

Sistemas Digitais

ET46B

Prof. Eduardo Vinicius Kuhn

kuhn@utfpr.edu.br

Curso de Engenharia Eletrônica

Universidade Tecnológica Federal do Paraná



Capítulo 5

Flip-Flops e dispositivos correlatos

Conteúdo

5.1 *Latch* com portas NAND

5.2 *Latch* com portas NOR

5.5 Sinais de clock

5.6 FF SR

5.7 FF JK

5.8 FF D

5.9 *Latch* D (transparente)

5.10 Entradas assíncronas

5.11 Temporização em FFs

5.13 Aplicações com FFs

5.17 Armazenamento e transferência

5.18 Transferência serial de dados

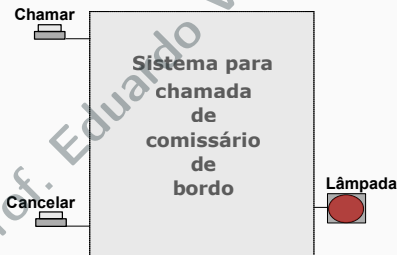
5.19 Divisão de frequência e
contagem

Objetivos

- Construir e analisar um *latch* com portas NOR ou NAND.
- Descrever o funcionamento de FFs (e.g., SR, JK, T e D).
- Conectar FFs D para formar registradores.
- Destacar diferenças entre carga e/ou transferência serial e paralela de dados.
- Apresentar características de sistemas síncronos e assíncronos.
- Construir circuitos divisores de frequência e contadores com FFs.
- Usar diagramas de transição de estado para descrever o funcionamento de contadores.
- Discutir aspectos centrais sobre a temporização em FFs.

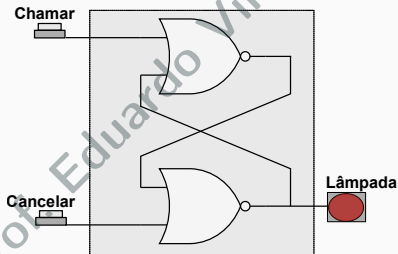
Exemplo: Projete um sistema com a seguinte lógica de operação:

- Um toque no botão “**chamar**” acende a lâmpada; e
- Um toque no botão “**cancelar**” apaga a lâmpada.



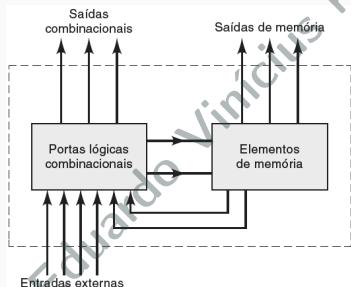
Exemplo: Projete um sistema com a seguinte lógica de operação:

- Um toque no botão “chamar” acende a lâmpada vermelha; e
- Um toque no botão “cancelar” apaga a lâmpada.



O sistema tem memória, visto que a saída não depende somente das entradas atuais, mas também do que aconteceu no passado.

Na prática, a maioria dos sistemas digitais é constituído de circuitos lógicos combinacionais e de **elementos de memória**.

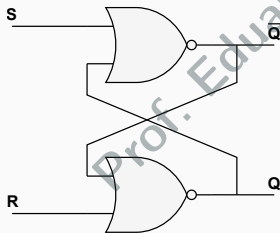


- **Circuito combinacional:** a saída depende apenas de valores atuais das entradas.
- **Circuito sequencial:** a saída depende do que aconteceu ao longo do tempo, i.e., do que consta em memória.

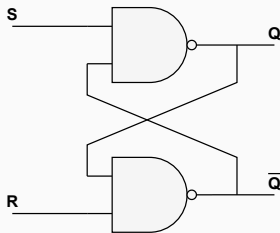
Introdução

- Embora uma porta lógica não tenha essa capacidade por si só, certas **topologias permitem armazenar informação**.
- Aplicando o **conceito de realimentação**, dá-se origem aos ***latches*** (assíncrono) e aos ***flip-flops*** (síncrono) (FFs).

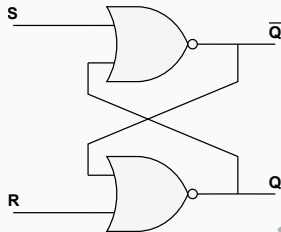
Latch com portas NOR



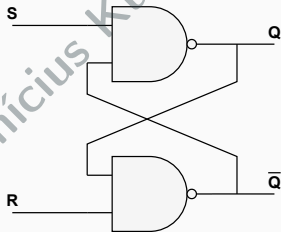
Latch com portas NAND



Latch com portas NOR



Latch com portas NAND

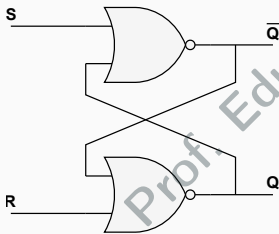


- A topologia é similar exceto pelas **saídas Q e \bar{Q} estarem em posições trocadas.**
- A lógica de operação das entradas S (*SET*) e R (*RESET*) é invertida de uma topologia para outra.

Latch SR com portas NOR

Com respeito à **topologia e operação**, tem-se

- S e R como entradas (repouso em nível BAIXO);
- Q e \overline{Q} como saídas; e
- Q_0 indica o estado prévio/anterior.

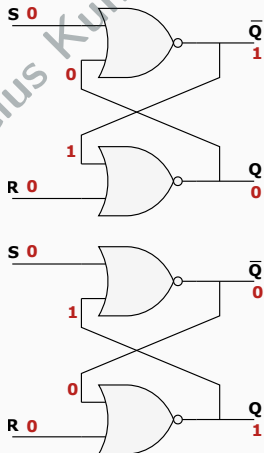


S	R	Q_0	Q	\overline{Q}
0	0	0		
0	0	1		
0	1	0		
0	1	1		
1	0	0		
1	0	1		
1	1	0		
1	1	1		

Latch SR com portas NOR

Caso S e R estejam em repouso (nível BAIXO):

S	R	Q_0	Q	\bar{Q}
0	0	0	0	1
0	0	1	1	0
0	1	0		
0	1	1		
1	0	0		
1	0	1		
1	1	0		
1	1	1		



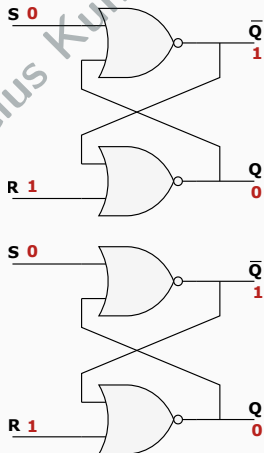
O estado atual Q da saída depende do estado prévio Q_0 .

kuhn@utfpr.edu.br | youtube.com/@eduardokuhn87

Latch SR com portas NOR

Caso R seja pulsada (nível ALTO):

S	R	Q_0	Q	\overline{Q}
0	0	0	0	1
0	0	1	1	0
0	1	0	0	1
0	1	1	0	1
1	0	0		
1	0	1		
1	1	0		
1	1	1		

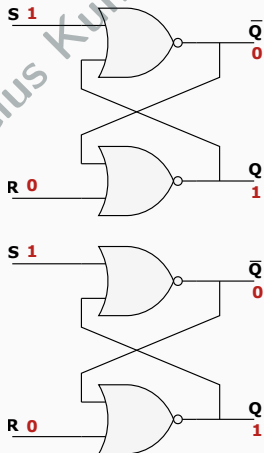


Quando $Q_0 = 1$, pode momentaneamente ocorrer $Q = 0$ e $\overline{Q} = 0$;
todavia, as saídas logo se estabilizam em $Q = 0$ e $\overline{Q} = 1$.

Latch SR com portas NOR

Caso S seja pulsada (nível ALTO):

S	R	Q_0	Q	\overline{Q}
0	0	0	0	1
0	0	1	1	0
0	1	0	0	1
0	1	1	0	1
1	0	0	1	0
1	0	1	1	0
1	1	0		
1	1	1		

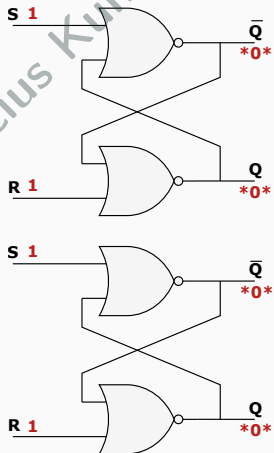


Quando $\overline{Q}_0 = 1$, pode momentaneamente ocorrer $Q = 0$ e $\overline{Q} = 0$;
todavia, as saídas logo se estabilizarão em $Q = 1$ e $\overline{Q} = 0$.

Latch SR com portas NOR

Caso S e R sejam pulsadas (nível ALTO):

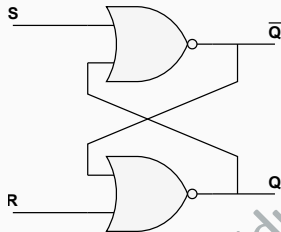
S	R	Q_0	Q	\bar{Q}
0	0	0	0	1
0	0	1	1	0
0	1	0	0	1
0	1	1	0	1
1	0	0	1	0
1	0	1	1	0
1	1	0	*	*
1	1	1	*	*



Não é possível determinar exatamente a saída; na prática, o estado resultante da saída depende de qual entrada retornou primeiro ao estado de repouso.

Latch SR com portas NOR

Portanto, a operação do *latch* SR com portas NOR pode ser resumida através da seguinte tabela-verdade:

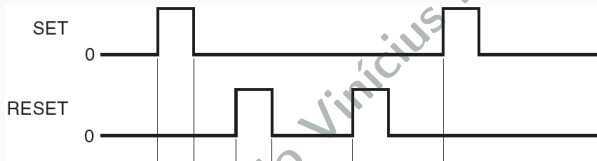


S	R	Q
0	0	Q_0 (mantém)
0	1	0 (reset)
1	0	1 (set)
1	1	* (inválida)

- S e R estão em repouso em nível BAIXO; e
- S ou R é pulsada em nível ALTO para alterar Q e \overline{Q} .

Latch SR com portas NOR

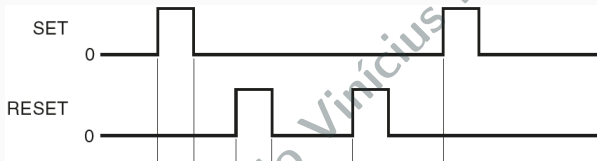
Exemplo: Determine o comportamento da saída Q frente as entradas SET e $RESET$ aplicadas em um *latch* com portas NOR.



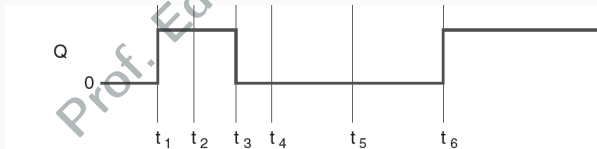
SET e $RESET$ são ativadas em nível ALTO.
kuhn@utfpr.edu.br | youtube.com/@eduardokuhn87

Latch SR com portas NOR

Exemplo: Determine o comportamento da saída Q frente as entradas SET e $RESET$ aplicadas em um *latch* com portas NOR.



R:



SET e $RESET$ são ativadas em nível ALTO.
kuhn@utfpr.edu.br | youtube.com/@eduardokuhn87

Quantos bits um *latch* é capaz armazenar?

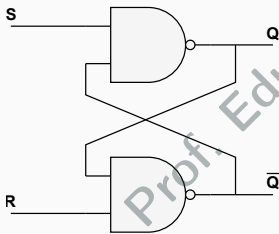
Quantos bits um *latch* é capaz armazenar?

1 bit (unidade elementar de memória)

Latch SR com portas NAND

Com respeito à **topologia e operação**, tem-se

- S e R como entradas (repouso em nível ALTO);
- Q e \overline{Q} como saídas; e
- Q_0 indica o estado prévio/anterior.

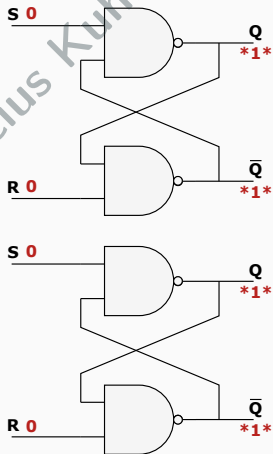


S	R	Q_0	Q	\overline{Q}
0	0	0		
0	0	1		
0	1	0		
0	1	1		
1	0	0		
1	0	1		
1	1	0		
1	1	1		

Latch SR com portas NAND

Caso S e R sejam pulsadas (nível BAIXO):

S	R	Q_0	Q	\bar{Q}
0	0	0	*	*
0	0	1	*	*
0	1	0		
0	1	1		
1	0	0		
1	0	1		
1	1	0		
1	1	1		

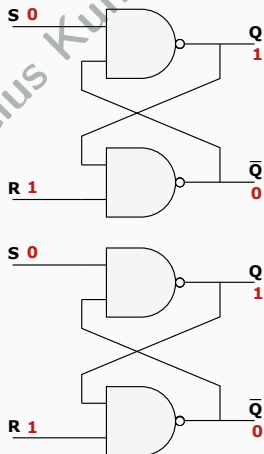


Não é possível determinar exatamente a saída; na prática, o estado resultante da saída dependerá de qual entrada retornou primeiro ao estado de repouso.

Latch SR com portas NAND

Caso S seja pulsada (nível BAIXO):

S	R	Q_0	Q	\bar{Q}
0	0	0	*	*
0	0	1	*	*
0	1	0	1	0
0	1	1	1	0
1	0	0		
1	0	1		
1	1	0		
1	1	1		

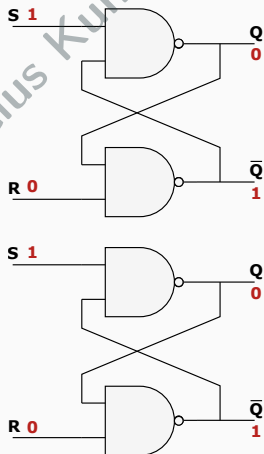


Quando $Q_0 = 0$, pode momentaneamente ocorrer $Q = 1$ e $\bar{Q} = 1$; todavia, as saídas logo se estabilizarão em $Q = 1$ e $\bar{Q} = 0$.

Latch SR com portas NAND

Caso R seja pulsada (nível BAIXO):

S	R	Q_0	Q	\bar{Q}
0	0	0	*	*
0	0	1	*	*
0	1	0	1	0
0	1	1	1	0
1	0	0	0	1
1	0	1	0	1
1	1	0		
1	1	1		

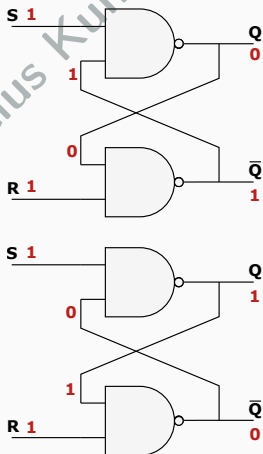


Quando $\bar{Q}_0 = 0$, pode momentaneamente ocorrer $Q = 1$ e $\bar{Q} = 1$;
todavia, as saídas logo se estabilizam em $Q = 0$ e $\bar{Q} = 1$.

Latch SR com portas NAND

Caso S e R estejam em repouso (nível ALTO):

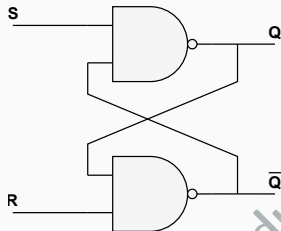
S	R	Q_0	Q	\bar{Q}
0	0	0	*	*
0	0	1	*	*
0	1	0	1	0
0	1	1	1	0
1	0	0	0	1
1	0	1	0	1
1	1	0	0	1
1	1	1	1	0



O estado atual Q da saída depende do estado prévio Q_0 .

Latch SR com portas NAND

Portanto, a operação do *latch* SR com portas NAND pode ser resumida através da seguinte tabela-verdade:

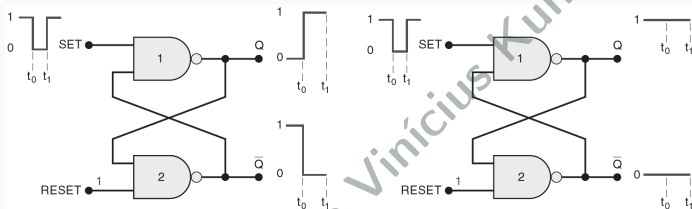


S	R	Q	
0	0	*	(inválida)
0	1	1	(set)
1	0	0	(reset)
1	1	Q_0	(mantém)

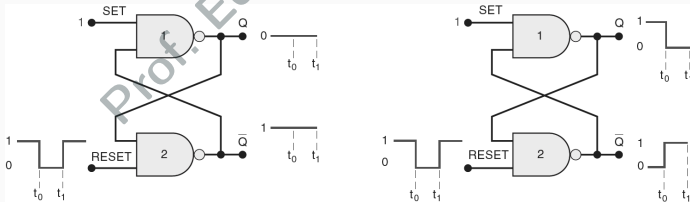
- S e R estão em repouso em nível ALTO; e
- S ou R é pulsada em nível BAIXO para alterar Q e \overline{Q} .

Latch SR com portas NAND

“Setando” o latch:

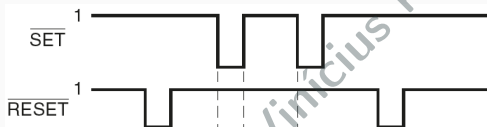


“Resetando” o latch:



Latch SR com portas NAND

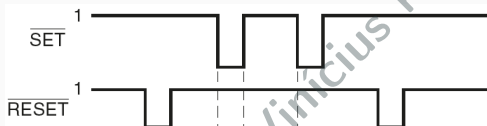
Exemplo: Considerando $Q_0 = 0$, determine a forma de onda na saída Q para as formas de onda aplicadas nas entradas do *latch* SR.



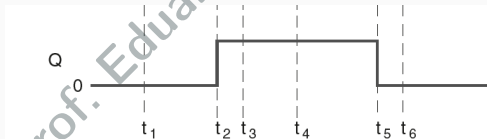
O uso de \overline{SET} e \overline{RESET} indica ativação em nível BAIXO.

Latch SR com portas NAND

Exemplo: Considerando $Q_0 = 0$, determine a forma de onda na saída Q para as formas de onda aplicadas nas entradas do *latch* SR.



R:



O uso de $\overline{\text{SET}}$ e $\overline{\text{RESET}}$ indica ativação em nível BAIXO.

Exemplo: Com respeito ao funcionamento do *latch* SR, responda:

- a) Qual é o estado de repouso das entradas \overline{S} e \overline{R} ? E, qual é o estado ativo?
- b) Quais serão os estados de Q e \overline{Q} após *RESET*?
- c) A afirmação “a entrada \overline{S} nunca pode ser usada para gerar $Q = 0$ ” é verdadeira ou falsa?
- d) O que poderia ser feito para garantir que um *latch* SR com portas NAND sempre comece no estado em que $Q = 1$?

Exemplo: Com respeito ao funcionamento do *latch* SR, responda:

- a) Qual é o estado de repouso das entradas \overline{S} e \overline{R} ? E, qual é o estado ativo?

R: ALTO; BAIXO.

- b) Quais serão os estados de Q e \overline{Q} após *RESET*?

R: $Q = 0$ e $\overline{Q} = 1$.

- c) A afirmação “a entrada \overline{S} nunca pode ser usada para gerar $Q = 0$ ” é verdadeira ou falsa?

R: Verdadeira.

- d) O que poderia ser feito para garantir que um *latch* SR com portas NAND sempre comece no estado em que $Q = 1$?

R: Aplicar um nível BAIXO momentaneamente em \overline{S} .

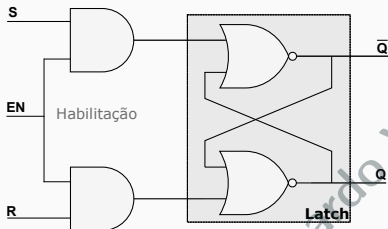
O que ocorre se as entradas S ou R de um *latch* exibirem oscilações momentâneas?

O que ocorre se as entradas S ou R de um *latch* exibirem oscilações momentâneas?

As saídas Q e \overline{Q} podem sofrer alterações indesejadas...

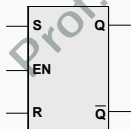
Latch SR sensível ao nível

A **entrada de habilitação EN** precisa ser ativada para que ocorra alguma mudança de estado.



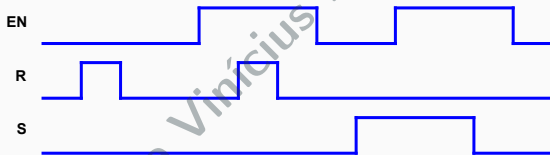
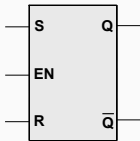
EN	S	R	Q
0	*	*	Q_0
1	0	1	0
1	1	0	1
1	1	1	*!*

Representação compacta:



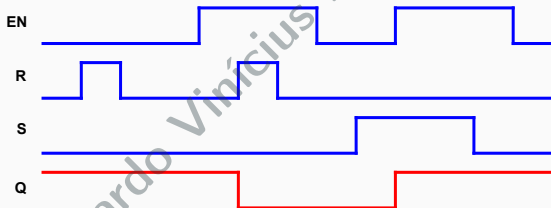
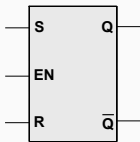
Latch SR sensível ao nível

Exemplo: Considerando que a operação agora envolve a entrada EN , determine a saída Q assumindo $Q_0 = 1$.



Latch SR sensível ao nível

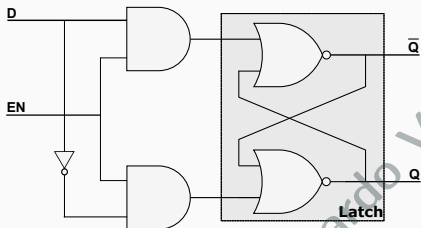
Exemplo: Considerando que a operação agora envolve a entrada EN , determine a saída Q assumindo $Q_0 = 1$.



E, como resolver o problema da condição inválida ($S = R = 1$) em um *latch* SR?

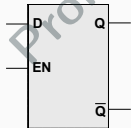
Latch D sensível ao nível (ou *latch* transparente)

O *latch* D é **construído** conectando as entradas *S* e *R* por uma **porta inversora**.



EN	D	Q
0	*	Q_0
1	0	0
1	1	1

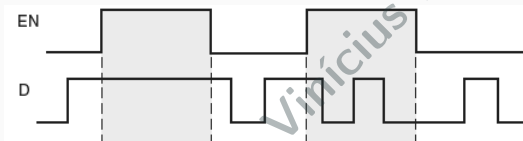
Representação compacta:



EN	D	Q
0	*	Q_0
1	D	D

Latch D sensível ao nível (ou *latch* transparente)

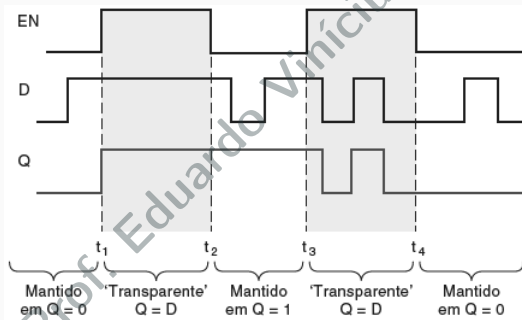
Exemplo: Determine a forma de onda da saída Q para um *latch* D sensível ao nível para as formas de onda EN e D , com $Q_0 = 0$.



Latch D sensível ao nível (ou *latch* transparente)

Exemplo: Determine a forma de onda da saída Q para um *latch* D sensível ao nível para as formas de onda EN e D , com $Q_0 = 0$.

R:



O termo "*latch* transparente" está relacionado ao fato de que $Q = D$ quando

$EN = 1$, i.e., a saída reproduz a entrada para $EN = 1$.

O que ocorre se D oscilar enquanto $EN = 1$?

O que ocorre se D oscilar enquanto $EN = 1$?

O valor armazenado Q pode ser diferente;
idealmente, o pulso EN deve ser feito tão
estreito quanto possível.

Considerações sobre sinais de *clock*

- **Em sistemas assíncronos:**

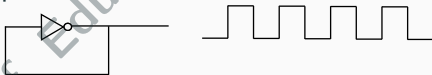
- as saídas podem mudar de estado a qualquer momento; e
- por isso, **o projeto e a análise de defeitos tornam-se complexas.**

- **Em sistemas síncronos:**

- **um sinal de clock é distribuído por todo o sistema;**
- as saídas podem mudar de estado apenas quando ocorrem transições de *clock*; e
- dessa forma, **o projeto e a análise de defeitos tornam-se mais fáceis já que os eventos são sincronizados.**

Considerações sobre sinais de *clock*

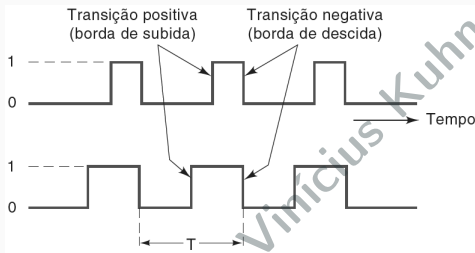
- Funções do *clock* em um circuito digital:
 - **Coordenar as operações/ações do circuito.**
 - Permitir sequenciamento de operações.
 - Criar, em conjunto com *latches* e *flip-flops*, barreiras de tempo.
- Usualmente, um oscilador é usado para gerar um sinal de *clock*; **de forma simplificada**, o oscilador pode ser representado por



- Na prática, **osciladores de quartzo** são utilizados:



Considerações sobre sinais de *clock*

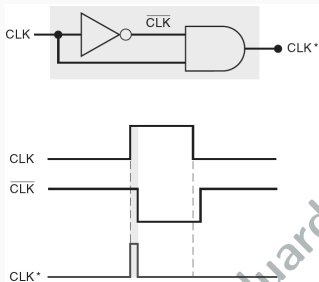


As **transições podem ocorrer** quando o pulso muda

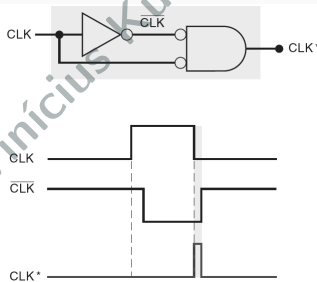
- de 0 para 1 \Rightarrow transição positiva (**borda de subida**);
- de 1 para 0 \Rightarrow transição negativa (**borda de descida**).

O tempo para completar um ciclo é chamado de período T ; e, a velocidade de operação do sistema depende da frequência com que ocorrem os ciclos de *clock*.

Transição positiva

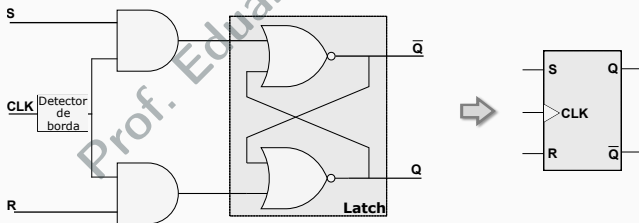


Transição negativa

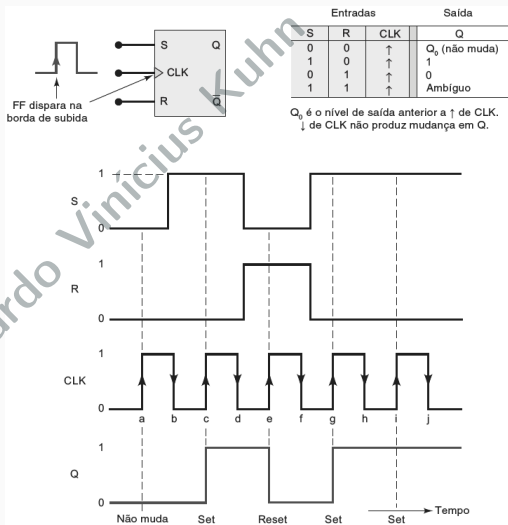


- Essa abordagem tira proveito do atraso de propagação dos elementos envolvidos...
- Quando $CLK^* = 1$, atualiza-se Q baseado em S e R .

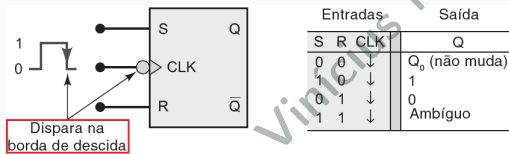
- A sincronização dos eventos é obtida usando *flip-flops* “com clock”, que mudam de estado apenas em transições do clock.
- O detector de borda produz um pulso estreito e positivo no instante da transição ativa do pulso em *CLK*.
- O objetivo é criar melhores barreiras temporais, em comparação ao *latch* sensível ao nível.



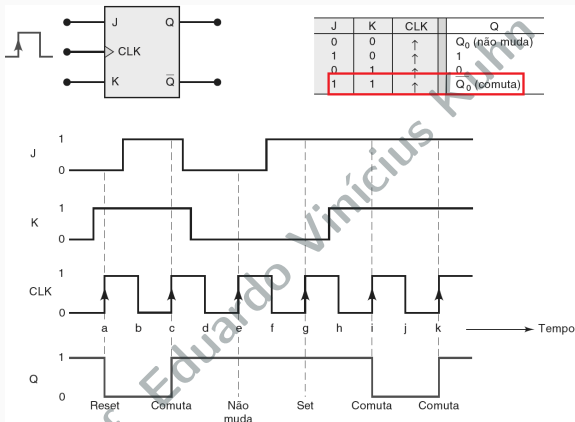
- As entradas S e R controlam o estado do FF.
- Têm efeito apenas na borda de subida do clock.
- Quando $S = R = 1$, tem-se uma condição ambígua.
- A saída Q muda apenas nas bordas de subida do CLK.



Disparo (*trigger*) na borda de descida:



- Observe o pequeno círculo na entrada CLK indicando disparo na borda de descida.
- Tanto os FFs disparados por borda de subida quanto os por borda de descida são usados em sistemas digitais.



- Note a diferença, em relação ao FF SR, quando $J = K = 1$.
- Nessa condição, Q muda para o estado oposto.
- **Modo de comutação (toggle mode)**

Sobre o funcionamento de FFs

Exemplo: Com respeito ao funcionamento de FFs, responda:

- a) Quais são os dois tipos de entradas de um FF?
- b) Qual é o significado do termo disparado por borda?
- c) Em um FF JK, o que ocorre quando $J = K = 1$?
- d) Quais condições para J e K fazem $Q = 1$ na transição ativa de CLK?

Sobre o funcionamento de FFs

Exemplo: Com respeito ao funcionamento de FFs, responda:

a) Quais são os dois tipos de entradas de um FF?

R: Entradas de controle e de *clock*.

b) Qual é o significado do termo disparado por borda?

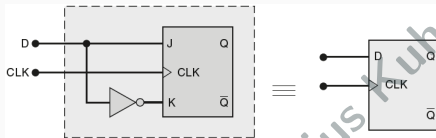
R: A saída muda quando ocorrer uma transição de *clock*.

c) Em um FF JK, o que ocorre quando $J = K = 1$?

R: A saída Q comuta para o estado oposto, i.e., $Q = \overline{Q_0}$.

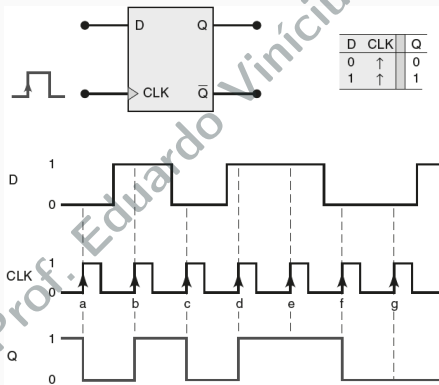
d) Quais condições para J e K fazem $Q = 1$ na transição ativa de CLK?

R: $J = 1$ e $K = 0$.



- Ao contrário dos FFs SR e JK, o **FF D** tem apenas uma entrada de controle D (dado).
- Um **FF D** pode ser construído conectando as entradas de um **FF JK** através de uma inversora.
- O mesmo procedimento pode ser usado para converter um **FF SR** em um **FF D**.

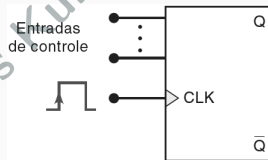
- No FF D, Q assumirá o mesmo estado lógico presente em D quando ocorrer uma borda de subida em CLK.



Entradas síncronas *versus* assíncronas

Entradas síncronas:

- Entradas de controle (e.g., S e R , J e K , T ou D).
- Efeito na saída sincronizado com a entrada CLK.
- Disparo na borda de subida/descida do sinal de clock.

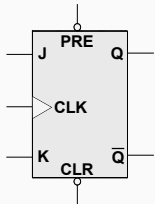


Entradas assíncronas:

- **Entradas que operam independentemente do clock.**
- Geralmente, $\overline{\text{PRESET}}$ e $\overline{\text{CLEAR}}$.
- **Sobrepõem as outras entradas e podem fixar o estado de saída do FF.**

Entradas assíncronas

Entradas assíncronas têm **efeito na saída independente do CLK**, i.e.,

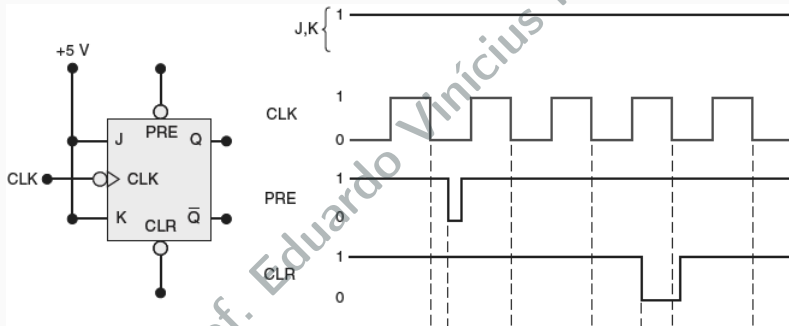


J	K	CLK	PRE	CLR	Q	
0	0	\uparrow	1	1	Q_0	(mantém)
0	1	\uparrow	1	1	0	(reset)
1	0	\uparrow	1	1	1	(set)
1	1	\uparrow	1	1	$\overline{Q_0}$	(comuta)
*	*	*	1	0	0	(CLEAR)
*	*	*	0	1	1	(PRESET)
*	*	*	0	0	*	(inválido)

Note que PRE e CLR são ativadas em nível BAIXO.

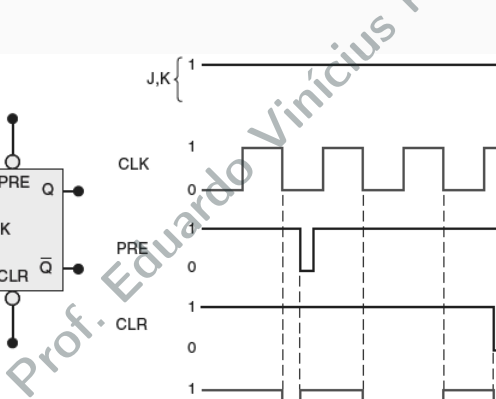
Entradas assíncronas

Exemplo: Determine a forma de onda Q levando em conta todas as entradas do FF JK.



Universidade Tecnológica Federal do Paraná

Determine a forma de onda de Q levando em conta o FF JK.



Quando $J = K = 1$, tem-se um FF T (toggle)!

Exemplo: Com respeito ao funcionamento de um FF, responda:

- a) Qual é a diferença entre a operação de uma entrada síncrona e a de uma entrada assíncrona?
- b) Um FF D pode responder à entrada D se $PRE = 1$?
- c) Relacione as condições necessárias para que um FF JK com entradas assíncronas ativas em nível BAIXO comute para o estado oposto.

Exemplo: Com respeito ao funcionamento de um FF, responda:

- a) Qual é a diferença entre a operação de uma entrada síncrona e a de uma entrada assíncrona?

R: Entradas assíncronas operam independentemente do *CLK*.

- b) Um FF D pode responder à entrada *D* se $PRE = 1$?

R: Sim, visto que *PRE* é ativo em nível BAIXO.

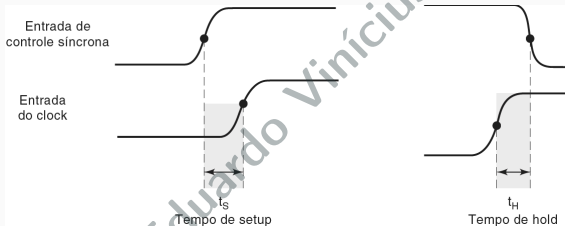
- c) Relacione as condições necessárias para que um FF JK com entradas assíncronas ativas em nível BAIXO comute para o estado oposto.

R: $J = K = 1$ e $PRE = CLR = 1$.

A temporização em FFs é importante sobretudo em situações que as entradas de controle mudam de estado “quase” ao mesmo tempo que CLK.

Temporização em FFs

Os **tempos de setup (preparação)** e de **hold (manutenção)** indicam a duração em que as entradas de controle devem ser mantidas em nível adequado.



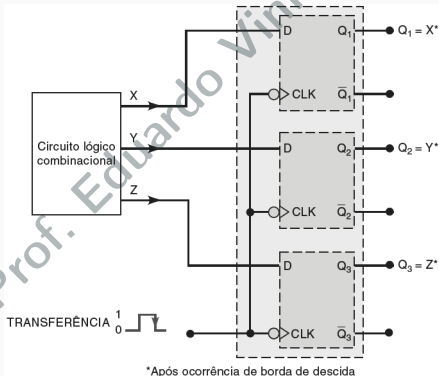
- Tempo de **setup** t_S : de 5 a 50 ns.
- Tempo de **hold** t_H : de 0 a 10 ns.
- **Importância:** Garantir operação confiável do FF.

Aplicações envolvendo FFs

- **Deteccão de sequências binárias:** Em muitas situações, uma saída é ativada apenas quando as entradas são ativadas em uma determinada sequência (pré-definida).
- **Armazenamento de dados:** Grupos de FFs formam os denominados **registradores** usados no armazenamento de dados.
- **Transferência de dados:** FFs são utilizados na transferência serial/paralela de dados.
- **Divisão de frequência e contagem:** Muitas aplicações requerem um divisor de frequência (e.g., relógio, μC) ou um contador (e.g., usando FF T)

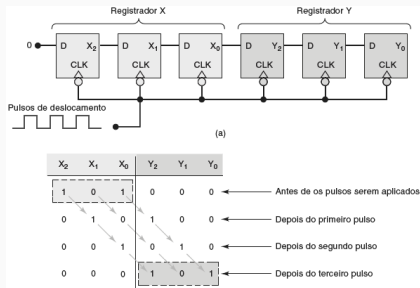
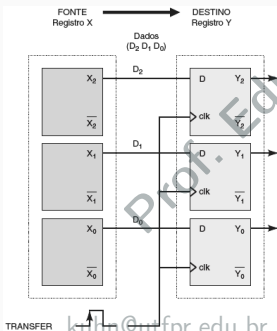
Armazenamento de dados usando FF D

- As saídas X , Y e Z servem como entradas de FFs D.
- Os valores X , Y e Z são armazenados (simultaneamente) em Q_1 , Q_2 e Q_3 , respectivamente, quando ocorre o pulso de TRANSFERÊNCIA.

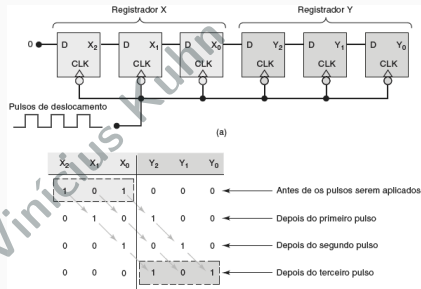
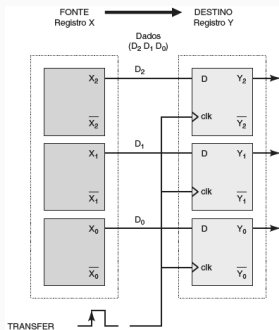


Transferência de dados paralela *versus* serial

- **Paralela:** A transferência do conteúdo de um registrador para outro é síncrona, i.e., **todos os bits são transferidos simultaneamente em um único pulso de clock.**
- **Serial:** O conteúdo de um registrador é transferido para outro, **um bit por vez a cada pulso de clock** (registrador de deslocamento).

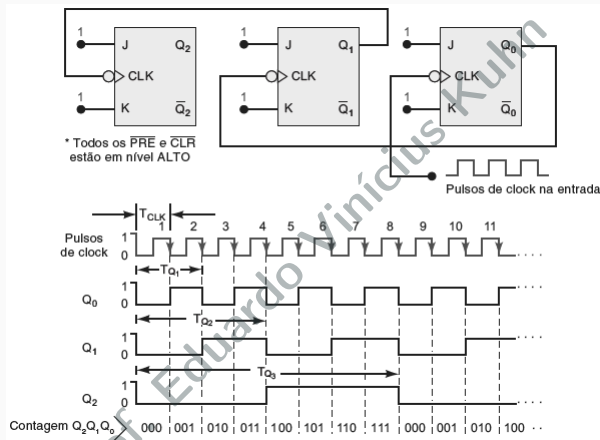


Transferência de dados paralela *versus* serial



- Transferência paralela requer mais conexões entre o registrador transmissor e o receptor do que a transferência serial.
- Pode ser problemático conforme o número de bits aumenta (linhas); sobretudo, quando os registradores estão distantes.
- Depende da aplicação; geralmente, ambas são usadas.

Divisão de frequência e contagem



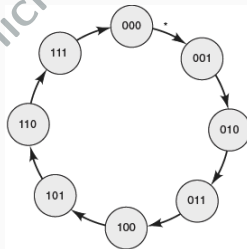
Módulo do contador: Se N FFs são interligados, o contador resultante tem 2^N estados possíveis; logo, módulo 2^N .

Divisão de frequência e contagem

Tabela de estados

2^2 Q_2	2^1 Q_1	2^0 Q_0	
0	0	0	Antes de aplicar os pulsos de clock
0	0	1	Depois do pulso #1
0	1	0	Depois do pulso #2
0	1	1	Depois do pulso #3
1	0	0	Depois do pulso #4
1	0	1	Depois do pulso #5
1	1	0	Depois do pulso #6
1	1	1	Depois do pulso #7
0	0	0	Depois do pulso #8 retorna para 000
0	0	1	Depois do pulso #9
0	1	0	Depois do pulso #10
0	1	1	Depois do pulso #11
.	.	.	.
.	.	.	.
.	.	.	.

Diagrama de transição de estados



Exemplo: Com respeito às aplicações, responda:

- a) A afirmação “a transferência paralela é mais rápida do que a serial” é verdadeira ou falsa?
- b) Qual é a maior vantagem da transferência serial?
- c) Um sinal de clock de 20 kHz é aplicado em um FF JK com $J = K = 1$. Qual é a frequência da forma de onda de saída?
- d) Quantos FFs são necessários para construir um contador de 0_{10} a 255_{10} ?

Exemplo: Com respeito às aplicações, responda:

- a) A afirmação “a transferência paralela é mais rápida do que a serial” é verdadeira ou falsa?

R: Verdadeira.

- b) Qual é a maior vantagem da transferência serial?

R: Menor número de linhas entre registradores.

- c) Um sinal de clock de 20 kHz é aplicado em um FF JK com $J = K = 1$. Qual é a frequência da forma de onda de saída?

R: 10 kHz

- d) Quantos FFs são necessários para construir um contador de 0_{10} a 255_{10} ?

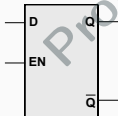
R: 8 FFs

Sugestão de leitura: Seções 5.22-5.24, as
quais tratam de circuitos geradores de clock.

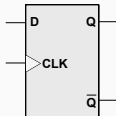
- **Latches e FFs têm característica de memória** (1 bit), i.e., mantêm o estado até que ocorra ação contrária.
- Os diferentes tipos de *latches* (e.g., SR e D) e FFs (e.g., SR, JK, T e D) exibem **características de operação distintas**.
- O **latch é assíncrono** e o **FF é síncrono**; por sua vez, multivibrador biestável é a terminologia técnica para um FF.
 - **Latch**: Utiliza **entrada de habilitação (EN)**; logo, sensível ao nível.
 - **FF**: Utiliza um **detector de borda**; logo, sensível à borda de subida/descida (transição do *clock*).

- Um FF vai para um novo estado em **resposta a um pulso de clock** e de acordo com as entradas de controle.
- **FFs possuem entradas assíncronas que podem “SETAR” ou “LIMPAR” o estado, independentemente do clock.**
- Aplicações incluem armazenamento e transferência de dados, deslocamento de dados, contagem e divisão de frequência.
- **Registradores são construídos agrupando/agregando latches ou FFs D.**

Latch D

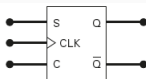


FF D



Resumo

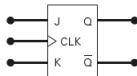
FF S-C com clock



S	C	CLK	Q
0	0	↑	Q_0 (Não muda)
1	0	↑	1
0	1	↑	0
1	1	↑	Ambíguo

↓ Não tem efeito em Q

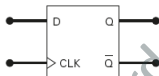
FF J-K com clock



J	K	CLK	Q
0	0	↑	Q_0 (Não muda)
1	0	↑	1
0	1	↑	0
1	1	↑	\bar{Q}_0 (Comuta)

↓ Não tem efeito em Q

FF D com clock



D	CLK	Q
0	↑	0
1	↑	1

↓ Não tem efeito em Q

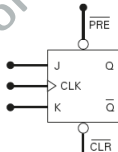
Latch D



EN	D	Q*
0	X	Não muda
1	0	0
1	1	1

*Segue a entrada D enquanto EN está em nível ALTO

Entradas assíncronas



$\overline{\text{PRE}}$	$\overline{\text{CLR}}$	Q*
1	1	Sem efeito; FF pode responder a J, K e CLK
1	0	Q = 0 independente de J, K, CLK
0	1	Q = 1 independente de J, K, CLK
0	0	Ambíguos (não usado)

*CLK pode estar em qualquer estado

Considerações finais

Exercícios sugeridos:

5.3, 5.8, 5.9, 5.11-5.16, 5.18, 5.20, 5.21, 5.26, 5.27, 5.30.

de R.J. Tocci, N.S. Widmer, G.L. Moss, *Sistemas digitais: princípios e aplicações*, 12a ed., São Paulo: Pearson, 2019. → (Capítulo 5)

Para a próxima aula:

R.J. Tocci, N.S. Widmer, G.L. Moss, *Sistemas digitais: princípios e aplicações*, 12a ed., São Paulo: Pearson, 2019. → (Capítulo 6)

Até a próxima aula... =)