

# Microeletrônica

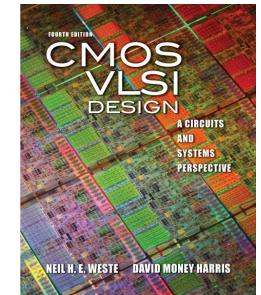
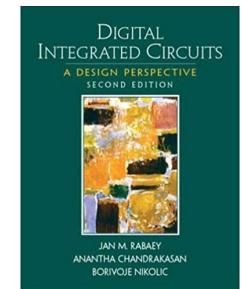
## Aula #7 → Circuitos sequenciais estáticos

□ Professor: Fernando Gehm Moraes

□ Livro texto:

Digital Integrated Circuits a Design Perspective - Rabaey

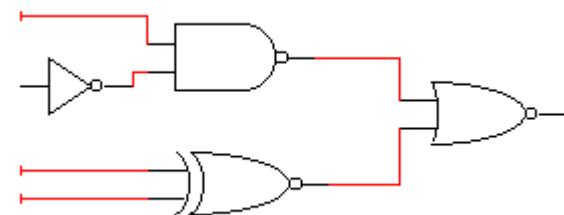
C MOS VLSI Design - Weste



# Lógica Sequencial

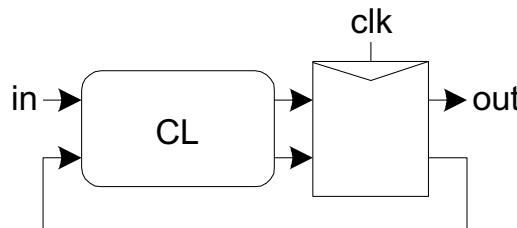
## Lógica Combinacional

- Os sinais de saída são resultados de uma combinação lógica dos sinais de entrada atuais

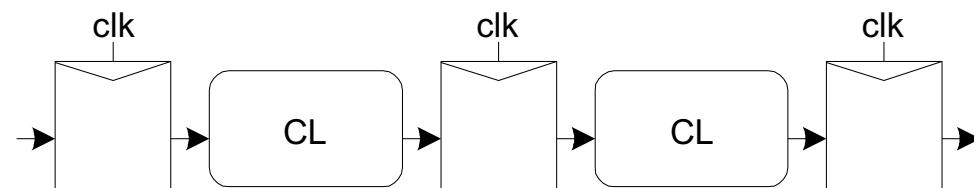


## Lógica Sequencial

- Os sinais de saída são resultados não apenas dos sinais de entrada atuais, mas também de sinais anteriores
- Flip Flops são elementos de memória que registram o estado do circuito

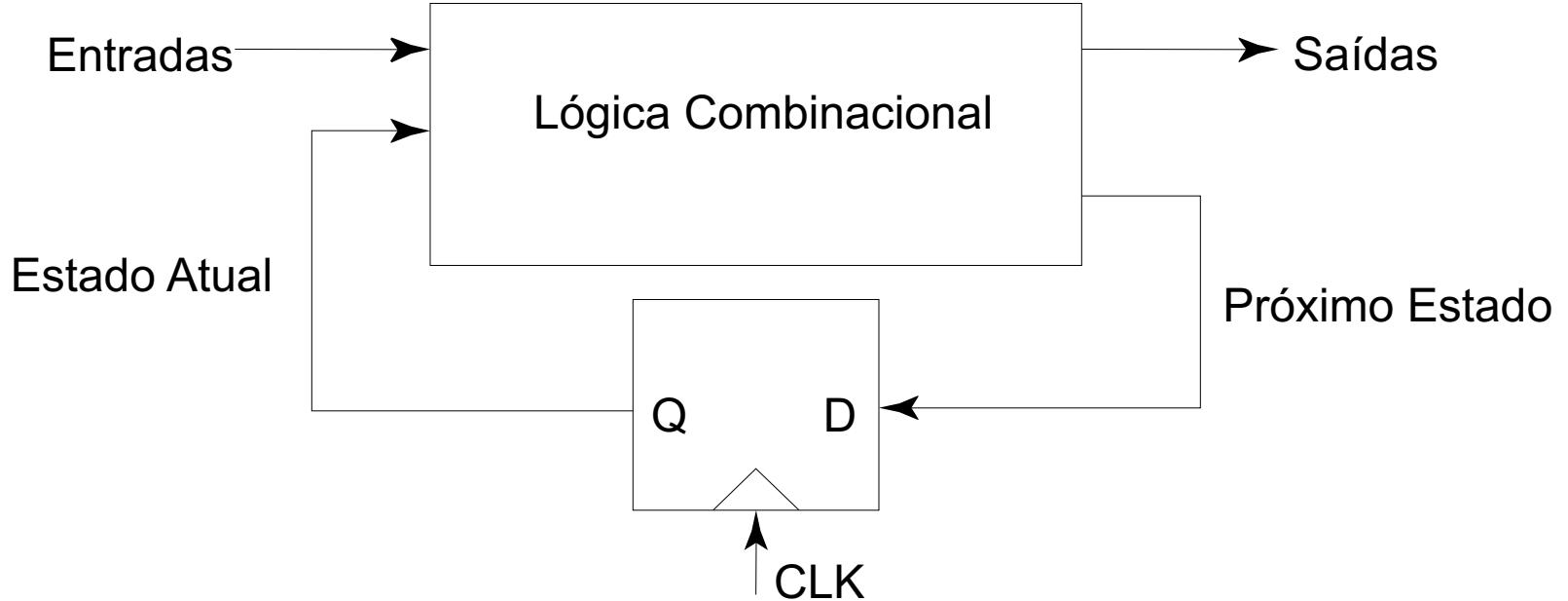


Finite State Machine



Pipeline

# Lógica Sequencial



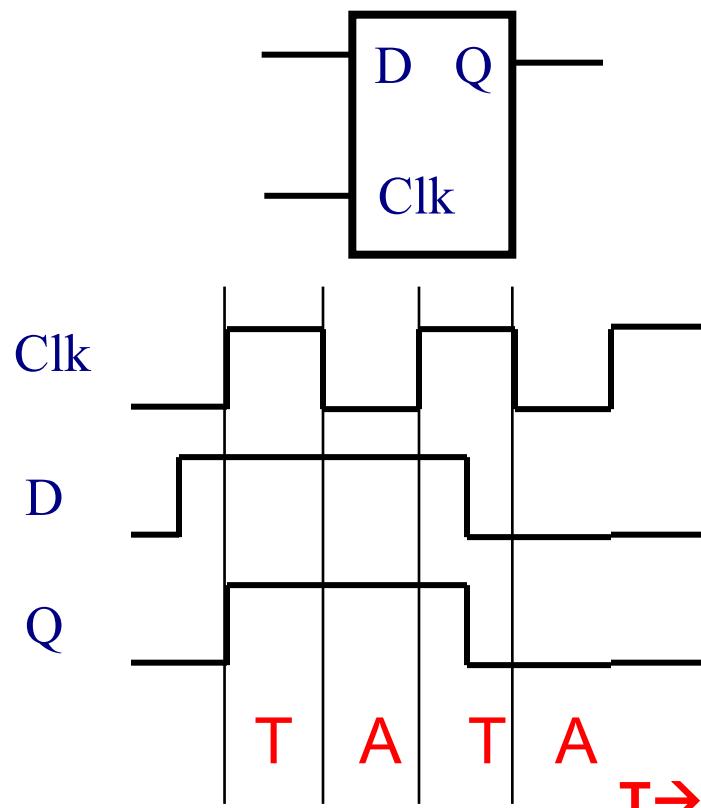
**2 mecanismos de armazenamento**

- positive feedback
- charge-based

# Latch versus Register

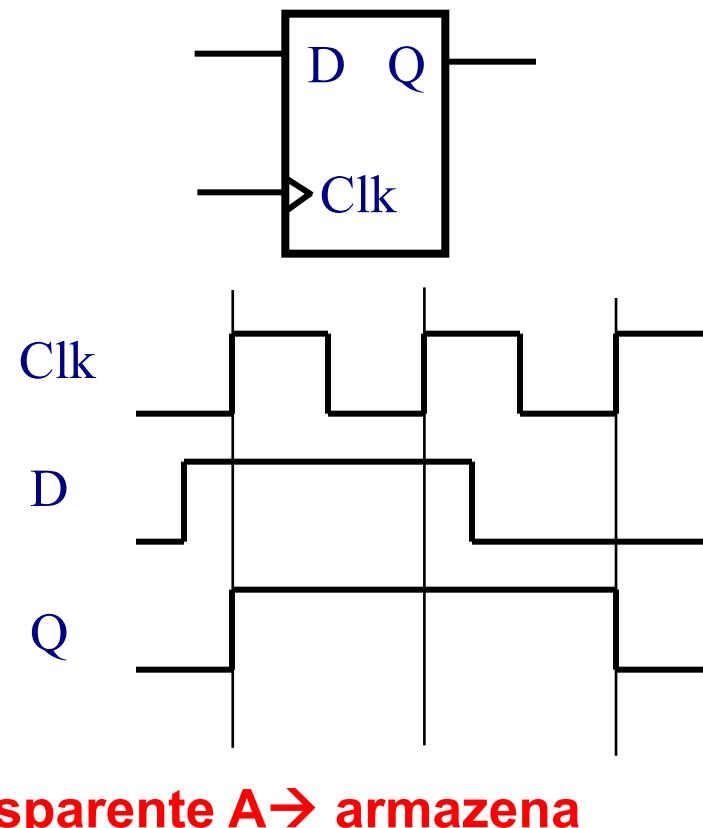
## □ Latch

**transparente** em um nível do clk e depois mantém o dado



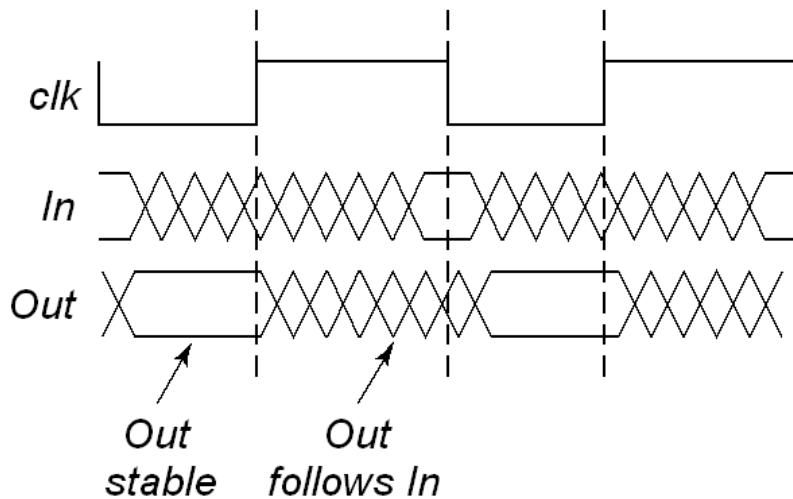
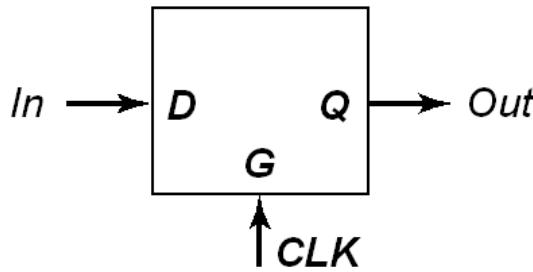
## □ Mestre-Escravo

armazenamento no momento de uma **borda**

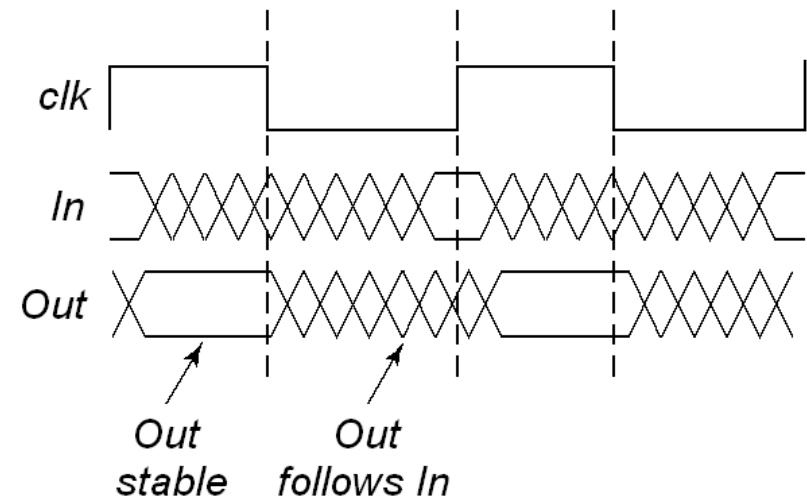
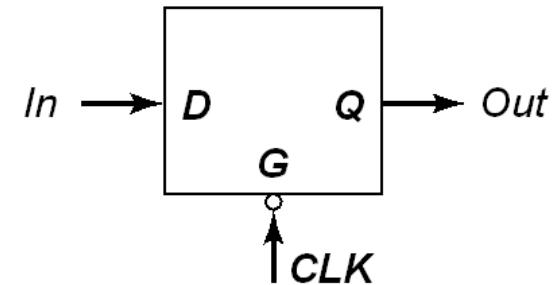


# Latches

*Positive Latch*

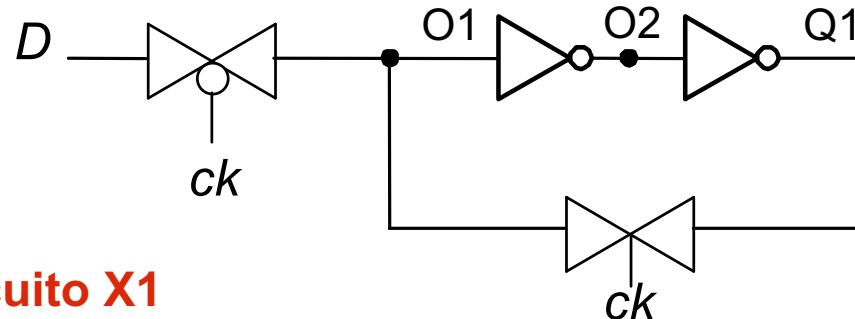


*Negative Latch*



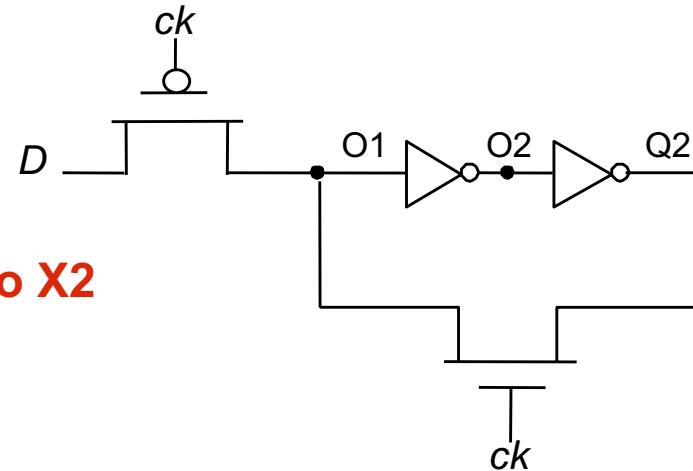
# Latches - implementações

Com portas de transmissão



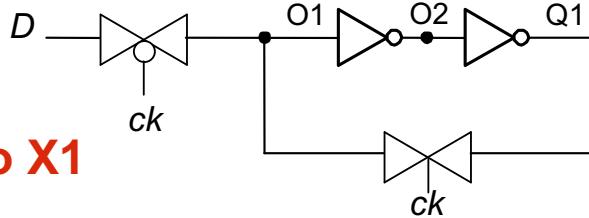
Círcuito X1

Com transistores (N e P)

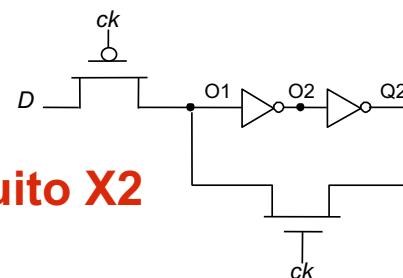


Círcuito X2

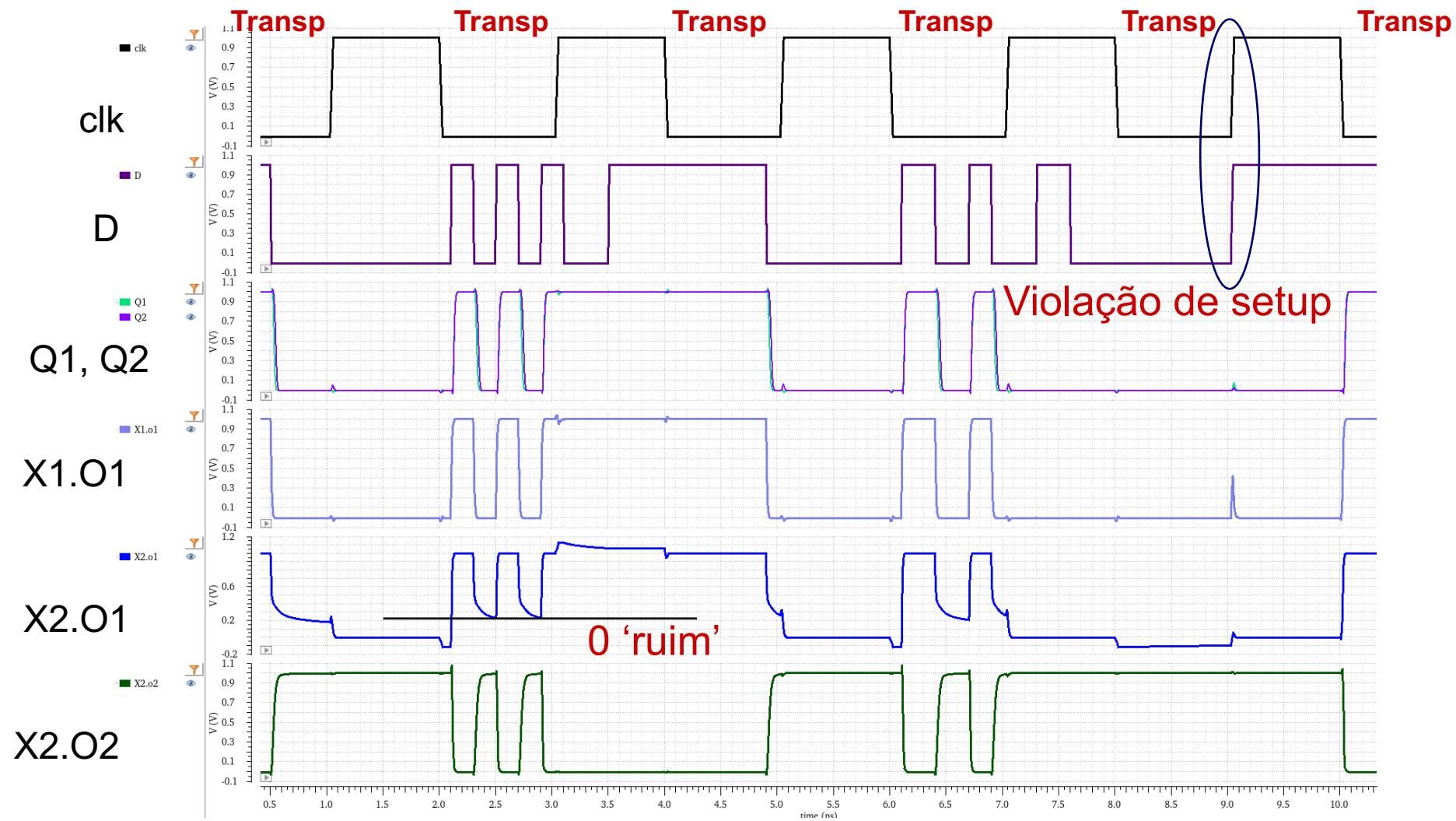
**Tempo de setup:** duração necessária para que os dados de entrada devam estar estáveis antes da borda do *clock*. Se os dados mudarem dentro dessa janela de tempo de configuração, os dados de entrada podem ser perdidos e não armazenados na latch/MS.

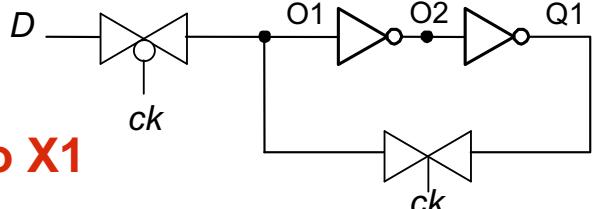


**Círcuito X1**

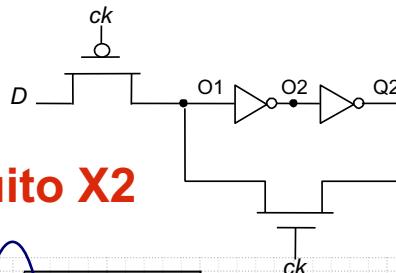


**Círcuito X2**

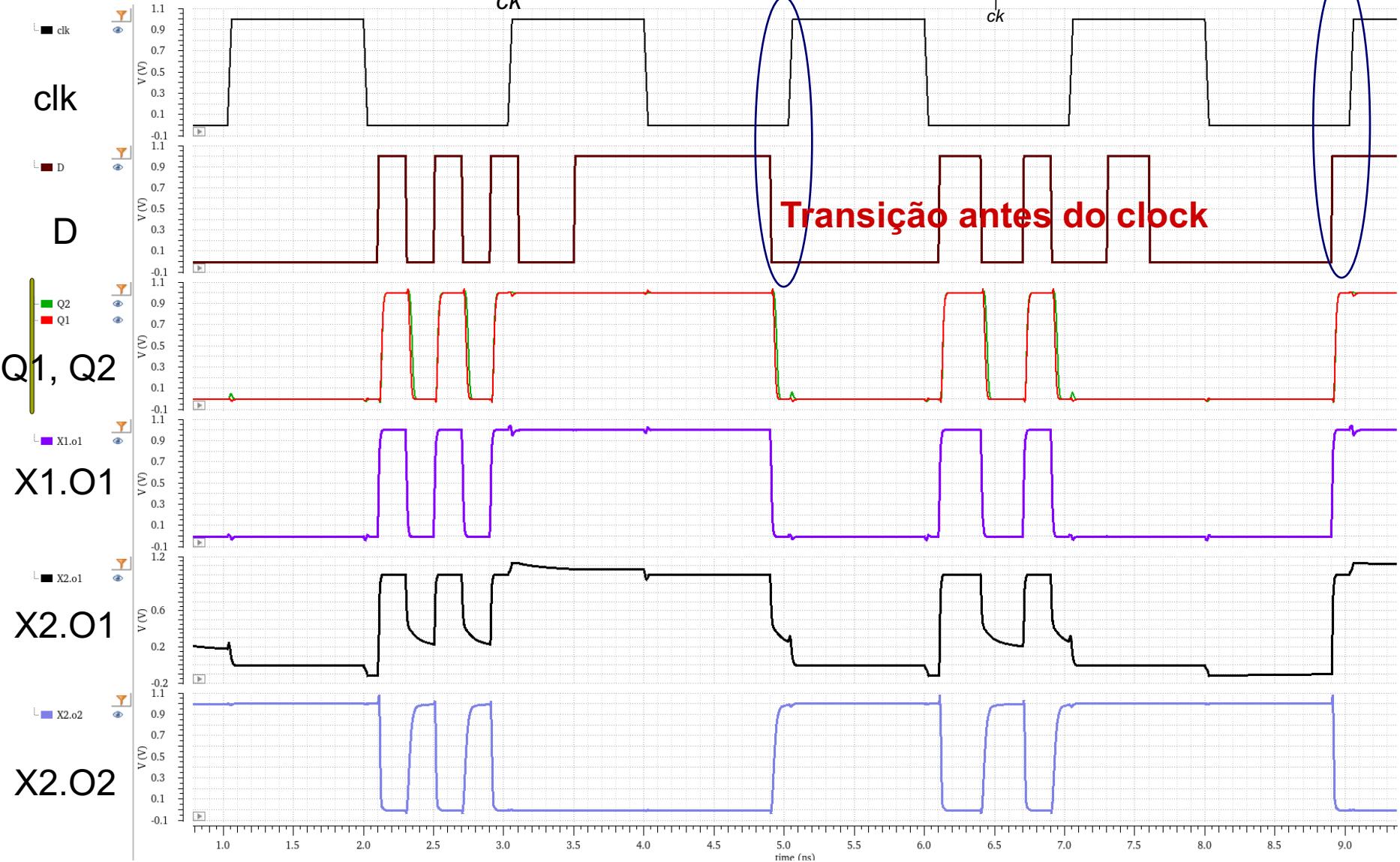




**Círcuito X1**

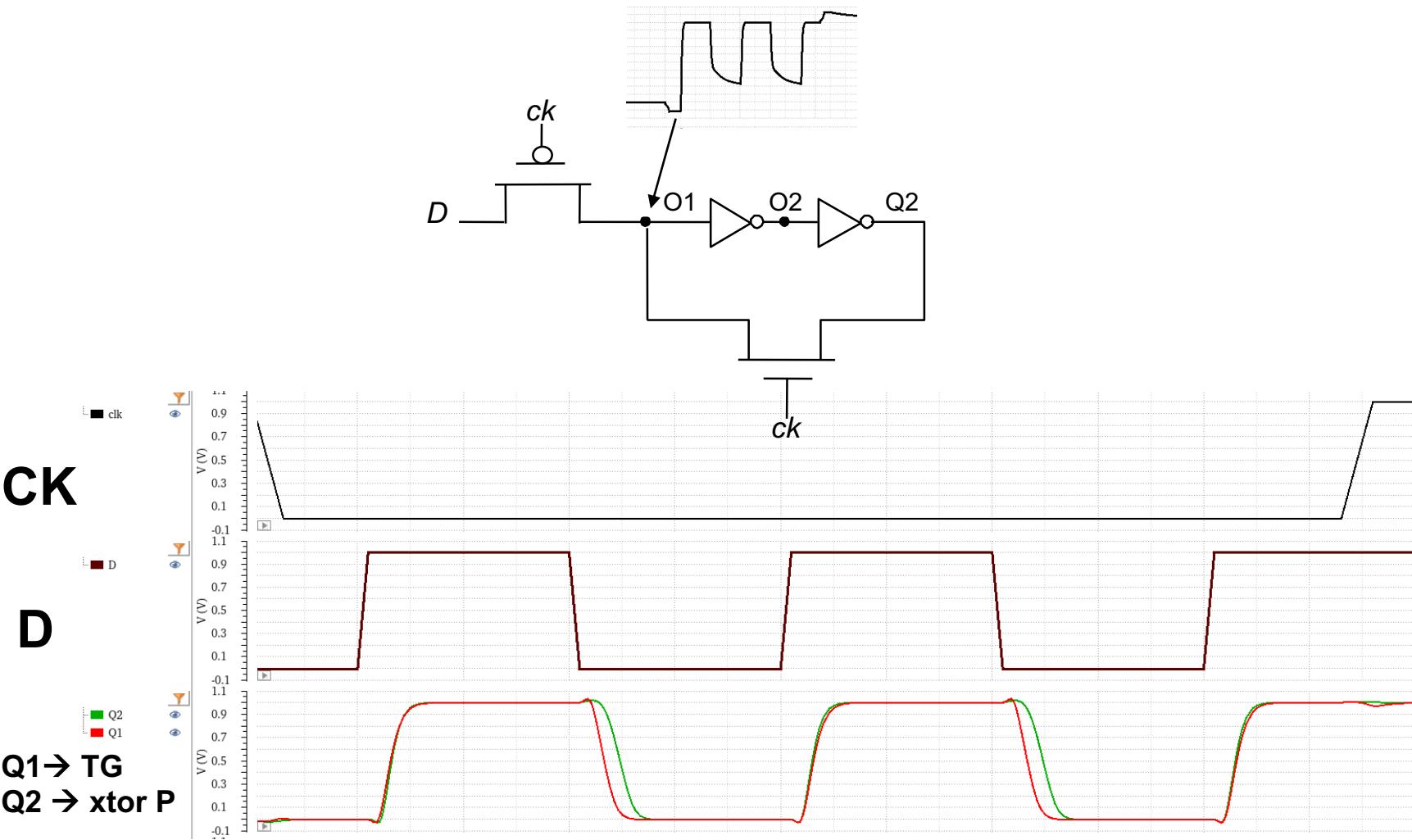


**Círcuito X2**



# Saída Q2: depende de um transistor P em modo transparente

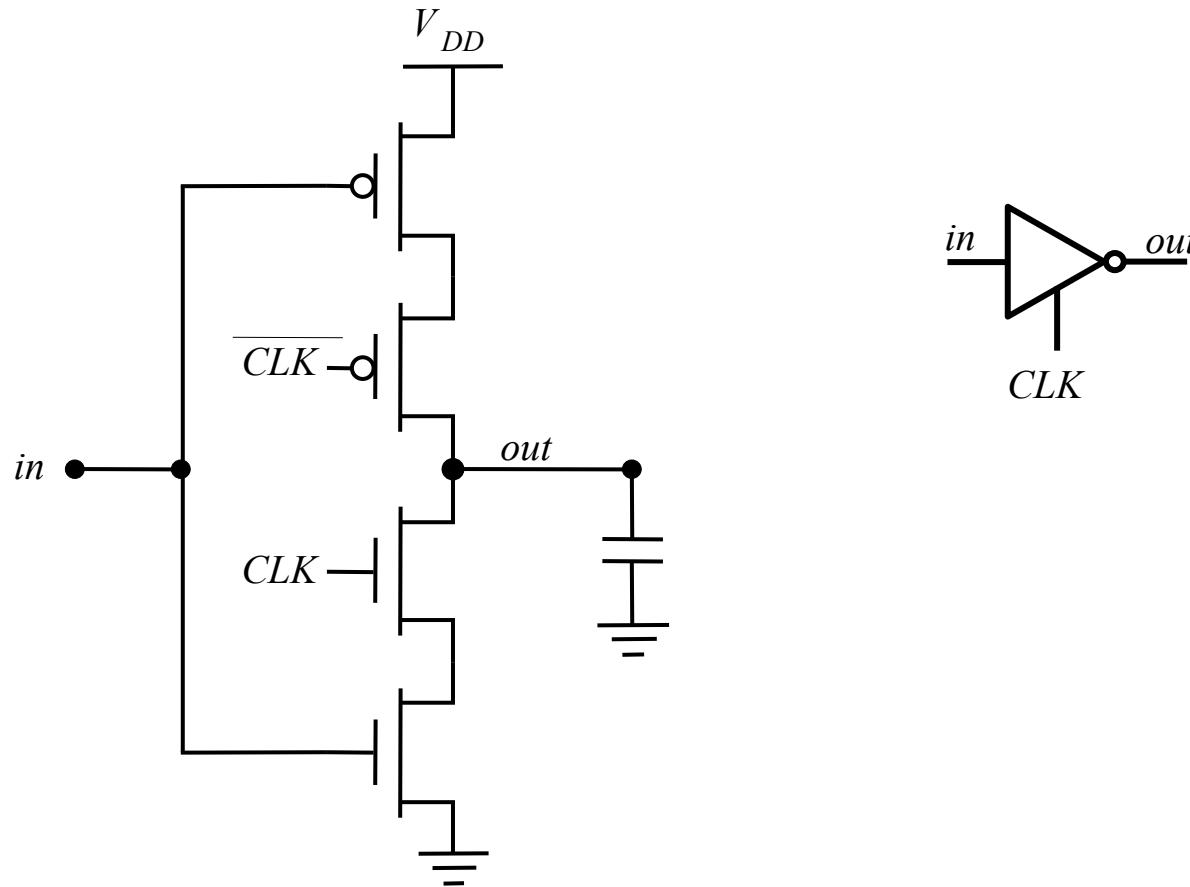
Mais lento no tempo de descida (0 ruim em 01, 02 lento para subir, Q2 desce mais lentamente)



# Tri-state buffer

Integra o TG ao inversor

Reduz sensibilidade de propagação de ruído entre a saída e a entrada

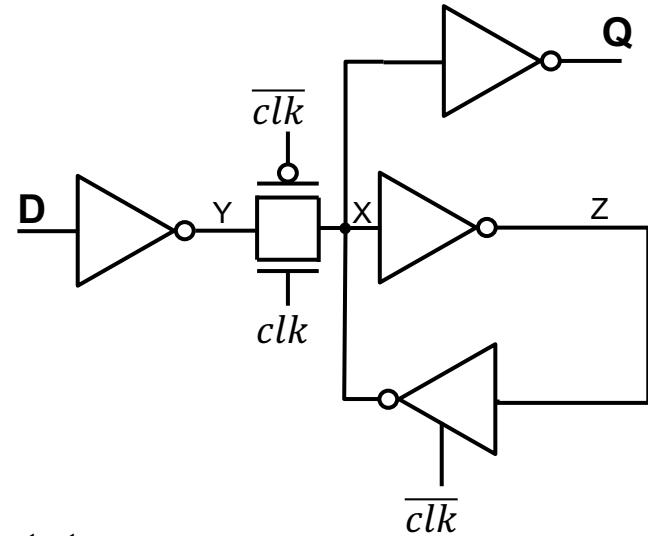


# Latches – com buffer tri-state

Amplamente utilizado em standard-cells

**Robustez:** saída Q é gerada usando um buffer separado, o que elimina problemas com *backdriving* (informação da saída propagada para a entrada)

A saída Q é mantida pelo loop que inclui o buffer tristate



⌚ Os inversores de entrada e saída aumentam a área do circuito

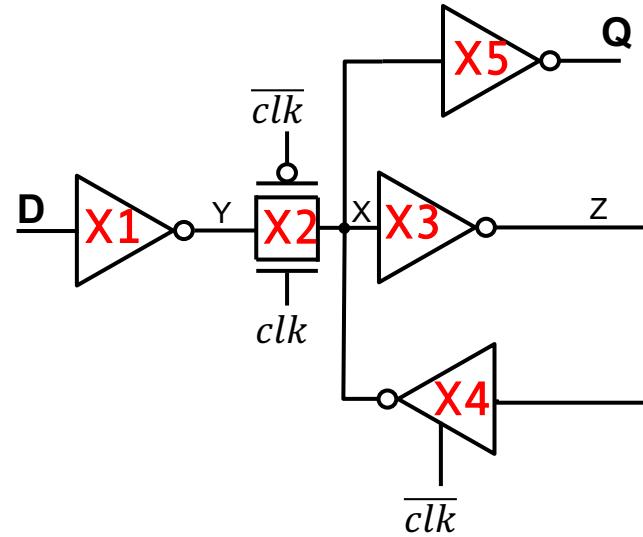
# Latches – com buffer tri-state

```
.subckt inv in out vcc
MP1 out in vcc vcc psvtgp w=wp l=0.06
MM2 out in 0 0 nsvtgp w=wn l=0.06
.ends inv
```

```
.subckt inv_tg in ck nck out vcc
MP1 1 in vcc vcc psvtgp w=wp l=0.06
MP2 out nck 1 vcc psvtgp w=wp l=0.06
MM1 out ck 2 0 nsvtgp w=wn l=0.06
MM2 2 in 0 0 nsvtgp w=wn l=0.06
.ends inv_tg
```

```
.subckt tg a b nb out vcc
MP1 a nb out vcc psvtgp w=wp l=0.06
MN1 a b out 0 nsvtgp w=wn l=0.06
.ends tg
```

```
.subckt latch D Q ck nck vcc
X1 D Y vcc inv
X2 Y ck nck X vcc tg
X3 X Z vcc inv
X4 Z nck ck X vcc inv_tg
X5 X Q vcc inv
.ends latch
```



# Mux-based latches

- A realimentação não precisa ser sobreescrita para gravar na latch
- Maior velocidade, maior área

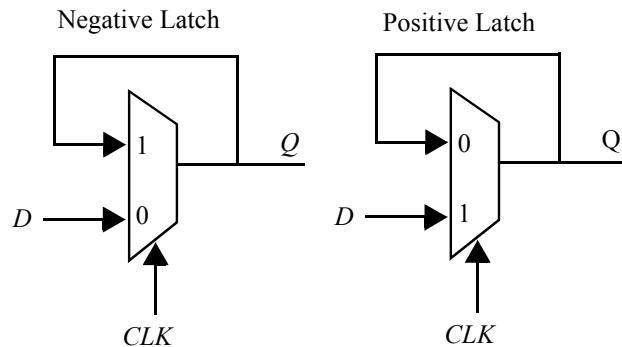


Figure 7.11 Negative and positive latches based on multiplexers.

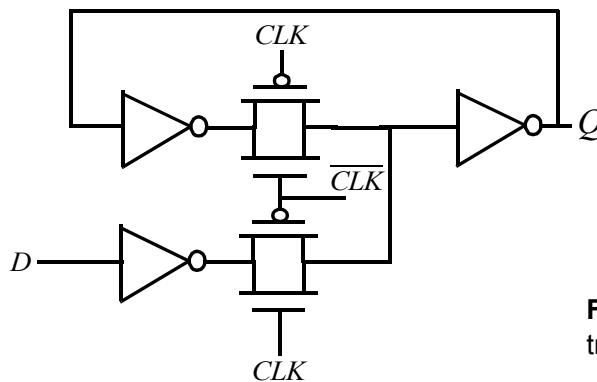
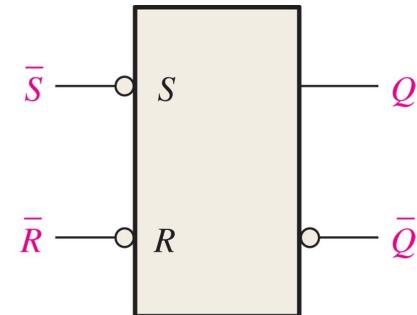
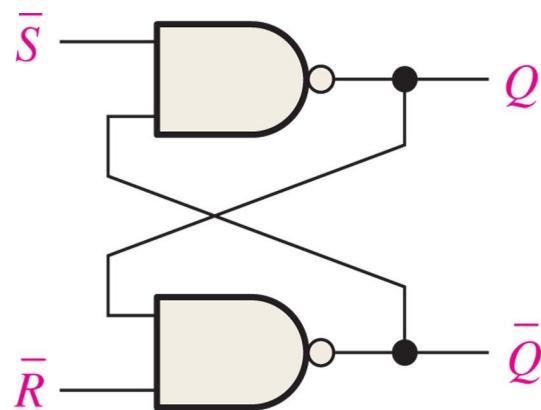
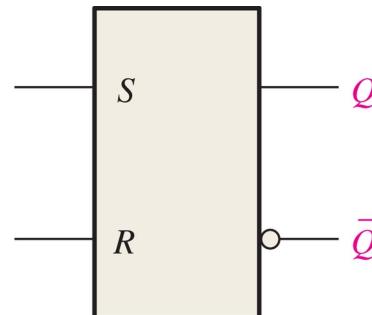
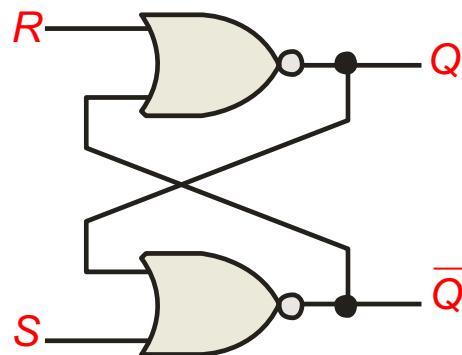


Figure 7.12 Positive latch built using transmission gates.

# Elementos de Memória

## SR - Flip Flop (latch)

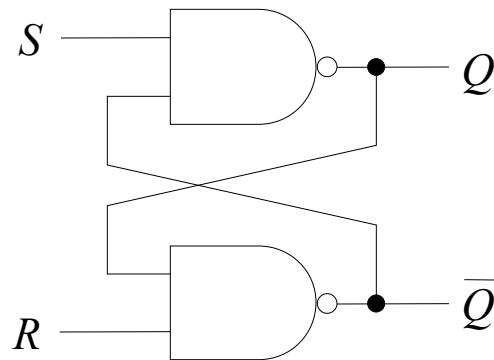


Inputs		Outputs	
$\bar{S}$	$\bar{R}$	$Q$	$\bar{Q}$
1	1	NC	NC
0	1	1	0
1	0	0	1
0	0	1	1

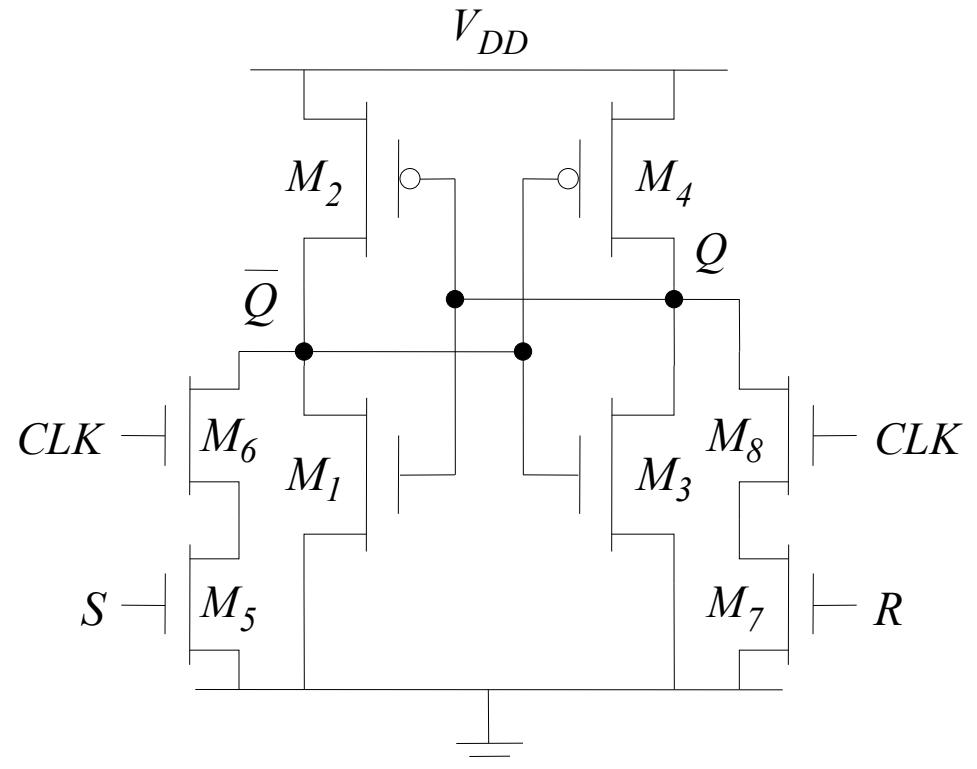
Diagrama Esquemático   Símbolo Lógico

# Cross-Coupled NAND

Cross-coupled NANDs



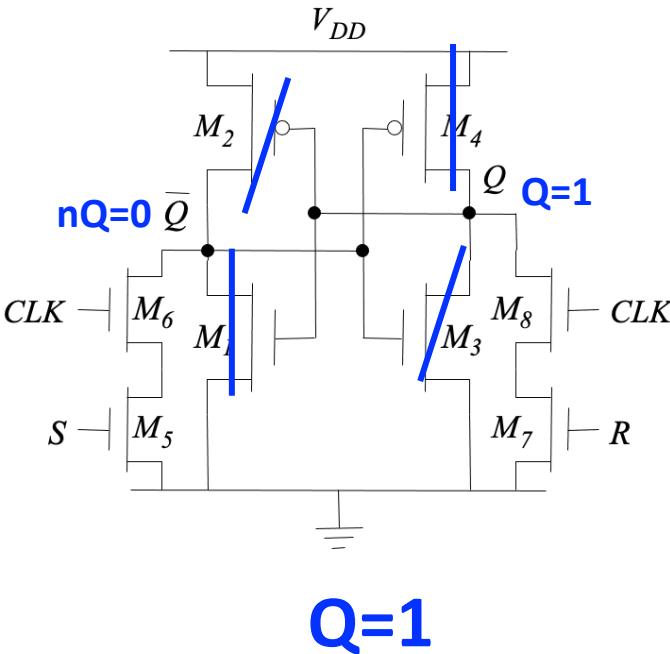
Added clock



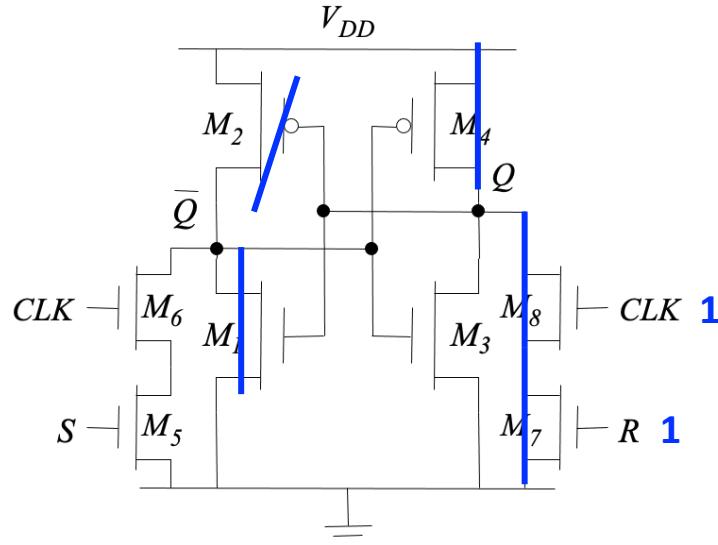
$CLK=0 \rightarrow$  não muda de estado

- M1-M2 inverter e M3-M4 inverter
- logo, 2 inversores realimentados

# Cross-Coupled NAND

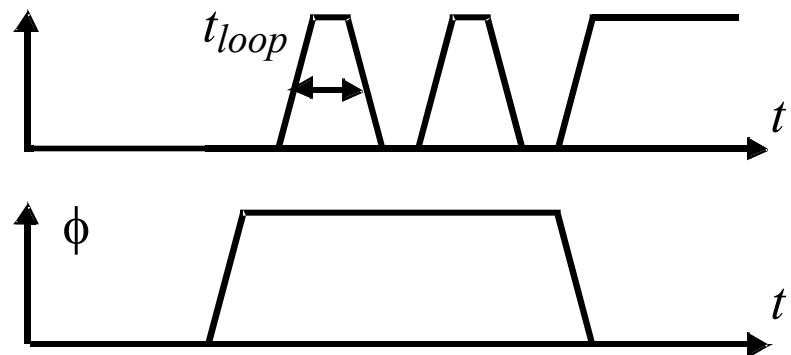
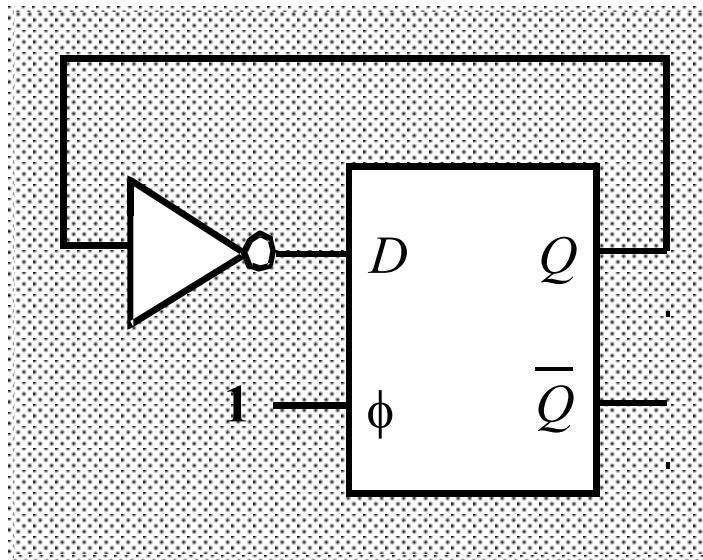


Considere o caso em que Q é alto e um pulso R é aplicado ( $CLK=1$ ):



- $M_4, M_8$  e  $M_7 \rightarrow$  curto (*ratiored inverter*)
- Para o circuito operar: maior ganho (W) em  $M_5, M_6, M_7$  e  $M_8$ .

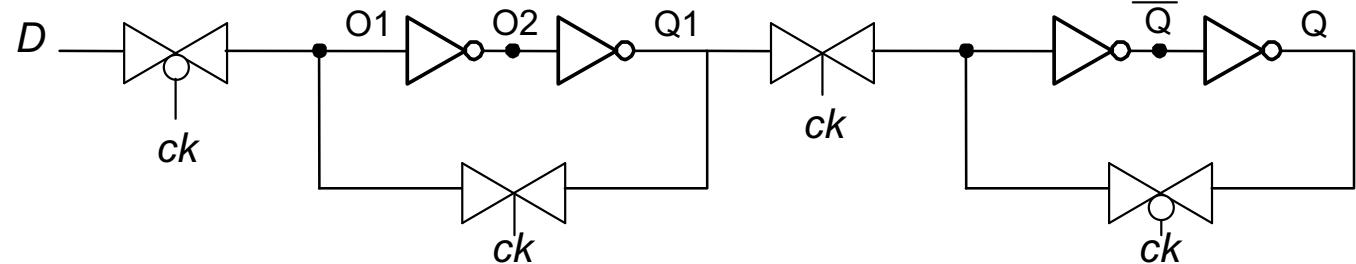
# Problema de Corrida (Race)



- A natureza transparente da *latch* causa o problema de ‘race’
- A *latch* é **transparente** quando  $\phi = 1$
- Uma solução ao problema de *race* são os FF mestre-escravo

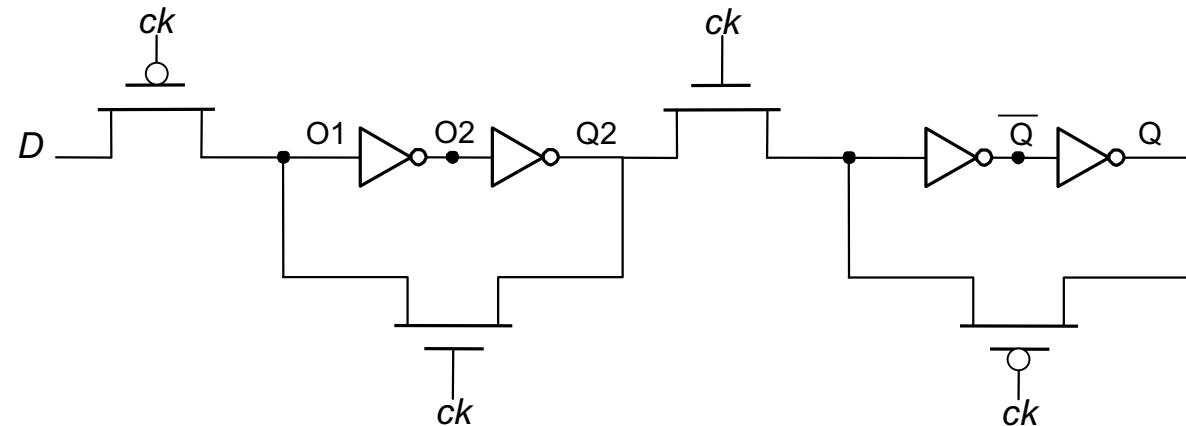
# Flip Flop Mestre Escravo Estático

Com portas de transmissão



Círcuito X1

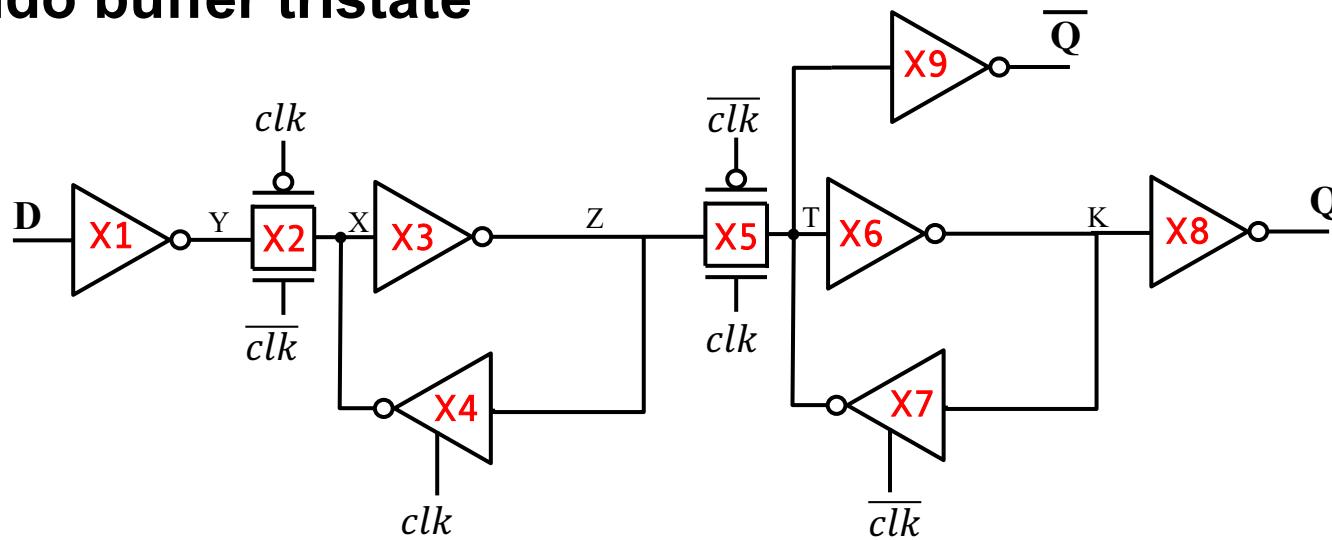
Sem portas de transmissão



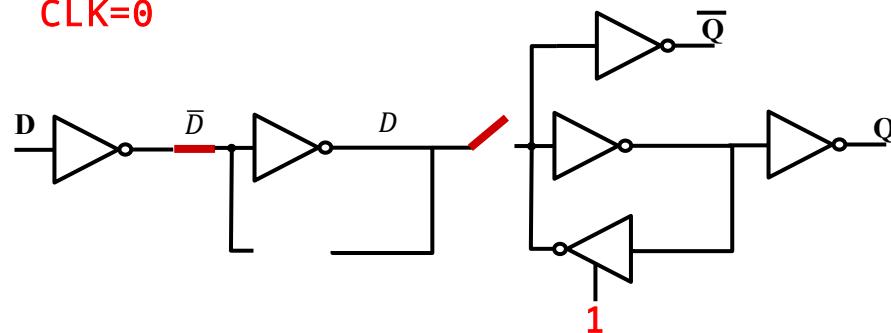
Círcuito X2

# Flip Flop Mestre Escravo Estático

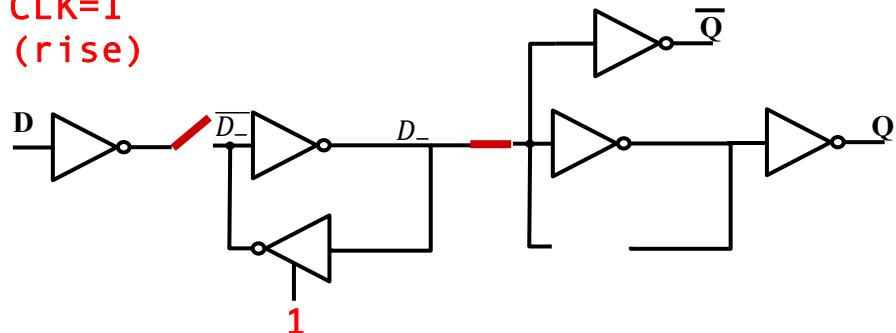
Utilizando buffer tristate



CLK=0



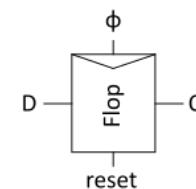
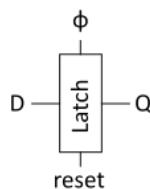
CLK=1  
(rise)



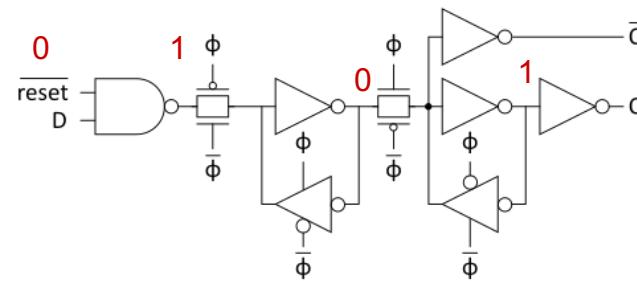
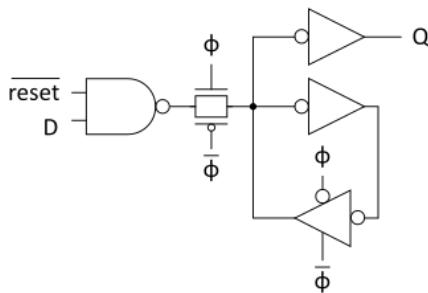
# Flip Flop Mestre Escravo Estático

## Reset

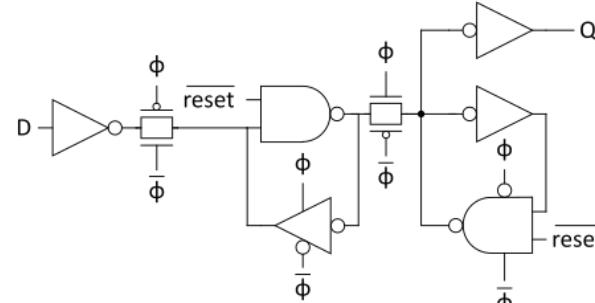
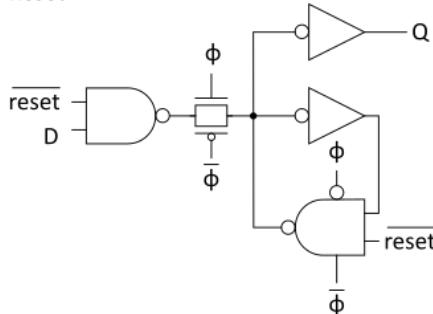
Symbol



Synchronous  
Reset



Asynchronous  
Reset



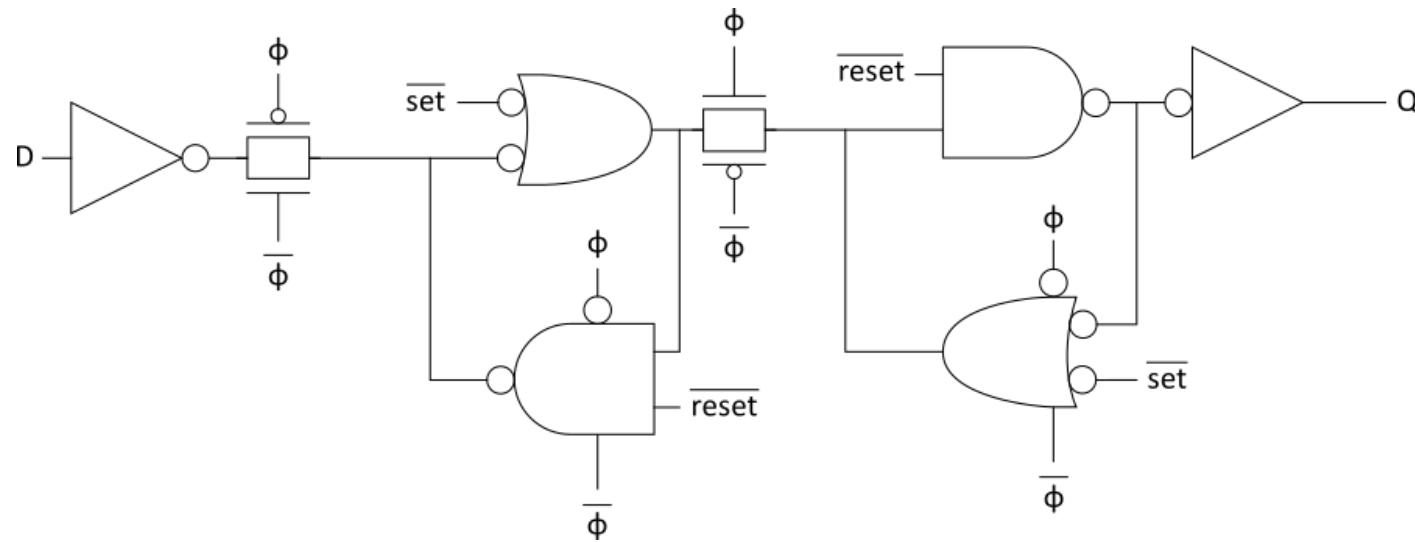
0	0	1
0	1	1
1	0	1
1	1	0

NAND

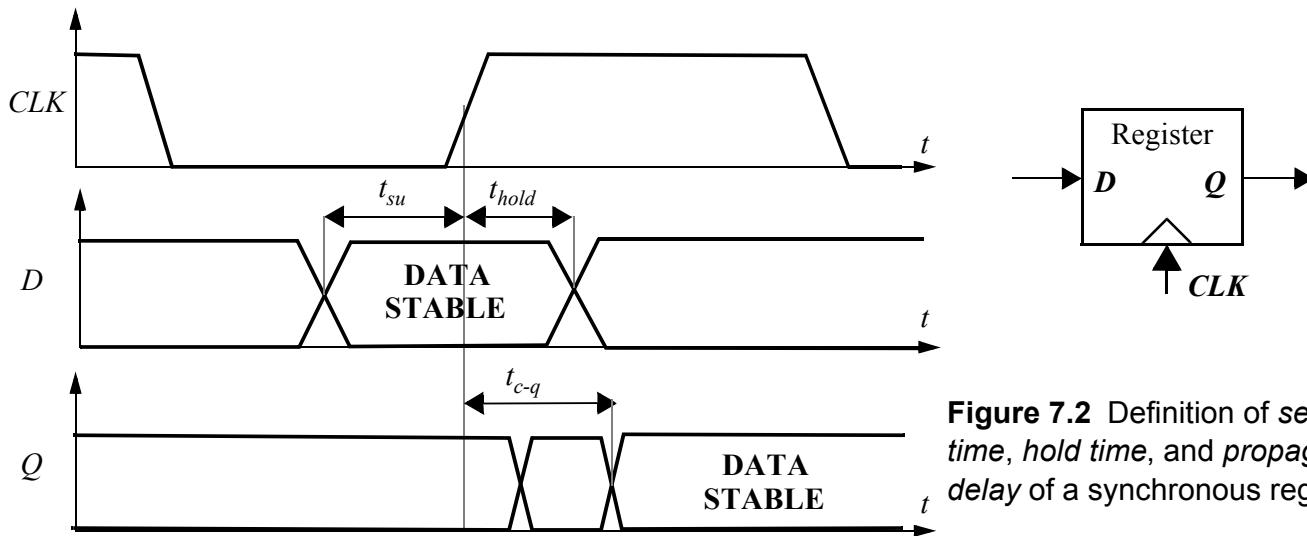
uma dada entrada em '0' saída em 1.  
uma dada entrada em '1' comporta-se como um inverter.

# Flip Flop Mestre Escravo Estático

# Flip-flop com set/reset assíncrono



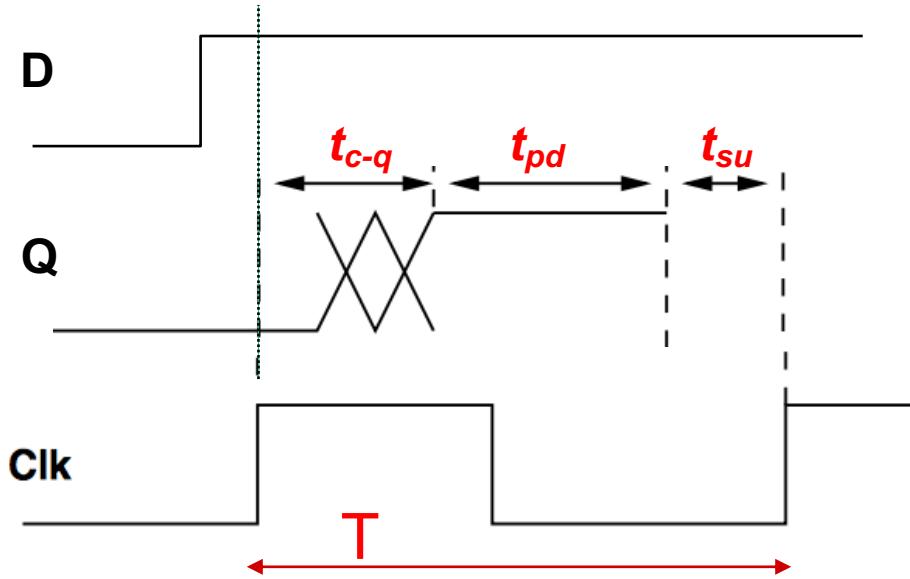
# Set-up and hold times



**Figure 7.2** Definition of set-up time, hold time, and propagation delay of a synchronous register.

- Tempo de **set-up** ( $t_{su}$ ): tempo que as entradas de dados (D) devem ser válidas antes da transição do relógio ( $0 \rightarrow 1$  para um registrador disparado por borda positiva)
- O tempo de **hold** ( $t_{hold}$ ): tempo que a entrada de dados deve permanecer válida após a transição do clock
- Atraso do registrador ( $t_{c-q}$ ): supondo que  $t_{su}$  e  $t_{hold}$  sejam atendidos, os dados na entrada D são copiados para a saída Q após  $t_{c-q}$

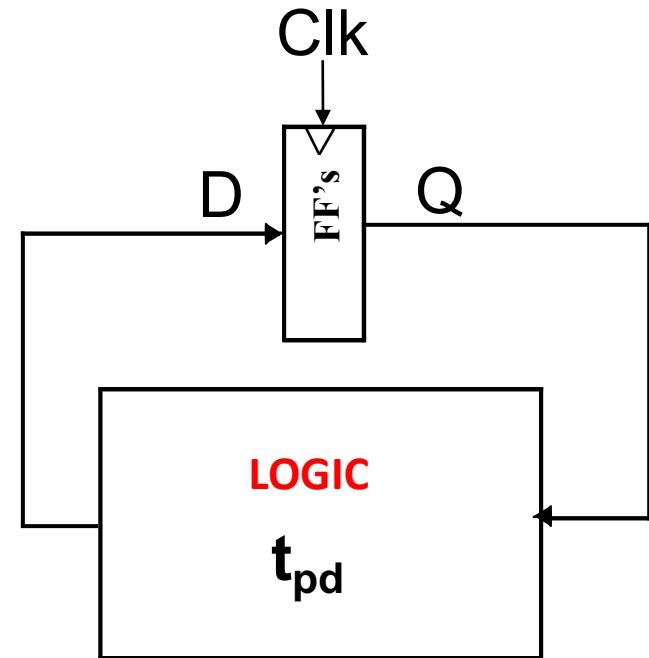
# Máxima freqüência de relógio



$t_{c-q}$ : tempo de propagação do flip-flop

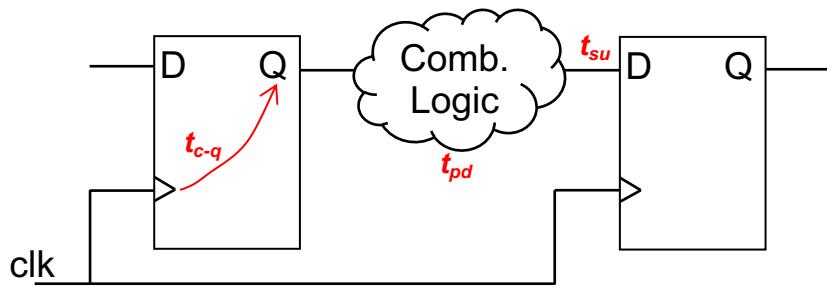
$t_{pd}$ : tempo de propagação da lógica combinacional

$t_{su}$ : tempo de setup do flip-flop



$$T \geq t_{c-q} + t_{pd} + t_{su}$$

# Exemplo



(1) Determina o tempo de *slack*

$$t_{c-q} = 200 \text{ ps}$$

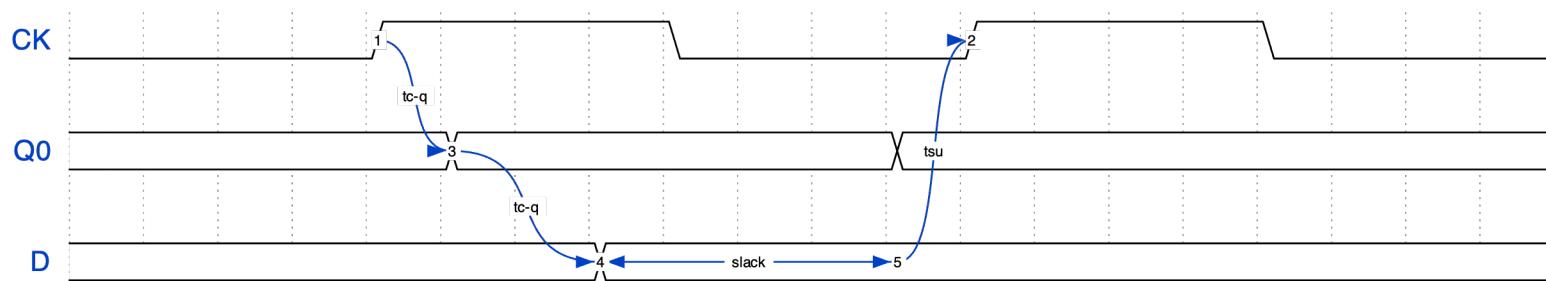
$$t_{su} = 100 \text{ ps}$$

$$t_{pd} = 300 \text{ ps}$$

$$T = 1000 \text{ ps}$$

(2) Qual o período mínimo?

$$T_{min} = 600 \text{ ps}$$



$$t_{slack} = 1000 - 600 = 400 \text{ ps}$$

# Relatório de síntese - exemplo

```
@genus:root: 15> report timing
```

Path 1: MET (544 ps) Setup Check with Pin EA\_reg[1]/CP->D

Group: Bus2IP\_Clkv – **2000 ps**

**Startpoint:** (R) address\_reg[2]/CP

Clock: (R) Bus2IP\_Clk

**Endpoint:** (R) EA\_reg[1]/D

Clock: (R) Bus2IP\_Clk

...

**Setup:- 72**

Required Time:= 1928

Launch Clock:- 0

Data Path:- 1383

**Slack:= 544**

#

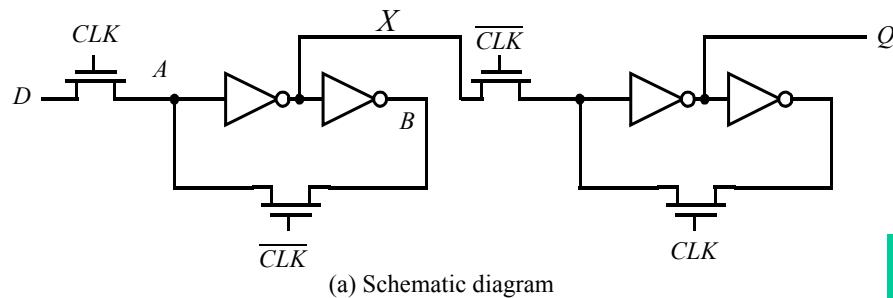
#	Timing Point	Flags	Arc	Edge	Cell	Fanout	Load (fF)	Trans (ps)	Delay (ps)	Arrival (ps)	Instance	Location
#	address_reg[2]/CP	-	-	R	(arrival)	116	-	0	-	0	(-, -)	
#	address_reg[2]/Q	-	CP->Q	R	HS65_GS_DFPRQX9	20	72.0	165	133	133	(-, -)	
#	g21671/Z	-	A->Z	F	HS65_GS_IVX9	19	70.0	118	122	255	(-, -)	
#	g21598/Z	-	A->Z	F	HS65_GS_AND2X4	10	42.3	107	128	383	(-, -)	
#	g21539/Z	-	B->Z	R	HS65_GS_NAND2X7	7	24.8	78	77	460	(-, -)	
#	g21301/Z	-	B->Z	F	HS65_GS_CBI4I6X5	1	5.8	44	41	502	(-, -)	
#	g21245/Z	-	E->Z	R	HS65_GS_A0I311X4	1	4.5	60	48	550	(-, -)	
#	g21168/Z	-	B->Z	R	HS65_GS_N0R4ABX2	1	6.1	113	102	652	(-, -)	
#	g21037/Z	-	B->Z	F	HS65_GS_OAI212X5	1	5.2	59	50	701	(-, -)	
#	g21027/Z	-	D->Z	R	HS65_GS_N0R4ABX2	1	4.8	98	75	776	(-, -)	
#	g21012/Z	-	B->Z	R	HS65_GS_AND2X4	12	58.3	260	181	958	(-, -)	
#	g20979/Z	-	B->Z	F	HS65_GSS_X0R2X6	1	6.1	68	127	1084	(-, -)	
#	g20956/Z	-	E->Z	R	HS65_GS_OAI212X5	1	5.8	68	40	1125	(-, -)	
#	g20946/Z	-	E->Z	F	HS65_GS_A0I212X4	1	4.5	56	37	1162	(-, -)	
#	g20938/Z	-	B->Z	F	HS65_GS_N0R4ABX2	1	4.8	39	60	1222	(-, -)	
#	g20927/Z	-	B->Z	R	HS65_GS_A0I12X2	1	4.5	84	60	1282	(-, -)	
#	g20922/Z	-	B->Z	R	HS65_GS_N0R4ABX2	1	5.0	98	101	1383	(-, -)	
#	EA_reg[1]/D	<<<	-	R	HS65_GS_DFPRQX9	1	-	-	0	1383	(-, -)	

# LIB file – setup and hold

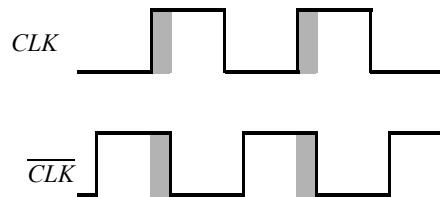
```
...
timing(){
    intrinsic_fall : 0.075;
    intrinsic_rise : 0.055;
    related_pin : "CP";
    sdf_edges : "both_edges";
    timing_label : "D_CP_SETUP";
    timing_type : setup_rising;
    fall_constraint(table_44){
        values("0.075000, 0.115000, 0.155000",\
              "0.040000, 0.080000, 0.125000",\
              "0.020000, 0.060000, 0.100000");
    }
    rise_constraint(table_44){
        values("0.055010, 0.095010, 0.125010",\
              "0.040010, 0.075010, 0.105010",\
              "0.030010, 0.070010, 0.095010");
    }
}

timing(){
    intrinsic_fall : -0.04;
    intrinsic_rise : -0.035;
    related_pin : "CP";
    sdf_edges : "both_edges";
    timing_label : "D_CP_HOLD";
    timing_type : hold_rising;
    fall_constraint(table_43){
        values("-0.040000, -0.015000, 0.000000",\
              "-0.080000, -0.060000, -0.040000",\
              "-0.120000, -0.095000, -0.075000");
    }
    rise_constraint(table_43){
        values("-0.034990, -0.019990, -0.014990",\
              "-0.074990, -0.059990, -0.054990",\
              "-0.104990, -0.089990, -0.079990");
    }
}
```

# Problema de corrida (race)



(a) Schematic diagram



(b) Overlapping clock pairs

Sobreposição do sinal do relógio pode causar

- Sinais indefinidos na entrada e saída
- Condições de 'Race' (High por um período longo)

Figure 7.20 Master-slave register based on NMOS-only pass transistors.

## Uma solução: clock sem sobreposição

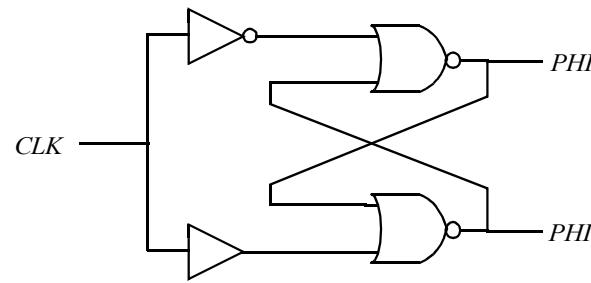
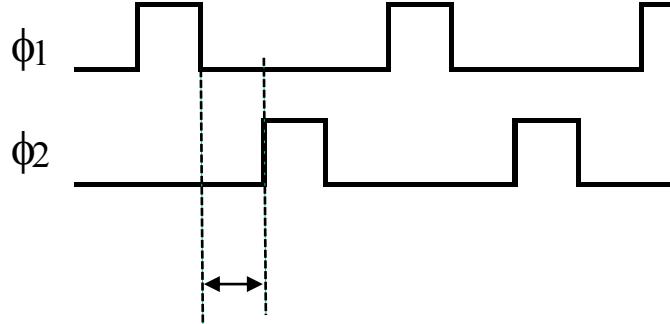
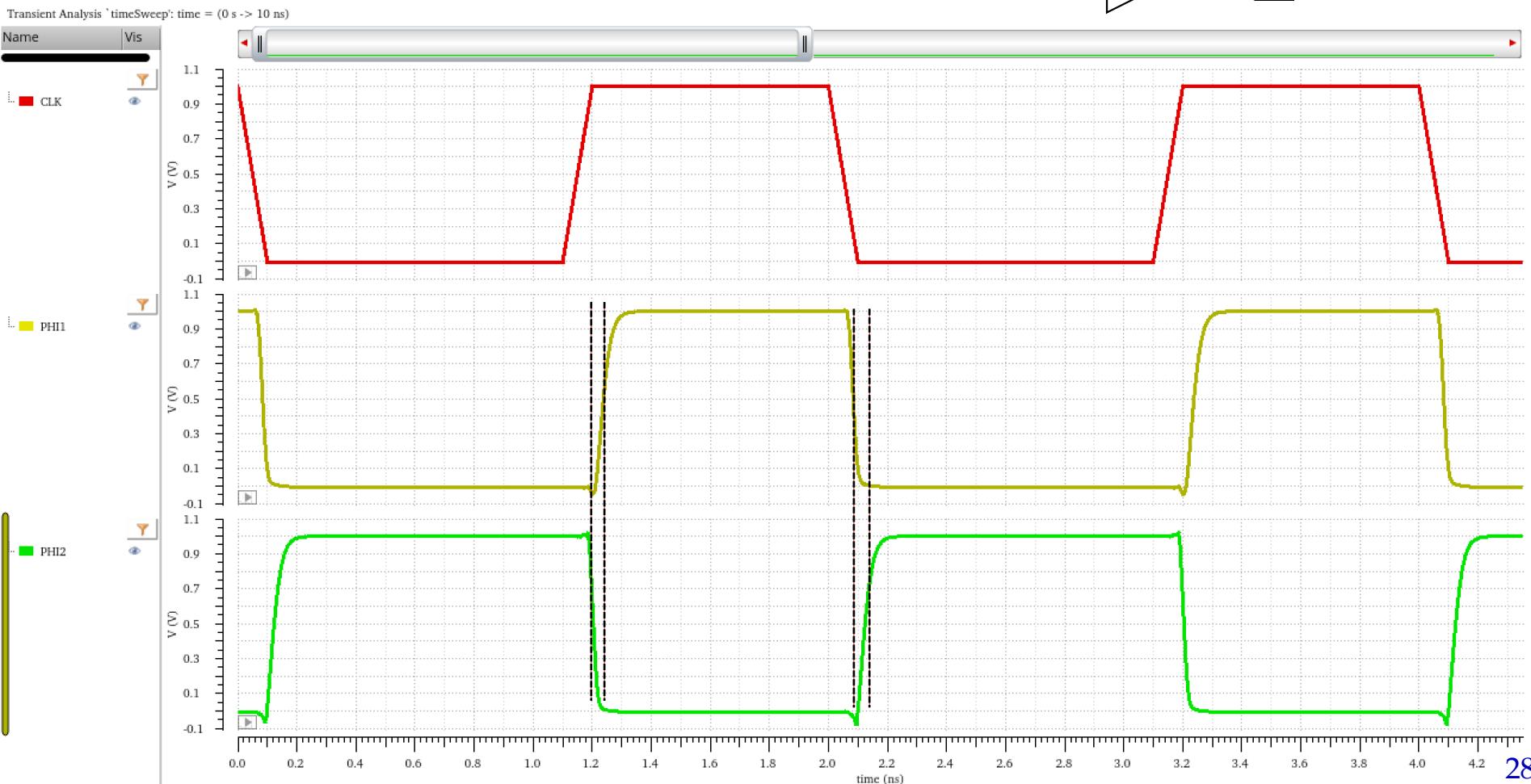
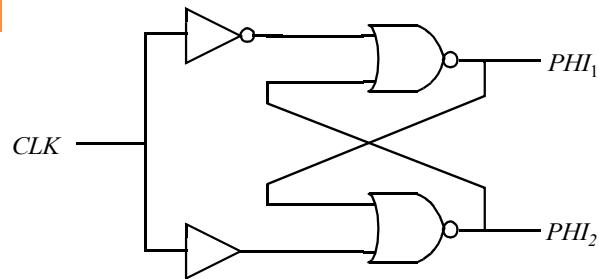
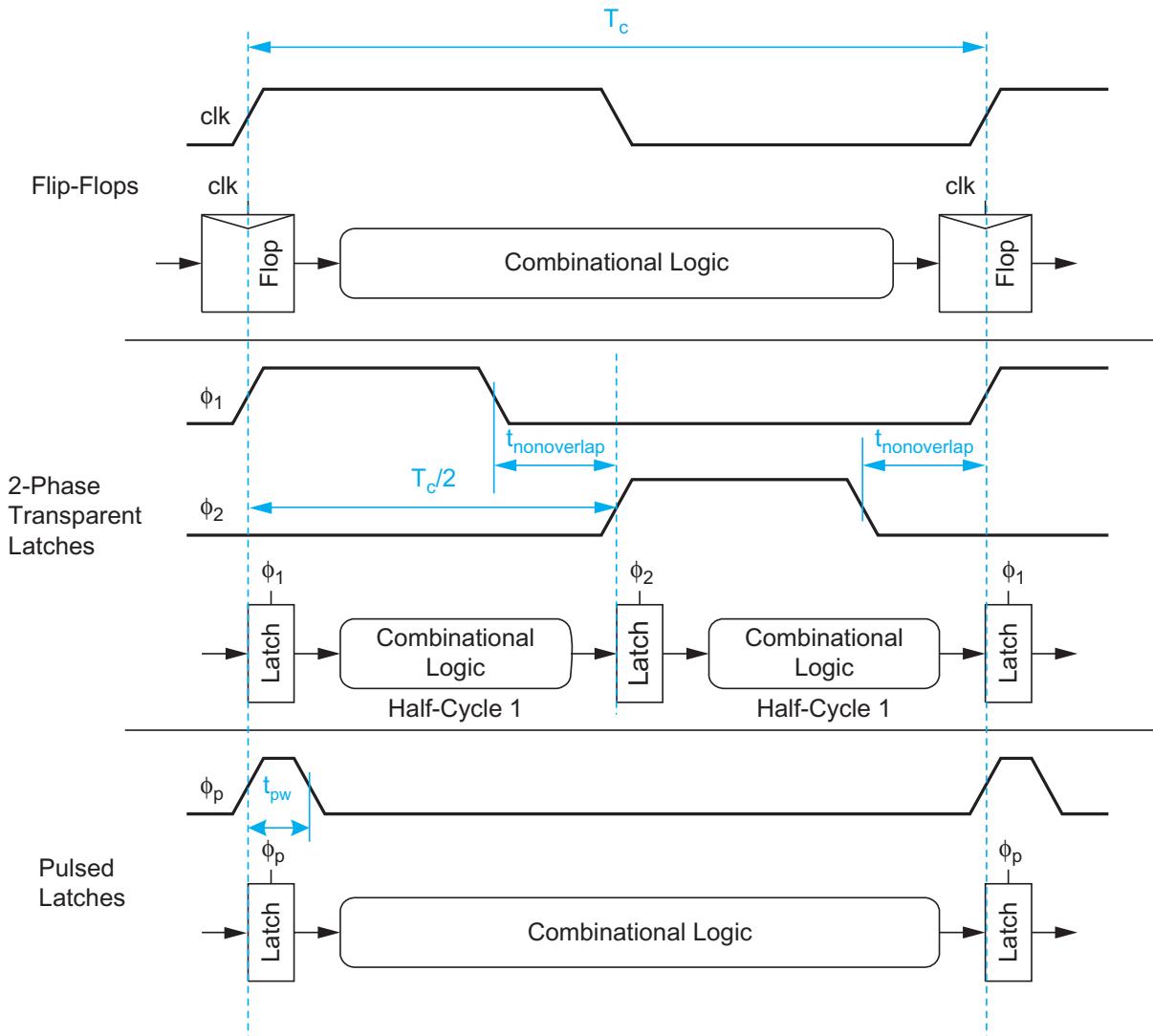


Figure 7.22 Circuitry for generating a two-phase non-overlapping clock.

# Fases não sobrepostas



# Métodos de sincronização



MS

**Latch-based  
design**

**Latch  
“pulsadas”  
(glitch)**

FIGURE 10.2 Static sequencing methods

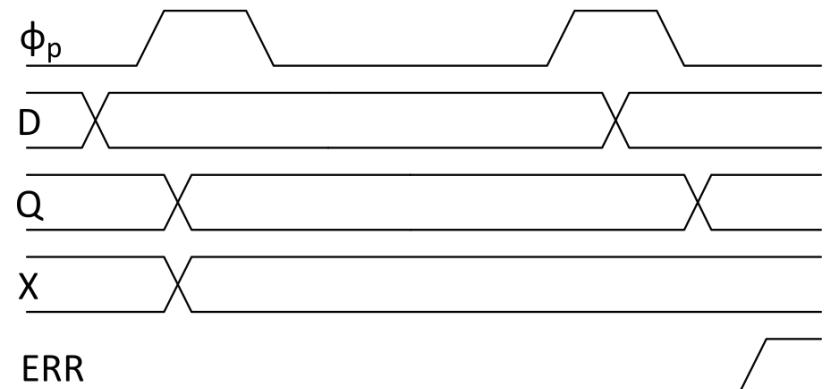
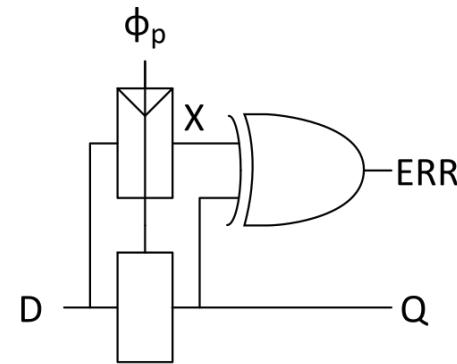
# Razor

Os projetistas devem incluir margem de tempo na frequência devido:

- tensão
- temperatura
- variação do processo
- dependência de dados
- imprecisões das ferramentas

Alternativa:

- utilizar frequência mais alta e verificar erros falhas
- técnica: **razor**
- pode reduzir o tempo o período em ~30%



# Razor (ENADE)

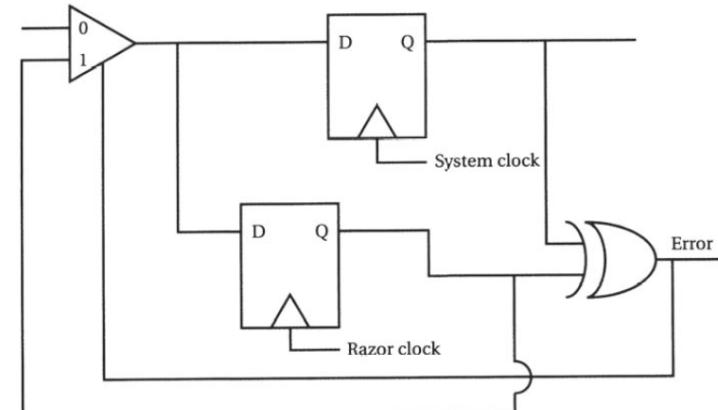
## QUESTÃO 43

O razor é uma arquitetura para desempenho better-than-worst-case que usa um registrador especializado, mostrado na figura, que mede e avalia os erros.

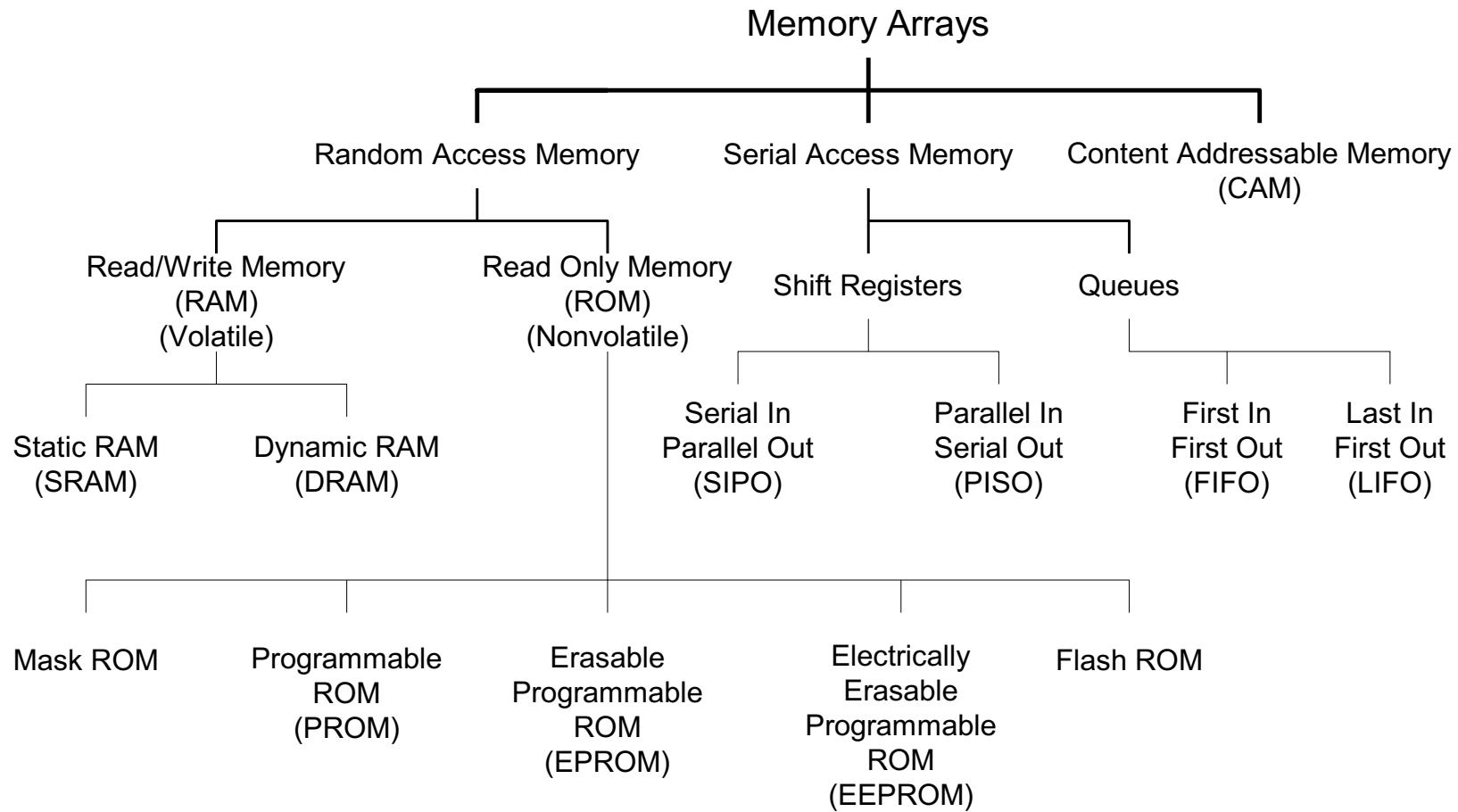
O registrador do sistema mantém o valor chaveado e é comandado por um *clock* de sistema *better-than-worst-case*. Um registrador adicional é comandado separadamente por um *clock* ligeiramente atrasado com relação ao do sistema. Se os resultados armazenados nos dois registradores são diferentes, então um erro ocorreu, provavelmente devido a temporização. A porta XOR detecta o erro e faz com que esse valor seja substituído por aquele do registrador do sistema.

Considerando essas informações, analise as afirmações a seguir.

- I. Sistemas digitais são tradicionalmente concebidos como sistemas assíncronos regidos por um clock.
- II. Better-than-worst-case é um estilo de projeto alternativo em que a lógica detecta e se recupera de erros, permitindo que o circuito possa operar com uma frequência maior.
- III. Nos sistemas digitais, o período de clock é determinado por uma análise cuidadosa para que os valores sejam armazenados corretamente nos registradores, com o período de clock alargado para abranger o atraso de pior caso.



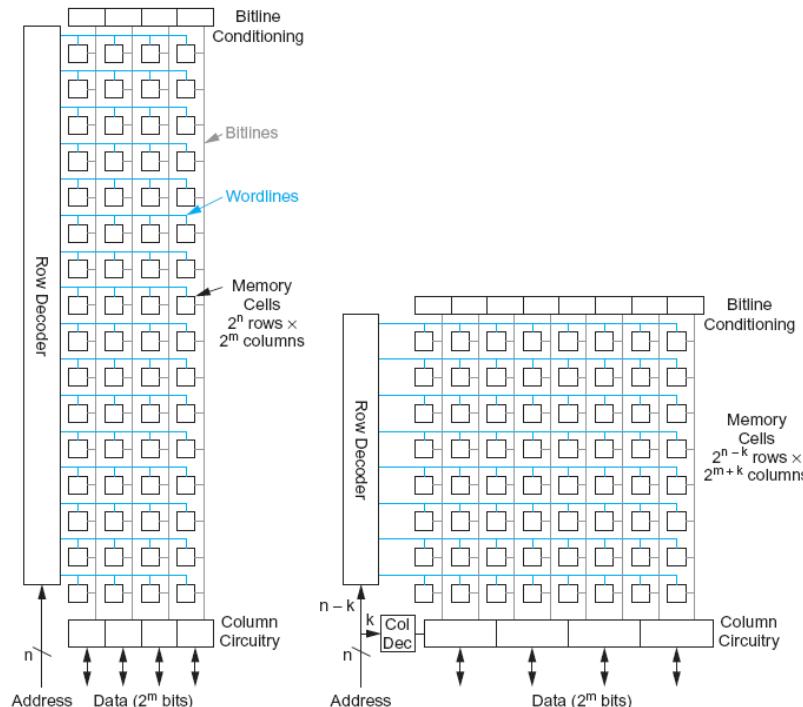
# Memórias



# Memórias

## Array Architecture

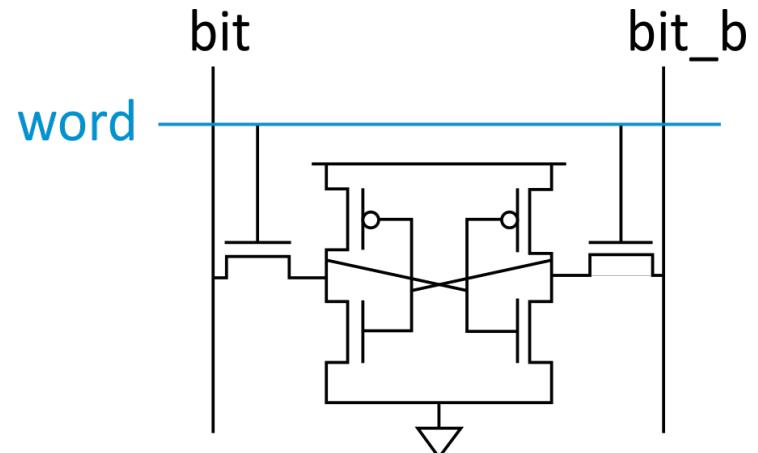
- $2^n$  words of  $2^m$  bits each
- If  $n \gg m$ , fold by  $2^k$  into fewer rows of more columns



- Good regularity – easy to design
- Very high density if good cells are used

## 6T SRAM Cell

- Cell size accounts for most of array size
  - Reduce cell size at expense of complexity
- 6T SRAM Cell
  - Used in most commercial chips
  - Data stored in cross-coupled inverters
- Read:
  - Precharge bit, bit\_b
  - Raise wordline
- Write:
  - Drive data onto bit, bit\_b
  - Raise wordline



- Considere o circuito abaixo, composto por 2 entradas de 48 bits (A e B), e duas saídas (*soma\_out* de 48 bits, e *carry\_out*).
- Os flip-flops (FFs – mestre-escravo, sensíveis à borda de subida) possuem um tempo de propagação D→Q igual a 0,8 ns e o tempo de setup igual a 0,2 ns.
- Um somador tem tempo de propagação igual a 10,4 ns

**Determine a frequência de operação do circuito.**

