


**NAND**

Inputs		Output
A	B	F
0	0	1
1	0	1
0	1	1
1	1	0


**AND**

A	B	F
0	0	0
1	0	0
0	1	0
1	1	1



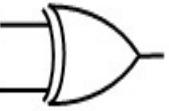
**OR**

Inputs		Output
A	B	F
0	0	0
1	0	1
0	1	1
1	1	1



**NOR**

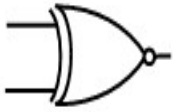
Inputs		Output
A	B	F
0	0	1
1	0	0
0	1	0
1	1	0



**EXCLUSIVE OR**

Inputs		Output
A	B	F
0	0	0
0	1	1
1	0	1
1	1	0

**EXCLUSIVE NOR**



Inputs		Output
A	B	F
0	0	1
0	1	0
1	0	0
1	1	1

# IC – Introdução a Computação

# Elementos de Memória



## Latch SR

- Na aula passada vimos que o Latch SR é o elemento de memória mais simples que temos.
- Vimos que o Latch SR pode mudar de estado a qualquer momento.
- Vimos que ele não pode receber a entrada  $SR = 11$  pois isso acarreta em um problema no Latch.

O que podemos fazer para evitar que aconteça as entradas  $SR = 11$  no nosso Latch?

Vamos fazer uma modificação, adicionando uma entrada nova em nosso Latch SR.  
Se estamos falando de controle, estamos falando de?

CLOCK!!!

Vamos incluir um sinal C (Clock), que vai indicar quando o Latch SR modifica seu estado. Com isso as nossas entradas SR irão estabilizar. Evitando assim a condição indesejada (como  $SR = 11$ ).

# Elementos de Memória

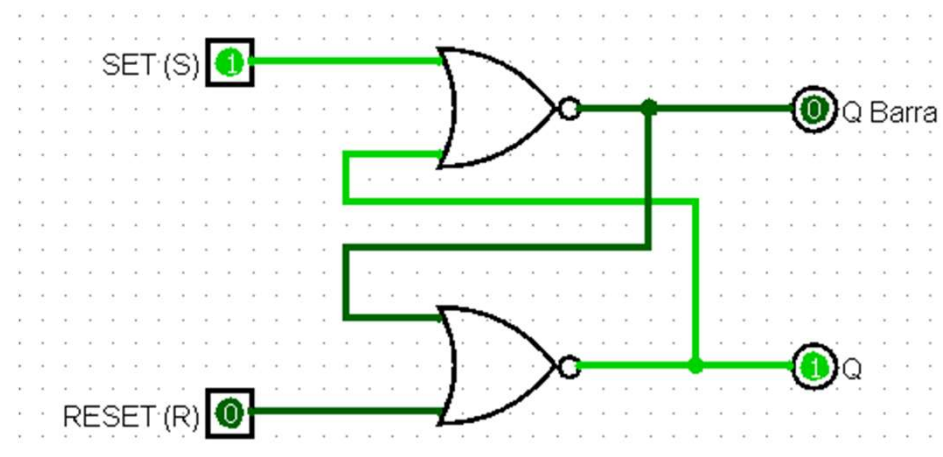
## Latch SR - Relembrando

Possui duas entradas, SET e RESET, que podem mudar os valores armazenados em Q e Q`.

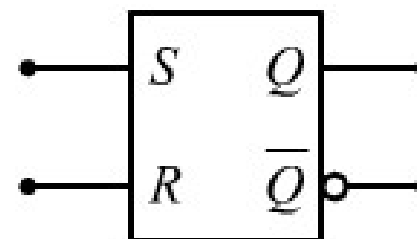
### Estados possíveis:

- ☐ Estado SET:  $Q = 1$  e  $Q' = 0$ ;
- ☐ Estado RESET  $Q = 0$  e  $Q' = 1$ ;

### Nosso circuito

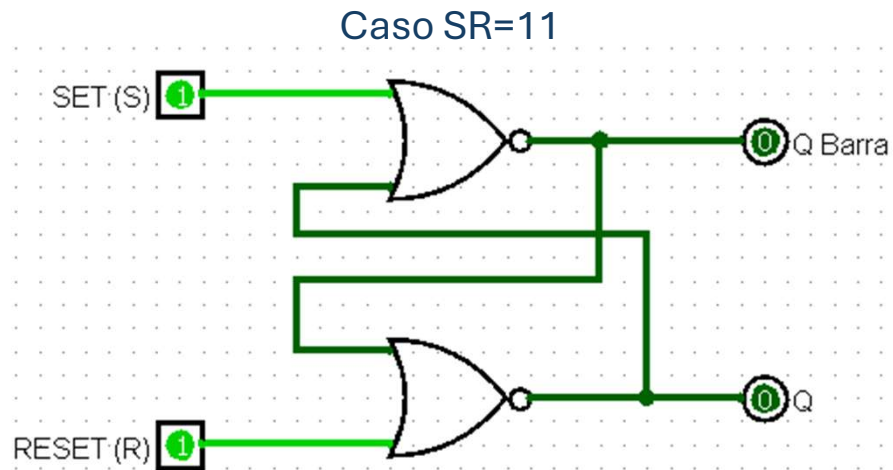


### Símbolo Esquemático



# Elementos de Memória

## LATCH SR



Estado Impossível ou indesejado

Resumindo:

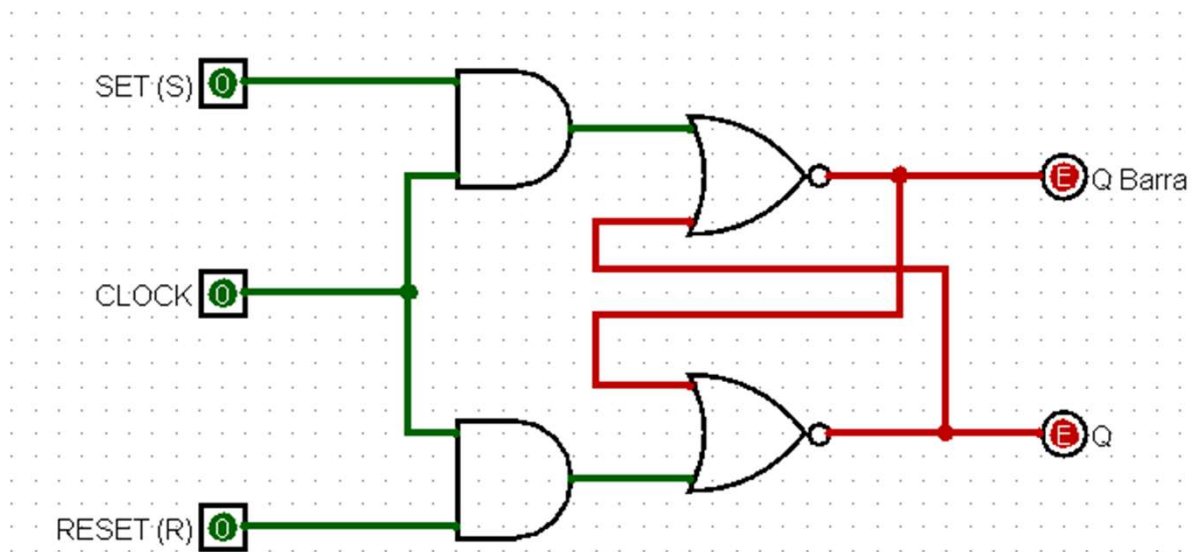
- ❑ R=1, S=0 faz com que o Latch vá para o estado de RESET;
- ❑ R=0, S=1 faz com que o Latch vá para o estado de SET;
- ❑ R=0, S=0 faz com que o Latch não mude o seu estado;
- ❑ R=1, S=1 é uma situação proibida;

# Elementos de Memória

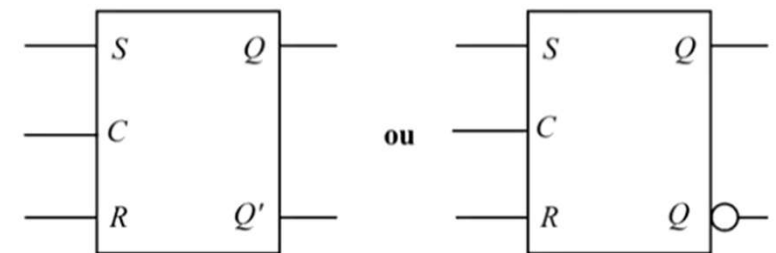
## LATCH SR – Controlado

Para resolver o problema da entrada impossível SR 11, vamos utilizar um Latch SR controlado, ou seja, vamos adicionar uma nova entrada no nosso Latch.

### Nosso circuito



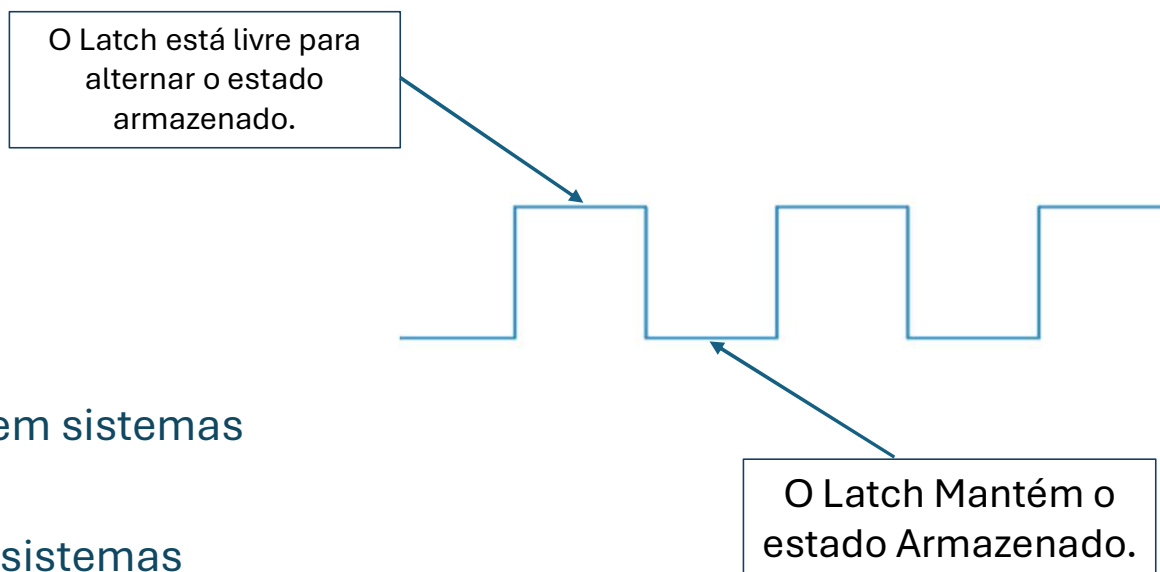
### Símbolo Esquemático



# Elementos de Memória

## LATCH SR – Controlado

O sinal C é o nosso sinal de controle, ou seja nosso Relógio.



- O Latch SR controlado é utilizado em sistemas de modo Pulsado
- O Latch SR comum é utilizado em sistemas de modo fundamental.
- Importância da frequência do Relógio: deve ser tal que permita que as entradas S e R atinjam a estabilidade.

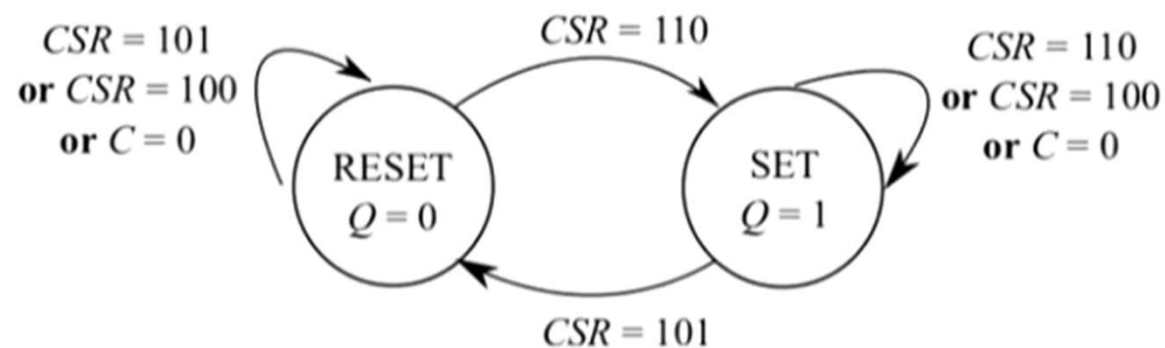
# Elementos de Memória

## LATCH SR – Controlado

Tabela de transição de Estados:

Entradas			E <sup>+</sup>	Comentários
C	S	R		
0	-	-	E	Latch Desabilitado
1	0	0	E	Manutenção do Estado
1	0	1	0	Estado de Reset
1	1	0	1	Estado de SET
1	1	1	Impossível	Condição Evitada

Diagrama de Transição de Estados:



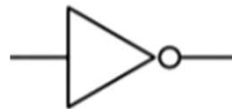
# Elementos de Memória

## LATCH SR – D (Transparente)

O Latch SR controlado ainda não resolve todos os nossos problemas em relação a uma entrada indesejada.

Qual seria a Solução então para que essa entrada indesejada nunca ocorra?

Ligamos um Inversor entre as entradas SR, garantindo assim que esse estado indesejado  $SR=11$  não ocorra nunca!

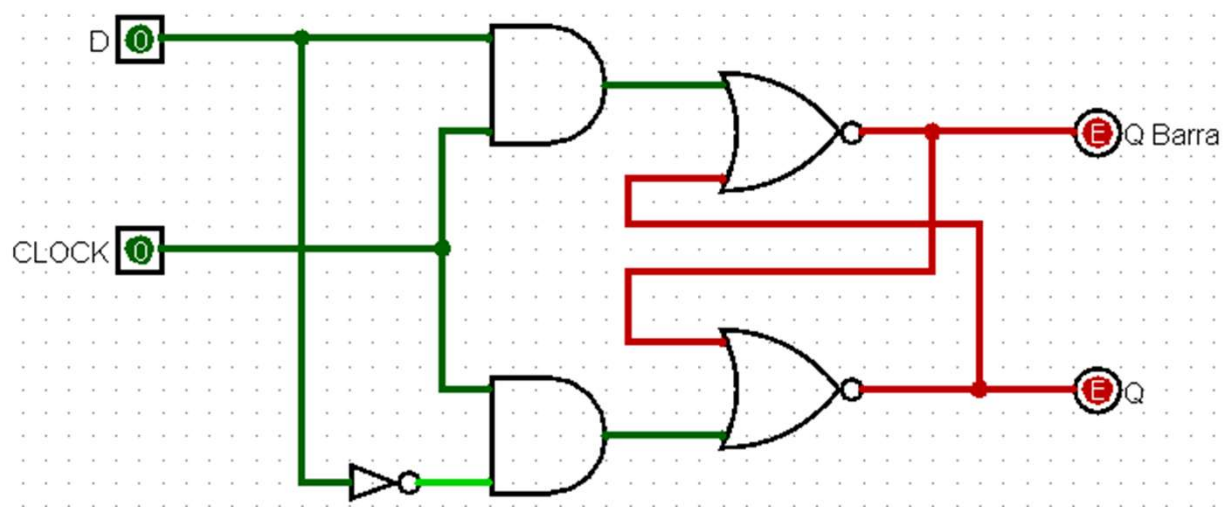




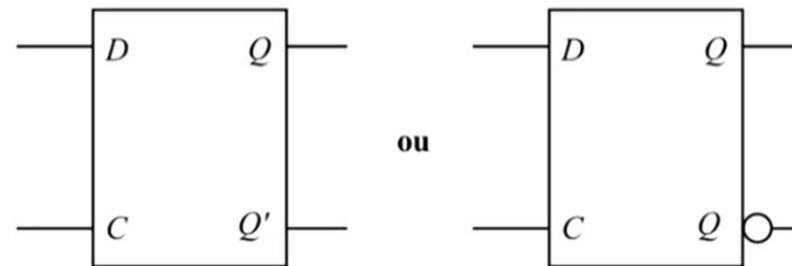
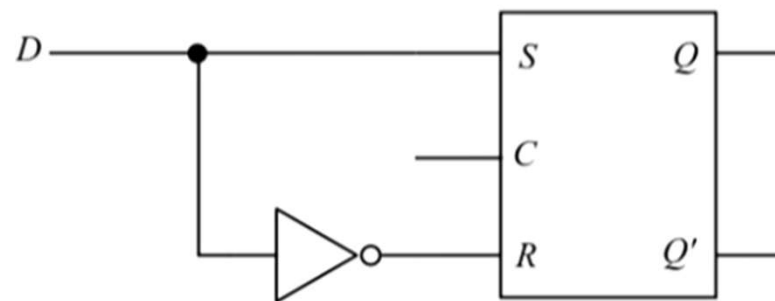
# Elementos de Memória

LATCH SR – D (Transparente)

## Nosso circuito



## Símbolo Esquemático



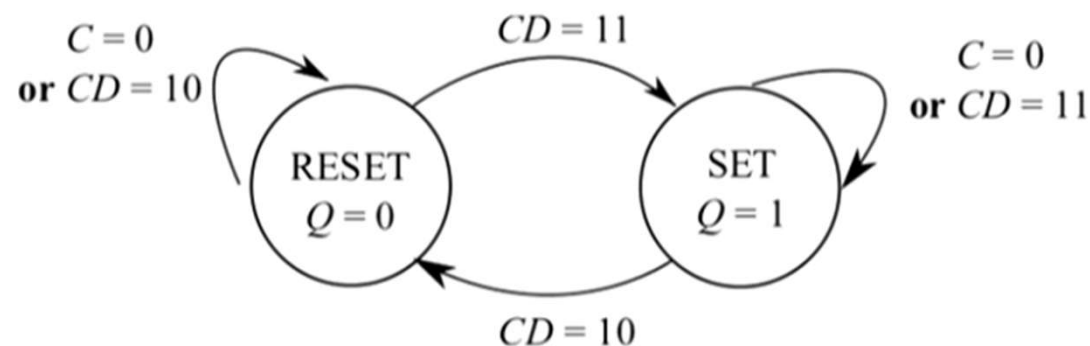
# Elementos de Memória

## LATCH SR – D (Transparente)

Tabela de Transição de estados:

Entradas		E <sup>+</sup>	Comentários
C	D		
0	-	E	Mantém o Estado
1	0	0	Estado de Reset
1	1	1	Estado de SET

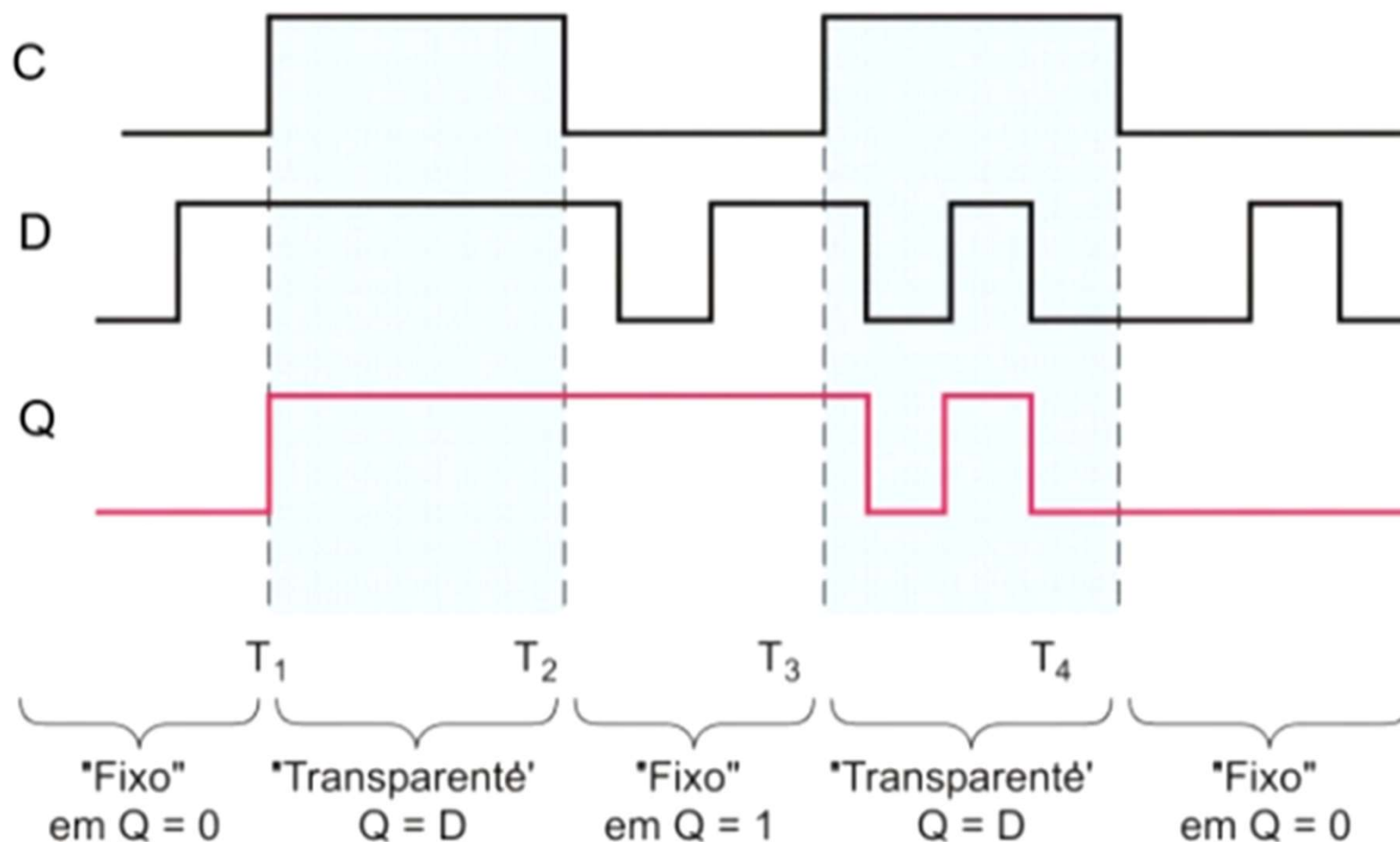
Diagrama de Transição de Estados:



# Elementos de Memória

LATCH SR – D (Transparente)

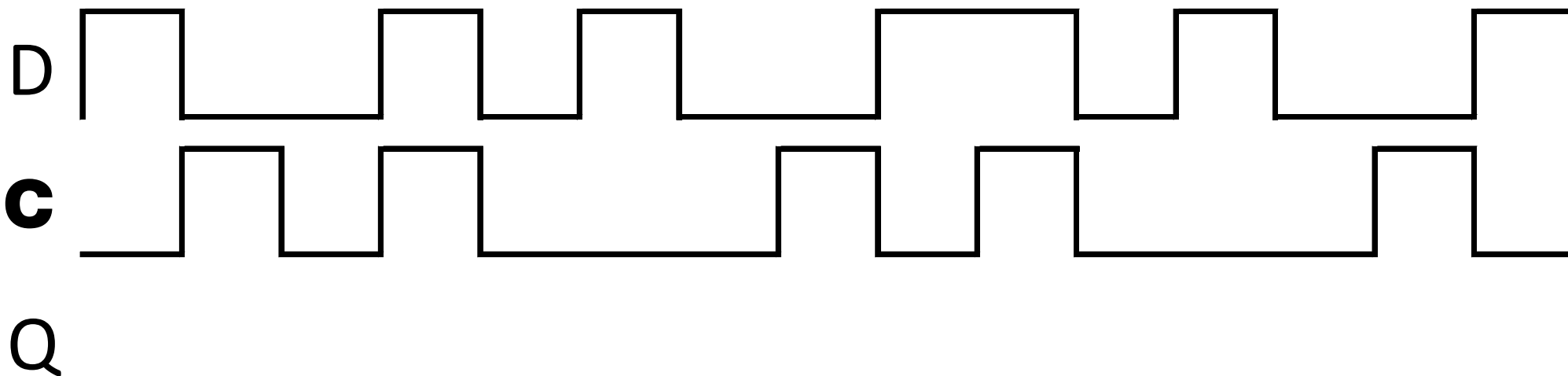
Diagrama de Temporização:



# Elementos de Memória

LATCH SR – D (Transparente)

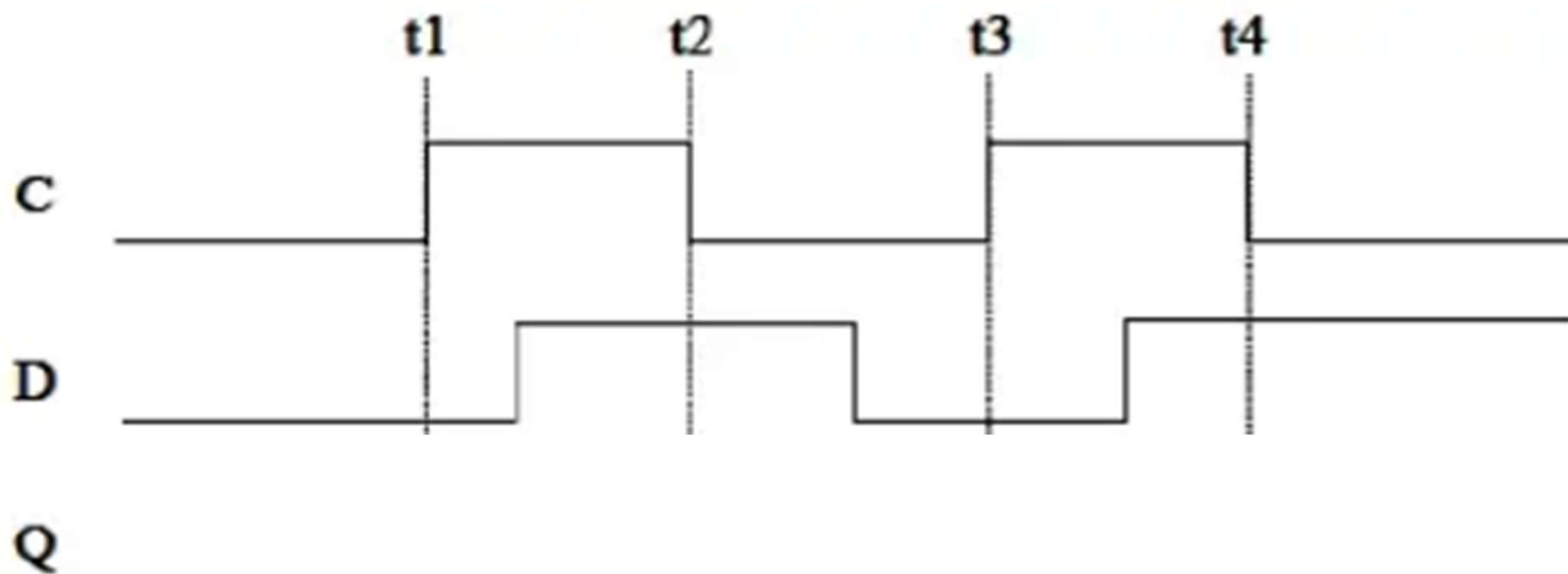
Exercícios de Diagrama de Temporização:



# Elementos de Memória

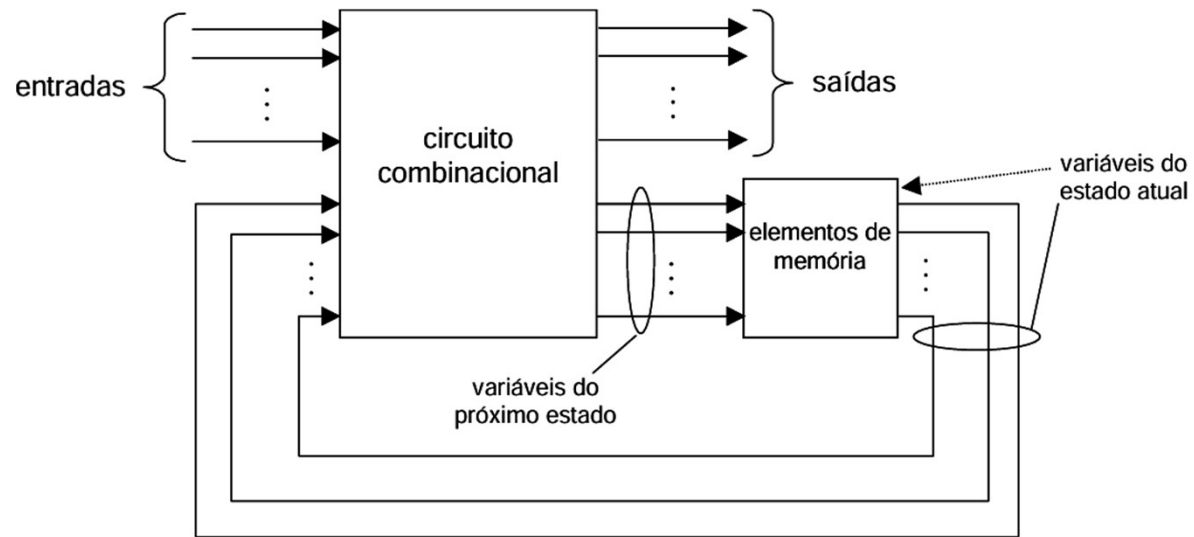
LATCH SR – D (Transparente)

Exercícios de Diagrama de Temporização:



# Elementos de Memória

## Flip-Flops



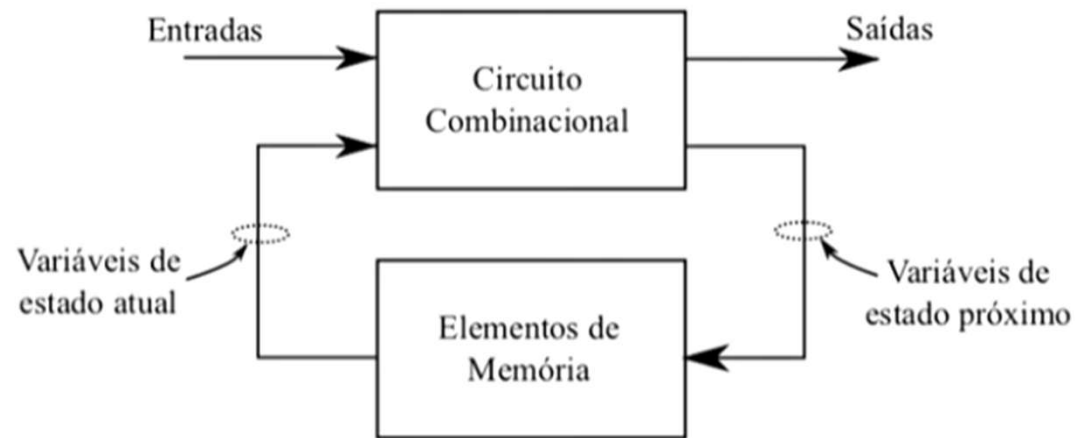
### ➤ Latch:

- Latch normais: Muda os valores armazenados toda vez que suas entradas mudam;
- Latch controlado: Sensível a um nível de sinal de controle;

### ➤ Flip-Flops: Sensível a uma borda de um sinal de controle;

# Elementos de Memória

## Flip-Flops



### **Temos Problemas em utilizar apenas Latches.**

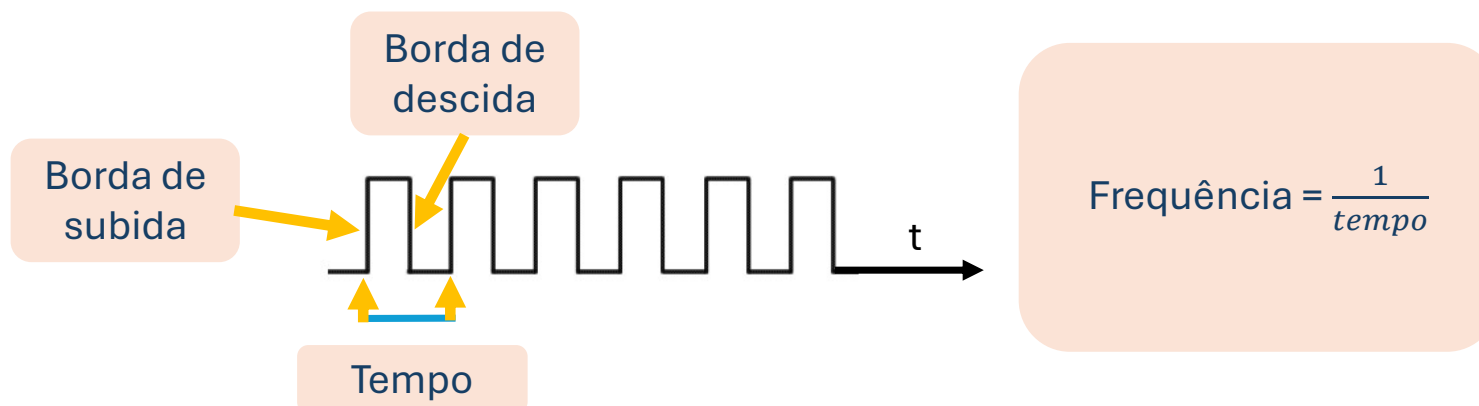
- As transições de estados dos Latches ocorrem quando o pulso do relógio está em nível lógico alto.
- Situação imprevisível, pois o estado dos Latches pode continuar mudando enquanto o pulso do relógio permanecer no nível lógico Alto.

# Elementos de Memória

## Flip-Flops

Vamos dar uma pausa e detalhar o que é clock.

Pulsos para que todos os controladores internos atuem no mesmo ritmo.  
Determina a velocidade com que o processamento será executado.



**Frequência:** Quantidade de pulsos por segundo, medida em Hertz (Hz)

- 1Hz = 1 pulso por segundo ou 1pps (GPS)
- 10Hz = 10 pulsos por segundo
- 1kHz = 1.000 pulsos por segundo
- 1mHz = 1 milhão de pulsos por segundo



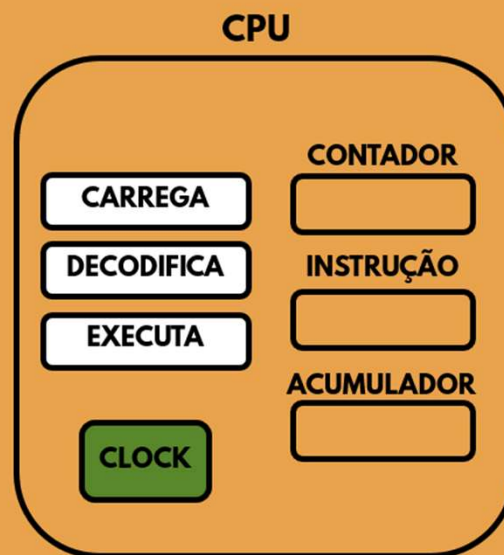
# Elementos de Memória

Flip-Flops

Clock então é a velocidade?

Não é só o clock, vai depender da sua arquitetura ... Vamos ver esse exemplo:

## O que seu computador está realmente fazendo?



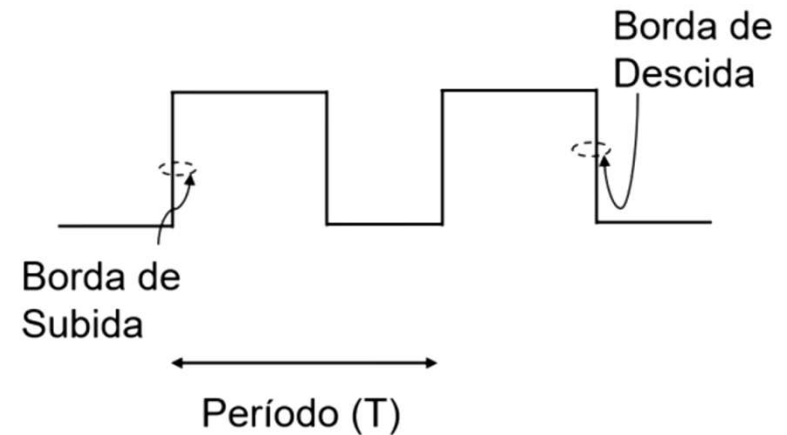
**MEMÓRIA**

ENDEREÇO	VALOR
0	LOAD 6
1	ADD 7
2	STORE 6
3	JUMP 1
4	0
5	0
6	1
7	1

# Elementos de Memória

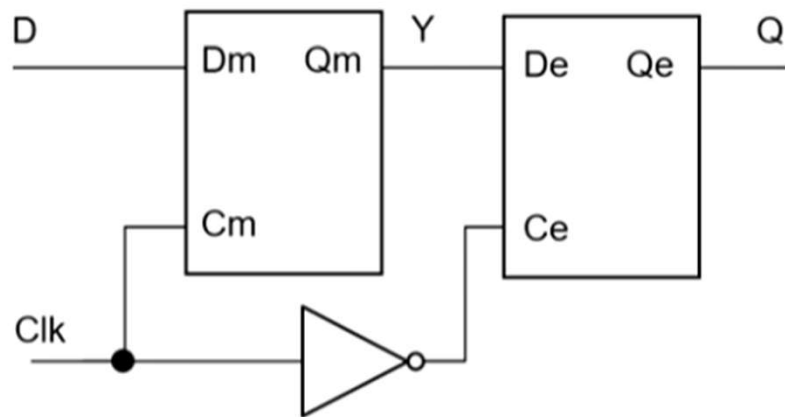
## Flip-Flops

- Vamos utilizar um circuito que seja sensível a borda do sinal de Clock, a esse dispositivo chamamos de FLIP-FLOP.
- O Flip-Flop vai permanecer ativo por um intervalo de tempo muito pequeno.
- Entre transições sucessivas do mesmo tipo, o Flip-Flop mantém o último estado adquirido.
  - Transição de subida do relógio;
  - Transição de descida do relógio;

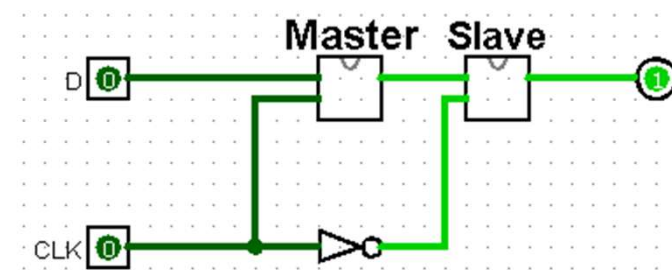


# Elementos de Memória

## Flip-Flops – D Master-Slave



### Nosso circuito

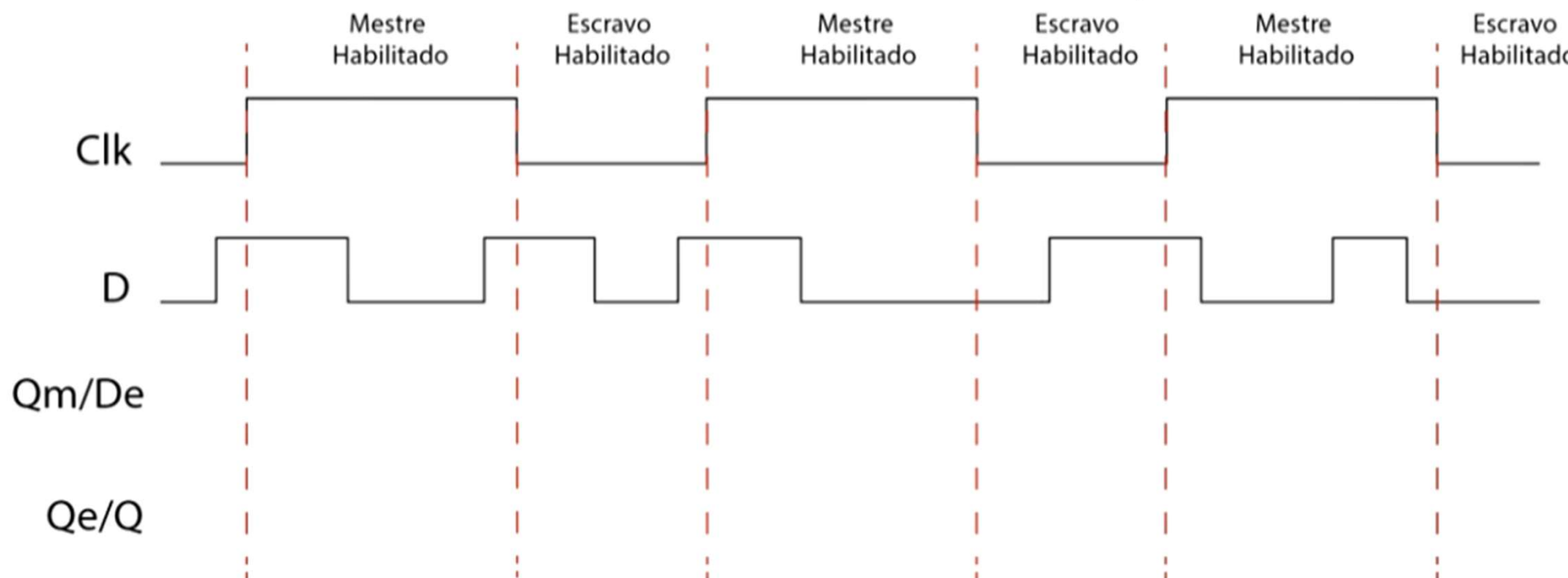
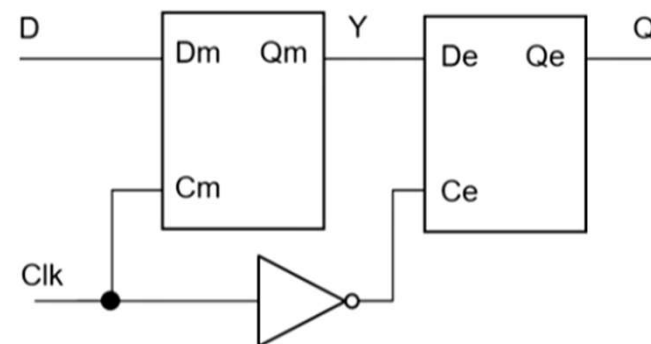


- Dois Latches D em cascata
- Os Latches estão funcionando de forma complementar
- Gatilhado na borda de descida do relógio.

# Elementos de Memória

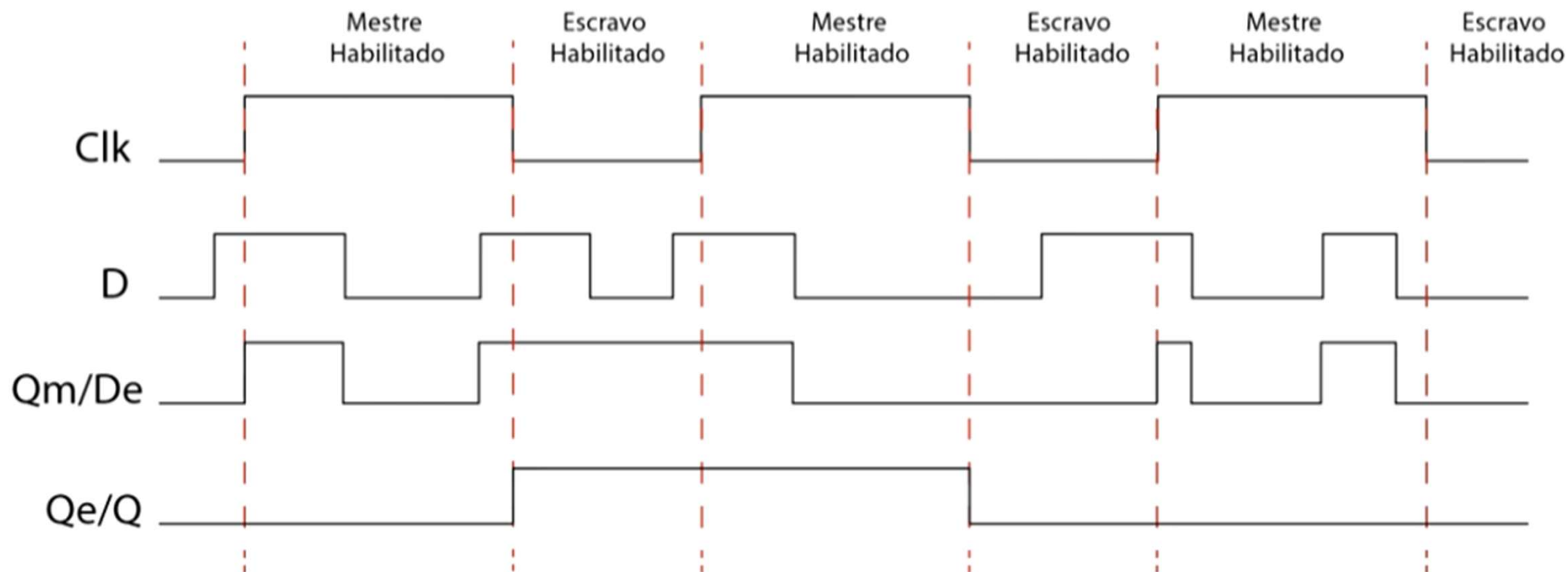
## Flip-Flops

### Análise do Flip-Flop D MS



# Elementos de Memória

## Flip-Flops



**ATÉ A  
PRÓXIMA  
AULA!**



## Bibliografia



TOCCI, R.; WIDMER, N.; MOSS, G. Sistemas Digitais – Princípios e Aplicações. [S.I.]: Pearson Education Limited, 2011.

FEDELI, Ricardo Daniel. Introdução à ciência da computação / Ricardo Daniel Fedeli, Erico Giulio Franco Polloni, Fernando Eduardo Peres. – 2. ed. – São Paulo: Cengage Learning, 2011.

TANENBAUM, Andrew S.. Organização Estruturada de Computadores. 6º Edição. São Paulo, Pearson Prentice Hall, 2013.



This work is licensed under a [Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License](https://creativecommons.org/licenses/by-nc-sa/4.0/).