

TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN
KHOA ĐIỆN TỬ - VIỄN THÔNG
BỘ MÔN KỸ THUẬT MẠCH ĐIỆN TỬ



ĐỒ ÁN CUỐI KÌ
FLIP FLOP USING LATCHES

Giảng viên hướng dẫn: ThS NGUYỄN THỊ THIÊN TRANG

Sinh viên thực hiện: ĐẶNG GIA BẢO

MSSV: 22200009

MAI TIẾN DŨNG

MSSV: 22200037

LỜI MỞ ĐẦU

Trong lĩnh vực điện tử số, các phần tử lưu trữ đóng vai trò quan trọng trong việc ghi nhớ và xử lý thông tin. Trong đó, flip-flop là một cấu trúc cơ bản được sử dụng rộng rãi trong nhiều hệ thống mạch số đồng bộ như bộ đếm, thanh ghi, bộ nhớ và bộ xử lý.

Trong khuôn khổ đề án này, đề tài yêu cầu khảo sát flip-flop sử dụng latch, do đó cấu trúc **D Flip-Flop kiểu master–slave latch** đã được lựa chọn làm cơ sở thiết kế. Nhằm làm rõ hơn tính ứng dụng và khả năng điều khiển của flip-flop trong các hệ thống thực tế, mạch còn được tích hợp thêm chức năng **Set/Reset bất đồng bộ**, vốn là tính năng phổ biến trong nhiều cell lưu trữ chuẩn hiện nay.

Việc nghiên cứu và thiết kế flip-flop không chỉ giúp hiểu rõ hơn về nguyên lý hoạt động của các mạch lưu trữ, mà còn góp phần nâng cao khả năng phân tích, đánh giá hiệu năng và độ tin cậy của hệ thống số ở cấp độ transistor.

Với mục tiêu đó, đề án này tập trung vào việc tìm hiểu và phân tích hoạt động của một loại flip-flop sử dụng cấu trúc latch, thông qua quá trình thiết kế, mô phỏng và đánh giá các đặc tính hoạt động trong nhiều khía cạnh khác nhau.

MỤC LỤC

CHƯƠNG 1: TỔNG QUAN	1
1.1. Giới thiệu về Flip-flop và Latch	1
1.1.1. Khái niệm các phần tử lưu trữ trong mạch số	1
1.1.2. Latch là gì?	2
1.1.3. Flip-flop là gì?	4
1.1.4. So sánh giữa Flip-flop và Latch	6
1.1.5. Cấu trúc Flip-flop Master-Slave sử dụng Latch	6
1.2. Vai trò và ứng dụng của Flip-flop	9
1.2.1. Vai trò của Flip-flop trong vi xử lý hiện đại	9
1.2.2. Ứng dụng của Flip-flop trong mạch số	10
1.3. Flip-flop với Set/Reset bất đồng bộ	11
1.3.1. Khái niệm về tín hiệu Set/Reset bất đồng bộ	11
1.3.2. Ý nghĩa Set/Reset trong hệ thống số	12
1.4. Các thông số quan trọng cần khảo sát.....	14
1.4.1. Thông số thời gian.....	14
1.4.2. Công suất tiêu thụ.....	18
1.4.3. Độ ổn định và khả năng chống nhiễu	18
1.5. Mục tiêu và phạm vi đồ án.....	18
1.5.1. Mục tiêu thực hiện.....	19
1.5.2. Phạm vi thực hiện.....	19
1.5.3. Giới hạn và hướng mở rộng	20
CHƯƠNG 2: THIẾT KẾ VÀ XÂY DỰNG MẠCH	21
2.1. Quy trình thực hiện	21
2.1.1. Mục tiêu của thiết kế Custom Flow.....	21
2.1.2. Các bước thực hiện	21
2.2. Thiết kế các phần tử logic cơ bản và kiểm tra chức năng	23
2.2.1. Thiết kế cổng INV	23
2.2.2. Thiết kế cổng NAND.....	25
2.2.3. Thiết kế cổng truyền(TG).....	27
2.3. Thiết kế Master Latch, Slave Latch kiểm tra chức năng	28
2.3.1. Kiến trúc Latch sử dụng TG	28
2.3.2. Tích hợp Set/Reset bất đồng bộ.....	30
2.4. Thiết kế D Flip-flop hoàn chỉnh	31
2.4.1. Kết nối Master Latch và Slave Latch thành DFF	31
2.4.2. Mô phỏng chức năng D Flip-flop với Set/Reset bất đồng bộ.....	32
2.4.3. Nhận xét sơ bộ	33

CHƯƠNG 3: KHẢO SÁT CÁC THÔNG SỐ	34
3.1. Khảo sát thông số thời gian(Timing)	34
3.1.1. Đo Propagation Delay(Clock-to-Q Delay)	34
3.1.2. Đo Rise Time và Fall Time.....	37
3.1.3. Đo Setup Time và Hold Time	40
3.1.4. Đo Recovery Time và Removal Time	44
3.1.5. Đo Maximun Frequency/Minimum Clock Period.....	49
3.2. Khảo sát thông số công suất(Power)	50
3.2.1. Công suất động	50
3.2.2. Công suất tĩnh.....	50
3.2.3. Công suất tổng	51
3.3. Khảo sát khả năng chống nhiễu(Noise)	52
3.3.1. Khái quát.....	52
3.3.2. Khảo sát nhiễu	52
3.4. So sánh với các giá trị tham chiếu.....	53
TỔNG KẾT	54
1. Tóm tắt mục tiêu và kết quả	54
2. Vì sao chọn thiết kế Latch dùng Transmission Gate	55
2.1. So sánh kiến trúc Logic với Transmission Gate	55
2.2. Vì sao thiết kế thuần Logic dễ bị Metastable	55
2.3. Vì sao thiết kế dùng Trasmision Gate hiệu quả hơn	55
2.4. Tiến hành thử nghiệm trên Latch Logic	56
3. Đánh giá thiết kế	57
4. Đề xuất cải thiện	57
5. Kết luận.....	57
PHỤ LỤC.....	58
1. Tài liệu tham khảo.....	58
2. Sơ đồ mạch D Flip-flop được sử dụng	59

CHƯƠNG 1: TỔNG QUAN

1.1. GIỚI THIỆU VỀ FLIP-FLOP VÀ LATCH

1.1.1. Khái niệm các phần tử lưu trữ trong mạch số

Trong lĩnh vực mạch số, phần tử lưu trữ được xem là một thành phần quan trọng và không thể thiếu, đóng vai trò cốt lõi trong việc duy trì và xử lý thông tin nhị phân, cụ thể là các giá trị 0 hoặc 1 trong một khoảng thời gian xác định. Khái niệm này đề cập đến các thành phần hoặc khối mạch điện tử được thiết kế đặc biệt nhằm đảm bảo rằng thông tin có thể được lưu giữ một cách hiệu quả và sử dụng lại khi cần thiết trong các giai đoạn xử lý dữ liệu tiếp theo.

Phần tử lưu trữ hoạt động dựa trên nguyên lý giữ vững trạng thái của bit cho đến khi có một tín hiệu điều khiển hoặc một sự kiện cụ thể được kích hoạt để thay đổi trạng thái đó, qua đó cung cấp khả năng duy trì thông tin một cách liên tục và đáng tin cậy. Điều này cho phép các hệ thống mạch số thực hiện các chức năng tuần tự với độ chính xác cao, biến phần tử lưu trữ trở thành nền tảng quan trọng trong việc hỗ trợ các hoạt động phức tạp.

Ứng dụng của phần tử lưu trữ rất đa dạng, từ việc lưu trữ tạm thời đến lưu trữ dài hạn, chúng đóng vai trò trung tâm trong nhiều cấu trúc mạch số, giúp quản lý dữ liệu một cách hiệu quả và đảm bảo tính đồng bộ trong hoạt động của hệ thống. Nhờ vậy, phần tử lưu trữ không chỉ giới hạn ở việc đơn thuần lưu giữ dữ liệu mà còn liên quan chặt chẽ đến hiệu suất tổng thể và độ tin cậy của toàn bộ hệ thống, tạo điều kiện thuận lợi cho sự phát triển của các ứng dụng điện tử hiện đại đòi hỏi khả năng xử lý thông tin nhanh chóng và chính xác.

Flip-flops và latches là hai phần tử lưu trữ phổ biến nhất trong lĩnh vực điện tử. Điểm khác biệt chính của hai loại mạch này là cách chúng phản ứng với sự thay đổi của tín hiệu đầu vào. Một phần tử latch (chốt) thay đổi tín hiệu ngõ ra mỗi khi ngõ vào thay đổi giúp nó luôn sẵn sàng phản hồi khi cần. Mặt khác, một phần tử flip-flop chỉ thay đổi tín hiệu ngõ ra ở một thời điểm cụ thể, chẳng hạn như khi tín hiệu điều khiển đi từ thấp lên cao hoặc ngược lại. Điều này giúp cho flip-flops trở nên ổn định hơn latch trong nhiều trường hợp. Nói ngắn gọn, latch luôn sẵn sàng thay đổi tín hiệu trong khi flip-flops chỉ hoạt động trong một thời điểm cụ thể.

1.1.2. Latch là gì?

1.1.2.a. Định nghĩa

Latch là một mạch số có khả năng lưu trữ 1 bit thông tin và giữ giá trị đó đến khi được cập nhật bởi một tín hiệu đầu vào mới. Latch là phần tử lưu trữ cơ bản hoạt động theo mức logic, hữu dụng cho các thiết kế mạch tuần tự bất đồng bộ. Latch có hai trạng thái ổn định là mức logic cao(1) và thấp(0).

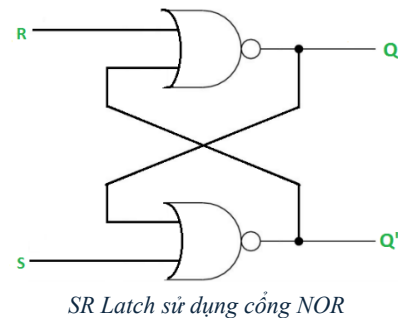
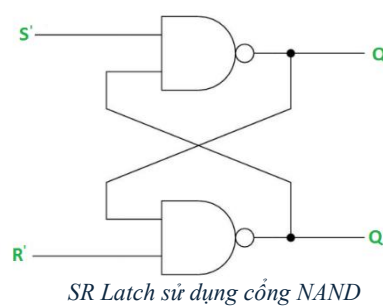
Latch được dùng trong các hệ thống số như một thành phần lưu trữ tạm thời dùng để lưu thông tin dưới dạng nhị phân. Latch có thể được xây dựng từ nhiều loại cổng logic như AND, OR, NOT, NAND, NOR và Transmission Gate.

1.1.2.b. Các loại Latch:

- **SR Latch:** hay còn gọi là Set-Reset Latch là loại latch đơn giản nhất được tạo thành từ 2 đầu vào là S(Set) và R(Reset). Tín hiệu S đưa ngõ ra lên mức cao(1), trong khi tín hiệu R đưa ngõ ra xuống mức thấp(0). Khi S và R không hoạt động, Latch sẽ giữ trạng thái trước đó. Khi cả S và R cùng hoạt động, Latch sẽ rơi vào trạng thái “không xác định”(undefined state). Latch SR là khối cơ bản nhất cấu thành nên các loại **flip-flop** khác.

S	R	Q	Qbar
0	0	Latch	Latch
0	1	0	1
1	0	1	0
1	1	0	0

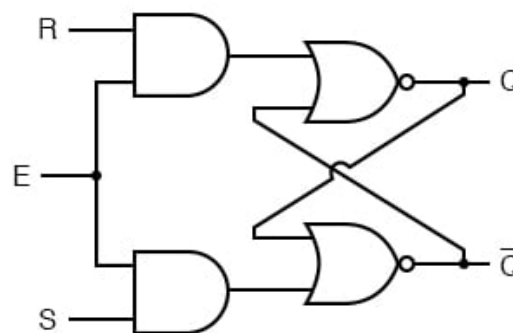
Bảng trạng thái SR Latch



- **SR Latch với tín hiệu Enable:** Khi tín hiệu Enable bằng 1, mạch hoạt động tương tự SR Latch. Khi tín hiệu Enable bằng 0, Latch sẽ giữ trạng thái trước đó.

Enable	S	R	Q_{n+1}
0	×	×	Q_n
1	0	0	Q_n
1	0	1	0
1	1	0	1
1	1	1	×

Bảng trạng thái SR Latch với Enable

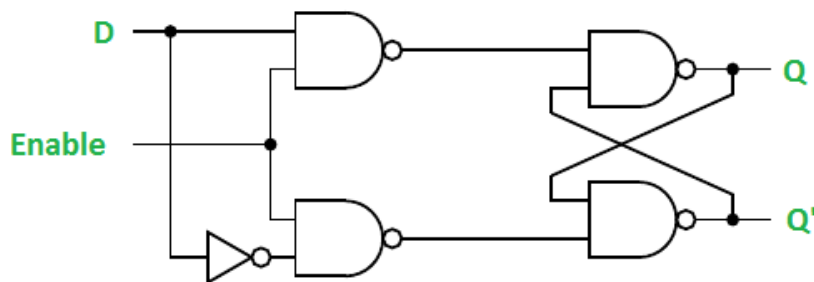


SR Latch với Enable

- **D Latch:** Khi tín hiệu Enable bằng 1, ngõ ra Q sẽ có giá trị bằng ngõ vào D. Khi tín hiệu Enable bằng 0, ngõ ra Q sẽ giữ giá trị trước đó.

Enable	D	Q	Qbar
0	×	Latch	Latch
1	0	0	1
1	1	1	0

Bảng trạng thái D Latch

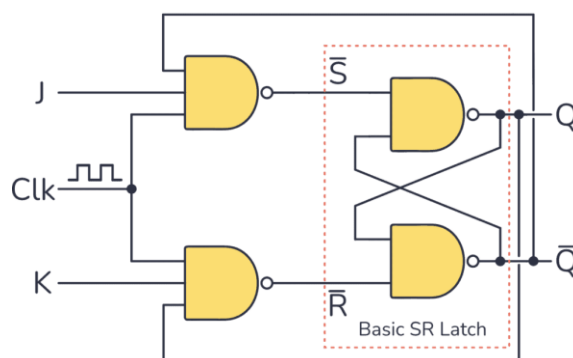


D Latch

- **JK Latch:** Hoạt động tương tự SR Latch nhưng khi $J = K = 1$ thì ngõ ra sẽ đảo trạng thái thay vì rơi vào trạng thái cấm.

J	K	Q
0	0	Latch
0	1	0
1	0	1
1	1	0

Bảng trạng thái JK Latch

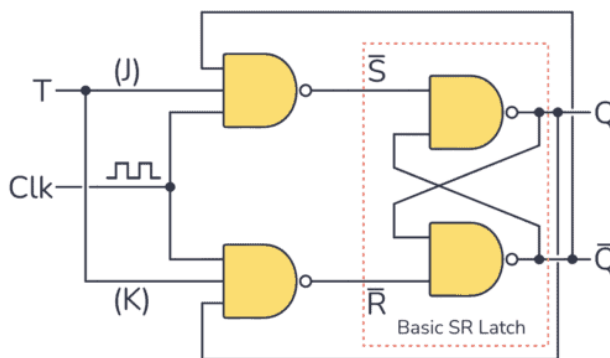


JK Latch

- **T Latch:** thực chất là JK Latch nhưng ngõ vào JK được điều khiển bởi một tín hiệu T duy nhất.

T	Q	Q _{n+1}
0	0	0
0	1	0
1	0	1
1	1	0

Bảng trạng thái T Latch



T Latch

1.1.3. Flip-Flop là gì?

1.1.3.a. Định nghĩa

Flip-flop là mạch duy trì trạng thái nó lưu trữ cho đến khi có lệnh từ đầu vào để thay đổi trạng thái. Một flip-flop cơ bản có thể được cấu thành bằng cách sử dụng 4 cổng NAND hoặc 4 cổng NOR. Flip-flop được biết tới như một phần tử lưu trữ dữ liệu cơ bản. Giống như Latch, Flip-flop cũng có hai trạng thái đó là logic cao(1) và thấp(0), cũng được dùng để lưu trữ 1 bit thông tin hay dữ liệu.

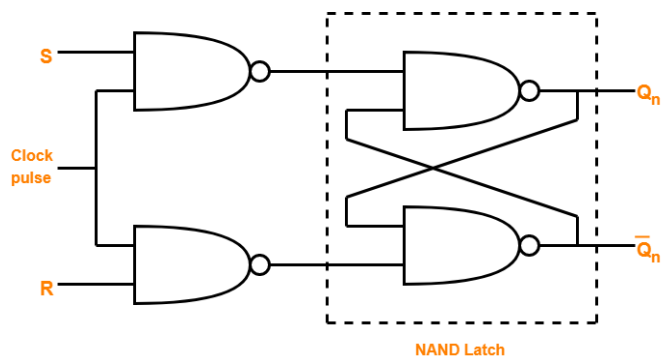
Flip-flop là một mạch tuần tự hoạt động dựa theo cạnh logic thay vì mức logic như Latch, điều này có nghĩa tại một chu kỳ Clock, Flip-flop chỉ lấy giá trị với số lần hữu hạn tại cạnh của xung Clock(cạnh lên hoặc xuống) và giữ nó đến cạnh tiếp theo(cạnh lên hoặc xuống). Điều này khiến cho Flip-flop có xu hướng ổn định hơn so với Latch(lấy giá trị liên tục) vốn nhạy cảm với nhiễu và vọt lố. Do đó, Flip-flop thường được ưu tiên sử dụng trong các thiết kế đồng bộ để đảm bảo tính toàn vẹn của dữ liệu.

1.1.3.b. Các loại Flip-flop:

- **SR Flip-flop:** là một Flip-flop với hai input S(Set) và R(Reset). Khi đến cạnh của xung Clock(cạnh lên hoặc xuống)
 - Khi $S = 1$, ngõ ra Q sẽ bằng 1
 - Khi $S = 0$, ngõ ra Q sẽ bằng 0
 - Khi $S = 0, R = 0$, ngõ ra Q sẽ giữ nguyên
 - Khi $S = 1, R = 1$, ngõ ra Q sẽ ở trạng thái cấm(invalid)

S	R	Q_{n+1}
0	0	Q_n
0	1	0
1	0	1
1	1	×

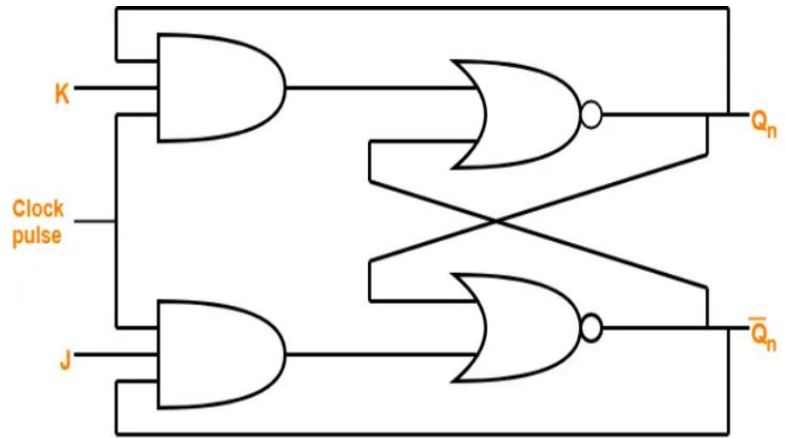
Bảng trạng thái SR Flip-flop



- **JK Flip-flop:** JK Flip-flop là phiên bản cải tiến hơn của SR Flip-flop khi mà trạng thái cấm xuất hiện ở SR Flip-flop đã được thay bằng trạng thái đảo(Toggle)

S	R	Q_{n+1}
0	0	Q_n
0	1	0
1	0	1
1	1	$\overline{Q_n}$

Bảng trạng thái JK Flip-flop

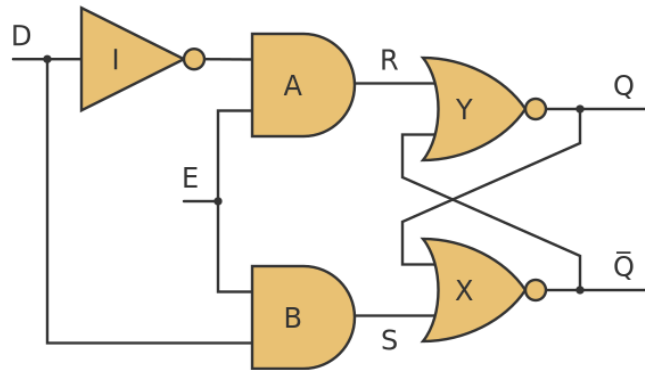


JK Flip-flop

- **D Flip-flop:** D(Data/Delay) Flip-flop là phiên bản chỉnh sửa của SR Flip-flop khi ta cấp cho tín hiệu R giá trị của S đảo. Khi có cạnh xung Clock(cạnh lên hoặc xuống), ngõ ra Q sẽ lấy giá trị của D và giữ đến cạnh xung Clock tiếp theo.

D	CLK	Q_n	Q_{n+1}
×	↓	Q_{n-1}	Q_n
0	↑	Q_{n-1}	0
1	↑	Q_{n-1}	1

Bảng trạng thái D Flip-flop(Cạnh lên)

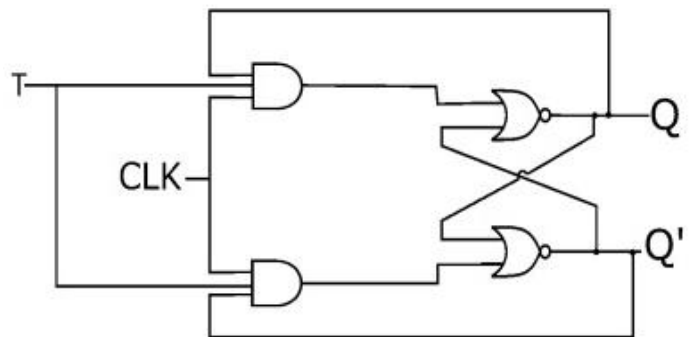


D Flip-flop

- **T Flip-flop:** T Flip-flop được tạo thành từ JK Flip-flop bằng cách nối chung ngõ vào J và K với nhau. Khi tín hiệu T = 1 tại cạnh xung Clock(cạnh lên hoặc xuống), tín hiệu ngõ ra sẽ đảo lại so với tín hiệu ngõ ra ở chu kỳ trước. Khi tín hiệu T = 0 thì ngõ ra giữ nguyên trạng thái.

T	CLK	Q_n	Q_{n+1}
0	×	Q_{n-1}	Q_n
1	↑	Q_{n-1}	$\overline{Q_n}$

Bảng trạng thái T Flip-flop(Cạnh lên)



T Flip-flop

1.1.4. So sánh giữa Latch và Flip-flop:

Tiêu chí	Latch	Flip-flop
Đồng bộ hóa	Không đồng bộ hoặc chỉ đồng bộ với tín hiệu enable (mức logic).	Đồng bộ với tín hiệu đồng hồ (clock), thường là cạnh lên/xuống.
Hoạt động	Đầu ra thay đổi khi tín hiệu enable hoạt động và đầu vào thay đổi.	Đầu ra chỉ thay đổi tại thời điểm cạnh đồng hồ (edge-triggered).
Tín hiệu điều khiển	Sử dụng tín hiệu enable (mức cao/thấp).	Sử dụng tín hiệu clock (cạnh lên hoặc xuống).
Độ phức tạp	Đơn giản hơn, thường sử dụng ít cổng logic hơn.	Phức tạp hơn, cần thêm mạch để xử lý cạnh đồng hồ (master-slave hoặc edge-triggered).
Tốc độ	Nhanh hơn do không cần đồng bộ với clock, dễ bị nhiễu (glitch).	Chậm hơn một chút do phải đợi cạnh clock, nhưng đáng tin cậy hơn.
Nhạy với nhiễu	Nhạy hơn với nhiễu (glitch) vì đầu ra có thể thay đổi ngay khi đầu vào thay đổi trong lúc enable.	Ít nhạy hơn với nhiễu vì chỉ cập nhật trạng thái tại cạnh clock.

1.1.5. Cấu trúc Flip-flop master-slave sử dụng Latch:

1.1.5.a. Định nghĩa, cấu trúc, nguyên lý:

- **Định nghĩa:**

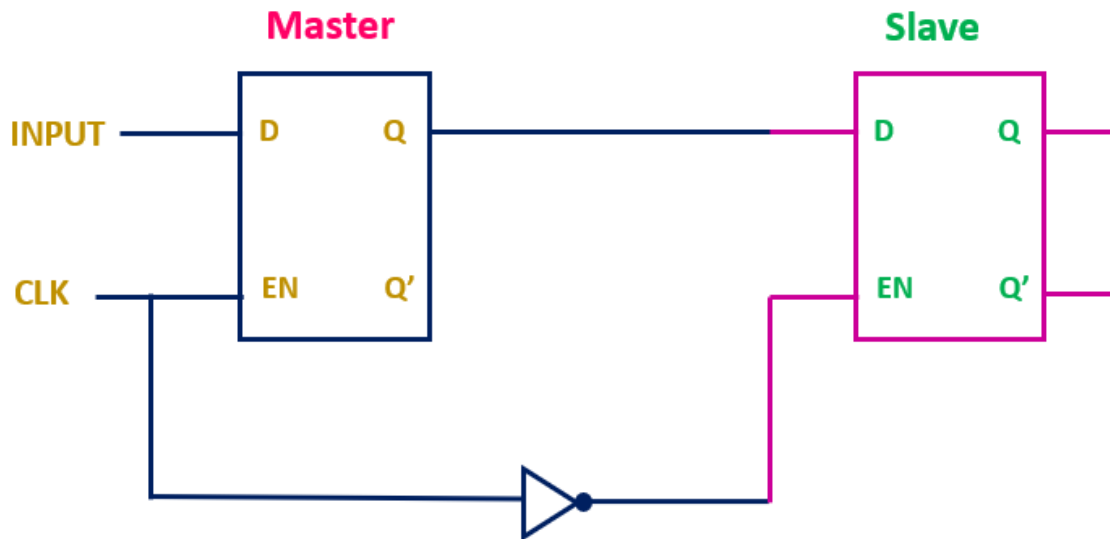
Flip-flop master-slave là một loại Flip-flop edge-triggered(kích hoạt theo cạnh), được xây dựng từ hai latch nối tiếp (master latch và slave latch).

Cấu trúc này sử dụng tín hiệu đồng hồ (clock) để đảm bảo đầu ra chỉ thay đổi tại các thời điểm cụ thể (thường là cạnh lên hoặc cạnh xuống của clock), giúp đồng bộ hóa dữ liệu trong các hệ thống số.

- **Cấu trúc:**

Flip-flop master-slave gồm hai latch được kết nối tuần tự. Master Latch: Là Latch đầu tiên, nhận dữ liệu đầu vào (D) và được điều khiển bởi tín hiệu Clock có mức logic thấp/cao. Slave Latch: Là Latch thứ hai, nhận đầu ra của Master Latch và cung cấp đầu ra cuối cùng (Q) của Flip-flop và được điều khiển bởi tín hiệu Clock có mức logic cao/thấp.

Tín hiệu Clock được sử dụng để điều khiển hai Latch sao cho chúng hoạt động thay phiên(bổ sung/complementary) nhau sau nửa chu kỳ Clock. Khi Clock ở mức thấp(hoặc cao, tùy thiết kế), Master Latch được kích hoạt, còn Slave Latch vào trạng thái giữ giá trị. Khi Clock chuyển sang mức cao(hoặc thấp), Master Latch vào trạng thái giữ giá trị và Slave Latch được kích hoạt.



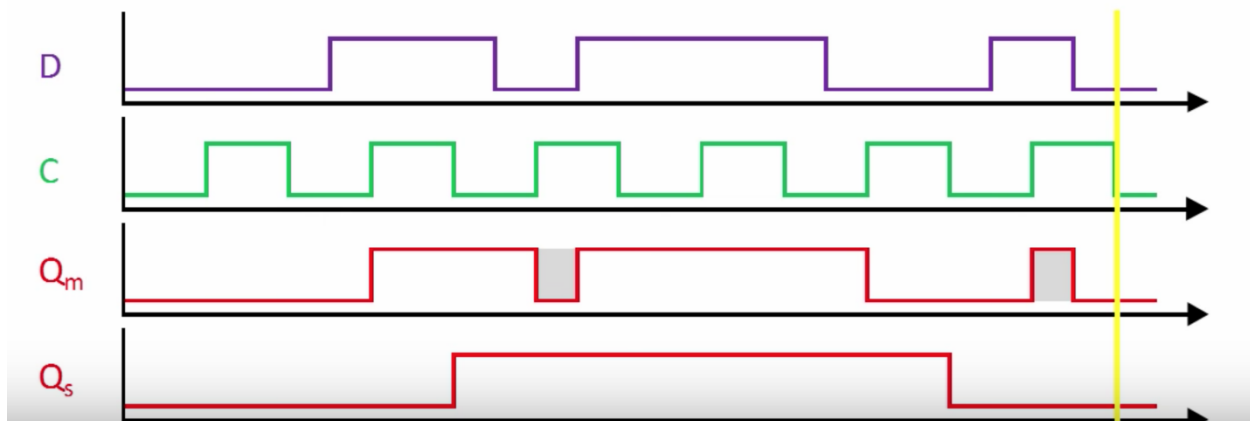
DFF sử dụng Latch với cấu trúc Master-Slave

- **Nguyên lý hoạt động:**

Trong giai đoạn Master Latch bật(master active), Master Latch nhận và lưu trữ giá trị đầu vào D, trong khi slave latch giữ nguyên giá trị trước đó.

Khi Clock chuyển mức logic(ví dụ: từ 1 sang 0), Master Latch khóa giá trị hiện tại, và Slave Latch lấy giá trị từ master latch để cập nhật đầu ra Q.

Kết quả: Đầu ra Q chỉ thay đổi tại cạnh clock (edge-triggered), đảm bảo đồng bộ và tránh nhiễu trong suốt chu kỳ clock.



Dạng sóng minh họa của một D Flip-flop

1.1.5.b. Vai trò của Latch trong Master-Slave Flip-flop:

Tính bổ sung(complementary): Hai latch hoạt động xen kẽ (master và slave không bao giờ đồng thời trong trạng thái transparent), đảm bảo rằng đầu vào không ảnh hưởng trực tiếp đến đầu ra trong cùng một chu kỳ clock. Điều này giúp loại bỏ vấn đề vọt lố thường gặp ở các Latch đơn lẻ.

Tăng độ tin cậy: Cấu trúc master-slave sử dụng latch để tạo ra một flip-flop có khả năng đồng bộ hóa, phù hợp với các hệ thống số phức tạp như thanh ghi, bộ đếm, hoặc máy trạng thái.

1.1.5.c. Ưu điểm và nhược điểm:

- **Ưu điểm:**

Đơn giản để triển khai từ các latch cơ bản (như D Latch).

Đồng bộ hóa tốt nhờ cơ chế edge-triggered, giảm thiểu nhiễu và lỗi do thay đổi đầu vào không mong muốn.

- **Nhược điểm:**

Cấu trúc phức tạp hơn Latch đơn lẻ, cần nhiều cổng logic hơn, dẫn đến tăng diện tích và tiêu thụ năng lượng.

Có thể gặp hiện tượng metastability nếu đầu vào thay đổi gần cạnh Clock, đòi hỏi thiết kế cẩn thận.

1.2. VAI TRÒ VÀ ỨNG DỤNG CỦA FLIP-FLOP

1.2.1. Vai trò của Flip-flop trong vi xử lý hiện đại

Flip-flop là một trong những thành phần quan trọng nhất trong vi xử lý hiện đại, đóng vai trò nền tảng trong việc đảm bảo hoạt động ổn định, chính xác và hiệu quả của các hệ thống số phức tạp. Với khả năng lưu trữ trạng thái và đồng bộ hóa tín hiệu theo tín hiệu đồng hồ (clock), flip-flop hỗ trợ vi xử lý thực hiện các tác vụ xử lý dữ liệu và điều khiển một cách đáng tin cậy. Dưới đây là các vai trò chính của flip-flop trong vi xử lý:

- **Lưu trữ, duy trì trạng thái:**

Flip-flop, đặc biệt là các loại như D Flip-flop, được sử dụng để lưu trữ các giá trị logic (0 hoặc 1) trong suốt quá trình xử lý của vi xử lý. Chúng hoạt động như các đơn vị bộ nhớ nhỏ, giữ các trạng thái tạm thời cần thiết cho các thao tác tính toán, điều khiển hoặc truyền dữ liệu. Khả năng này đảm bảo rằng vi xử lý có thể duy trì thông tin quan trọng giữa các chu kỳ đồng hồ, tạo nền tảng cho việc xử lý dữ liệu liên tục và chính xác.

- **Đồng bộ hóa tín hiệu và hoạt động**

Vi xử lý hiện đại hoạt động trong môi trường đồng bộ, nơi các thao tác được thực hiện theo các chu kỳ đồng hồ chính xác. Flip-flop, với cơ chế kích hoạt theo cạnh (edge-triggered), đảm bảo rằng các tín hiệu và dữ liệu được cập nhật chỉ tại các thời điểm cụ thể (cạnh lên hoặc cạnh xuống của clock). Điều này loại bỏ nguy cơ nhiễu tín hiệu, hiện tượng chạy đua (race condition), hoặc lỗi do thời gian trễ, giúp vi xử lý hoạt động ổn định ngay cả ở tốc độ cao.

- **Tăng cường độ tin cậy và hiệu suất:**

Nhờ khả năng đồng bộ hóa và lưu trữ trạng thái, flip-flop giúp vi xử lý đạt được độ tin cậy cao trong các ứng dụng đòi hỏi hiệu suất lớn. Chúng giảm thiểu lỗi do nhiễu tín hiệu hoặc thay đổi không đồng bộ, đồng thời hỗ trợ vi xử lý xử lý các tác vụ phức tạp với tốc độ nhanh và chính xác. Trong các hệ thống số hiện đại, flip-flop là yếu tố không thể thiếu để đảm bảo vi xử lý đáp ứng các yêu cầu khắt khe về hiệu năng.

- **Linh hoạt trong thiết kế vi xử lý**

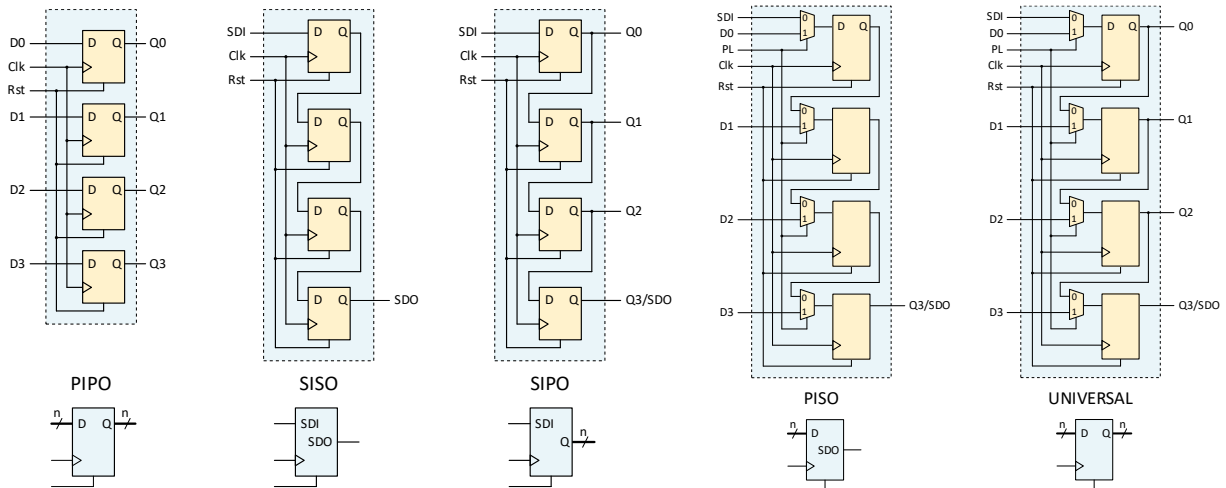
Flip-flop cung cấp sự linh hoạt trong thiết kế vi xử lý, cho phép các nhà thiết kế tích hợp chúng vào nhiều khối chức năng khác nhau với các yêu cầu cụ thể. Sự đa dạng của các loại flip-flop (như D, JK, T) và khả năng tùy chỉnh của chúng giúp vi xử lý đáp ứng được các nhu cầu đa dạng, từ xử lý tín hiệu số tốc độ cao đến quản lý trạng thái trong các hệ thống điều khiển phức tạp.

1.2.2. Ứng dụng của Flip-flop trong mạch số

Flip-flop là thành phần cốt lõi trong các mạch số nhờ khả năng lưu trữ trạng thái và đồng bộ hóa với tín hiệu đồng hồ (clock). Các ứng dụng chính của flip-flop bao gồm thanh ghi, bộ đếm, máy trạng thái hữu hạn (FSM), và pipeline, v.v...

• Thanh ghi(Register):

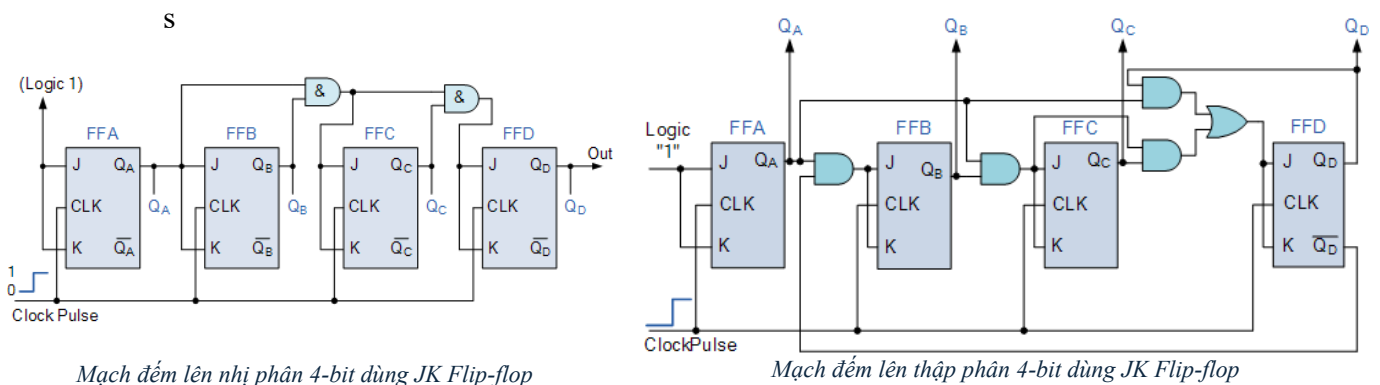
Thanh ghi là tập hợp các Flip-flop(thường là Flip-flop D) mắc nối tiếp hoặc song song nhau, được sử dụng để lưu trữ tạm thời một dữ liệu số trong một chu kỳ Clock.



Register(PIPO,SISO,SIPO,PISO) 4-bit sử dụng D Flip-flop

• Bộ đếm(Counter):

Flip-flop(thường là T Flip-flop hoặc JK Flip-flop) được sử dụng để tạo ra các bộ đếm, đếm số lần xuất hiện xung Clock hoặc các sự kiện khác.

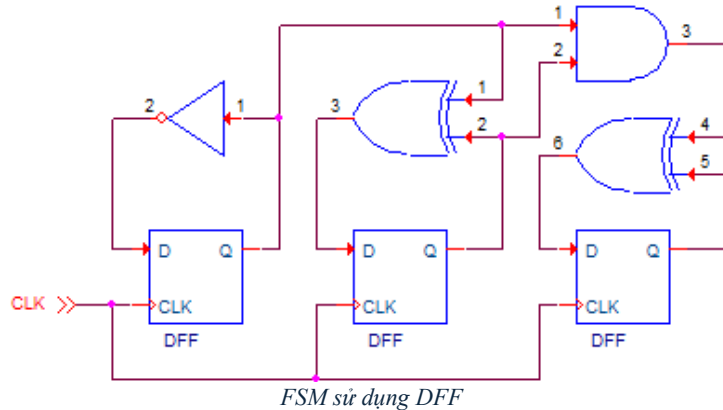
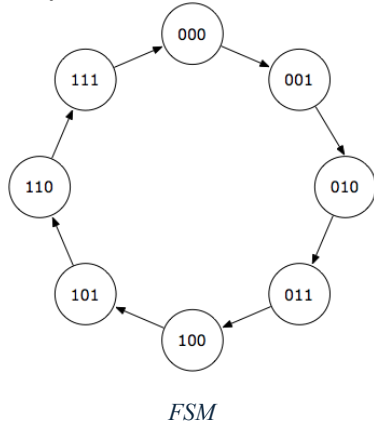


Mạch đếm lên nhị phân 4-bit dùng JK Flip-flop

Mạch đếm lên thập phân 4-bit dùng JK Flip-flop

- **Máy trạng thái hữu hạn(Finite State Machine/FSM):**

Flip-flop lưu trữ trạng thái hiện tại của FSM, cho phép chuyển đổi trạng thái dựa trên đầu vào và tín hiệu Clock. Mỗi khi có cạnh lên xung Clock, trạng thái của FSM sẽ thay đổi theo thứ tự đã được tính toán trước.



1.3. FLIP-FLOP VỚI SET/RESET BẤT ĐỒNG BỘ

1.3.1. Khái niệm về tín hiệu Set/Reset bất đồng bộ:

- **Khái niệm:**

Tín hiệu bất đồng bộ(asynchronous signal) là tín hiệu điều khiển hoặc dữ liệu không phụ thuộc vào tín hiệu đồng hồ(clock) trong một mạch số. Trong bối cảnh Flip-flop, tín hiệu bất đồng bộ thường được sử dụng để điều khiển các chức năng như Set(đặt đầu ra về trạng thái logic 1) hoặc Reset(đặt đầu ra về trạng thái logic 0) mà không cần đợi đến cạnh đồng hồ(edge-triggered).

- **Đặc điểm:**

Không đồng bộ với tín hiệu Clock: Tín hiệu bất đồng bộ hoạt động độc lập với chu kỳ hoặc cạnh của tín hiệu đồng hồ, cho phép flip-flop thay đổi trạng thái ngay lập tức khi tín hiệu Set hoặc Reset được kích hoạt.

Ưu tiên cao: Trong các flip-flop có Set/Reset bất đồng bộ, tín hiệu Set hoặc Reset có quyền ưu tiên cao hơn tín hiệu đồng hồ, nghĩa là trạng thái đầu ra sẽ thay đổi ngay khi các tín hiệu này được kích hoạt, bất kể trạng thái của clock.

Cơ chế hoạt động: Tín hiệu Set bất đồng bộ đặt đầu ra $Q = 1$, còn tín hiệu Reset bất đồng bộ đặt $Q = 0$, thường được tích hợp thông qua các cổng logic bổ sung trong cấu trúc flip-flop (như D Flip-flop hoặc JK Flip-flop).

1.3.2. Ý nghĩa của Set/Reset trong hệ thống số:

Tín hiệu Set và Reset bất đồng bộ trong flip-flop đóng vai trò quan trọng trong việc điều khiển trạng thái của các mạch số, đảm bảo hệ thống hoạt động đúng theo yêu cầu trong các tình huống đặc biệt. Các tín hiệu này cho phép đặt hoặc xóa trạng thái đầu ra của flip-flop ($Q = 1$ với Set, $Q = 0$ với Reset) mà không cần phụ thuộc vào tín hiệu đồng hồ, mang lại sự linh hoạt và hiệu quả trong thiết kế.

Ưu điểm:

- **Khởi tạo trạng thái ban đầu:**

Tín hiệu Set/Reset bất đồng bộ được sử dụng để thiết lập trạng thái ban đầu cho flip-flop khi hệ thống khởi động (power-on) hoặc sau một sự kiện đặc biệt (như lỗi hệ thống). Đảm bảo rằng các thanh ghi, bộ đếm, hoặc máy trạng thái bắt đầu ở một trạng thái xác định, giúp hệ thống hoạt động ổn định và dự đoán được.

- **Phản ứng nhanh với các sự kiện bất thường**

Tín hiệu Set/Reset cho phép flip-flop thay đổi trạng thái ngay lập tức khi nhận được tín hiệu điều khiển, không cần đợi cạnh đồng hồ. Điều này rất hữu ích trong các tình huống cần xử lý khẩn cấp, như ngắt hệ thống (interrupt), khôi phục sau lỗi, hoặc điều chỉnh trạng thái trong các mạch điều khiển.

- **Tăng tính linh hoạt trong thiết kế mạch**

Tín hiệu Set/Reset bất đồng bộ cung cấp khả năng điều khiển trực tiếp trạng thái của flip-flop, phù hợp với các ứng dụng không yêu cầu đồng bộ hóa chặt chẽ với clock. Cho phép các nhà thiết kế tùy chỉnh trạng thái của hệ thống trong các kịch bản đặc biệt, như chế độ kiểm tra (test mode) hoặc vận hành không đồng bộ.

- **Hỗ trợ quản lý lỗi và khôi phục hệ thống**

Tín hiệu Reset bất đồng bộ thường được sử dụng để đưa hệ thống về trạng thái an toàn khi xảy ra lỗi, như treo hệ thống hoặc dữ liệu không hợp lệ. Giúp tăng độ tin cậy của hệ thống bằng cách cung cấp cơ chế khôi phục nhanh chóng, giảm thiểu nguy cơ hỏng hóc hoặc hoạt động sai lệch.

Nhược điểm:

- **Nguy cơ nhiễu tín hiệu (glitch):**

Vì tín hiệu Set/Reset bất đồng bộ không đồng bộ với clock, chúng có thể gây ra nhiễu hoặc trạng thái không mong muốn nếu được kích hoạt không đúng thời điểm, đặc biệt trong các hệ thống tốc độ cao.

- **Tăng độ phức tạp trong thiết kế:**

Việc thêm Set/Reset bất đồng bộ đòi hỏi tích hợp các công logic bổ sung vào Flip-flop, làm tăng diện tích chip, tiêu thụ năng lượng, và chi phí sản xuất.

- **Khó khăn trong phân tích thời gian (timing analysis):**

Tín hiệu bất đồng bộ không tuân theo chu kỳ clock, gây khó khăn trong việc dự đoán và kiểm soát thời gian truyền tín hiệu, có thể dẫn đến lỗi đồng bộ hóa trong các hệ thống phức tạp.

- **Nguy cơ metastability:**

Nếu tín hiệu Set/Reset được kích hoạt gần cạnh clock, Flip-flop có thể rơi vào trạng thái metastability, nơi đầu ra không ổn định, ảnh hưởng đến độ tin cậy của vi mạch.

- **Yêu cầu quản lý tín hiệu cẩn thận:**

Để tránh kích hoạt không mong muốn, tín hiệu Set/Reset cần được điều khiển chính xác bằng các mạch chống nhiễu (debouncing) hoặc lọc tín hiệu, làm tăng độ phức tạp của hệ thống.

Kết luận:

Tín hiệu Set/Reset bất đồng bộ trong Flip-flop mang lại khả năng khởi tạo, điều khiển nhanh, và khôi phục trạng thái hệ thống một cách linh hoạt và hiệu quả. Chúng đóng vai trò quan trọng trong việc đảm bảo hệ thống số hoạt động đúng cách trong các điều kiện khởi động, sự kiện bất thường, hoặc khi cần quản lý lỗi, góp phần nâng cao độ tin cậy và tính linh hoạt của các thiết kế mạch hiện đại nhưng cũng đi kèm với các thách thức về nhiễu, độ phức tạp, và phân tích thời gian. Để tận dụng tối đa ưu điểm và giảm thiểu nhược điểm, các nhà thiết kế cần cân nhắc kỹ lưỡng trong việc sử dụng Set/Reset bất đồng bộ, đảm bảo tích hợp các biện pháp kiểm soát tín hiệu phù hợp trong các ứng dụng thực tế.

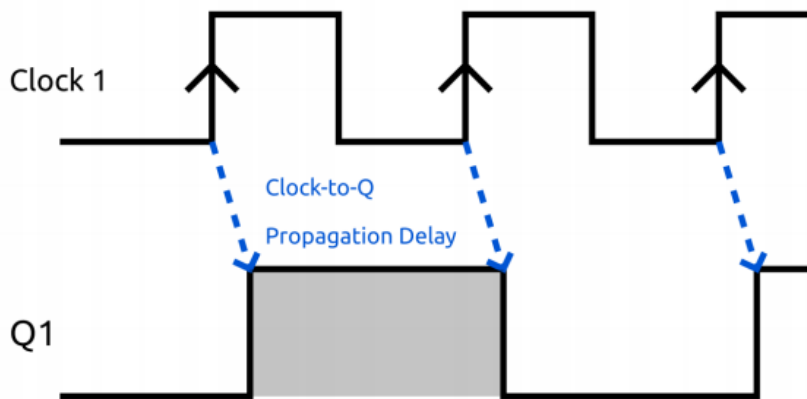
1.4. CÁC THÔNG SỐ QUAN TRỌNG CẦN KHẢO SÁT

1.4.1. Thông số thời gian:

1.4.1.a. Propagation Delay(Clock-to-Q delay/ t_{pd}/t_{CQ}):

Là thời gian từ lúc cạnh Clock kích hoạt đến khi đầu ra Q của Flip-flop ổn định ở giá trị mới. Trong Flip-flop Edge-Triggered, t_{pd} biểu thị thời gian cần thiết để dữ liệu đầu vào D được truyền sang đầu ra Q sau khi tín hiệu đồng bộ được kích hoạt.

t_{pd} thường được đo từ thời điểm 50% của biên độ cạnh đồng hồ đến điểm 50% của biên độ tín hiệu đầu ra, đảm bảo tính chính xác trong phân tích thời gian.



Ý nghĩa:

- **Đo lường tốc độ phản hồi:** t_{pd} xác định tốc độ mà flip-flop có thể cung cấp dữ liệu ra các khối logic tiếp theo trong vi mạch, chẳng hạn như các cổng logic hoặc thanh ghi khác. Một t_{pd} nhỏ cho thấy flip-flop có khả năng phản hồi nhanh, phù hợp với các hệ thống tốc độ cao.
- **Ảnh hưởng đến thời gian truyền tín hiệu:** Trong một chuỗi logic (logic path), t_{pd} góp phần vào tổng thời gian trễ (total propagation delay) của mạch, ảnh hưởng đến chu kỳ đồng hồ tối thiểu ($t_{clockmin}$) và tần số hoạt động tối đa (f_{max}).
- **Đồng bộ hóa trong hệ thống:** t_{pd} là yếu tố quan trọng trong việc đảm bảo dữ liệu được truyền đúng thời điểm giữa các khối chức năng, đặc biệt trong các kiến trúc pipeline hoặc máy trạng thái hữu hạn (FSM).

1.4.1.b. Rise Time và Fall Time(t_{rise}/t_{fall})

- **Thời gian tăng (t_{rise}):**

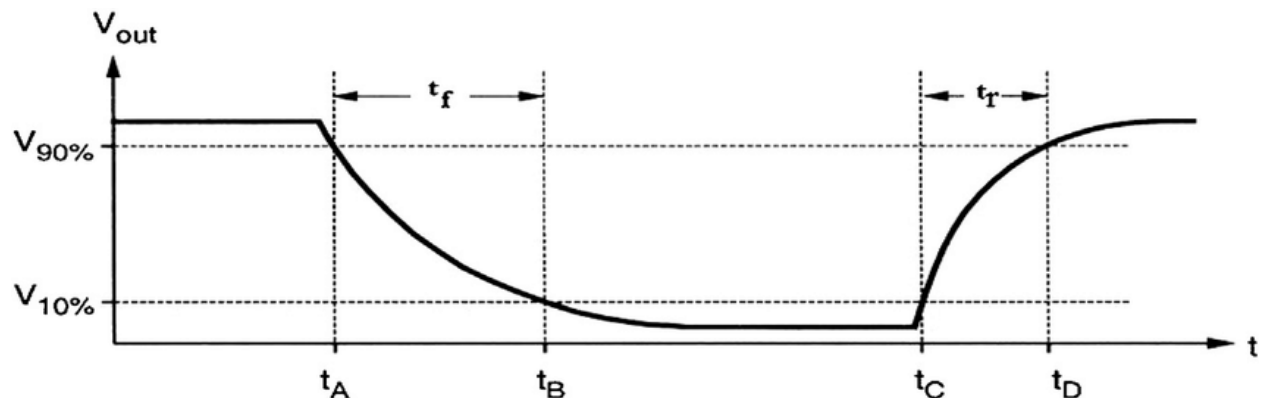
Là thời gian cần thiết để tín hiệu đầu ra Q (hoặc \bar{Q}) chuyển từ mức logic thấp (0, thường là 10% V_{DD}) lên mức logic cao (1, thường là 90% V_{DD}).

Ý nghĩa: Phản ánh tốc độ chuyển trạng thái của DFF, ảnh hưởng đến độ sắc nét của tín hiệu đầu ra và khả năng hoạt động ở tần số cao. Trong công nghệ CMOS 50nm, t_{rise} phụ thuộc vào điện dung tải và khả năng dẫn dòng của transistor PMOS trong cổng NAND hoặc Inverter.

- **Thời gian giảm (t_{fall}):**

Là thời gian cần thiết để tín hiệu đầu ra Q (hoặc \bar{Q}) chuyển từ mức logic cao (1, 90% V_{DD}) xuống mức logic thấp (0, 10% V_{DD}).

Ý nghĩa: Tương tự t_{rise} , t_{fall} ảnh hưởng đến tốc độ chuyển trạng thái và hiệu suất DFF. Trong CMOS 50nm, t_{fall} phụ thuộc vào khả năng dẫn dòng của transistor NMOS. Các yếu tố ảnh hưởng đến t_{rise} và t_{fall} là thư viện cmos.txt và kích thước W.



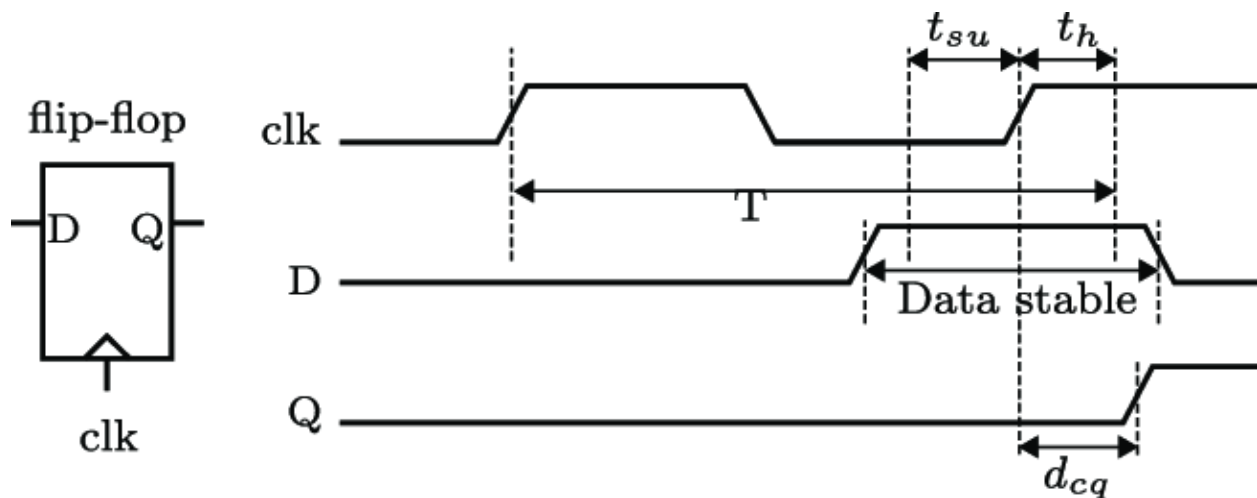
1.4.1.c. Setup Time và Hold Time (t_{setup}/t_{hold}):

- **Setup time(t_{setup}):**

Là khoảng thời gian tối thiểu mà tín hiệu đầu vào D của flip-flop phải ổn định (ở trạng thái logic 0 hoặc 1) trước khi cạnh đồng hồ (clock edge, thường là cạnh lên hoặc cạnh xuống) kích hoạt để đảm bảo dữ liệu được ghi chính xác vào flip-flop. Nếu tín hiệu thay đổi trong khoảng thời gian này, ngõ ra Q sẽ không thể lấy đúng tín hiệu ngõ vào D.

- **Hold time(t_{hold}):**

Là khoảng thời gian tối thiểu mà tín hiệu đầu vào D của flip-flop phải giữ ổn định (ở trạng thái logic 0 hoặc 1) sau khi cạnh đồng hồ (clock edge, thường là cạnh lên hoặc cạnh xuống) kích hoạt để đảm bảo dữ liệu được ghi chính xác vào flip-flop. Nếu tín hiệu thay đổi trong khoảng thời gian này, ngõ ra Q sẽ không thể lấy đúng tín hiệu ngõ vào D.



1.4.1.d. Removal Time và Recovery Time(t_{rem}/t_{rec}):

- **Recovery Time(t_{rec}):**

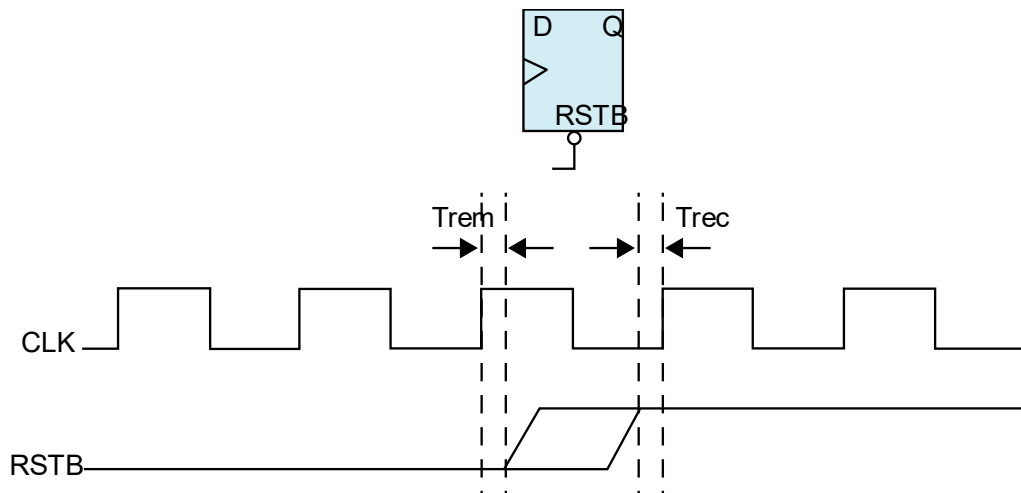
Là khoảng thời gian tối thiểu từ khi tín hiệu điều khiển bất đồng bộ (như Set hoặc Reset) bị vô hiệu hóa (chuyển từ trạng thái active sang inactive) đến khi cạnh đồng hồ tiếp theo (clock edge, thường là cạnh lên hoặc cạnh xuống) có thể kích hoạt mà không gây lỗi trong hoạt động của flip-flop.

Trong flip-flop có tín hiệu Set/Reset bất đồng bộ, $t_{recovery}$ đảm bảo rằng flip-flop đã thoát hoàn toàn khỏi trạng thái điều khiển bất đồng bộ (Set/Reset) và sẵn sàng để hoạt động bình thường theo tín hiệu đồng hồ.

- **Removal Time(t_{rem}):**

Là khoảng thời gian tối thiểu mà tín hiệu điều khiển bất đồng bộ (như Set hoặc Reset) phải được giữ ở trạng thái active (asserted) sau cạnh đồng hồ để đảm bảo flip-flop hoạt động đúng mà không bị lỗi.

Trong flip-flop có Set/Reset bất đồng bộ, $t_{removal}$ đảm bảo rằng tín hiệu Set/Reset có đủ thời gian để hoàn tất việc đặt trạng thái đầu ra ($Q = 1$ với Set, $Q = 0$ với Reset) trước khi bị vô hiệu hóa.



1.4.1.e. Chu kỳ tối thiểu/Tần số tối đa của Clock(t_{min}/f_{max}):

- **Chu kỳ tối thiểu(t_{min}):**

Là chu kỳ đồng hồ tối thiểu, tức khoảng thời gian ngắn nhất giữa hai cạnh đồng hồ liên tiếp (thường là từ cạnh lên đến cạnh lên tiếp theo hoặc từ cạnh xuống đến cạnh xuống) mà flip-flop có thể hoạt động ổn định mà không vi phạm các yêu cầu thời gian (như Setup Time, Hold Time, Clock-to-Q delay).

Ta có công thức: $t_{min} \geq t_{pd} + t_{setup}$.

- **Tần số tối đa(f_{max}):**

Là tần số đồng hồ tối đa mà flip-flop và hệ thống vi mạch có thể hoạt động ổn định mà không vi phạm các yêu cầu thời gian (như Setup Time, Hold Time, Clock-to-Q delay).

f_{max} được tính bằng công thức: $f_{max} = \frac{1}{t_{min}}$.

Thể hiện khả năng xử lý dữ liệu ở tốc độ cao của vi mạch, là một chỉ số quan trọng để đánh giá hiệu suất của các hệ thống đồng bộ như vi xử lý, FPGA, hoặc ASIC.

1.4.2. Công suất tiêu thụ:

Công suất tiêu thụ là một thông số quan trọng trong thiết kế flip-flop, ảnh hưởng đến hiệu suất năng lượng, thời lượng pin, và khả năng tản nhiệt của vi mạch. Công suất tiêu thụ bao gồm hai thành phần chính: công suất động và công suất tĩnh.

- **Công suất động:**

Công suất tiêu thụ do sự chuyển mạch (switching) của các tín hiệu trong Flip-flop, xảy ra khi trạng thái đầu ra (Q) hoặc các node nội bộ thay đổi ($0 \leftrightarrow 1$).

$$\text{Công thức: } P_{dynamic} = \alpha \times C \times V_{DD}^2 \times f$$

Trong đó:

α : Hệ số hoạt động, biểu thị tỷ lệ chuyển mạch của node ($0 \leq \alpha \leq 1$).

C : Tổng điện dung tại các node chuyển mạch.

V_{DD} : Điện áp cung cấp.

f : Tần số đồng hồ (liên quan đến mục 1.4.1).

- **Công suất tĩnh:**

Công suất tiêu thụ do dòng rò trong flip-flop, ngay cả khi không có chuyển mạch.

$$\text{Công thức: } P_{static} = I_{leak} \times V_{DD}$$

Trong đó: I_{leak} là Dòng rò qua transistor.

1.4.3. Độ ổn định và khả năng chống nhiễu:

Độ ổn định và khả năng chống nhiễu là các yếu tố quan trọng trong thiết kế flip-flop, đảm bảo hoạt động đáng tin cậy trong các điều kiện thực tế, nơi tín hiệu nhiễu, biến đổi công nghệ, và lỗi thời gian có thể xảy ra. Nội dung dưới đây phân tích các khía cạnh chính liên quan đến độ ổn định và khả năng chống nhiễu của flip-flop:

- **Metastability**

Là trạng thái không ổn định của flip-flop, xảy ra khi đầu ra Q dao động giữa 0 và 1 do vi phạm các thông số thời gian, đặc biệt là Tần số tối đa, Setup Time hoặc Hold Time (1.4.1).

- **Khả năng chống nhiễu(Noise Immunity)**

Khả năng chống nhiễu là khả năng của flip-flop duy trì trạng thái đúng khi gặp nhiễu tín hiệu, như nhiễu điện từ (EMI), nhiễu nguồn (power supply noise), hoặc nhiễu xuyên âm (crosstalk).

1.5. MỤC TIÊU VÀ PHẠM VI ĐỒ ÁN

1.5.1. Mục tiêu thực hiện:

Đồ án nhằm đạt được mục tiêu sau: Thiết kế và phân tích flip-flop hiệu quả: Phát triển hoặc tối ưu hóa một loại flip-flop (như D Flip-flop, JK Flip-flop, hoặc T Flip-flop) để đáp ứng các yêu cầu về hiệu suất, bao gồm tần số hoạt động cao (f_{max}), thời gian trễ thấp ($t_{pd}, t_{setup}, t_{hold}$).

1.5.2. Phạm vi thực hiện:

Phạm vi thực hiện của đồ án tập trung vào các hoạt động cụ thể để đạt được các mục tiêu đã đề ra trong mục 1.5.1, bao gồm thiết kế, mô phỏng, và phân tích flip-flop trong vi mạch số, sử dụng phần mềm LTspice và thư viện mô hình CMOS. Các nội dung chính bao gồm:

Thiết kế flip-flop:

Thiết kế một D Flip-flop có Set/Reset bất đồng bộ với cấu trúc tối ưu, đáp ứng các yêu cầu về thông số thời gian ($t_{pd}, t_{setup}, t_{hold}, t_{rem}, t_{hold}, f_{max}$).

Xây dựng sơ đồ mạch (schematic) của flip-flop dựa trên công nghệ CMOS, sử dụng các thông số transistor NMOS và PMOS từ thư viện cmos.txt (Chi tiết ở phần Phụ Lục).

Mô phỏng và phân tích hiệu suất bằng LTspice:

Sử dụng LTspice để mô phỏng hành vi của flip-flop ở mức transistor, dựa trên mô hình CMOS trong thư viện cmos.txt, nhằm đánh giá:

- Thông số thời gian: Đo lường $t_{pd}, t_{setup}, t_{hold}, t_{rem}, t_{hold}, f_{max}$ thông qua phân tích dạng sóng.
- Công suất tiêu thụ: Tính toán công suất động và công suất tĩnh.
- Độ ổn định: Đánh giá khả năng chống nhiễu (như nhiễu xuyên âm, nhiễu nguồn) thông qua mô phỏng tín hiệu nhiễu.

So sánh kết quả mô phỏng với các thông số lý thuyết hoặc flip-flop tiêu chuẩn để đánh giá tính tối ưu của thiết kế.

1.5.3. Giới hạn và hướng mở rộng:

Do phạm vi thời gian, công cụ đồ án tồn tại các giới hạn sau:

- Giới hạn về loại flip-flop và thiết kế: Chỉ tập trung vào thiết kế **D Flip-Flop có Set/Reset bất đồng bộ** sử dụng các latch, không bao gồm phân tích hoặc thiết kế các loại flip-flop khác (như JK Flip-Flop, T Flip-Flop) hoặc các cấu trúc flip-flop không có Set/Reset.
- Giới hạn về công nghệ chế tạo: Mô phỏng được thực hiện trên công nghệ **CMOS 50nm** với mô hình transistor NMOS và PMOS từ thư viện **cmos.txt**, không xem xét các công nghệ chế tạo tiên tiến hơn (như 7nm, 5nm) hoặc tác động của biến đổi công nghệ (process variation).
- Giới hạn về quy trình thiết kế: Thiết kế theo hướng custom flow chỉ dừng ở mức mô phỏng schematic trong LTspice, không bao gồm bố trí vật lý (layout) hoặc phân tích sau bố trí (post-layout simulation) để đánh giá tác động của điện dung ký sinh, điện trở dây dẫn, hoặc nhiễu xuyên âm lên các đặc tính thời gian hoặc công suất tiêu thụ.

Hướng mở rộng:

- Thiết kế và so sánh các loại flip-flop khác: Mở rộng thiết kế để bao gồm JK Flip-Flop hoặc T Flip-Flop sử dụng latch, so sánh với D Flip-Flop có Set/Reset về thời gian, công suất, độ ổn định, và diện tích để đánh giá ưu, nhược điểm.
- Áp dụng công nghệ chế tạo tiên tiến: Chuyển đổi thiết kế D Flip-Flop sang công nghệ CMOS tiên tiến (như 7nm hoặc 5nm) với mô hình transistor mới, tối ưu hóa $t_{clockmin}$, f_{max} , và công suất tiêu thụ, đồng thời phân tích tác động của dòng rò và biến đổi công nghệ.
- Thực hiện bố trí vật lý và phân tích sau bố trí: Tiến hành bố trí vật lý (layout) của D Flip-Flop bằng công cụ như Cadence Innovus hoặc Synopsys IC Compiler, và thực hiện mô phỏng sau bố trí để đánh giá chính xác điện dung ký sinh, nhiễu xuyên âm, và diện tích, cải thiện độ ổn định và hiệu suất.
- Phân tích chi tiết khả năng chống nhiễu: Mở rộng phân tích noise margin (NML, NMH) bằng cách sử dụng LTspice để mô phỏng đặc tính truyền điện áp (VTC), hoặc sử dụng công cụ như Cadence Spectre để đánh giá khả năng chống nhiễu phức tạp (như EMI, nhiễu nguồn) trong các ứng dụng thực tế.

CHƯƠNG 2: THIẾT KẾ VÀ XÂY DỰNG MẠCH

2.1. QUY TRÌNH THỰC HIỆN

Thiết kế và mô phỏng schematic của D Flip-Flop (DFF) có Set/Reset bất đồng bộ theo hướng custom flow sử dụng LTspice và thư viện cmos.txt (công nghệ CMOS 50nm) để đảm bảo mạch hoạt động đúng theo chức năng đồng bộ và bất đồng bộ. Quy trình bao gồm mục tiêu của thiết kế custom và các bước thực hiện chi tiết.

2.1.1. Mục tiêu của thiết kế Custom:

Thiết kế D Flip-Flop có Set/Reset bất đồng bộ theo hướng custom flow nhằm đạt được mục tiêu sau:

- Xây dựng mạch ở cấp transistor trong LTspice sử dụng mô hình NMOS/PMOS từ cmos.txt (CMOS 50nm) để kiểm soát chi tiết hành vi mạch, phù hợp với môn Kỹ thuật mạch điện tử (1.5.2).
- Tạo ra một DFF hoạt động đúng.
- Đảm bảo DFF thực hiện chức năng đồng bộ, lưu giá trị đầu vào D tại cạnh lên của xung đồng hồ (Clock) và xuất ra đầu ra Q chính xác (1.4.1).
- Hỗ trợ chức năng Set/Reset bất đồng bộ, cho phép đặt $Q = 1$ (Set) hoặc $Q = 0$ (Reset) độc lập với tín hiệu đồng hồ khi các đầu vào Set/Reset được kích hoạt.

Ý nghĩa: Thiết kế custom flow cho phép tạo ra DFF hoạt động đúng theo yêu cầu chức năng, cung cấp hiểu biết về quy trình thiết kế vi mạch số ở cấp transistor và làm nền tảng cho các phân tích

2.1.2. Các bước thực hiện

Quy trình thiết kế và mô phỏng D Flip-Flop có Set/Reset bất đồng bộ được thực hiện theo các bước sau để đảm bảo mạch hoạt động đúng:

2.1.2.a. Phân tích yêu cầu và thiết kế cấu trúc DFF:

Xác định chức năng của DFF: Lưu giá trị đầu vào D tại cạnh lên của đồng hồ và xuất ra Q, đồng thời hỗ trợ Set/Reset bất đồng bộ để đặt $Q = 1$ hoặc $Q = 0$ bất kỳ lúc nào.

Lựa chọn cấu trúc DFF dựa trên hai latch (master-slave) với các cổng logic bổ sung (như NOR hoặc NAND) để thực hiện chức năng Set/Reset bất đồng bộ, đảm bảo tín hiệu Set/Reset không phụ thuộc vào đồng hồ.

2.1.2.b. Xây dựng schematic trong LTspice:

Tạo sơ đồ mạch (schematic) của DFF ở cấp transistor trong LTspice, sử dụng mô hình transistor NMOS (N_50n) và PMOS (P_50n) từ thư viện cmos.txt.

Thực hiện thiết kế các khối cơ bản:

- Latch master và slave sử dụng cổng truyền (transmission gates) với cấu trúc feedback với cổng NAND/NOR.
- Thêm mạch logic cho Set/Reset bất đồng bộ, ví dụ, sử dụng cổng NOR để tích hợp Set/Reset vào latch.

Gán tỷ lệ W/L cơ bản cho transistor theo cấu hình **un-skew** (với $L = 50\text{nm}$ dựa trên cmos.txt) để đảm bảo mạch hoạt động.

2.1.2.c. Mô phỏng chức năng trong LTspice:

Thực hiện phân tích thời gian (transient analysis) trong LTspice để kiểm tra chức năng của DFF:

- Chức năng đồng bộ: Mô phỏng với các tín hiệu D và Clock khác nhau để xác nhận Q lưu đúng giá trị D tại cạnh lên của Clock.
- Chức năng Set/Reset bất đồng bộ: Mô phỏng với tín hiệu Set/Reset được kích hoạt độc lập với Clock, kiểm tra Q chuyển sang 1 (Set) hoặc 0 (Reset) đúng yêu cầu.

Cấu hình nguồn tín hiệu:

- Clock: Xung vuông với tần số phù hợp (3GHz để tương thích CMOS 50nm).
- D, Set, Reset: Tín hiệu logic 0/1 với biên độ $V_{DD} = 1\text{V}$ (công nghệ 50nm).
- Kiểm tra dạng sóng đầu ra Q và \bar{Q} để xác nhận mạch hoạt động đúng trong mọi trường hợp.

2.1.2.d. Kiểm tra và điều chỉnh schematic:

Phân tích kết quả mô phỏng để phát hiện lỗi chức năng, ví dụ: Q không lưu đúng D, hoặc Set/Reset không hoạt động bất đồng bộ.

Điều chỉnh schematic nếu cần (ví dụ, sửa lỗi kết nối, thêm buffer cho tín hiệu Clock, hoặc thay đổi cấu trúc cổng logic Set/Reset) để đảm bảo DFF hoạt động đúng.

Lặp lại mô phỏng transient trong LTspice để xác nhận mạch đã sửa chữa hoạt động theo yêu cầu.

2.1.2.e. Kết quả mô phỏng:

Tổng hợp kết quả mô phỏng từ LTspice, bao gồm các biểu đồ dạng sóng (waveforms) minh họa:

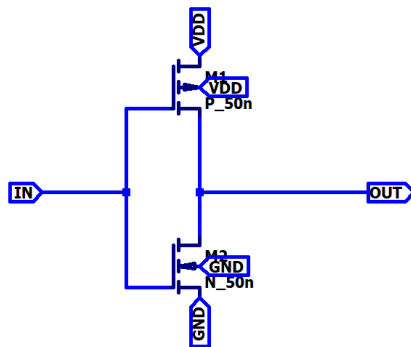
- D, Clock, Set, Reset, và Q/\bar{Q} trong các trường hợp đồng bộ và bất đồng bộ.
- Các trường hợp kiểm tra đặc biệt (ví dụ, Set/Reset kích hoạt khi Clock đang hoạt động).
- Xác nhận DFF hoạt động đúng theo chức năng, làm cơ sở cho các phân tích tiếp theo ở chương 3.

2.2. THIẾT KẾ CÁC PHẦN TỬ LOGIC CƠ BẢN VÀ KIỂM TRA CHỨC NĂNG

Thực hiện tạo các cổng logic, truyền cơ bản để sử dụng cho thiết kế Flip-flop và Latch, chọn kích thước theo cấu hình un-skew ($\frac{W_p}{W_n} = 2$ với $W_{p,INV} = 2\mu$, $W_{n,INV} = 1\mu$).

2.2.1. Thiết kế cổng INV

- **Inverter:** Giá trị của OUT chính là giá trị đảo của IN



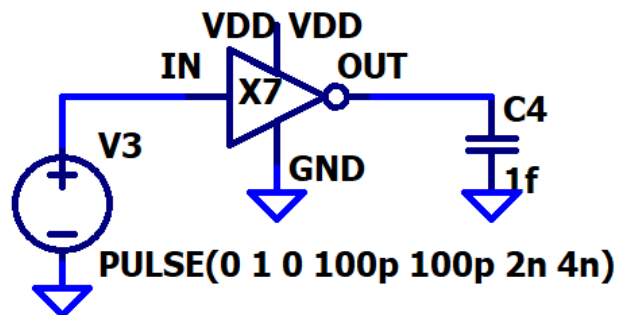
Schematic CMOS Inverter

IN	OUT
0	1
1	0

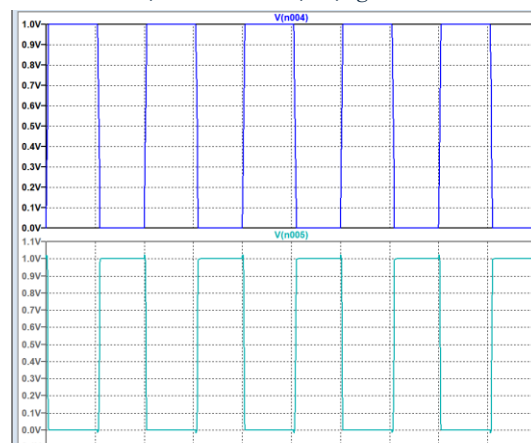
Bảng trạng thái Inverter

Nhận xét:

- Mạch hoạt động đúng với bảng trạng thái của Inverter, với $V_{DD} = 1V$.

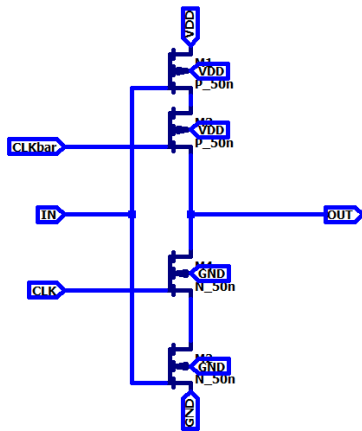


Mạch kiểm tra hoạt động Inverter

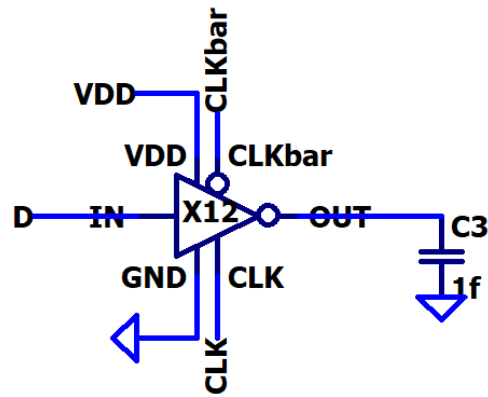


Kết quả mô phỏng hoạt động Inverter

- **Tri-state Inverter:** Khi CLK = 1, hoạt động như một Inverter thông thường, khi CLK = 0, rơi vào trạng thái high-impedence(Z).



Schematic Tri-state Inverter



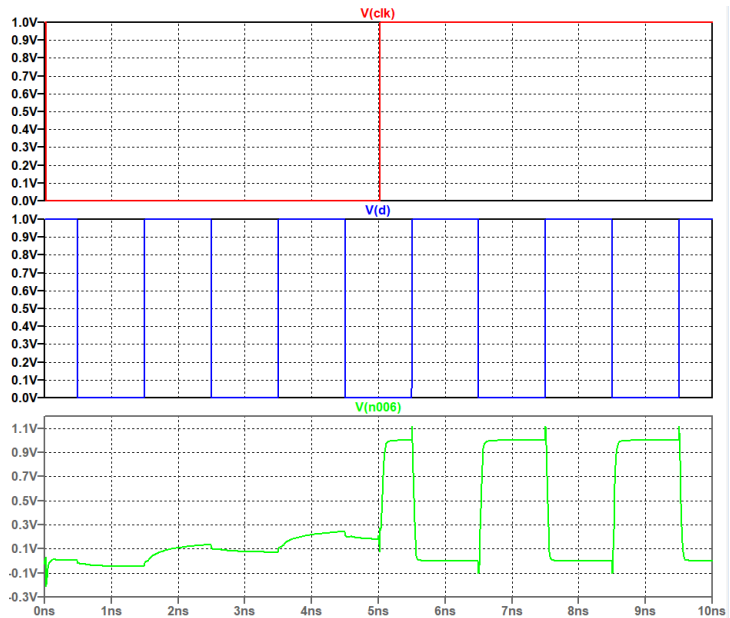
Mạch kiểm tra hoạt động Tri-state Inverter

IN	CLK	OUT
×	0	Z
0	1	1
1	1	0

Bảng trạng thái Tri-state Inverter

Nhận xét:

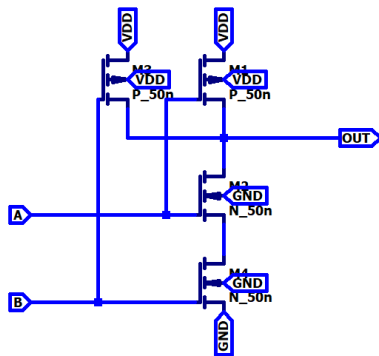
- Khi CLK = 1, mạch hoạt động như một Inverter.
- Khi CLK = 0, mạch rơi vào trạng thái high-impedence



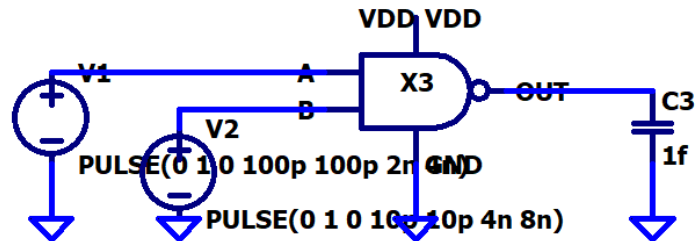
Kết quả mô phỏng Tri-state Inverter

2.2.2. Thiết kế cổng NAND

• **NAND2:** Thực hiện logic NAND cho hai ngõ ra vào đưa kết quả đến ngõ ra. Phần pull-up gồm 2 PMOS mắc song song nên kích thước $W_p = 2\mu$ không đổi, phần pull-down gồm 2 NMOS mắc nối tiếp nên nhân đôi kích thước $W_n = 2 \times 1\mu = 2\mu$.



Schematic NAND2



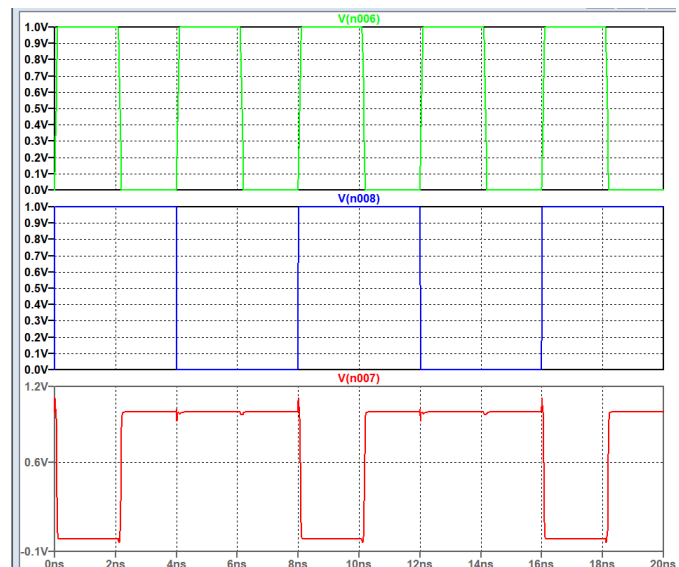
Mạch kiểm tra hoạt động NAND2

A	B	OUT
0	0	1
0	1	1
1	0	1
1	1	0

Bảng trạng thái NAND2

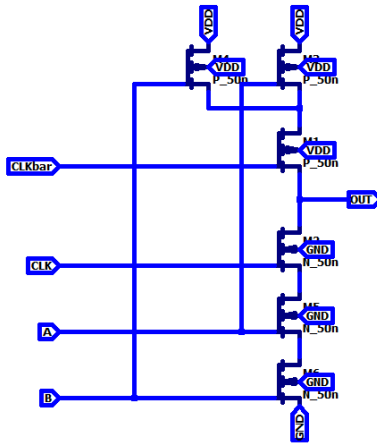
Nhận xét:

- Mạch hoạt động đúng với bảng trạng thái của NAND2, với $V_{DD} = 1V$.

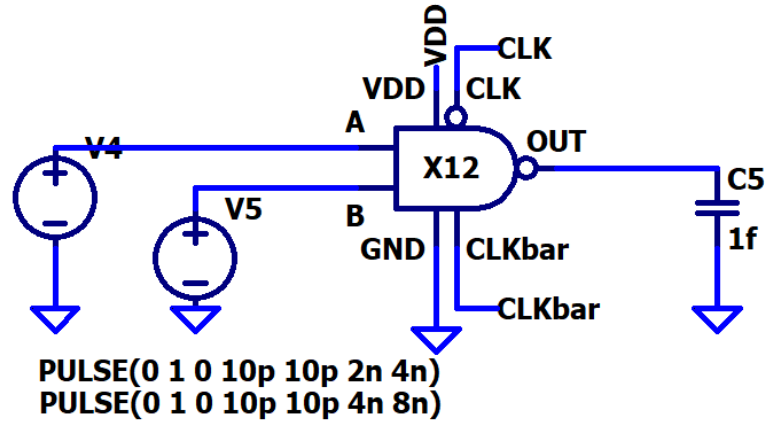


Kết quả mô phỏng NAND2

• **Tri-state NAND2:** Tích hợp chức năng đồng bộ với Clock, khi Clock = 1, hoạt động như một cổng NAND2 thông thường, khi Clock = 0, rơi vào trạng thái high-impedance(Z). Cấu trúc CMOS tương tự NAND2 nhưng tại ngõ ra OUT mắc thêm một cặp NMOS/PMOS được điều bởi CLK và \overline{CLK} . Phần pull-up có 2 PMOS mắc nối tiếp nên $W_p = 2 \times 2\mu = 4\mu$. Phần pull-down có 3 NMOS mắc nối tiếp nên $W_n = 3 \times 1\mu = 3\mu$.



Schematic Tri-state NAND2



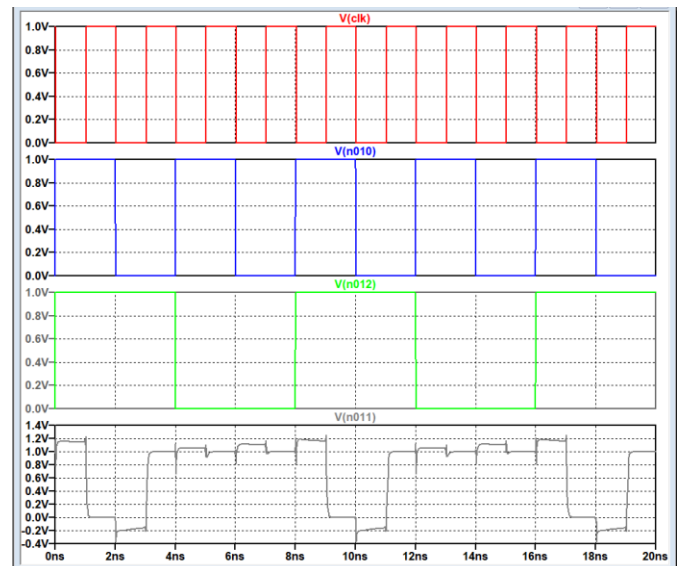
Mạch kiểm tra hoạt động Tri-state NAND2

A	B	CLK	OUT
×	×	0	Z
0	0	1	1
0	1	1	1
1	0	1	1
1	1	1	0

Bảng trạng thái Tri-state NAND2

Nhận xét:

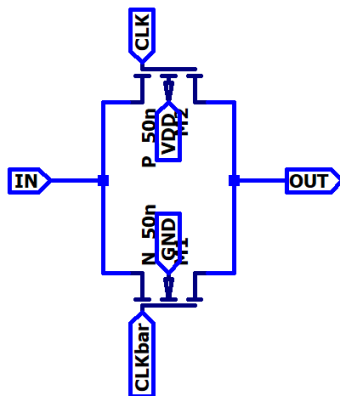
- Khi Clock = 1, mạch hoạt động như một cổng NAND2.
- Khi Clock = 0, mạch rơi vào trạng thái Z(biểu hiện bởi các giá trị khác 0 và 1 tại CLK = 0)



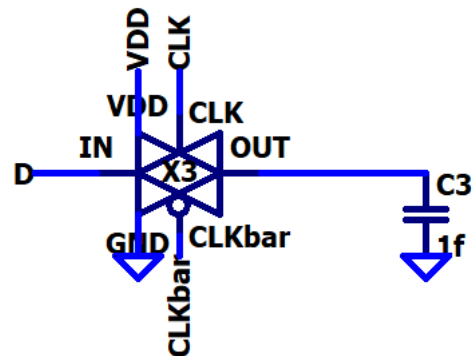
Kết quả mô phỏng Tri-state NAND2

2.2.3. Thiết kế cổng truyền(TG)

Hoạt động như một Buffer, khi CLK = 0 tín hiệu từ đi từ IN sang OUT với một lượng delay nhỏ, khi CLK = 1 sẽ rơi vào trạng thái high- impedance(Z). Kích thước NMOS,PMOS tương tự Inverter: $W_p = 2\mu$, $W_n = 1\mu = 2\mu$.



Schematic Transmission Gate



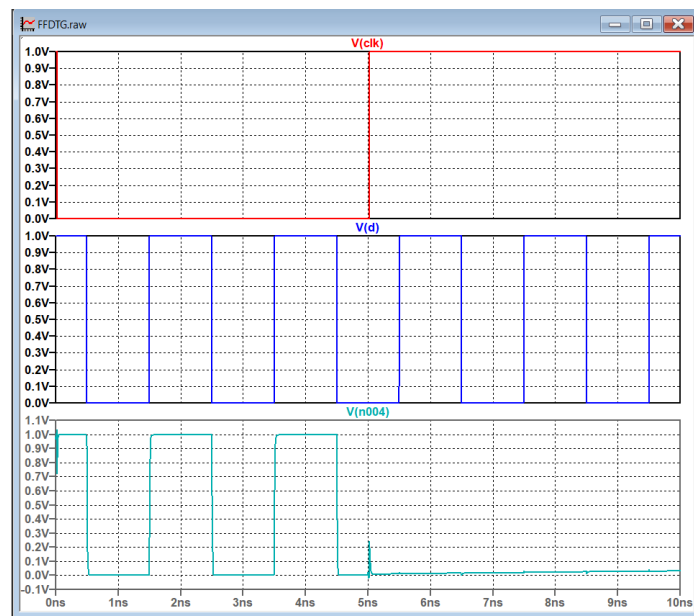
Mạch kiểm tra hoạt động Transmission Gate

IN	CLK	OUT
×	1	Z
×	0	IN

Bảng trạng thái TG

Nhận xét:

- Khi Clock = 0, mạch cho tín hiệu từ IN sang OUT.
- Khi Clock = 1, mạch rơi vào trạng thái high-impedance.



Kết quả mô phỏng Transmission Gate

2.3. THIẾT KẾ MASTER LATCH VÀ SLAVE LATCH

2.3.1. Kiến trúc latch sử dụng TG

Kiến trúc này được triển khai ở cấp transistor trong LTspice với mô hình NMOS (N_50n) và PMOS (P_50n) từ thư viện cmos.txt.

- **Cấu trúc:**

Latch sử dụng TG là Latch kết hợp **Transmission Gate (TG)** và **Inverter (INV)** để điều khiển luồng dữ liệu và duy trì trạng thái bằng một Tristate Inverter được mắc hồi tiếp.

TG hoạt động như một công tắc điện tử, mở/đóng dựa trên tín hiệu CLK và \overline{CLK} , trong khi Inverter và Tri-state Inverter tạo vòng phản hồi để lưu trữ dữ liệu.

- **Hoạt động:**

Khi $CLK = 1$, TG bật, D truyền qua TG và Inverter rồi đến ngõ ra (Q đảo/ \overline{Q}).

Khi $CLK = 0$, TG tắt, vòng phản hồi Tri-state Inverter sẽ giữ trạng thái Q .

- **Lý do chọn kiến trúc Latch sử dụng TG:**

Hiệu quả trong thiết kế mức transistor: TG truyền tín hiệu logic 0 và 1 hiệu quả hơn so với cổng logic đơn lẻ (như NAND/NOR), giảm suy hao tín hiệu trong công nghệ CMOS đồng thời sử dụng ít transistor hơn so với Latch tạo nên hoàn toàn từ NAND/NOR, phù hợp với thiết kế custom flow.

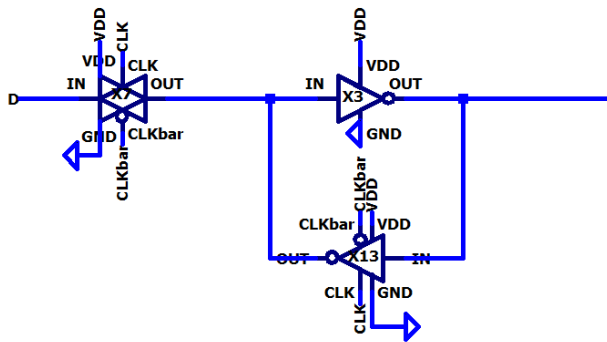
Đáp ứng yêu cầu đồng bộ của DFF: Latch sử dụng TG cho phép Master Latch và Slave Latch hoạt động bổ sung cho nhau (complementary), đảm bảo DFF có thể lưu và truyền tín hiệu tại đúng cạnh lên của tín hiệu Clock.

- **Kết luận:**

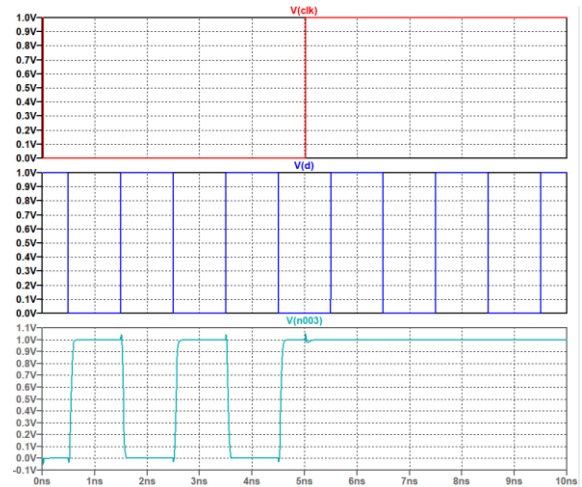
Kiến trúc latch sử dụng TG cung cấp giải pháp hiệu quả và linh hoạt để xây dựng master và slave latch cho DFF, đảm bảo chức năng đồng bộ đúng và làm nền tảng để tích hợp Set/Reset bất đồng bộ.

Thiết kế này tận dụng ưu điểm của CMOS 50nm và **LTspice**, phù hợp với yêu cầu **custom flow** của đồ án.

- **TG-based Latch:**

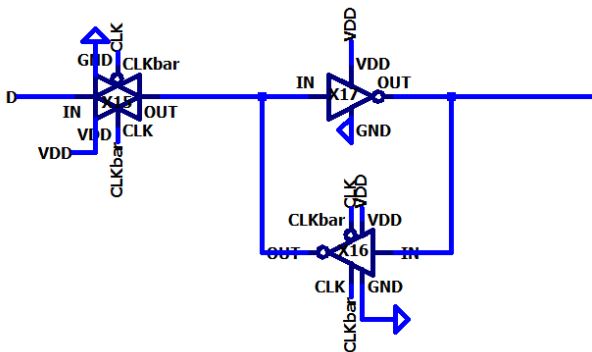


Master Latch(hoạt động khi CLK=0)

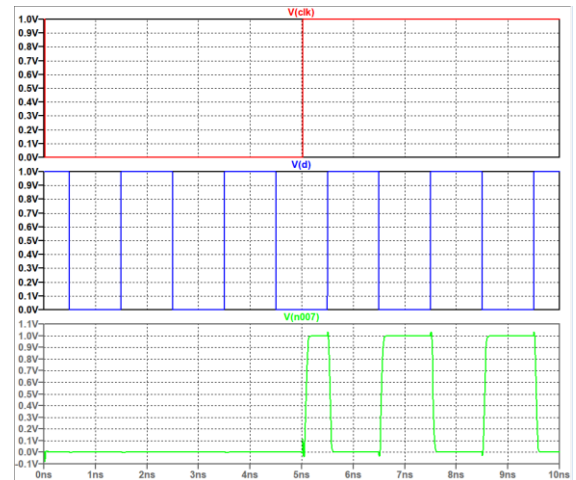


Kết quả mô phỏng Master Latch

Nhận xét: Mạch hoạt động đúng khi ngõ ra \bar{Q} nhận giá trị đảo của D khi CLK = 0.



Slave Latch(hoạt động khi CLK=1)



Kết quả mô phỏng Slave Latch

Nhận xét: Mạch hoạt động đúng khi ngõ ra \bar{Q} nhận giá trị đảo của D khi CLK = 1.

2.3.2. Tích hợp set/reset bất đồng bộ

Chức năng Set và Reset bất đồng bộ được tích hợp vào master latch và slave latch sử dụng Transmission Gate (TG) để đảm bảo D Flip-Flop (DFF) có thể đặt đầu ra $Q = 1$ (Set) hoặc $Q = 0$ (Reset) độc lập với tín hiệu đồng hồ (Clock). Mục tiêu là sửa đổi kiến trúc latch (2.3.1) để hỗ trợ Set/Reset bất đồng bộ mà vẫn duy trì chức năng đồng bộ của DFF.

- **Cấu trúc:**

Thay thế các Inverter và Tri-state Inverter bằng các cổng NAND2 và Tri-state NAND2 với ngõ vào thứ hai của các cổng NAND2 trên là tín hiệu Set và Reset. Do thay thế Inverter bằng NAND do đó tín hiệu Set/Reset là tín hiệu tích cực thấp, nghĩa là chỉ thực hiện khi mức logic là thấp.

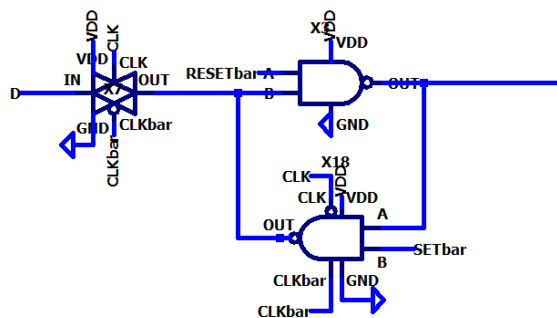
- **Hoạt động:**

Khi tín hiệu Set và Reset đều bằng 1, Latch hoạt động như bình thường.

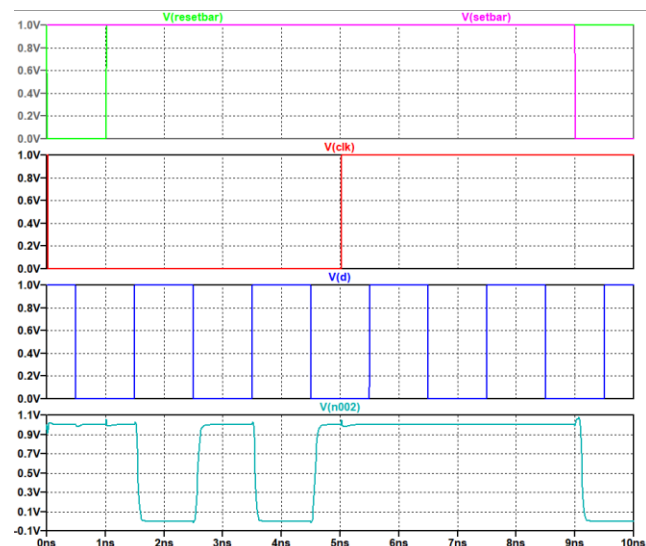
Khi tín hiệu Set = 0, ngõ ra $Q = 1$ bất kể có Clock hay không.

Khi tín hiệu Reset = 0, ngõ ra $Q = 0$ bất kể có Clock hay không.

- **TG-based Latch với Set/Reset bất đồng bộ:**



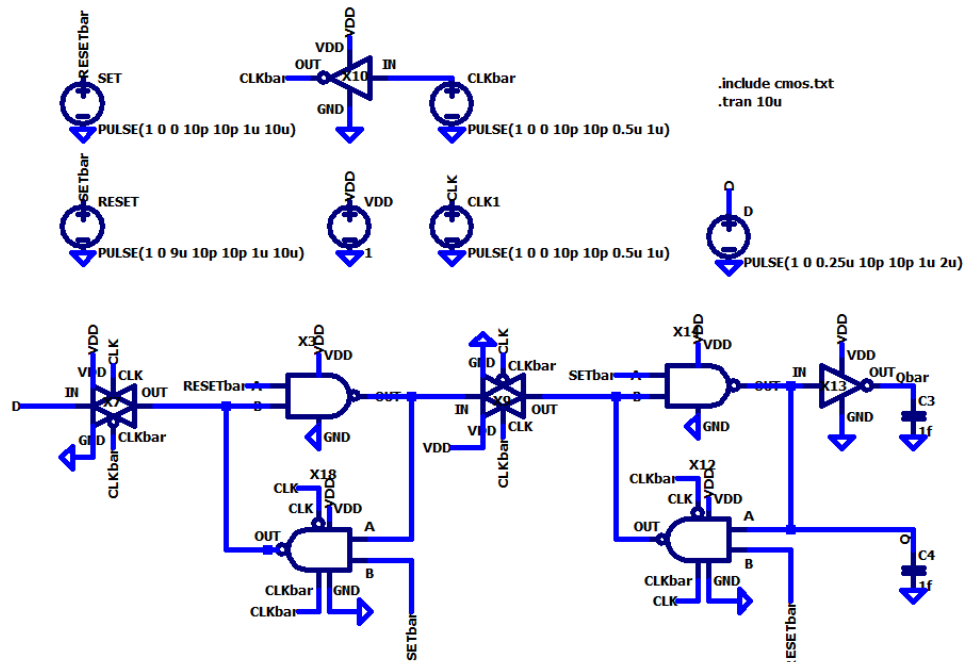
Master Latch với SR bất đồng bộ



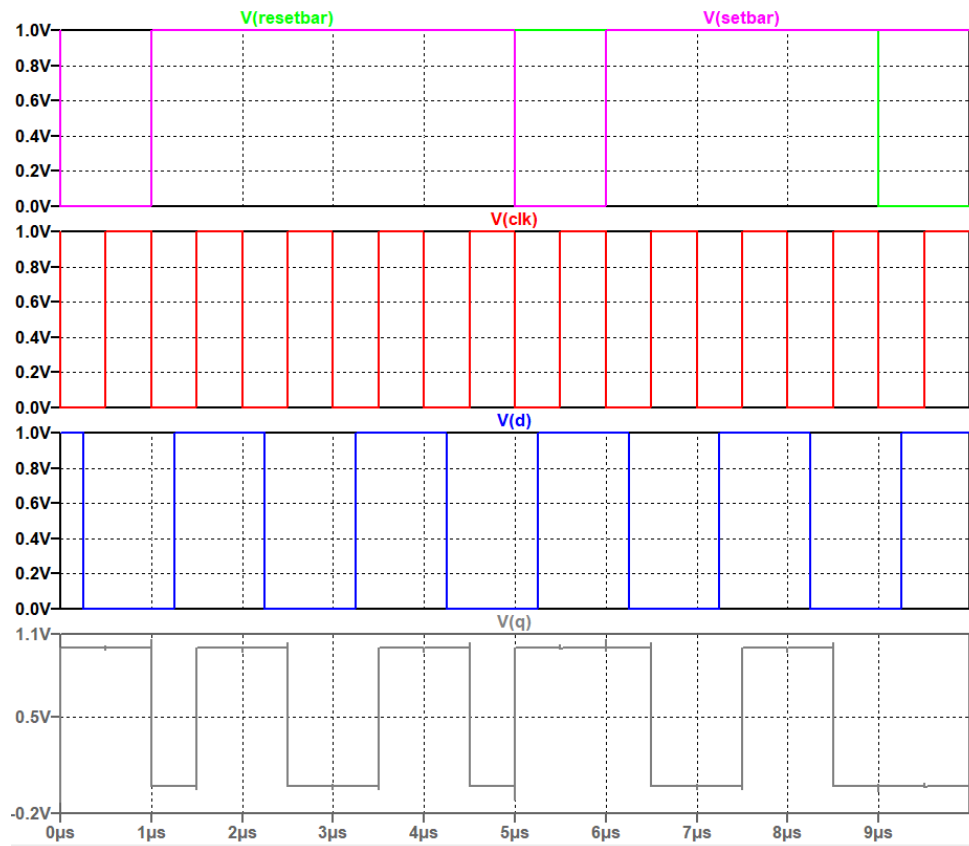
Kết quả mô phỏng Master Latch với SR bất đồng bộ

Nhận xét: Master Latch hoạt động đúng khi ngõ ra \bar{Q} nhận giá trị đảo của D khi $CLK = 0$ và ưu tiên tín hiệu Set/Reset bất đồng bộ.

2.4.2. Mô phỏng chức năng D Flip-flop với Set/Reset bất đồng bộ



Schematic DFF theo cấu trúc master-slave Latch với SR bất đồng bộ



Kết quả mô phỏng DFF theo cấu trúc master-slave latch với SR bất đồng bộ

2.4.3. Nhận xét sơ bộ

Đối với tín hiệu bất đồng bộ:

- Setbar kích hoạt tại lúc $0\mu s$ đến $1\mu s$ và $5\mu s$ đến $6\mu s$.
- Resetbar kích hoạt tại lúc $0\mu s$ đến $1\mu s$ và $9\mu s$ đến $10\mu s$.
- ➔ Tại thời điểm $5\mu s$ đến $6\mu s$, Setbar kích hoạt làm cho Q mang giá trị 1.
- ➔ Tại thời điểm $9\mu s$ đến $10\mu s$, Resetbar kích hoạt làm cho Q mang giá trị 0.
- ➔ Tại thời điểm $0\mu s$ đến $1\mu s$, Setbar và Resetbar cùng kích hoạt, do Setbar và Resetbar cùng được đưa vào cổng NAND2, do đó khi cả 2 cùng kích hoạt, của 4 cổng NAND đều có giá trị bằng 1 (do khi Setbar/Resetbar = 0, ngõ ra luôn luôn bằng 1), vì thế nên ngõ ra Q khi này sẽ bằng 1.

Đối với tín hiệu đồng bộ:

- Đây là mạch D Flip-flop nhảy cạnh lên (0 lên 1), do đó khi có cạnh lên xung CLK, mạch thực hiện chuyển giá trị D sang Q, khi không có cạnh lên xung CLK, Q giữ nguyên giá trị đến chu kỳ cạnh lên xung CLK tiếp theo.

Kết luận: Mạch hoàn toàn hoạt động đúng chức năng.

Nguyên nhân vì sao chọn kiến trúc D-FF cạnh lên có Set/Reset bất đồng bộ tích cực thấp theo cấu trúc Master-Slave sử dụng Latch Transmission Gate này?

Việc lựa chọn kiến trúc trước khi thiết kế là yêu cầu tối quan trọng nhất là đối với các mạch số vốn có nhiều cách tiếp cận khác nhau. Về nguyên nhân vì sao thiết kế trên được sử dụng để khảo sát, vui lòng tham khảo tại mục TỔNG KẾT (trang 59).

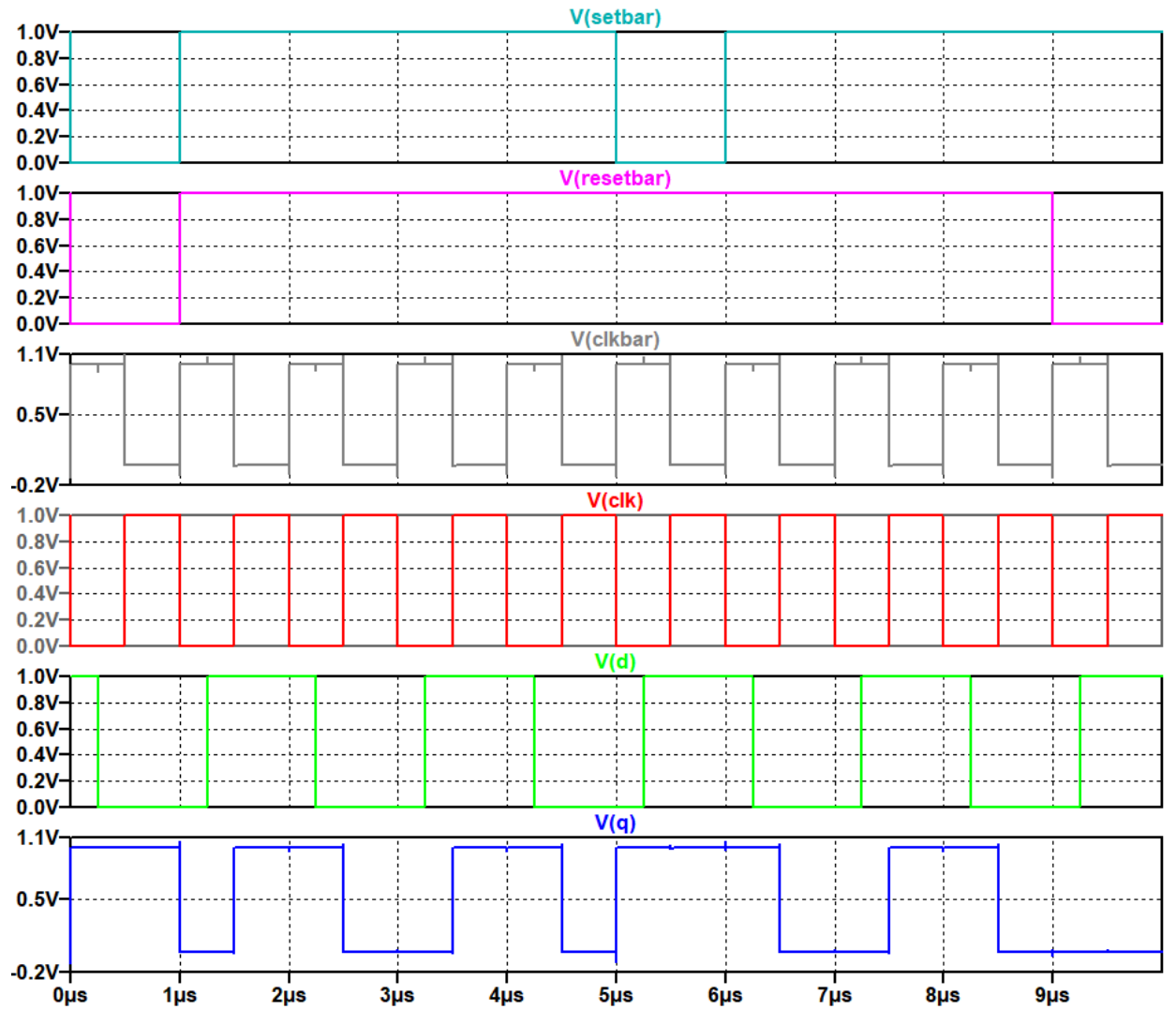
CHƯƠNG 3: KHẢO SÁT CÁC THÔNG SỐ

3.1. KHẢO SÁT THÔNG SỐ THỜI GIAN

3.1.1. Đo Propagation Delay(Clock-to-Q delay/ t_{cq}/t_{pd}):

Là thời gian từ lúc cạnh lên xung CLK đến lúc Q cập nhật giá trị mới.

Đo từ thời điểm 50% của biên độ cạnh CLK đến điểm 50% của biên độ đầu ra Q.



Dạng sóng được dùng để đo Propagation Delay

Chọn thời điểm đo:

- Q-rise1: xung CLK tại $1.5\mu\text{s}$, cạnh xung CLK thứ hai làm thời điểm bắt đầu đo, thời điểm ngừng đo tại cạnh lên thứ hai của tín hiệu Q.
- Q-rise2 xung CLK tại $3.5\mu\text{s}$, cạnh xung CLK thứ tư làm thời điểm bắt đầu đo, thời điểm ngừng đo tại cạnh lên thứ ba của tín hiệu Q.
- Q-fall1: xung CLK tại $2.5\mu\text{s}$, cạnh xung CLK thứ hai làm thời điểm bắt đầu đo, thời điểm ngừng đo tại cạnh xuống thứ hai của tín hiệu Q.
- Q-fall2: xung CLK tại $1.5\mu\text{s}$, cạnh xung CLK thứ hai làm thời điểm bắt đầu đo, thời điểm ngừng đo tại cạnh xuống thứ ba của tín hiệu Q.

Sử dụng lệnh `.measure` để thực hiện đo tự động quá trình trên:

```
.meas TRAN tcqrise1 TRIG V(clk)=0.5 RISE 2 TARG V(q)=0.5 RISE 2
.meas TRAN tcqrise2 TRIG V(clk)=0.5 RISE=4 TARG V(q)=0.5 RISE=3
.meas TRAN tcqfall1 TRIG V(clk)=0.5 RISE=3 TARG V(q)=0.5 FALL=2
.meas TRAN tcqfall2 TRIG V(clk)=0.5 RISE=5 TARG V(q)=0.5 FALL=3
```

Trong LTSpice, mở công cụ View, chọn SPICE output log để kiểm tra kết quả:

```
SPICE Output Log: C:\Documents\LTspice\Project\FFDTG.log X
LTspice 24.1.8 for Windows
Circuit: C:\Documents\LTspice\Project\FFDTG.net
Start Time: Thu Jun 12 09:40:23 2025
solver = Normal
Maximum thread count: 16
tnom = 27
temp = 27
method = modified trap
Direct Newton iteration for .op point succeeded.
Total elapsed time: 0.333 seconds.

Files loaded:
C:\Documents\LTspice\Project\FFDTG.net
C:\Users\BAO\AppData\Local\LTspice\lib\cmp\standard.mos
C:\Documents\LTspice\Project\cmos.txt

tcqrise1=1.14170394823e-10 FROM 1.500015e-06 TO 1.50012917039e-06
tcqrise2=1.14170444247e-10 FROM 3.500015e-06 TO 3.50012917044e-06
tcqfall1=1.29139961125e-10 FROM 2.500015e-06 TO 2.50014413996e-06
tcqfall2=1.29139961151e-10 FROM 4.500015e-06 TO 4.50014413996e-06
```

Kết quả:

- Thời gian delay cạnh lên ngõ ra $CQ_{rise1} = CQ_{rise2} = 111.17\text{ps}$.
- Thời gian delay cạnh xuống ngõ ra $CQ_{fall1} = CQ_{fall2} = 129.1399611\text{ps}$

Nhận xét:

- Ta nhận thấy CQrise1 và CQrise2 bằng nhau cho thấy sự nhất quán trong tín hiệu delay cạnh lên giúp cho giá trị t_{cqrise} càng trở nên tin cậy.
- Tương tự với CQfall1 và CQfall2 cũng bằng nhau khiến cho giá trị t_{cqfall} tin cậy hơn.

Cách giảm t_{cq}/t_{pd} :

Trong điều kiện không được can thiệp vào kích thước transistor, cấu trúc của Flip-flop cũng như file mô hình cmos.txt, việc giảm t_{cq}/t_{pd} phụ thuộc vào các yếu tố bên ngoài như nguồn điện áp V_{DD} , tín hiệu đầu vào như CLK, D, SR, điện dung tải C_{load} cũng như cải thiện các điều kiện mô phỏng như nhiệt độ,...

Tiến hành thử nghiệm thay đổi t_{cq}/t_{pd} :

• Tăng giảm điện dung tụ tải:

```
LTspice 24.1.8 for Windows
Circuit: C:\Documents\LTspice\Project\FFDTG.net
Start Time: Thu Jun 12 10:09:39 2025
solver = Normal
Maximum thread count: 16
tnom = 27
temp = 27
method = modified trap
Direct Newton iteration for .op point succeeded.
Total elapsed time: 0.325 seconds.

Files loaded:
C:\Documents\LTspice\Project\FFDTG.net
C:\Users\BAO\AppData\Local\LTspice\lib\cmp\standard.mos
C:\Documents\LTspice\Project\cmos.txt

tcqrise1=1.13351590308e-10 FROM 1.500015e-06 TO 1.50012835159e-06
tcqrise2=1.1335163747e-10 FROM 3.500015e-06 TO 3.50012835164e-06
tcqfall1=1.2833763037e-10 FROM 2.500015e-06 TO 2.50014333763e-06
tcqfall2=1.28337630396e-10 FROM 4.500015e-06 TO 4.50014333763e-06
```

Với $C_{load} = 0.1fF$

```
LTspice 24.1.8 for Windows
Circuit: C:\Documents\LTspice\Project\FFDTG.net
Start Time: Thu Jun 12 10:10:57 2025
solver = Normal
Maximum thread count: 16
tnom = 27
temp = 27
method = modified trap
Direct Newton iteration for .op point succeeded.
Total elapsed time: 0.325 seconds.

Files loaded:
C:\Documents\LTspice\Project\FFDTG.net
C:\Users\BAO\AppData\Local\LTspice\lib\cmp\standard.mos
C:\Documents\LTspice\Project\cmos.txt

tcqrise1=1.22962296054e-10 FROM 1.500015e-06 TO 1.5001379623e-06
tcqrise2=1.2296233802e-10 FROM 3.500015e-06 TO 3.50013796233e-06
tcqfall1=1.3696888774e-10 FROM 2.500015e-06 TO 2.50015196889e-06
tcqfall2=1.36968887763e-10 FROM 4.500015e-06 TO 4.50015196889e-06
```

Với $C_{load} = 10fF$

• Tăng giảm nhiệt độ:

```
LTspice 24.1.8 for Windows
Circuit: C:\Documents\LTspice\Project\FFDTG.net
Start Time: Thu Jun 12 10:14:54 2025
solver = Normal
Maximum thread count: 16
tnom = 27
temp = 80
method = modified trap
Direct Newton iteration for .op point succeeded.
Total elapsed time: 0.317 seconds.

Files loaded:
C:\Documents\LTspice\Project\FFDTG.net
C:\Users\BAO\AppData\Local\LTspice\lib\cmp\standard.mos
C:\Documents\LTspice\Project\cmos.txt

tcqrise1=1.24504994421e-10 FROM 1.500015e-06 TO 1.50013950499e-06
tcqrise2=1.24504994142e-10 FROM 3.500015e-06 TO 3.50013950499e-06
tcqfall1=1.39323754402e-10 FROM 2.500015e-06 TO 2.50015432375e-06
tcqfall2=1.39323754417e-10 FROM 4.500015e-06 TO 4.50015432375e-06
```

Với $C_{load} = 0.1fF$, $temp = 80$

```
LTspice 24.1.8 for Windows
Circuit: C:\Documents\LTspice\Project\FFDTG.net
Start Time: Thu Jun 12 10:12:46 2025
solver = Normal
Maximum thread count: 16
tnom = 27
temp = 80
method = modified trap
Direct Newton iteration for .op point succeeded.
Total elapsed time: 0.317 seconds.

Files loaded:
C:\Documents\LTspice\Project\FFDTG.net
C:\Users\BAO\AppData\Local\LTspice\lib\cmp\standard.mos
C:\Documents\LTspice\Project\cmos.txt

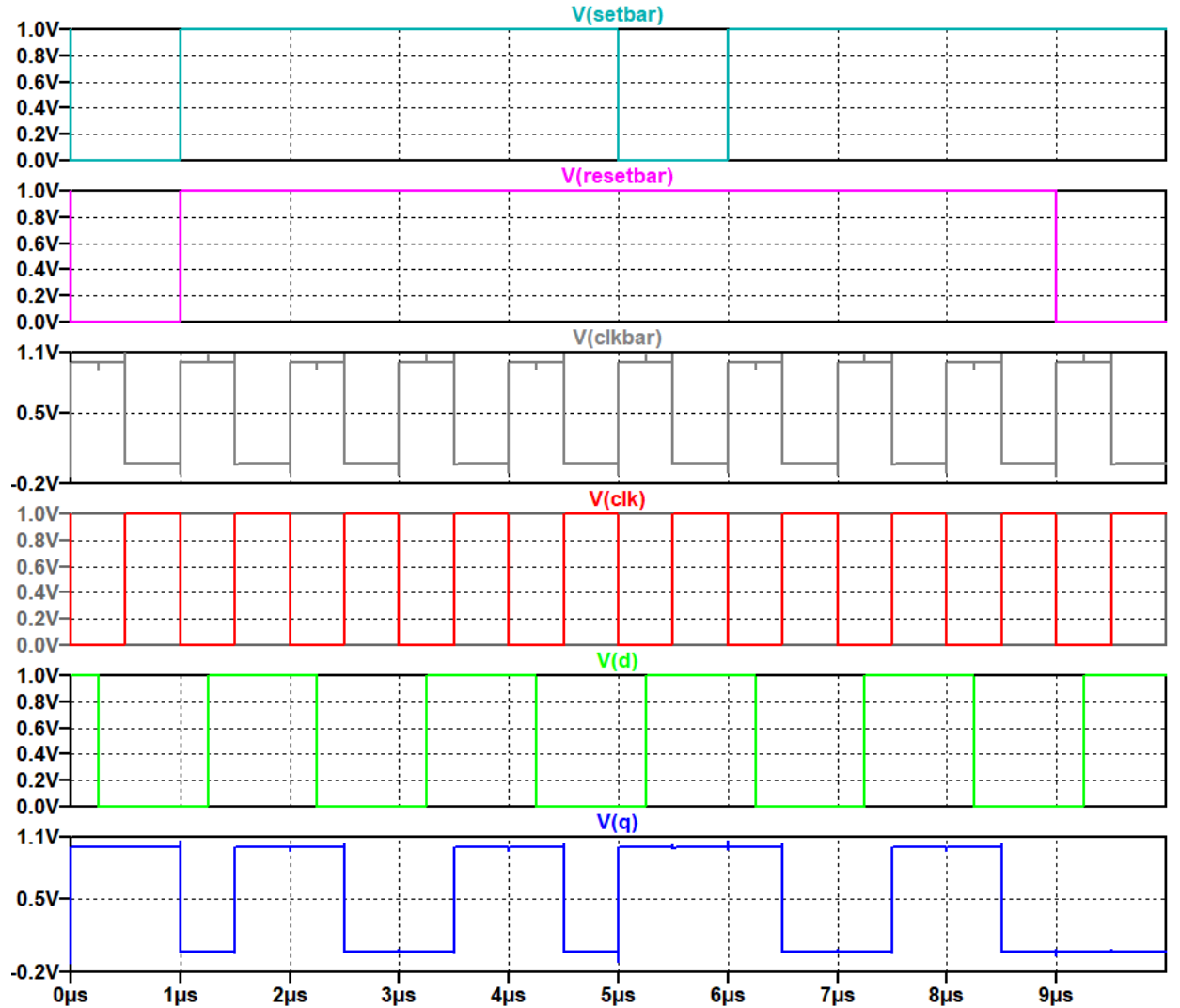
tcqrise1=1.3506302404e-10 FROM 1.500015e-06 TO 1.50015006302e-06
tcqrise2=1.3506302459e-10 FROM 3.500015e-06 TO 3.50015006302e-06
tcqfall1=1.48622573487e-10 FROM 2.500015e-06 TO 2.50016362257e-06
tcqfall2=1.48622573502e-10 FROM 4.500015e-06 TO 4.50016362257e-06
```

Với $C_{load} = 10fF$, $temp = 80$

3.1.2. Đo Rise Time và Fall Time(t_{rise}/t_{fall}):

Là thời gian từ lúc Q thay đổi logic từ cao sang thấp hoặc ngược lại.

Đo từ thời điểm 10% của biên độ cạnh Q đến điểm 90% của biên độ cạnh Q.




Dạng sóng được dùng để đo Rise Time/Fall Time

Chọn thời điểm đo:

- Q-rise1: xung CLK tại $1.5\mu\text{s}$, cạnh xung CLK thứ hai làm thời điểm bắt đầu đo, thời điểm ngừng đo tại cạnh lên thứ hai của tín hiệu Q.
- Q-rise2 xung CLK tại $3.5\mu\text{s}$, cạnh xung CLK thứ tư làm thời điểm bắt đầu đo, thời điểm ngừng đo tại cạnh lên thứ ba của tín hiệu Q.
- Q-fall1: xung CLK tại $2.5\mu\text{s}$, cạnh xung CLK thứ hai làm thời điểm bắt đầu đo, thời điểm ngừng đo tại cạnh xuống thứ hai của tín hiệu Q.
- Q-fall2: xung CLK tại $1.5\mu\text{s}$, cạnh xung CLK thứ hai làm thời điểm bắt đầu đo, thời điểm ngừng đo tại cạnh xuống thứ ba của tín hiệu Q.

Sử dụng lệnh `.measure` để thực hiện đo tự động quá trình trên:

```
.meas TRAN tqrise1 TRIG V(q)=0.1 RISE=2 TARG V(q)=0.9 RISE=2  
.meas TRAN tqrise2 TRIG V(q)=0.1 RISE=3 TARG V(q)=0.9 RISE=3  
.meas TRAN tqfall1 TRIG V(q)=0.9 FALL=2 TARG V(q)=0.1 FALL=2  
.meas TRAN tqfall2 TRIG V(q)=0.9 FALL=3 TARG V(q)=0.1 FALL=3
```

 SPICE Output Log: C:\Documents\LTspice\Project\FFDTG.log

LTspice 24.1.8 for Windows

Circuit: C:\Documents\LTspice\Project\FFDTG.net

Start Time: Thu Jun 12 10:22:59 2025

solver = Normal

Maximum thread count: 16

tnom = 27

temp = 27

method = modified trap

Direct Newton iteration for .op point succeeded.

Total elapsed time: 0.326 seconds.

Files loaded:

C:\Documents\LTspice\Project\FFDTG.net

C:\Users\BAO\AppData\Local\LTspice\lib\cmp\standard.mos

C:\Documents\LTspice\Project\cmos.txt

tqrise1=1.21859652036e-10 FROM 1.50007619924e-06 TO 1.50019805889e-06

tqrise2=1.21859657281e-10 FROM 3.50007619928e-06 TO 3.50019805893e-06

tqfall1=1.02877935348e-10 FROM 2.50009944282e-06 TO 2.50020232075e-06

tqfall2=1.02877935331e-10 FROM 4.50009944282e-06 TO 4.50020232075e-06

Kết quả:

- Thời gian cạnh lên ngõ ra $Q_{rise1} = Q_{rise2} = 121.85965\text{ps}$.
- Thời gian cạnh xuống ngõ ra $Q_{fall1} = Q_{fall2} = 102.8779353\text{ps}$

Nhận xét:

- Ta nhận thấy Q_{rise1} và Q_{rise2} bằng nhau cho thấy sự nhất quán trong thời gia cạnh lên giúp cho giá trị $t_{q_{rise}}$ càng trở nên tin cậy.
- Tương tự với Q_{fall1} và Q_{fall2} cũng bằng nhau khiến cho giá trị $t_{q_{fall}}$ tin cậy hơn.

Cách giảm t_{rise}/t_{fall} :

Trong điều kiện không được can thiệp vào kích thước transistor, cấu trúc của Flip-flop cũng như file mô hình cmos.txt, việc giảm t_{rise}/t_{fall} phụ thuộc vào các yếu tố bên ngoài như nguồn điện áp V_{DD} , tín hiệu đầu vào như CLK, D, điện dung tụ tải C_{load} .

Tiến hành thử nghiệm thay đổi t_{fall}/t_{rise} :

- Tăng giảm điện dung tụ tải:

```
LTspice 24.1.8 for Windows
Circuit: C:\Documents\LTspice\Project\FFDTG.net
Start Time: Thu Jun 12 10:40:39 2025
solver = Normal
Maximum thread count: 16
tnom = 27
temp = 27
method = modified trap
Direct Newton iteration for .op point succeeded.
Total elapsed time: 0.328 seconds.

Files loaded:
C:\Documents\LTspice\Project\FFDTG.net
C:\Users\BAO\AppData\Local\LTspice\lib\cmp\standard.mos
C:\Documents\LTspice\Project\cmos.txt

tqrise1=1.20640272443e-10 FROM 1.50007589675e-06 TO 1.50019653703e-06
tqrise2=1.20640313613e-10 FROM 3.50007589679e-06 TO 3.5001965371e-06
tqfall1=1.01349149483e-10 FROM 2.50009903613e-06 TO 2.50020038528e-06
tqfall2=1.01349147575e-10 FROM 4.50009903613e-06 TO 4.50020038527e-06
```

Với $C_{load} = 0.1\text{fF}$

```
LTspice 24.1.8 for Windows
Circuit: C:\Documents\LTspice\Project\FFDTG.net
Start Time: Thu Jun 12 10:41:56 2025
solver = Normal
Maximum thread count: 16
tnom = 27
temp = 27
method = modified trap
Direct Newton iteration for .op point succeeded.
Total elapsed time: 0.329 seconds.

Files loaded:
C:\Documents\LTspice\Project\FFDTG.net
C:\Users\BAO\AppData\Local\LTspice\lib\cmp\standard.mos
C:\Documents\LTspice\Project\cmos.txt

tqrise1=1.40452130473e-10 FROM 1.50007912394e-06 TO 1.50021957607e-06
tqrise2=1.40452132074e-10 FROM 3.50007912396e-06 TO 3.50021957609e-06
tqfall1=1.1387304805e-10 FROM 2.5001035415e-06 TO 2.50021741455e-06
tqfall2=1.13873048023e-10 FROM 4.5001035415e-06 TO 4.50021741455e-06
```

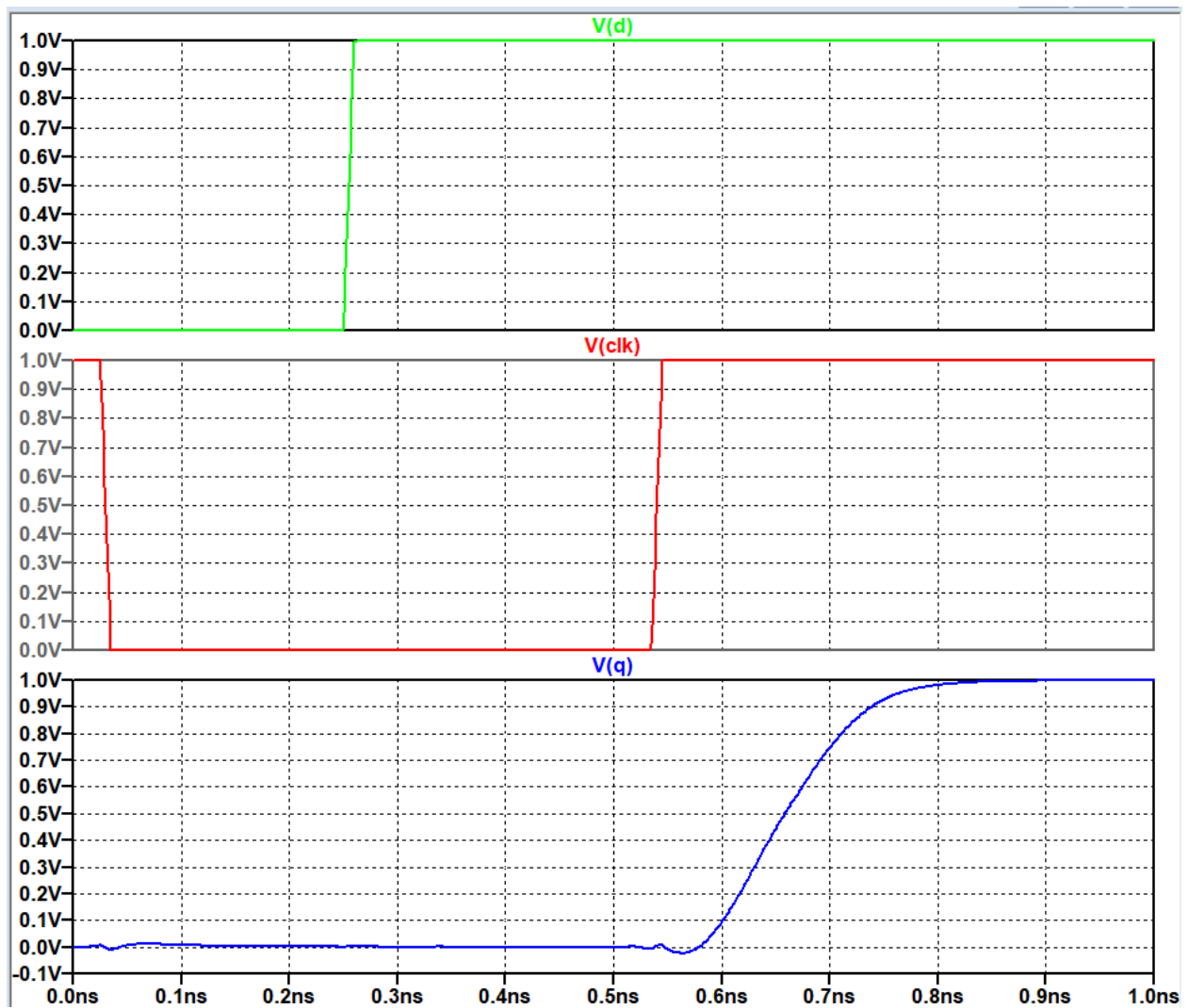
Với $C_{load} = 10\text{fF}$

3.1.4. Đo Setup Time và Hold Time

- **Setup Time:**

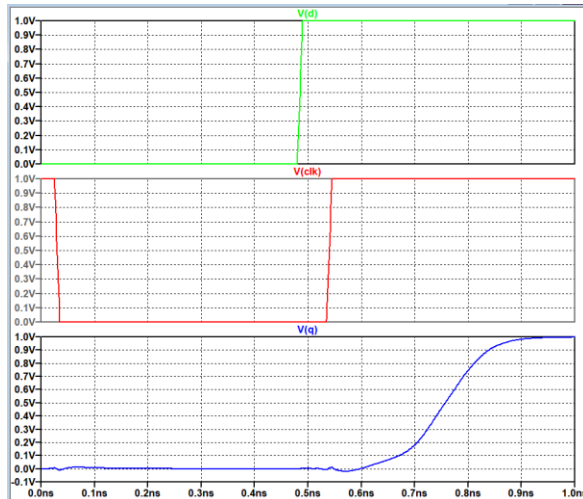
Là thời gian tín hiệu phải ổn định trước xung CLK để ngõ ra Q có thể nhận đúng giá trị. Giá trị này phụ thuộc vào đặc tính của cmos.txt.

Cách đo: Cho tín hiệu D thay đổi trước cạnh CLK một khoảng thời gian nhỏ dần đến khi Flip-flop không nhận được đúng giá trị nữa.

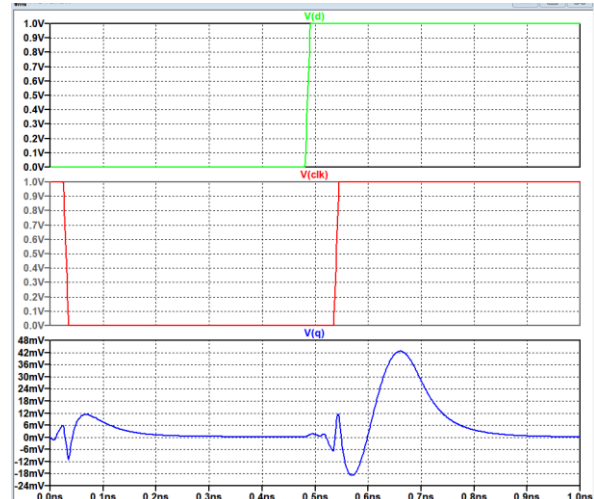


Dạng sóng được dùng để đo Setup Time

Cho D thay đổi tại thời điểm càng ngày càng gần cạnh lên CLK đến khi ngõ ra không nhận mức logic 1 nữa:



Cho D delay trong 480ps



Cho D delay trong 481ps

Nhận xét:

Có thể thấy khi D thay đổi tiến gần đến CLK, mặc dù vẫn ở trước cạnh CLK, tín hiệu vẫn không được truyền đến đầu ra Q do đã vi phạm về Setup Time.

Tiến hành đo t_{setup} bằng lệnh `.meas` ở thời điểm trước khi vi phạm Setup Time :

```
.meas TRAN tsetup TRIG V(d)=0.5 RISE=1 TARG V(clk)=0.5 RISE=1
```

```
LTspice 24.1.8 for Windows
Circuit: C:\Documents\LTspice\Project\FFDTG.net
Start Time: Thu Jun 12 12:43:25 2025
solver = Normal
Maximum thread count: 16
tnom = 27
temp = 27
method = modified trap
Direct Newton iteration for .op point succeeded.
Total elapsed time: 0.205 seconds.

Files loaded:
C:\Documents\LTspice\Project\FFDTG.net
C:\Users\BAO\AppData\Local\LTspice\lib\cmp\standard.mos
C:\Documents\LTspice\Project\cmos.txt

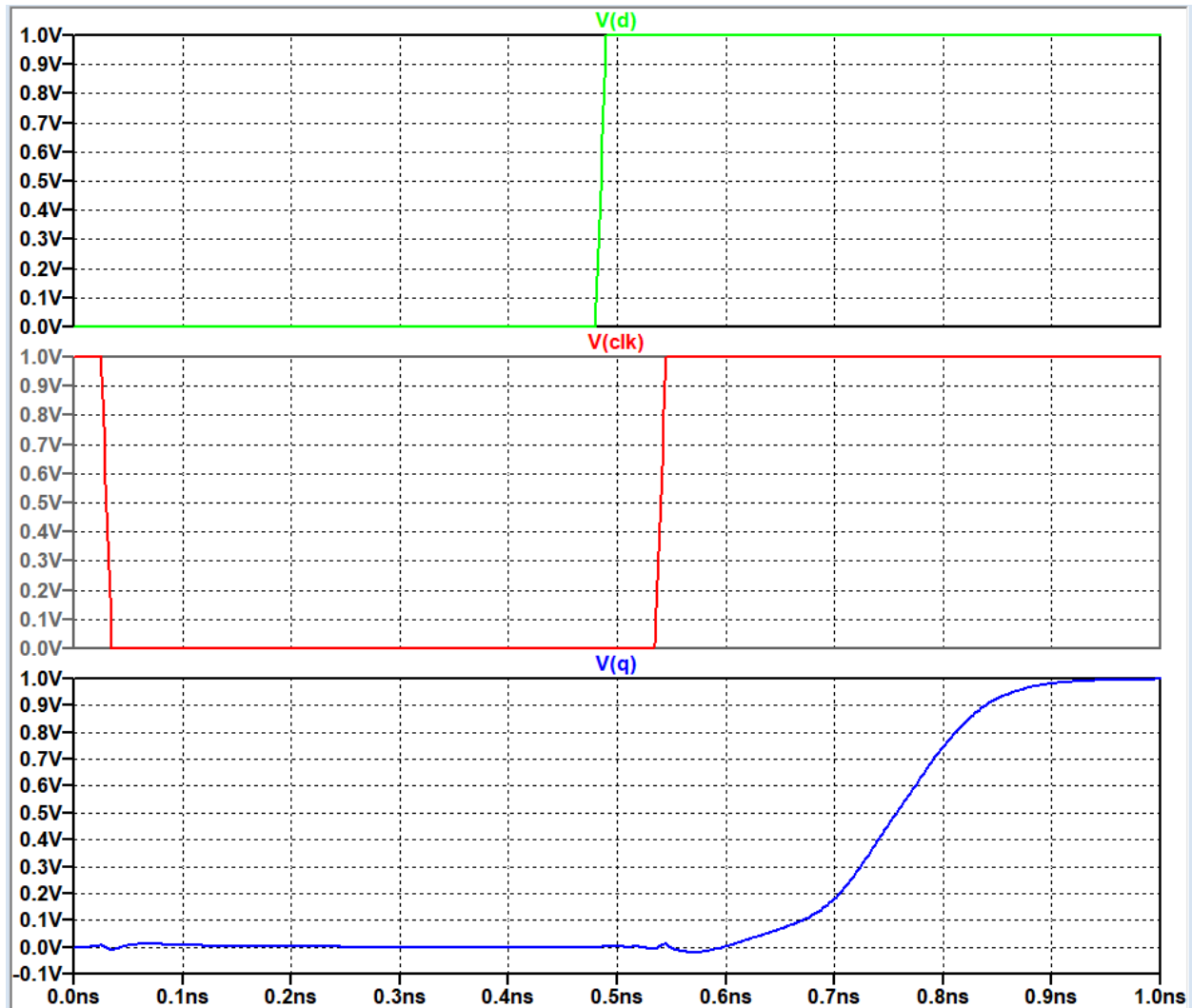
tsetup=2.99999998453e-11 FROM 4.85000000047e-10 TO 5.14999999893e-10
```

Kết luận: Setup time là 30ps

- **Hold Time:**

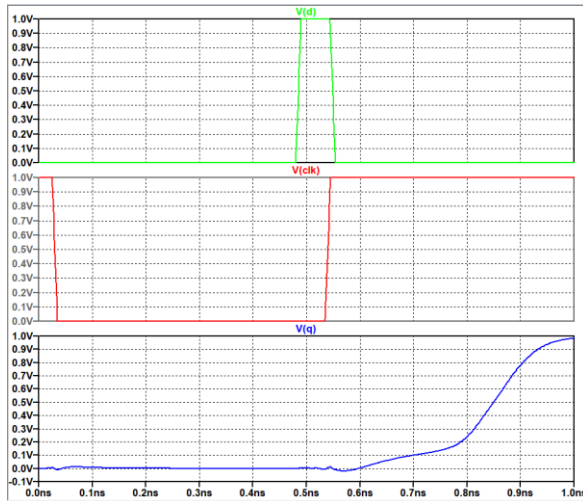
Là thời gian tín hiệu giữ ổn định sau cạnh lên CLK để Q có thể lấy đúng giá trị. Giá trị này phụ thuộc vào đặc tính của cmos.txt.

Cách đo: Cho tín hiệu D thay đổi sau cạnh lên CLK một khoảng thời gian nhỏ dần đến khi Flip-flop không nhận được đúng giá trị nữa.

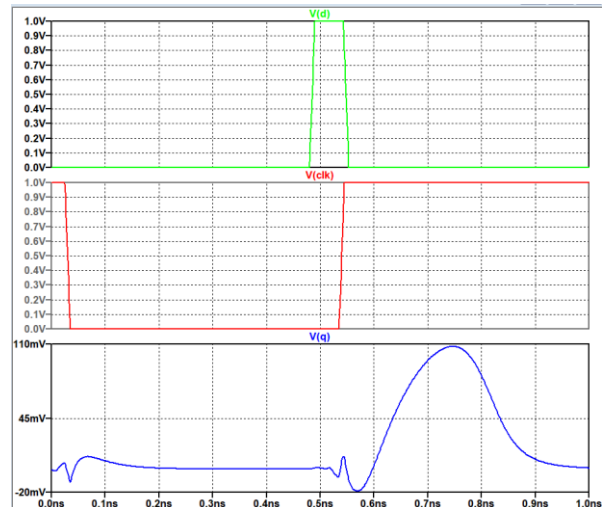


Dạng sóng được dùng để đo Hold Time

Sử dụng dạng sóng có được ở phần tìm t_{setup} , tiến hành giảm thời gian hoạt động của D nhỏ dần đến khi Q không nhận được giá trị logic bằng 1 nữa.



Cho D hoạt động trong 54ps



Cho D hoạt động trong 53ps

Nhận xét:

Có thể thấy khi D thay đổi tiền gần đến CLK, mặc dù vẫn ở sau cạnh CLK, tín hiệu vẫn không được truyền đến đầu ra Q do đã vi phạm về Hold Time.

Tiến hành đo t_{hold} bằng lệnh .meas ở thời điểm trước khi vi phạm Hold Time :

```
.meas TRAN thold TRIG V(clk)=0.5 RISE=1 TARG V(d)=0.5 FALL=1
```

```
LTspice 24.1.8 for Windows
Circuit: C:\Documents\LTspice\Project\FFDTG.net
Start Time: Thu Jun 12 13:11:08 2025
solver = Normal
Maximum thread count: 16
tnom = 27
temp = 27
method = modified trap
Direct Newton iteration for .op point succeeded.
Total elapsed time: 0.209 seconds.

Files loaded:
C:\Documents\LTspice\Project\FFDTG.net
C:\Users\BAO\AppData\Local\LTspice\lib\cmp\standard.mos
C:\Documents\LTspice\Project\cmos.txt

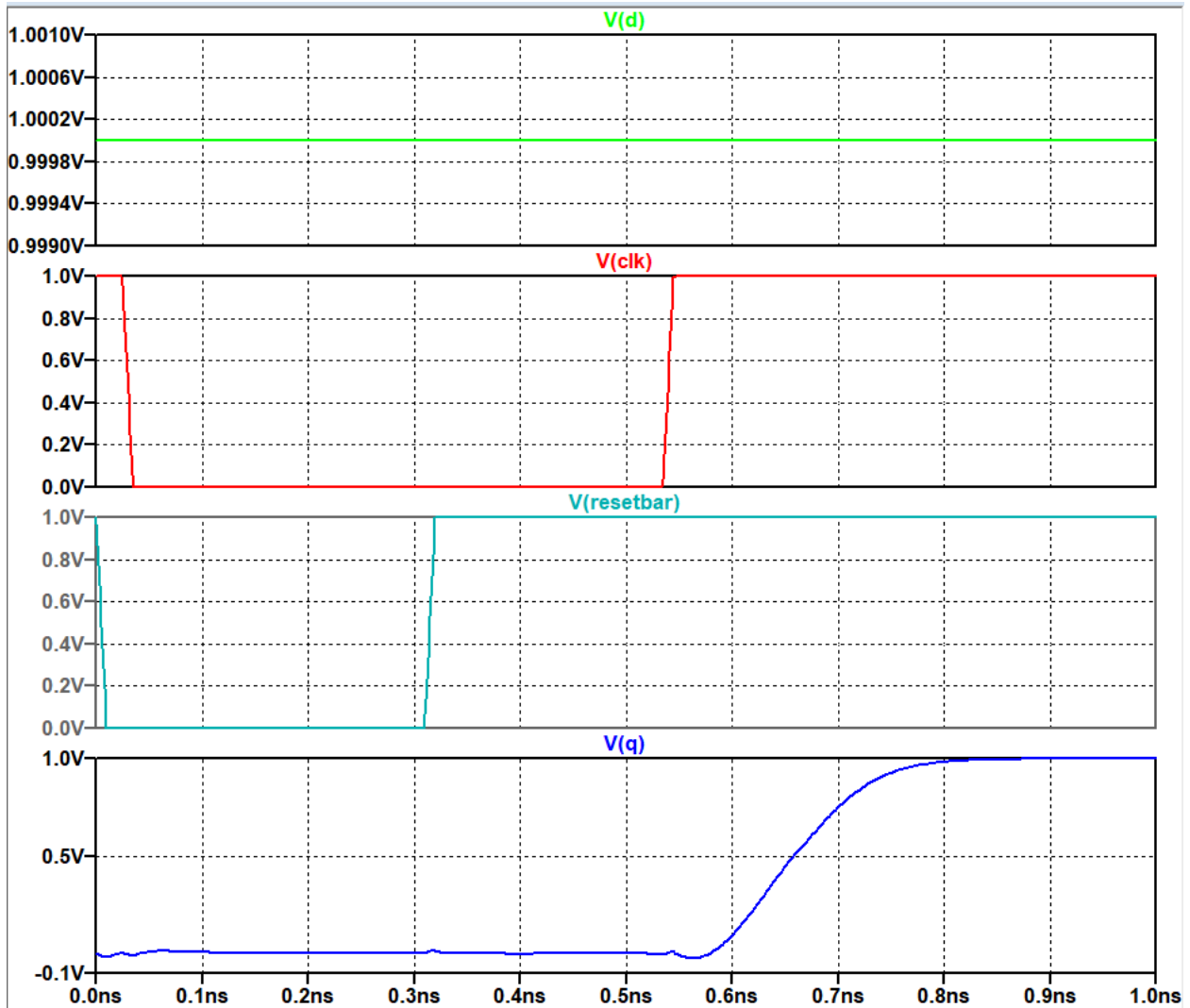
thold=8.9999999645e-12 FROM 5.40000000125e-10 TO 5.49000000009e-10
```

Kết luận: Thời gian Hold Time(t_{hold}) là 9ps.

3.1.5. Đo Recovery Time và Removal Time

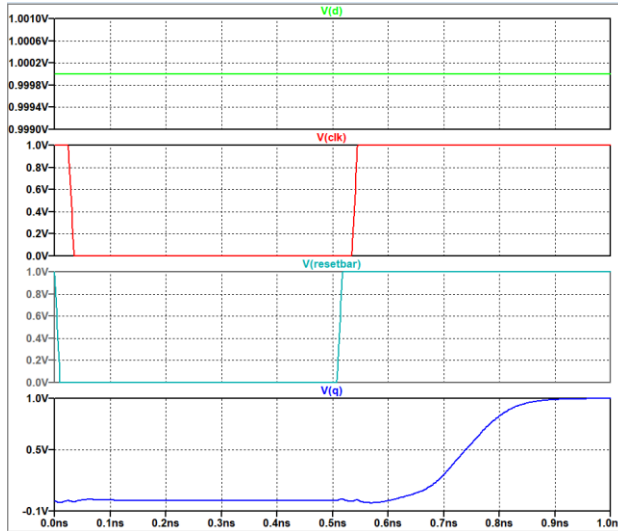
- **Recovery Time:**

Là khoảng thời gian tối thiểu từ khi tín hiệu điều khiển bất đồng bộ (như Set hoặc Reset) bị vô hiệu hóa (chuyển từ trạng thái active sang inactive) đến khi cạnh đồng hồ tiếp theo.

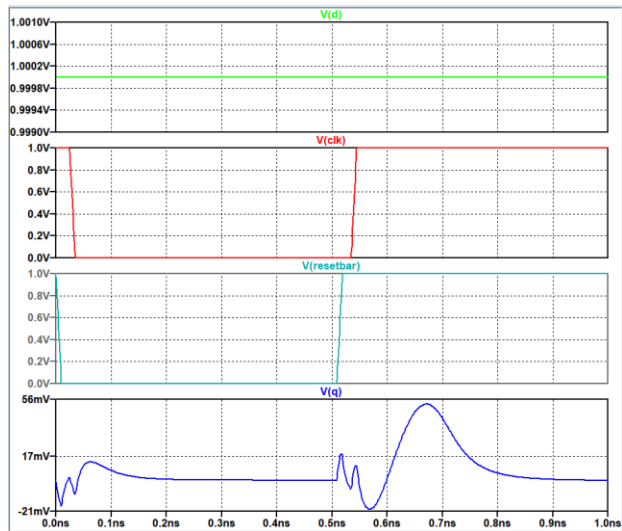


Dạng sóng được dùng để đo Recovery Time cho tín hiệu RESETbar

Cho thời gian inactive của Resetbar tiến gần đến trước xung CLK đến khi tín hiệu ngõ ra Q không ghi được giá trị 1 nữa.



Tín hiệu RESETbar tích cực trong 0.498ns



Tín hiệu RESETbar tích cực trong 0.499ns

Nhận xét:

Có thể thấy khi Resetbar trở nên inactive tiến gần đến cạnh lên xung CLK, ngõ ra không nhận được giá trị 1 nữa mặc dù Resetbar kết thúc trước cạnh CLK do mạch đã vi phạm Recovery Time.

Tiến hành đo t_{hold} bằng lệnh .meas ở thời điểm trước khi vi phạm Hold Time :

```
.meas TRAN trec TRIG V(resetbar)=0.5 RISE=1 TARG V(clk)=0.5 RISE=1
```

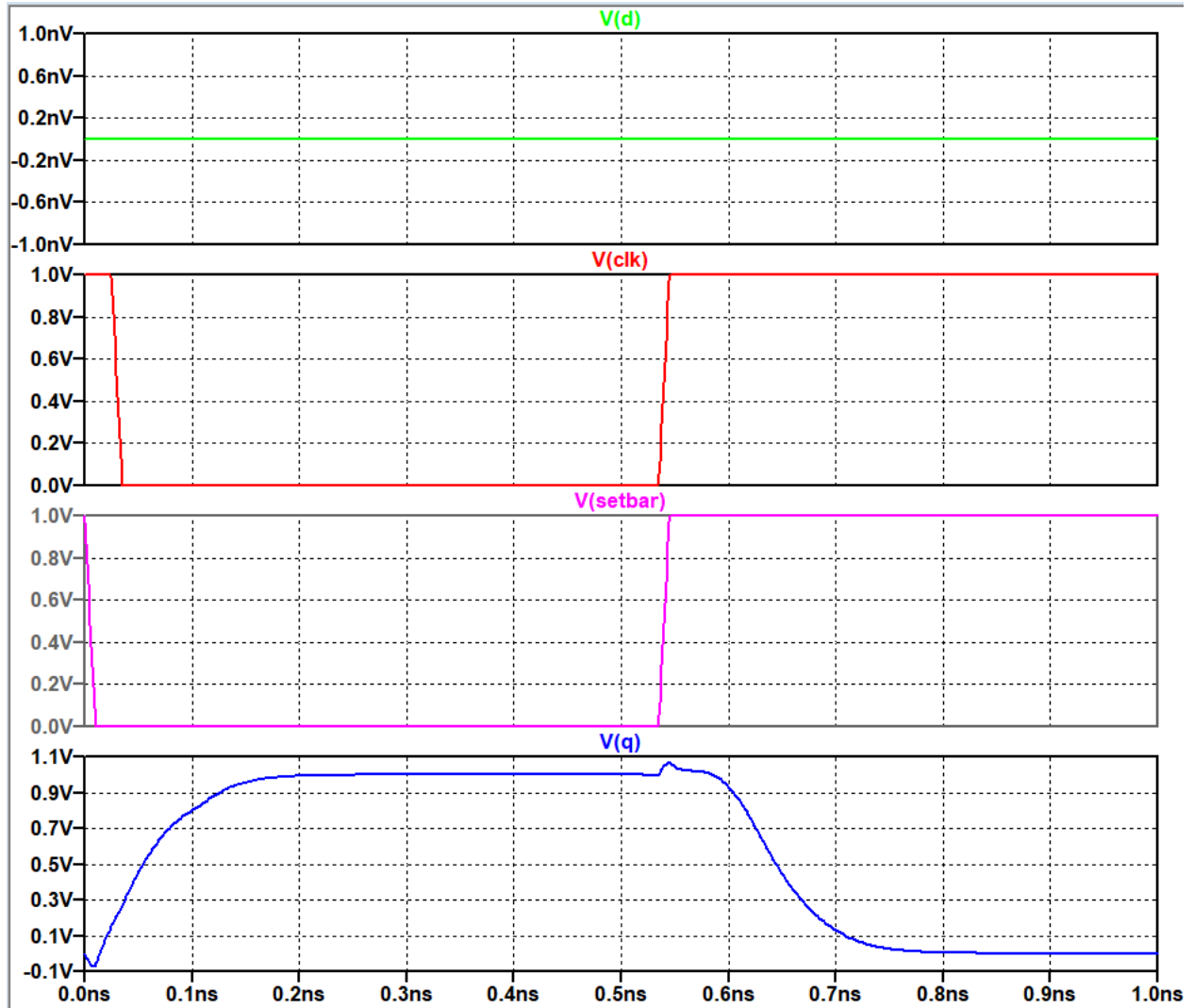
```
LTspice 24.1.8 for Windows
Circuit: C:\Documents\LTspice\Project\FFDTG.net
Start Time: Thu Jun 12 14:41:31 2025
solver = Normal
Maximum thread count: 16
tnom = 27
temp = 27
method = modified trap
Direct Newton iteration for .op point succeeded.
Total elapsed time: 0.202 seconds.

Files loaded:
C:\Documents\LTspice\Project\FFDTG.net
C:\Users\BAO\AppData\Local\LTspice\lib\cmp\standard.mos
C:\Documents\LTspice\Project\cmos.txt

trec=2.69999997314e-11 FROM 5.1300000014e-10 TO 5.39999999871e-10
```

Kết luận: Thời gian Recovery Time($t_{rec,reset}$) là 27ps.

Thực hiện tương tự với tín hiệu Setbar:



Dạng sóng được dùng để đo Recovery Time cho tín hiệu SETbar

Nhận xét:

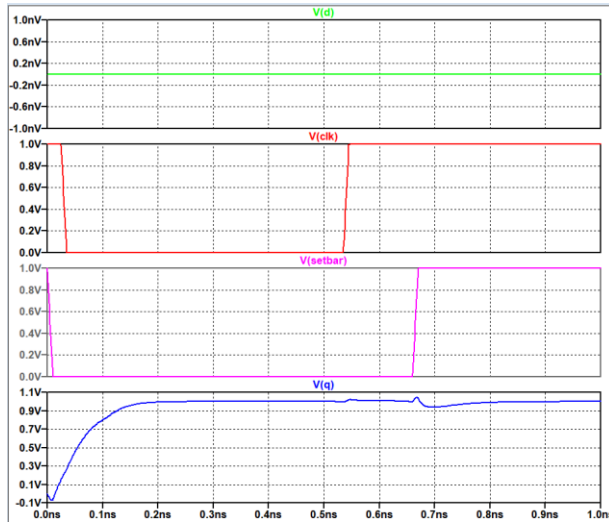
Có thể thấy dù SETbar có inactive tại cạnh lên của xung CLK, mạch vẫn hoạt động đúng cho thấy không tồn tại thời gian Recovery cho tín hiệu SETbar.

Kết luận: Thời gian Recovery Time($t_{rec,set}$) là 0ps.

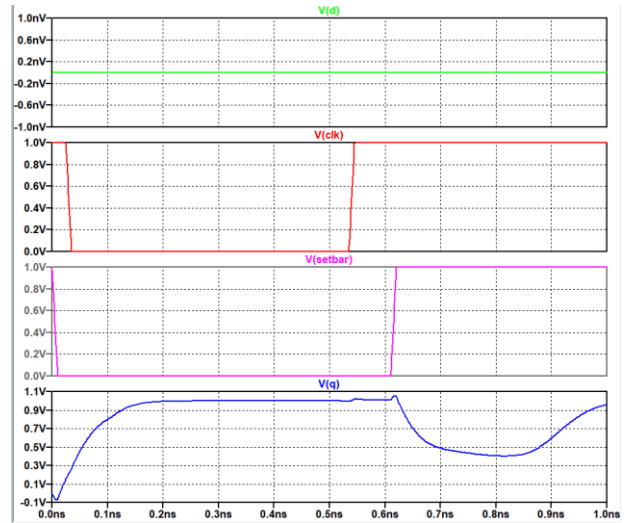
- **Removal Time:**

Là khoảng thời gian tối thiểu mà tín hiệu điều khiển bất đồng bộ (như Set hoặc Reset) phải được giữ ở trạng thái active (asserted) sau cạnh đồng hồ để đảm bảo flip-flop hoạt động đúng mà không bị lỗi.

Cho thời gian inactive của SETbar tiến gần tới sau cạnh lên xung CLK đến khi tín hiệu ngõ ra Q không ghi được giá trị 1 nữa.



Tín hiệu SETbar tích cực trong 0.65ns



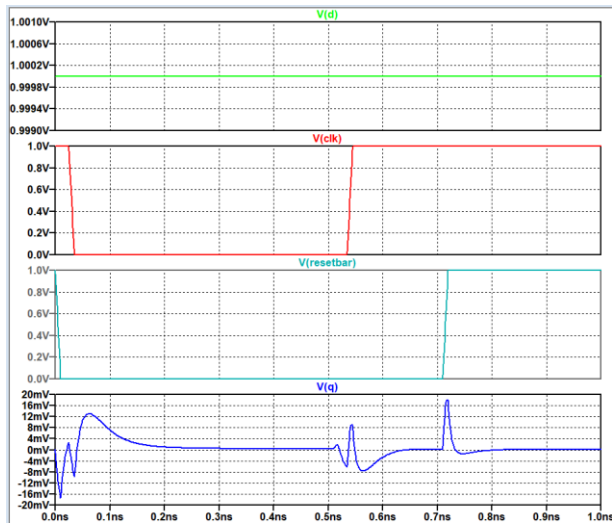
Tín hiệu SETbar tích cực trong 0.60ns

Nhận xét:

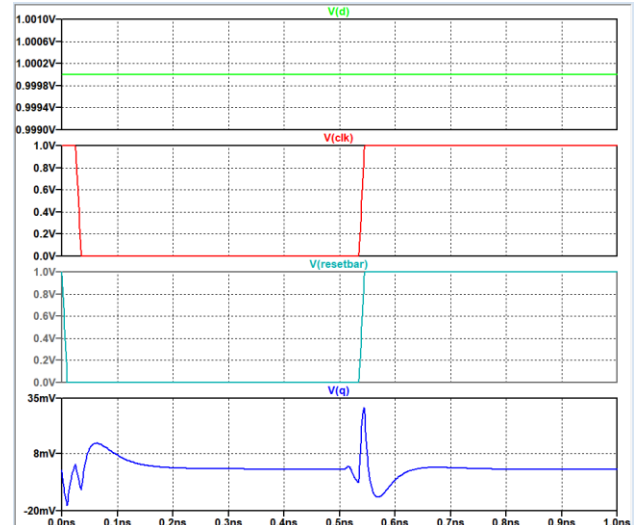
Có thể thấy khi SETbar trở nên inactive tiến gần đến sau cạnh lên xung CLK, ngõ ra không nhận được giá trị 1 nữa mặc dù SETbar kết thúc sau cạnh CLK do mạch đã vi phạm Removal Time.

Kết luận: Thời gian Removal Time($t_{rem,set}$) là 75ps

Thực hiện tương tự với RESETbar:



Tín hiệu RESETbar tích cực trong 0.7ns



Tín hiệu RESETbar tích cực trong 0.525ns

Nhận xét: Có thể thấy dù RESETbar có inactive tại cạnh lên của xung CLK, mạch vẫn hoạt động đúng cho thấy không tồn tại thời gian Removal cho tín hiệu RESETbar.

Kết luận: Thời gian Recovery Time($t_{rec,set}$) là 0ps.

3.1.6. Đo Maximun Frequency/Minumum Clock Period

- **Chu kì tối thiểu(t_{min}):**

Là chu kỳ đồng hồ tối thiểu, tức khoảng thời gian ngắn nhất giữa hai cạnh đồng hồ liên tiếp (thường là từ cạnh lên đến cạnh lên tiếp theo hoặc từ cạnh xuống đến cạnh xuống) mà flip-flop có thể hoạt động ổn định mà không vi phạm các yêu cầu thời gian (như Setup Time, Hold Time, Clock-to-Q delay). t_{min} được tính bằng công thức:

$$t_{min} = t_{pd} + t_{setup} + t_{rise} + t_{fall}.$$

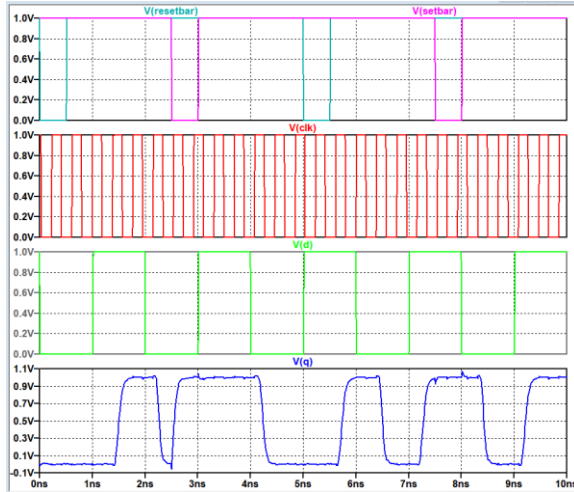
Với các giá trị đã có ở phân tích phía trên, ta có thể tính được t_{min} theo lý thuyết là:

$$t_{min} = 130 + 30 + 122 + 103 = 385ps$$

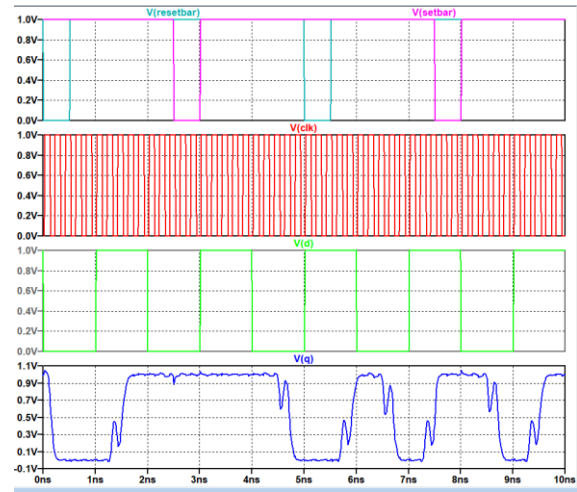
- **Tần số tối đa(f_{max}):**

Là tần số đồng hồ tối đa mà flip-flop và hệ thống vi mạch có thể hoạt động ổn định mà không vi phạm các yêu cầu thời gian (như Setup Time, Hold Time, Clock-to-Q delay). f_{max} được tính bằng công thức: $f_{max} = \frac{1}{t_{min}} = \frac{1}{385 \cdot 10^{-12}} \approx 2.6GHz$

Tiến hành kiểm tra trên LTSpice với xung CLK và \overline{CLK} có $t_{min} = 385ps$:



Kết quả mô phỏng hoạt động của DFF với
SR bất đồng bộ với $t_{min} = 385ps$



Kết quả mô phỏng hoạt động của DFF với
SR bất đồng bộ với $t_{min} = 300ps$

Nhận xét: Với chu kì $t_{min} = 385ps$, Dạng sóng của DFF bắt đầu có dấu hiệu metastable nhưng chức năng logic vẫn chưa bị vi phạm. Tuy nhiên với $t_{min} = 300ps$, mạch bắt đầu hoạt động sai chức năng và xuất hiện tình trạng metastable tại cạnh lên/xuống.

3.2. KHẢO SÁT THÔNG SỐ ĐIỆN NĂNG

3.2.1. Công suất động:

Công suất tiêu thụ do sự chuyển mạch (switching) của các tín hiệu trong Flip-flop, xảy ra khi trạng thái đầu ra (Q) hoặc các node nội bộ thay đổi ($0 \leftrightarrow 1$).

Công suất động phụ thuộc rất nhiều vào hoạt động của mạch cũng như tổng điện dung kí sinh trong từng transistor do đó không thể đo trực tiếp trên LTSpice.

Ta có công thức tính công suất động theo lý thuyết: $P_{dynamic} = \alpha \times C \times V_{DD}^2 \times f$

3.2.2. Công suất tĩnh:

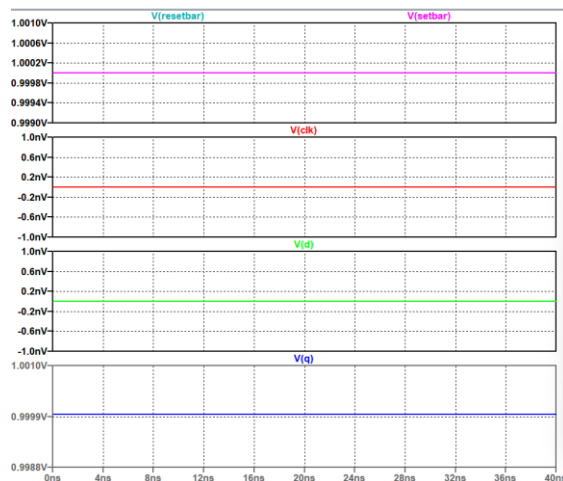
Công suất tiêu thụ do dòng rò trong flip-flop, ngay cả khi không có chuyển mạch.

Ta có công thức tính công suất tĩnh theo lý thuyết: $P_{static} = I_{leak} \times V_{DD}$.

Đối với phạm vi đồ án, thực hiện đo công suất với CLK = 0, D = 0 và SR = 1 để có được công suất tĩnh.

Sử dụng lệnh .meas để đo công suất tĩnh tự động:

```
.meas TRAN P_static AVG V(VDD)*-I(VDD) FROM 0 TO 80ns
```



Mạch ở trạng thái tĩnh

```
LTspice 24.1.8 for Windows
Circuit: C:\Documents\LTspice\Project\FDGTG.net
Start Time: Thu Jun 12 16:35:08 2025
solver = Normal
Maximum thread count: 16
tnom = 27
temp = 27
method = modified trap
Direct Newton iteration for .op point succeeded.
Total elapsed time: 0.999 seconds.

Files loaded:
C:\Documents\LTspice\Project\FDGTG.net
C:\Users\BAO\AppData\Local\LTspice\lib\cmp\standard.mos
C:\Documents\LTspice\Project\cmos.txt

p_static: AVG(V(VDD)*-I(VDD))=3.70722668784e-07 FROM 0 TO 8e-08
```

Công suất tĩnh đo được

Nhận xét: Công suất tĩnh của mạch là $P_{static} = 37\mu W$.

3.2.3. Công suất tổng:

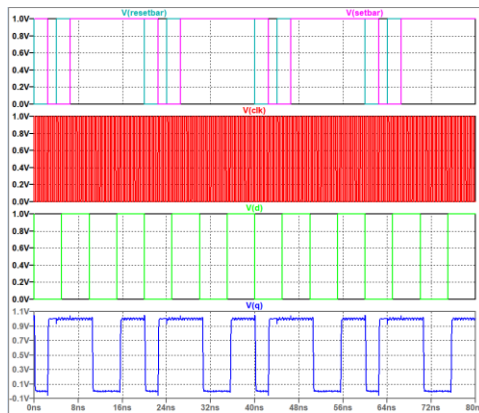
Là tổng công suất của công suất tĩnh và công suất động hay nói cách khác là công suất của mạch.

$$P_{total} = P_{static} + P_{dynamic}$$

Thực hiện đo công suất với CLK có chu kỳ là 0.5ns, tín hiệu D có chu kỳ 10ns, tín hiệu SET/RESET có chu kỳ 20ns, thời gian mô phỏng trong 80ns.

Sử dụng lệnh `.meas` để đo công suất tổng tự động:

```
.meas TRAN P_dynamic AVG V(VDD)*-I(VDD) FROM 0 TO 80ns
```



Mạch ở trạng thái hoạt động

```
LTspice 24.1.8 for Windows
Circuit: C:\Documents\LTspice\Project\FFDTG.net
Start Time: Thu Jun 12 16:47:18 2025
solver = Normal
Maximum thread count: 16
tnom = 27
temp = 27
method = modified trap
Direct Newton iteration for .op point succeeded.
Total elapsed time: 0.924 seconds.

Files loaded:
C:\Documents\LTspice\Project\FFDTG.net
C:\Users\BAO\AppData\Local\LTspice\lib\cmp\standard.mos
C:\Documents\LTspice\Project\cmos.txt

p_static: AVG(V(VDD)*-I(VDD))=9.68812081474e-05 FROM 0 TO 8e-08
```

Công suất tổng đo được

Nhận xét: Công suất tổng đo được là $P_{total} = 97\mu W$.

Tính công suất động:

$$P_{total} = P_{static} + P_{dynamic}$$

Suy ra:

$$P_{dynamic} = P_{total} - P_{static} = 97 - 0.37 = 96.63\mu W$$

Kết luận:

Công suất tĩnh đo được là $0.37\mu W$, trong khi công suất động trung bình khi mạch chuyển trạng thái đạt $97\mu W$ tại tần số 2GHz. Điều này cho thấy thiết kế hoạt động hiệu quả và tiết kiệm năng lượng trong cả hai trạng thái tĩnh và động, phù hợp với yêu cầu của các ứng dụng tốc độ cao và tiêu thụ thấp.

3.3. KHẢO SÁT KHẢ NĂNG CHỐNG NHIỄU:

3.3.1. Khái quát:

4. Mục tiêu:

Đánh giá khả năng hoạt động ổn định của DFF khi có nhiễu tại các đầu vào: V_{DD} , CLK, D, Set/Reset.

Xác định mức nhiễu tối đa mà DFF vẫn hoạt động đúng.

5. Nguyên tắc khảo sát:

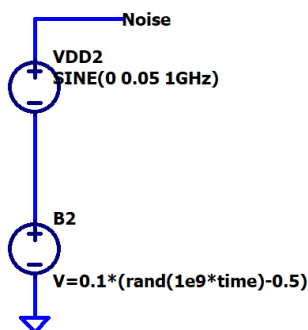
Nhiễu là dao động không mong muốn làm sai lệch tín hiệu, có thể gây lỗi logic.

Loại nhiễu khảo sát:

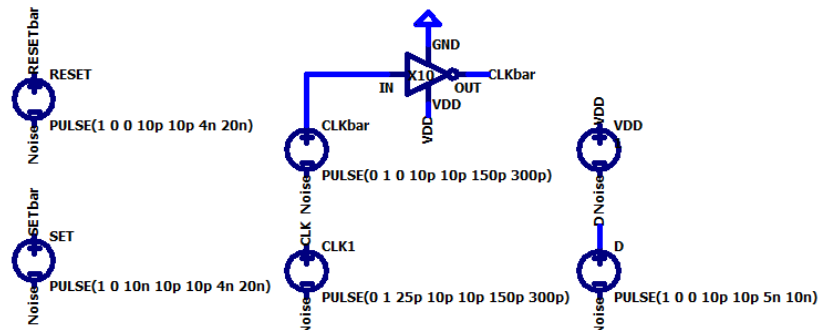
- Nhiễu tại V_{DD} : Dao động điện áp nguồn (ví dụ, $\pm 10\% V_{DD}$).
- Nhiễu tại CLK: Dao động biên độ hoặc jitter trên xung đồng hồ.
- Nhiễu tại D: Biến dạng tín hiệu dữ liệu.
- Nhiễu tại Set/Reset: Biến dạng tín hiệu điều khiển bất đồng bộ (active-low).

3.3.2. Khảo sát nhiễu:

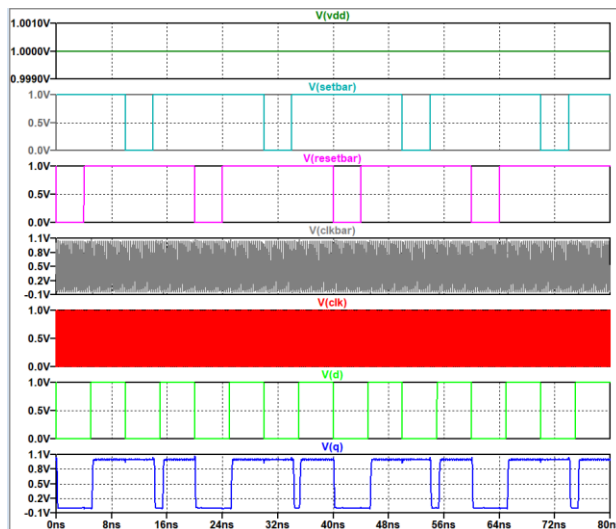
Tạo nguồn nhiễu: tích hợp nhiễu ngẫu nhiên rand() cùng với nhiễu sine() thành một nguồn nhiễu với biên độ 0.1. Cấp tín hiệu nhiễu vào các tín hiệu CLK, \overline{CLK} , Set/Reset và D, khảo sát hoạt động sau khi tích hợp nhiễu.



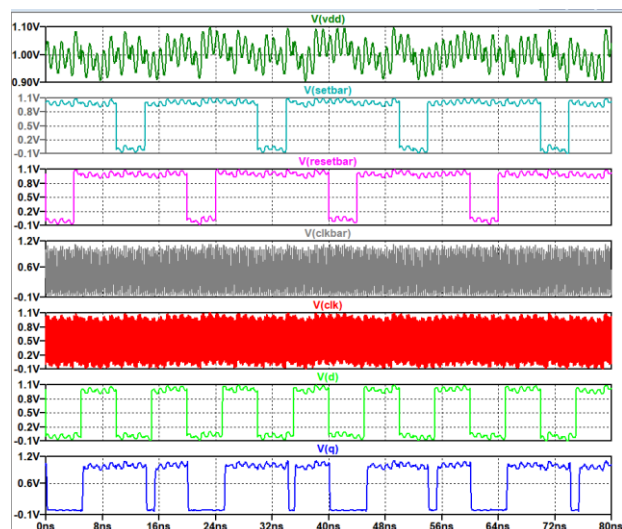
Nguồn nhiễu



Tích hợp nguồn nhiễu vào các tín hiệu



Tín hiệu ngõ vào và ra trước khi thêm nhiễu



Tín hiệu ngõ vào và ra sau khi thêm nhiễu

Nhận xét: Với nhiễu biên độ nhiễu 10%, hoạt động của Flip-flop chỉ bị ảnh hưởng nhỏ tại mức logic 1, ngoài ra về chức năng vẫn được đảm bảo.

3.4. SO SÁNH VỚI CÁC GIÁ TRỊ THAM CHIẾU

Thông số	Ký hiệu	Giá trị đo được	Giá trị tham chiếu
Clock-to-Q Delay (Propagation delay)	t_{cq}/t_{pd}	130ps	80 – 150ps
Rise Time	t_{rise}	121ps	30 – 80ps
Fall Time	t_{fall}	102ps	30 – 80ps
Setup Time	t_{setup}	50ps	50 – 90ps
Hold Time	t_{hold}	9ps	–20 – 20ps
Recovery Time	t_{rec}	27ps	60 – 100ps
Removal Time	t_{rem}	75ps	0 – 50ps
Maximun Frequency	f_{max}	2.5GHz	1.5 – 2.5GHz
Dynamic Power	$P_{dynamic}$	97μW	50 – 200μW
Static Power	P_{static}	0.37μW	0.1 – 5μW

Chú thích: Các số liệu tham chiếu được thu thập từ nhiều nguồn với độ tin cậy tương đối. Chi tiết ở Phụ Lục.

TỔNG KẾT

1. TÓM TẮT MỤC TIÊU VÀ KẾT QUẢ:

Đồ án tập trung thiết kế và khảo sát một D Flip-Flop (DFF) sử dụng cấu trúc master-slave latch, tích hợp chức năng Set/Reset bất đồng bộ tích cực mức thấp, dựa trên công nghệ CMOS 50nm, mô phỏng bằng LTspice với thư viện cmos.txt. Mục tiêu là phân tích các thông số thời gian, công suất tiêu thụ, khả năng chống nhiễu, và so sánh với các giá trị tham chiếu của cell tiêu chuẩn.

Các kết quả chính từ Chương 3 bao gồm:

1.1. Thông số thời gian:

- Clock-to-Q Delay: 130 ps (trong khoảng tham chiếu 80–150 ps).
- Setup Time: 50 ps (trong khoảng 50–90 ps).
- Hold Time: 9 ps (trong khoảng -20–20 ps).
- Recovery Time: 27 ps (thấp hơn tham chiếu 60–100 ps).
- Removal Time: 75 ps (cao hơn tham chiếu 0–50 ps).
- Tần số tối đa: 2.5 GHz (gần mức tối đa tham chiếu 1.5–2.5 GHz).

1.2. Công suất tiêu thụ:

- Static Power: 0.37 μ W (trong khoảng tham chiếu 0.1–5 μ W).
- Dynamic Power: 96.63 μ W.
- Total Power: 97 μ W (trong khoảng tham chiếu 50–200 μ W).

1.3. Khả năng chống nhiễu:

- Chấp nhận biên độ nhiễu <10%.

2. VÌ SAO CHỌN THIẾT KẾ LATCH DÙNG TRANSMISSION GATE

2.1 So sánh kiến trúc cổng logic với transmission gate

Tiêu chí	Cổng logic (NAND/NOR)	Transmission Gate (TG)
Số lượng transistor	Nhiều hơn (thường ≥ 40)	Ít hơn (thường 24–30)
Mức độ logic (logic depth)	Nhiều tầng, delay lớn	Ngắn gọn, ít tầng
Đường truyền tín hiệu	Dễ bị lệch pha, skew	Truyền trực tiếp, delay đối xứng
Tạo xung clock điều khiển latch	Thủ công, dễ mất đồng bộ	Tự nhiên khi dùng CLK/CLK_bar
Khả năng truyền dữ liệu	Phụ thuộc delay logic	Truyền tín hiệu gần như tuyến tính

2.2 Vì sao thiết kế thuần logic dễ bị metastable:

Độ trễ không cân bằng

- Thiết kế bằng NAND2/NAND3 thường có các chuỗi logic phức tạp → đường truyền từ D đến Q có nhiều tầng logic → độ trễ không đồng đều.
- Điều này khiến cho dữ liệu chưa ổn định tại thời điểm clock chốt vào → gây metastability.

Khó tạo xung điều khiển chính xác

- Nếu điều khiển latch bằng clock thủ công (không có TG), rất dễ có skew giữa xung điều khiển, làm thời điểm chốt không chính xác.

Tín hiệu feedback không được kiểm soát tốt

- Flip-flop dùng latch logic dễ bị chòng tín hiệu phản hồi và tín hiệu đầu vào → gây trạng thái hỗn hợp tại Q → metastable.

2.3 Vì sao thiết kế dùng Transmission Gate hiệu quả hơn?

Đường tín hiệu đơn giản và cân bằng

- Transmission gate cho phép **truyền trực tiếp tín hiệu D vào latch** khi TG mở → không cần nhiều tầng logic → delay thấp, ít nhiễu.

Đồng bộ tốt hơn

- TG điều khiển bằng **CLK và CLK_bar**, nên **hoạt động ăn khớp tự nhiên**, đảm bảo đúng pha giữa master và slave.

Ít nguy cơ feedback không ổn định

- Kiến trúc latch dạng TG có **vùng chốt rõ ràng**: chỉ khi TG tắt, dữ liệu bị giữ. Không có xung động hoặc nhiễu phản hồi không mong muốn.

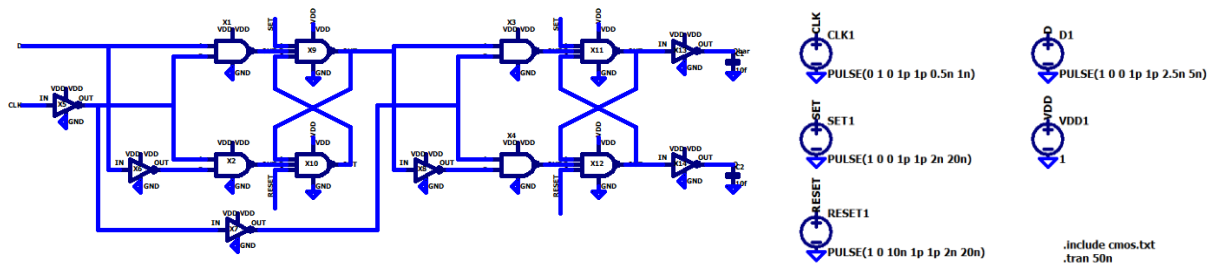
Nhận xét:

Thiết kế DFF sử dụng cổng logic hoàn toàn (52 transistor) cho thấy tín hiệu đầu ra xuất hiện hiện tượng metastability kể cả ở tần số 1GHz. Nguyên nhân được cho là do đường truyền tín hiệu qua nhiều tầng logic dẫn đến độ trễ không đồng đều và khó kiểm soát thời điểm chốt dữ liệu giữa master và slave latch.

Trong khi đó, flip-flop sử dụng kiến trúc transmission gate (28 transistor) hoạt động ổn định hơn do có đường tín hiệu ngắn gọn, điều khiển pha bằng CLK/CLK_bar trực tiếp, giúp giảm thiểu skew và đảm bảo tính đồng bộ giữa hai latch.

Kết quả cho thấy việc chọn kiến trúc hợp lý đóng vai trò quan trọng hơn là số lượng transistor trong hiệu năng và độ tin cậy của DFF.

2.4 Tiến hành thử nghiệm trên Latch Logic:



Tiến hành đo các thông số như Propagation Delay, Rise Time, Fall Time, Maximum Frequency thu được bảng giá trị sau:

Các cổng logic sử dụng có kích thước và cấu tạo tương đương với thiết kế sử dụng transmission gate.

Thông số	Giá trị
Propagation Delay	222ps
Rise Time	40ps
Fall Time	50ps
Maximum Frequency	1GHz

Có thể thấy mạch DFF sử dụng hoàn toàn bằng cổng logic không đáp ứng được nhu cầu về tốc độ cao đã đề ra từ đầu.

3. ĐÁNH GIÁ THIẾT KẾ

Ưu điểm:

- Static Power ($0.37 \mu\text{W}$) thấp, phù hợp với ứng dụng tiêu thụ thấp.
- Total Power ($97 \mu\text{W}$) hợp lý tại 2 GHz, tương đương cell tiêu chuẩn.
- Maximum Frequency (2.5 GHz) cho thấy khả năng hoạt động ở tốc độ cao.

Hạn chế:

- Rise/fall time cao.
- Removal time (75 ps) cao, ảnh hưởng đến hiệu suất Set/Reset.

4. ĐỀ XUẤT CẢI TIẾN(Đã cải tiến trong phần thuyết trình)

Cải thiện Timing.

- Thêm mạch lọc nhiễu cho V_{DD} và tín hiệu Set/Reset.

Mở rộng nghiên cứu:

- Khảo sát DFF ở các công nghệ tiên tiến (22nm, 7nm) hoặc tích hợp vào hệ thống lớn hơn (thanh ghi, bộ đếm).

5. KẾT LUẬN

Thiết kế DFF đạt các mục tiêu đề ra, với các thông số thời gian, công suất, và chống nhiễu nằm trong hoặc gần phạm vi tham chiếu của cell tiêu chuẩn. Công suất tĩnh thấp và tần số tối đa cao cho thấy tiềm năng ứng dụng trong các hệ thống số tốc độ cao, tiêu thụ thấp. Tuy nhiên, các hạn chế về rise/fall time cần được khắc phục. Đồ án không chỉ giúp hiểu rõ nguyên lý hoạt động của DFF mà còn rèn luyện kỹ năng thiết kế, mô phỏng, và phân tích mạch điện tử, tạo nền tảng cho các nghiên cứu tiếp theo.

PHỤ LỤC

1. Tài liệu tham khảo

Các giá trị tham chiếu (trang 58) được tổng hợp từ các nguồn sau:

1. Weste, N. H. E., & Harris, D. (2010). *CMOS VLSI Design: A Circuits and Systems Perspective (4th Ed.)*. Addison-Wesley: Cung cấp thông số công suất và thời gian của DFF chuẩn (Chương 5, 7).
2. Rabaey, J. M., Chandrakasan, A., & Nikolić, B. (2003). *Digital Integrated Circuits: A Design Perspective (2nd Ed.)*. Prentice Hall: Thông tin về công suất tĩnh/động(Chương 6).
3. Predictive Technology Model (PTM). (<http://ptm.asu.edu/>): Mô hình CMOS 45nm/65nm, cung cấp thông số dòng rò và điện dung.
4. TSMC 45nm Standard Cell Library Datasheet: Tham chiếu công suất tĩnh (0.1–5 μ W), công suất động (50–200 μ W), và thông số thời gian.
5. Uyemura, J. P. (2002). *Introduction to VLSI Circuits and Systems*. Wiley: Thông tin về thiết kế latch và flip-flop (Chương 8).
6. Khác:
 - Nangate 45nm
 - Nangate simulation @ 1V, 25°C, 50nm
 - Nangate 45nm .lib, thực nghiệm trong bài báo IEEE
 - IEEE JSSC papers (2018–2022)
 - IEEE "Low Power DFF" papers
 - IEEE flip-flop metastability papers
 - MOSIS OSU libraries
 - Synopsys Liberty cells
 - Synopsys Design Compiler reports

Các hình ảnh tham khảo(Chương 1) được tổng hợp từ các nguồn sau:

1. www.geeksforgeeks.org
2. www.build-electronic-circuits.com
3. www.realdigital.org
4. www.electronics-tutorials.ws

2. SƠ ĐỒ MẠCH D FLIP-FLOP SỬ DỤNG TRONG ĐỒ ÁN: Netlist LTSpice:

```
VDD VDD Noise 1
V$D D Noise PULSE(1 0 0 10p 10p 5n 10n)
V$RESET RESETbar Noise PULSE(1 0 0 10p 10p 4n 20n)
V$SET SETbar Noise PULSE(1 0 10n 10p 10p 4n 20n)
V$CLKbar N001 Noise PULSE(0 1 0 10p 10p 150p 300p)
X$X10 0 N001 VDD CLKbar Inverter
V$CLK1 CLK Noise PULSE(0 1 25p 10p 10p 150p 300p)
X$X7 CLK CLKbar 0 D VDD N005 TG
X$X14 SETbar N004 0 VDD Q NAN2
X$X9 CLKbar CLK 0 N003 VDD N004 TG
X$X3 RESETbar N005 0 VDD N003 NAN2
X$X18 N003 SETbar CLK CLKbar 0 VDD N005 TristateNAND2
X$X12 Q RESETbar CLKbar CLK 0 VDD N004 TristateNAND2
C3 Qbar 0 10f
C4 Q 0 10f
X$X13 0 Q VDD Qbar Inverter
B2 N002 0 V=0.2*(rand(1e9*time)-0.5)
VDD2 Noise N002 SINE(0 0.1 1GHz)

* block symbol definitions
.subckt Inverter GND IN VDD OUT
M1 VDD IN OUT VDD P_50n l=50n w=2u
M2 OUT IN GND GND N_50n l=50n w=1u
.ends Inverter

.subckt TG CLK CLKbar GND IN VDD OUT
M1 OUT CLKbar IN GND N_50n l=50n w=1u
M2 OUT CLK IN VDD P_50n l=50n w=2u
.ends TG

.subckt NAN2 A B GND VDD OUT
M1 VDD A OUT VDD P_50n l=50n w=2u
M2 OUT A N001 GND N_50n l=50n w=2u
M3 VDD B OUT VDD P_50n l=50n w=2u
M4 N001 B GND GND N_50n l=50n w=2u
.ends NAN2

.subckt TristateNAND2 A B CLK CLKbar GND VDD OUT
M1 N001 CLKbar OUT VDD P_50n l=50n w=4u
M2 OUT CLK N002 GND N_50n l=50n w=3u
M3 VDD A N001 VDD P_50n l=50n w=4u
M4 VDD B N001 VDD P_50n l=50n w=4u
M5 N002 A N003 GND N_50n l=50n w=3u
M6 N003 B GND GND N_50n l=50n w=3u
.ends TristateNAND2

.model NMOS NMOS
.model PMOS PMOS
```



```
.lib C:\Users\BAO\AppData\Local\LTspice\lib\cmp\standard.mos
.include cmos.txt
.tran 80n
.meas TRAN P_static AVG V(VDD)*-I(VDD) FROM 0 TO 80ns
.backanno
.end
```

File cmos.txt:

```
.model N_50n nmos level = 54
+binunit = 1          paramchk= 1          mobmod = 0
+capmod = 2           igcmod = 1           igbmod = 1          geomod = 0
+diomod = 1           rdsmod = 0           rbodymod= 1         rgatemod= 1
+permod = 1           acnqsmod= 0          trnqsmod= 0
+tnom = 27            toxex = 1.4e-009     toxp = 7e-010        toxm = 1.4e-009
+epsrox = 3.9         wint = 5e-009        lint = 1.2e-008
+ll = 0              wl = 0                llm = 1             wln = 1
+lw = 0              ww = 0                lwn = 1             wwn = 1
+lw1 = 0             wwl = 0              xpart = 0           toxref = 1.4e-009
+vth0 = 0.22         k1 = 0.35              k2 = 0.05           k3 = 0
+k3b = 0             w0 = 2.5e-006         dvt0 = 2.8          dvt1 = 0.52
+dvt2 = -0.032       dvt0w = 0          dvt1w = 0           dvt2w = 0
+dsb = 2             minv = 0.05             voff1 = 0           dvtp0 = 1e-007
+dvtp1 = 0.05        lpe0 = 5.75e-008      lpeb = 2.3e-010     xj = 2e-008
+ngate = 5e+020       ndep = 2.8e+018      nsd = 1e+020        phin = 0
+cdsc = 0.0002        cdsb = 0             cdsd = 0            cit = 0
+voff = -0.15         nfactor = 1.2         eta0 = 0.15         etab = 0
+vfb = -0.55         u0 = 0.032            ua = 1.6e-010       ub = 1.1e-017
+uc = -3e-011         vsat = 1.1e+005       a0 = 2              ags = 1e-020
+a1 = 0              a2 = 1                b0 = -1e-020        b1 = 0
+keta = 0.04          dwg = 0              dwb = 0             pclm = 0.18
+pdiblc1 = 0.028      pdiblc2 = 0.022       pdiblc3 = -0.005    drout = 0.45
+pvag = 1e-020        delta = 0.01          pscbe1 = 8.14e+008  pscbe2 = 1e-007
+fprout = 0.2         pdits = 0.2           pditsd = 0.23       pdits1 = 2.3e+006
+rsh = 3              rdsw = 150            rsw = 150           rdw = 150
+rdswmin = 0          rdwmin = 0            rswmin = 0          prwg = 0
+prwb = 6.8e-011      wr = 1                alpha0 = 0.074       alpha1 = 0.005
+beta0 = 30           agidl = 0.0002        bgidl = 2.1e+009    cgidl = 0.0002
+egidl = 0.8
+aigbacc = 0.012      bigbacc = 0.0028      cigbacc = 0.002
+nigbacc = 1          aigbinv = 0.014       bigbinv = 0.004      cigbinv = 0.004
+eigbinv = 1.1        nigbinv = 3           aigc = 0.017        bigc = 0.0028
+cigc = 0.002         aigsd = 0.017        bigsd = 0.0028      cigsd = 0.002
+nigc = 1            poxedg = 1           pigcd = 1           ntox = 1
+xrcrg1 = 12          xrcrg2 = 5
+cgso = 6.238e-010    cgdo = 6.238e-010    cgbo = 2.56e-011     cgd1 = 2.495e-10
+cgs1 = 2.495e-10     ckappas = 0.02        ckappad = 0.02       acde = 1
+moin = 15            noff = 0.9            voffcv = 0.02
+kt1 = -0.21          kt11 = 0              kt2 = -0.042         ute = -1.5
+ua1 = 1e-009         ub1 = -3.5e-019       uc1 = 0              prt = 0
+at = 53000
+fnoimod = 1          tnoimod = 0
+jss = 0.0001         jsws = 1e-011         jswgs = 1e-010      njs = 1
+ijthsfwd = 0.01      ijthsrev = 0.001      bvs = 10            xjbvs = 1
+jsd = 0.0001         jswd = 1e-011         jswgd = 1e-010      njd = 1
+ijthd fwd = 0.01     ijthdrev = 0.001      bvd = 10            xjbvd = 1
+pbs = 1              cjs = 0.0005          mjs = 0.5           pbsws = 1
+cjsws = 5e-010       mjsws = 0.33          pbswgs = 1          cjswgs = 5e-010
+mjswgs = 0.33        pbd = 1               cjd = 0.0005        mjd = 0.5
+pbswd = 1            cjswd = 5e-010        mjswd = 0.33        pbswgd = 1
+cjswgd = 5e-010     mjswgd = 0.33         tpb = 0.005          tcj = 0.001
+tpbsw = 0.005        tcjsw = 0.001         tpbswg = 0.005       tcjswg = 0.001
+xtis = 3             xtld = 3
```

+dmcg	= 0e-006	dmci	= 0e-006	dmdg	= 0e-006	dmcgt	= 0e-007
+dwj	= 0e-008	xgw	= 0e-007	xgl	= 0e-008		
+rshg	= 0.4	gbmin	= 1e-010	rbpb	= 5	rbpd	= 15
+rbps	= 15	rbdb	= 15	rbsb	= 15	ngcon	= 1
*							
.model P_50n pmos level = 54							
+binunit	= 1	paramchk	= 1	mobmod	= 0		
+capmod	= 2	igcmod	= 1	igbmod	= 1	geomod	= 0
+diomod	= 1	rdsmode	= 0	rbodymod	= 1	rgatemod	= 1
+permod	= 1	acnqsmode	= 0	trnqsmode	= 0		
+tnom	= 27	toxe	= 1.4e-009	toxp	= 7e-010	toxm	= 1.4e-009
+epsrox	= 3.9	wint	= 5e-009	lint	= 1.2e-008		
+ll	= 0	wl	= 0	lln	= 1	wln	= 1
+lw	= 0	ww	= 0	lwn	= 1	wwn	= 1
+lwl	= 0	wwl	= 0	xpart	= 0	toxref	= 1.4e-009
+vth0	= -0.22	k1	= 0.39	k2	= 0.05	k3	= 0
+k3b	= 0	w0	= 2.5e-006	dvt0	= 3.9	dvt1	= 0.635
+dvt2	= -0.032	dvt0w	= 0	dvt1w	= 0	dvt2w	= 0
+dsusb	= 0.7	minv	= 0.05	voffl	= 0	dvt0	= 0.5e-008
+dvtp1	= 0.05	lpe0	= 5.75e-008	lpeb	= 2.3e-010	xj	= 2e-008
+ngate	= 5e+020	ndep	= 2.8e+018	nsd	= 1e+020	phin	= 0
+cdsc	= 0.000258	cdscb	= 0	cdscd	= 6.1e-008	cit	= 0
+voff	= -0.15	nfactor	= 2	eta0	= 0.15	etab	= 0
+vfb	= 0.55	u0	= 0.0095	ua	= 1.6e-009	ub	= 8e-018
+uc	= 4.6e-013	vsat	= 90000	a0	= 1.2	ags	= 1e-020
+a1	= 0	a2	= 1	b0	= -1e-020	b1	= 0
+keta	= -0.047	dwg	= 0	dwb	= 0	pclm	= 0.55
+pdiblc1	= 0.03	pdiblc2	= 0.0055	pdiblc3	= 3.4e-008	drout	= 0.56
+pvag	= 1e-020	delta	= 0.014	pscbe1	= 8.14e+008	pscbe2	= 9.58e-007
+fprout	= 0.2	pdits	= 0.2	pditsd	= 0.23	pdits1	= 2.3e+006
+rsh	= 3	rdsw	= 250	rsd	= 160	rdw	= 160
+rdswmin	= 0	rdwmin	= 0	rsdmin	= 0	prwg	= 3.22e-008
+prwb	= 6.8e-011	wr	= 1	alpha0	= 0.074	alpha1	= 0.005
+beta0	= 30	agidl	= 0.0002	bgidl	= 2.1e+009	cgidl	= 0.0002
+egidl	= 0.8						
+aigbacc	= 0.012	bigbacc	= 0.0028	cigbacc	= 0.002		
+nigbacc	= 1	aigbinv	= 0.014	bigbinv	= 0.004	cigbinv	= 0.004
+eigbinv	= 1.1	nigbinv	= 3	aigc	= 0.69	bigc	= 0.0012
+cigc	= 0.0008	aigsd	= 0.0087	bigsd	= 0.0012	cigsd	= 0.0008
+nigc	= 1	poxedge	= 1	bigcd	= 1	ntox	= 1
+xrcrg1	= 12	xrcrg2	= 5				
+cgso	= 7.43e-010	cgdo	= 7.43e-010	cgbo	= 2.56e-011	cgdl	= 1e-014
+cgsl	= 1e-014	ckappas	= 0.5	ckappad	= 0.5	acde	= 1
+moin	= 15	noff	= 0.9	voffcv	= 0.02		
+kt1	= -0.19	kt11	= 0	kt2	= -0.052	ute	= -1.5
+ual	= -1e-009	ub1	= 2e-018	uc1	= 0	prr	= 0
+at	= 33000						
+fnoimod	= 1	tnoimod	= 0				
+jss	= 0.0001	jsws	= 1e-011	jswgs	= 1e-010	njs	= 1
+ijthsfdw	= 0.01	ijthsrev	= 0.001	bvs	= 10	xjbvs	= 1
+jsd	= 0.0001	jswd	= 1e-011	jswgd	= 1e-010	njd	= 1
+ijthdfwd	= 0.01	ijthdrev	= 0.001	bvd	= 10	xjbvd	= 1
+pbs	= 1	cjs	= 0.0005	mjs	= 0.5	pbsws	= 1
+cjsws	= 5e-010	mjsws	= 0.33	pbswgs	= 1	cjswgs	= 5e-010
+mjswgs	= 0.33	pbd	= 1	cjd	= 0.0005	mjd	= 0.5
+pbswd	= 1	cjswd	= 5e-010	mjswd	= 0.33	pbswgd	= 1
+cjswgd	= 5e-010	mjswgd	= 0.33	tpb	= 0.005	tcj	= 0.001
+tpbsw	= 0.005	tcjsw	= 0.001	tpbswg	= 0.005	tcjswg	= 0.001
+xtis	= 3	xtid	= 3				
+dmcg	= 0e-006	dmci	= 0e-006	dmdg	= 0e-006	dmcgt	= 0e-007
+dwj	= 0e-008	xgw	= 0e-007	xgl	= 0e-008		
+rshg	= 0.4	gbmin	= 1e-010	rbpb	= 5	rbpd	= 15
+rbps	= 15	rbdb	= 15	rbsb	= 15	ngcon	= 1