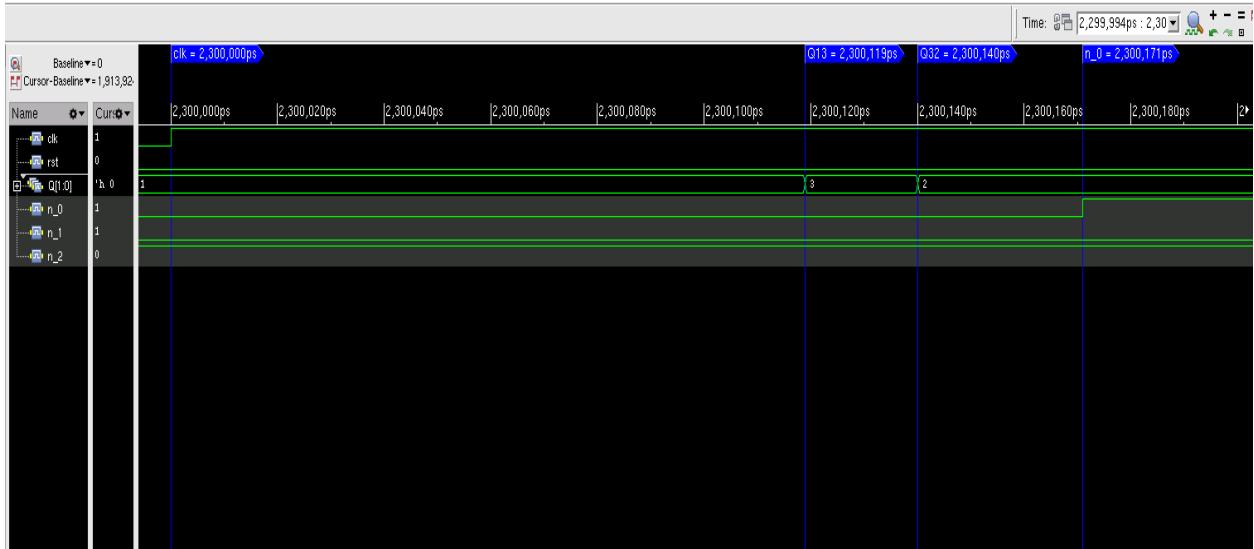


Hình 4.2.5.2.b – Cảnh clock khoảng 1500 ns

Hình 4.2.5.2.b – Dạng sóng gate-level tại $T = 200$ ns quanh cảnh clock $t \approx 1500$ ns.

- Một lần nữa, trước cảnh clk ta có $Q[1:0] = 01$.

- Ngay sau cạnh, dạng sóng lặp lại hành vi giống hệt:
 - Q nhảy qua 11 rồi mới ổn định ở 10.
- Việc chuỗi $1 \rightarrow 3 \rightarrow 2$ lặp lại ở một chu kỳ khác cho thấy đây là đặc trưng của mạch khi chuyển từ 1 sang 2, không phải glitch ngẫu nhiên.



Hình 4.2.5.2.c – Cạnh clock khoảng 2300 ns

Hình 4.2.5.2.c – Dạng sóng gate-level tại $T = 200$ ns quanh cạnh clock $t \approx 2300$ ns, zoom chi tiết thứ tự đổi bit.

Hình này cho thấy rõ thứ tự thay đổi của từng bit trên bus Q:

- Trước cạnh lên của clk: $Q[1:0] = 01$ (1).
- Sau cạnh:
 - Q_1 (bit MSB) chuyển từ 0 lên 1 nhanh hơn, trong khi Q_0 vẫn là 1 \rightarrow bus Q = 11 (3).
 - Sau đó một khoảng rất ngắn, Q_0 mới rơi từ 1 xuống 0 \rightarrow Q = 10 (2).

Như vậy tại các cạnh này, thứ tự chính xác là:

- $Q[1:0]: 01 (1) \rightarrow 11 (3) \rightarrow 10 (2)$.

Hiện tượng này phù hợp với các thông số trễ trong SDF:

- CK \rightarrow Q ($0 \rightarrow 1$) $\approx 0,119$ ns
- CK \rightarrow Q ($1 \rightarrow 0$) $\approx 0,140$ ns

Trong bước đếm từ 1 lên 2:

- Q_0 phải đổi $1 \rightarrow 0 \rightarrow$ dùng trễ $1 \rightarrow 0 = 0,140$ ns.

- Q1 phải đổi $0 \rightarrow 1$ → dùng trễ $0 \rightarrow 1 = 0,119$ ns.

Nếu gọi thời điểm cạnh clk là t_0 :

- $t \approx t_0 + 0,119$ ns: Q1 đã lên 1, Q0 vẫn $1 \rightarrow Q = 11$ (3).
- $t \approx t_0 + 0,140$ ns: Q0 rơi xuống $0 \rightarrow Q = 10$ (2).

Khoảng thời gian $Q = 11$ tồn tại:

$$\Delta t \approx 0,140 \text{ ns} - 0,119 \text{ ns} = 0,021 \text{ ns} = 21 \text{ ps}$$

So với chu kỳ $T = 200$ ns, 21 ps là cực kỳ nhỏ, nhưng simulator vẫn vẽ được một đoạn ngắn có giá trị 3 trên bus, tạo đếm sai $1 \rightarrow 3 \rightarrow 2$ khi zoom rất sâu.

Giải thích vì sao glitch chỉ nỗi bật ở bước $1 \rightarrow 2$:

- $0 \rightarrow 1$: $00 \rightarrow 01$, chỉ Q0 $0 \rightarrow 1$, Q1 giữ $0 \rightarrow$ bus đi thẳng $0 \rightarrow 1$.
- $2 \rightarrow 3$: $10 \rightarrow 11$, chỉ Q0 $0 \rightarrow 1$, Q1 giữ $1 \rightarrow$ bus đi thẳng $2 \rightarrow 3$.
- $3 \rightarrow 0$: $11 \rightarrow 00$, cả Q0 và Q1 đều $1 \rightarrow 0$, cùng dùng trễ $1 \rightarrow 0 = 0,140$ ns, cùng loại cell và tải tương đương nên hai bit rơi gần như đồng thời → không thấy rõ trạng thái 01 hoặc 10 trung gian.

Riêng ở $1 \rightarrow 2$:

- Cả hai bit cùng đổi và đổi ngược chiều (Q1 tăng, Q0 giảm).
- Hai hướng chuyển mức sử dụng hai giá trị trễ CK $\rightarrow Q$ khác nhau ($0 \rightarrow 1$ nhanh hơn $1 \rightarrow 0$).

Do đó bus Q[1:0] có thời gian ngắn đi qua mã 11 trước khi ổn định tại 10, tạo glitch $1 \rightarrow 3 \rightarrow 2$.

Kết luận cho $T = 200$ ns

- Chu kỳ clock $T = 200$ ns còn lớn hơn $T = 50$ và $T = 100$ rất nhiều so với tổng trễ của đường dữ liệu, nên ở mỗi cạnh posedge clk, Q đã ổn định và đếm đúng giống mô phỏng RTL không trễ.
- Glitch $1 \rightarrow 3 \rightarrow 2$ vẫn xuất hiện khi zoom kỹ quanh các cạnh mà Q chuyển $1 \rightarrow 2$, nhưng tồn tại rất ngắn (khoảng 21 ps), không gây vi phạm setup/hold.
- Trường hợp $T = 200$ ns vì vậy minh họa rõ ràng: khi chu kỳ clock đủ lớn so với trễ lan truyền trong thư viện slow_vdd1v2_basicCells_hvt.lib, mạch tuần tự vẫn hoạt động đúng chức năng, còn các trạng thái trung gian chỉ là đặc trưng của mô phỏng gate-level có SDF, không phải lỗi thiết kế.

4.2.6 Tổng kết các dạng sóng và đánh giá hoạt động của mạch

Từ các kết quả mô phỏng ở trên (RTL, T = 50; 100; 200 ns), ta rút ra một số nhận xét sau:

1. Vai trò của mô phỏng RTL (không có trễ)
 - o Dạng sóng RTL cho chuỗi đếm lý tưởng: 00 → 01 → 10 → 11 → 00 → , các bit Q[1:0] thay đổi “đồng thời” đúng tại cạnh lên của clk.
 - o Không có trạng thái trung gian, không có glitch.
 - o Đây là “kết quả chuẩn” dùng làm mốc so sánh cho mọi mô phỏng gate-level.
2. Ảnh hưởng của SDF và trễ CK→Q / trễ logic
 - o Khi gắn SDF từ thư viện slow_vdd1v2_basicCells_hvt.lib, mỗi phần tử trong mạch mang một giá trị trễ cụ thể (CK→Q, IOPATH...).
 - o Trễ CK→Q của flip-flop DFFHQX1HVT không đổi xứng: 0 → 1 nhanh hơn 1 → 0 ($\approx 0,119$ ns so với $\approx 0,140$ ns). Trong bước đếm từ 1 lên 2 (01 → 10): Q1 phải tăng 0 → 1 (trễ nhỏ hơn); Q0 phải giảm 1 → 0 (trễ lớn hơn).
 - o Do hai bit đổi ngược chiều và có trễ khác nhau, bus Q[1:0] đi qua trạng thái trung gian 11 (3) trong thời gian ~ 21 ps, tạo ra chuỗi 1 → 3 → 2 ở cả ba trường hợp T = 50, 100, 200.
 - o Ở các bước đếm khác (0 → 1, 2 → 3, 3 → 0), hoặc chỉ một bit thay đổi, hoặc hai bit đổi cùng chiều với trễ tương đương, nên gần như không xuất hiện mã trung gian rõ ràng trên waveform.
3. So sánh giữa các chu kỳ T = 50 ns, 100 ns, 200 ns
 - o Ở cả ba giá trị T, tổng trễ lan truyền của đường dữ liệu (CK→Q + logic) chỉ khoảng 0,2–0,3 ns, nhỏ hơn T rất nhiều. Vì vậy, mặc dù bên trong mỗi chu kỳ có glitch 1 → 3 → 2, nhưng trước cạnh clk tiếp theo Q luôn kịp ổn định về giá trị đúng. Khi testbench có chèn một khoảng chờ nhỏ sau cạnh clk rồi mới so sánh (ví dụ #1;), cả ba trường hợp T = 50, 100, 200 đều PASS, chứng tỏ mạch đạt yêu cầu về mặt chức năng với ba chu kỳ clock này.
 - o Sự khác nhau giữa T = 50, 100 và 200 chủ yếu nằm ở mật độ cạnh clock trên trực thời gian; hình dạng glitch (1 → 3 → 2) gần như giống nhau vì nó phụ thuộc vào trễ cell trong SDF chứ không phụ thuộc vào T.
4. Liên hệ giữa mô phỏng và ràng buộc timing thực tế
 - o Các thí nghiệm cho thấy: chỉ nhìn RTL sẽ không thấy glitch, trong khi gate-level + SDF cho cái nhìn thực tế hơn về trễ và trạng thái trung gian.
 - o Ba giá trị T được chọn đều lớn hơn nhiều so với trễ đường dữ liệu ở corner “slow”, nên mạch vẫn thỏa timing (không vi phạm setup/hold).
 - o Nếu T tiếp tục giảm xuống gần với tổng trễ đường dài nhất, glitch và việc Q cập nhật trễ sau cạnh clk có thể dẫn đến lỗi chức năng (mô phỏng sẽ FAIL hoặc xuất hiện thông báo vi phạm timing).

Phần 4.2 cho thấy khác biệt giữa mô phỏng không trễ và mô phỏng gate-level có SDF: mạch counter về mặt chức năng vẫn đúng với cả ba chu kỳ T = 50; 100; 200 ns, nhưng dạng sóng chi tiết lộ rõ trễ CK→Q, trễ logic và các trạng thái trung gian trên bus Q.

V. KẾT LUẬN

Trong Lab 2, quy trình thiết kế số với thư viện standard cell đã được thực hiện trọn vẹn: từ mô tả RTL, tổng hợp sang netlist gate-level, phân tích timing/area/power trên nhiều corner PVT, đến mô phỏng chức năng và mô phỏng có trễ với SDF. Các bài thực hành từ cell tổ hợp đơn (NOT, NAND, NOR), ALU, DFF cho tới bộ đếm tuần tự giúp liên kết lý thuyết mạch số với flow thiết kế ASIC thực tế.

Kết quả các thí nghiệm cho thấy rõ ảnh hưởng của corner PVT và loại cell (LVT/HVT) lên độ trễ và công suất: corner “slow, VDD thấp, HVT” cho trễ lớn hơn nhưng rò nhỏ hơn, trong khi corner “fast, VDD cao, LVT” cho mạch nhanh hơn nhưng công suất tăng. Từ các báo cáo timing/area/power, có thể thấy bài toán thiết kế luôn là sự đánh đổi giữa tốc độ, diện tích và công suất, không có cấu hình nào tối ưu tuyệt đối cho mọi mục tiêu.

So sánh dạng sóng RTL với gate-level + SDF ở các bài DFF và bộ đếm cho thấy sự khác biệt quan trọng giữa mô phỏng lý tưởng và mô phỏng có trễ: RTL không xuất hiện trạng thái trung gian, trong khi gate-level bộc lộ trễ CK→Q, trễ qua logic và các glitch như chuỗi “1 → 3 → 2” ở bộ đếm 2 bit. Tuy nhiên, khi chu kỳ clock lớn hơn đáng kể so với tổng trễ đường dữ liệu (các trường hợp $T = 50$ ns, 100 ns, 200 ns), mạch vẫn đáp ứng timing và đếm hoàn toàn đúng tại các cạnh clock, minh họa rõ khái niệm “hoạt động đúng chức năng nhưng bên trong vẫn có glitch”.

Cuối cùng, việc xây dựng testbench tự kiểm tra và dùng chung cho cả RTL lẫn gate-level giúp kiểm chứng tính tương đương chức năng sau tổng hợp và sau khi gắn SDF. Qua Lab 2, người làm bài không chỉ nắm được cách sử dụng công cụ (Genus, Xcelium, thư viện gpdk045...) mà còn hiểu rõ hơn mối quan hệ giữa mô tả RTL, netlist standard cell, file SDF và các ràng buộc timing trong một flow thiết kế VLSI thực tế.