FUNDAÇÃO EDSON QUEIROZ
UNIVERSIDADE DE FORTALEZA
ENSINANDO E APRENDENDO

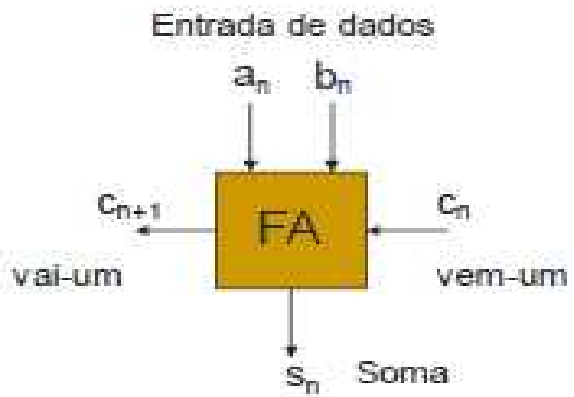
T566 –SISTEMAS DIGITAIS AVANÇADOS

Aula 6 - Circuitos combinacionais e seqüenciais

Prof. Danilo Reis

Somadores

Também chamado: full-adder. O somador completo tem 3 bits de entrada a_n e b_n, utilizados pelos dados,e c_n, utilizado como bit de entrada do vai-um da coluna imediatamente à direita. O circuito produz dois bits de saída, a soma s_n e o vai-um de saída c_{n+1} .



a_n	b_n	c_n	s_n	c_{n+1}
0	0	0	0	0
0	1	0	1	0
1	0	0	1	0
1	1	0	0	1
0	0	1	1	0
0	1	1	0	1
1	0	1	0	1
1	1	1	1	1

Somadores

- Na forma SDP:
  - $s_n = a'_n.b_n.c'_n + a_n.b'_n.c'_n + a'_n.b'_n.c_n + a_n.b_n.c_n$
- Simplificando:
  - $s_n = (a'_n.b_n + a_n.b'_n).\ c'_n + (a'_n.b'_n + a_n.b_n).\ c_n$
  - $= \underbrace{(a_n \oplus b_n)}_{x}.\ c'_n + \underbrace{(a_n \oplus b_n)'}_{x'}.\ c_n$

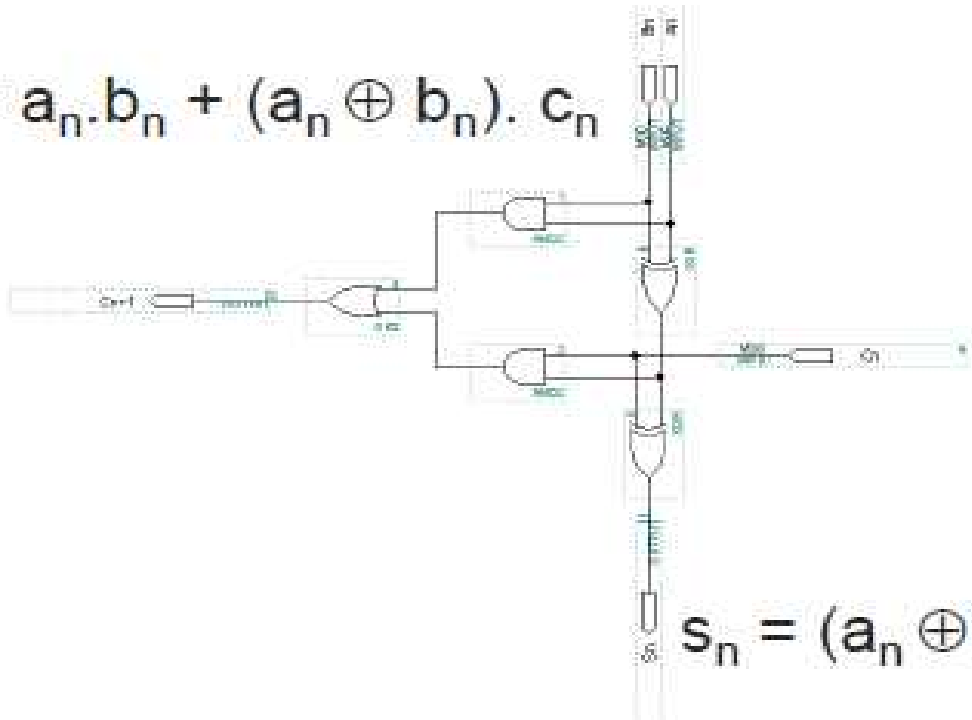
$$s_n = (a_n \oplus b_n \oplus c_n)$$

$$\begin{cases} x \oplus y = x'.y + x.y' \\ (x \oplus y)' = x.y + x'.y' \end{cases}$$

Somadores

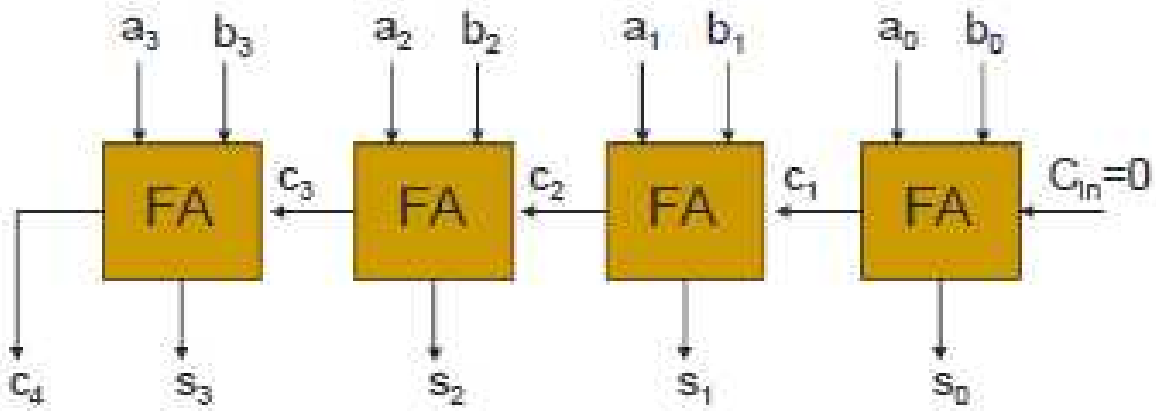
- Da mesma forma para c_{n+1}
  - $c_{n+1} = a_n.b_n.c'_n + a'_n.b_n.c_n + a_n.b'_n.c_n + a_n.b_n.c_n$
  - $= a_n.b_n.(c'_n + c_n) + (a'_n.b_n + a_n.b_n).c_n$
  - $= a_n.b_n + (a_n \oplus b_n).\ c_n$

$c_{n+1} = a_n.b_n + (a_n \oplus b_n).\ c_n$



$s_n = (a_n \oplus b_n \oplus c_n)$

Somadores



Subtratores

Módulo somador com números negativos expressos em complemento de 2.
Complemento 2 = complemento 1 + 1

1 0 1 1 0 1 Número binário original

↓ ↓ ↓ ↓ ↓ ↓

0 1 0 0 1 0 Complemento de 1

```
  1 0 1 1 0 1   (binário (45)10)
  0 1 0 0 1 0   (complemento de 1)
+             1 (adiciona-se 1)
  ------------
  0 1 0 0 1 1   (Complemento de 2)
```

Subtratores

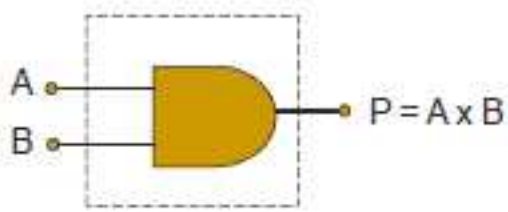
Representa-se o número negativo com 1 bit de sinal
0 - Número positivo
1 - Número negativo

Multiplicadores

A multiplicação binária é definida pelas seguintes regras:

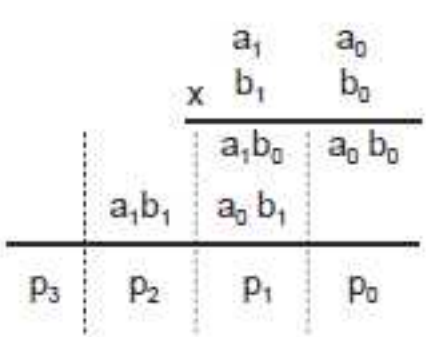
0 x 0 = 0
0 x 1 = 0
1 x 0 = 0
1 x 1 = 1

A	B	(AxB)
0	0	0
0	1	0
1	0	0
1	1	1

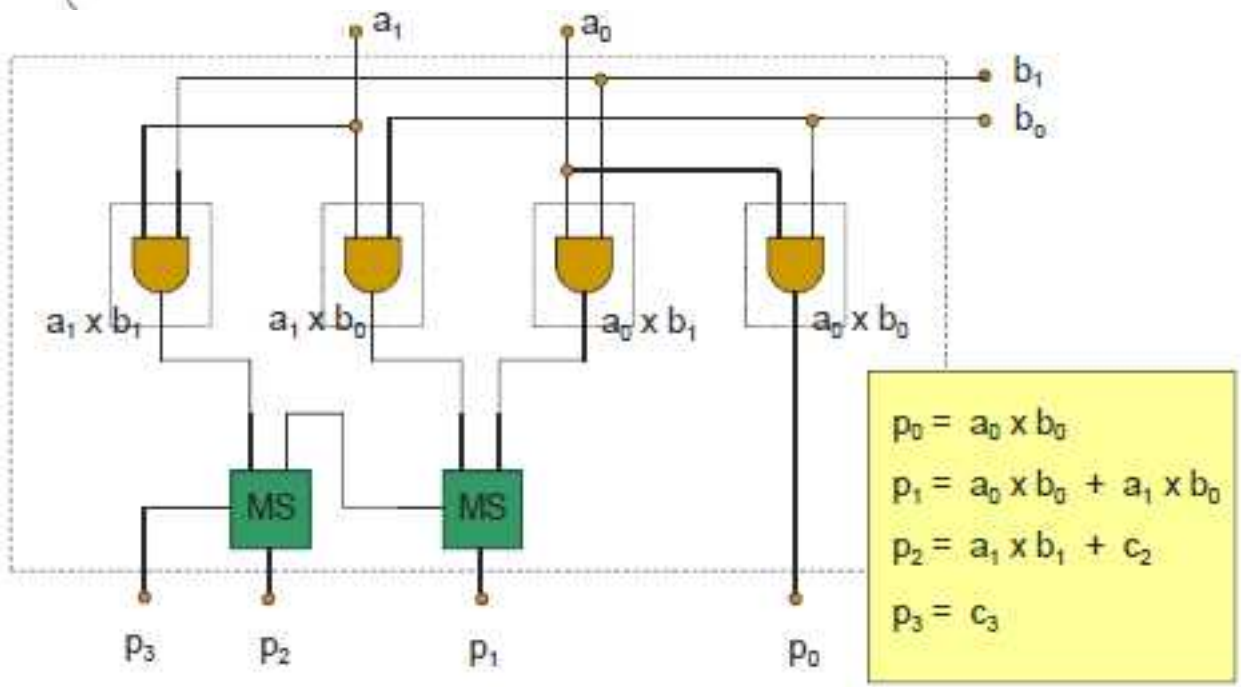


Analizando as regras acima, podemos perceber que a operação de multiplicação é o mesmo que a operação AND.

Multiplicadores

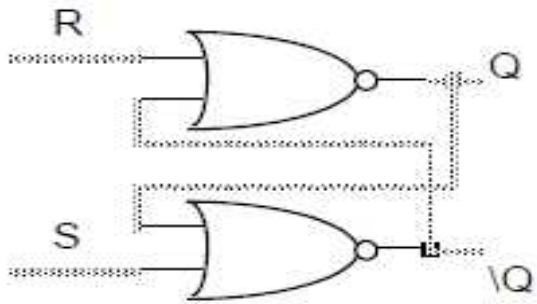


$$\begin{cases} p_0 = a_0 \times b_0 \\ p_1 = a_0 \times b_1 + a_1 \times b_0 \\ p_2 = a_1 \times b_1 + c_2 \\ p_3 = c_3 \end{cases}$$



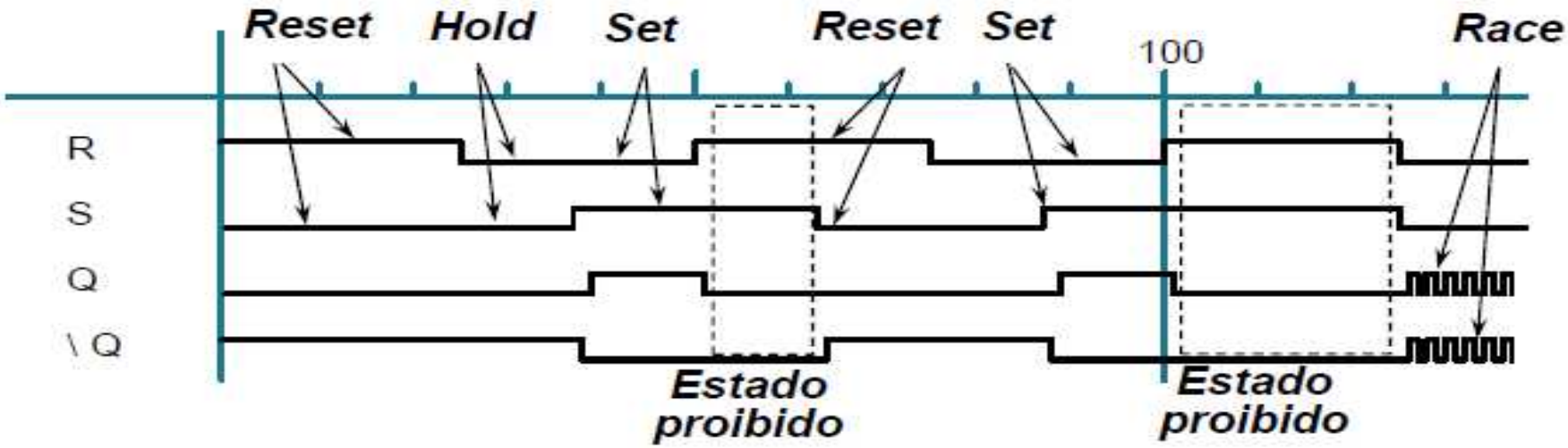
Latches e Flip-Flops

☑ *Latch RS*



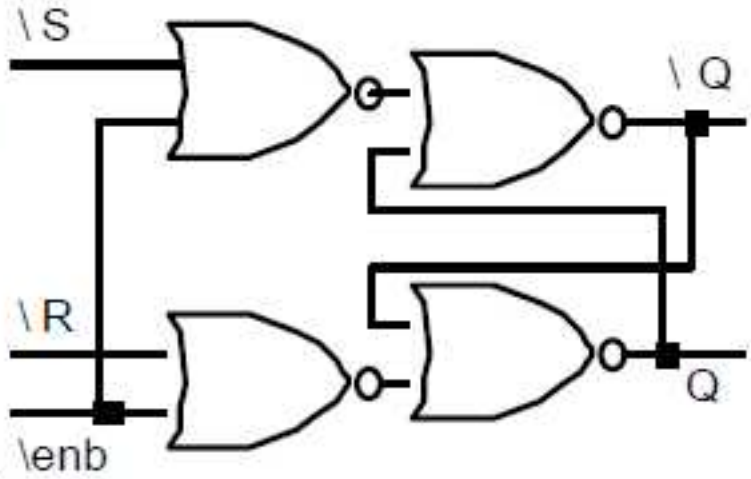
S	R	Q
0	0	Mantém Q
0	1	0
1	0	1
1	1	$\overline{Q}$ (não usado)

Diagrama de tempo do latch RS



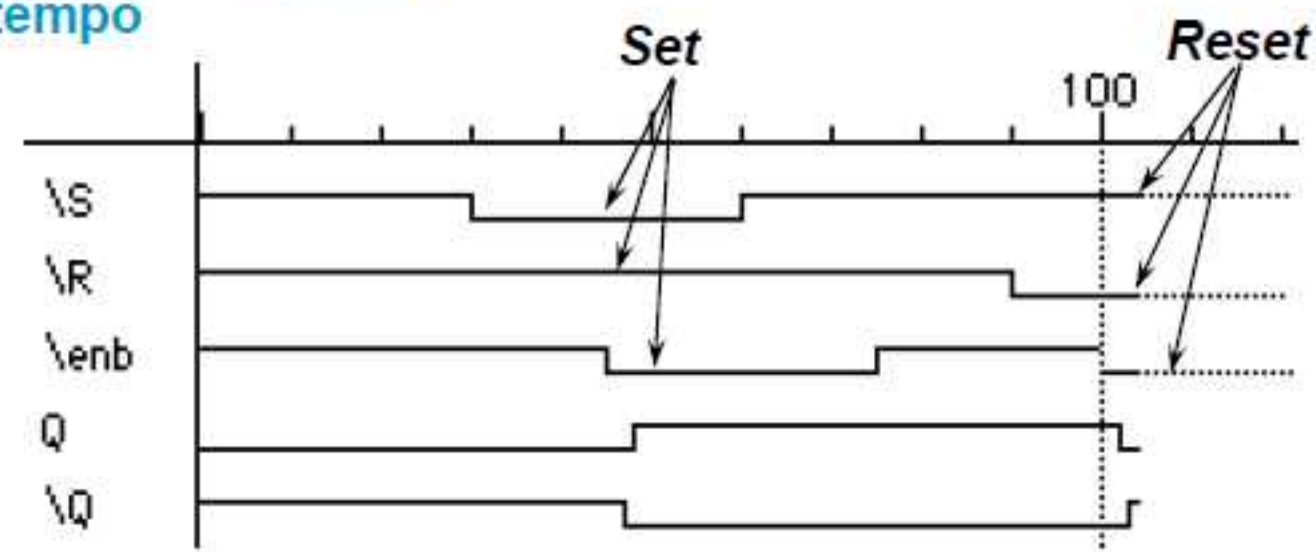
☑ *Latch RS - sensível a nível*

- ☑ ***Latches sensíveis a nível mostram continuamente suas entradas enquanto são habilitados (enb = 0)***



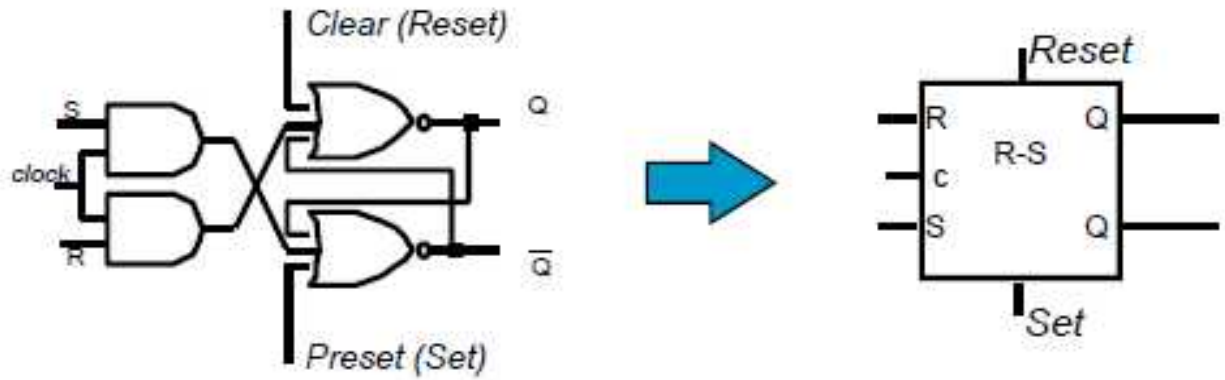
Enable

Diagrama de tempo



Latches

- Clear e Preset



Clear (Reset)	Preset (Set)	Q	Q'
0	0	normal	normal
1	0	0	1
0	1	1	0
1	1	Não usado	Não usado

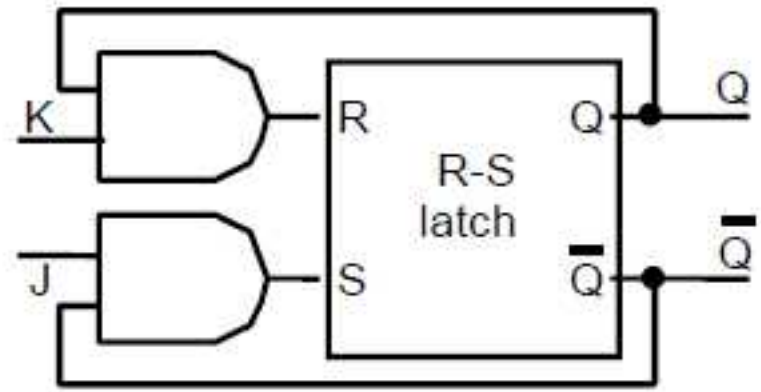
⇐ *com clock = ´0′

⇐ *com clock = ´0′

Latch JK

Como eliminar o estado proibido dos Latches tipo RS?

Usar uma re-alimentação para garantir que R e S nunca são "1".



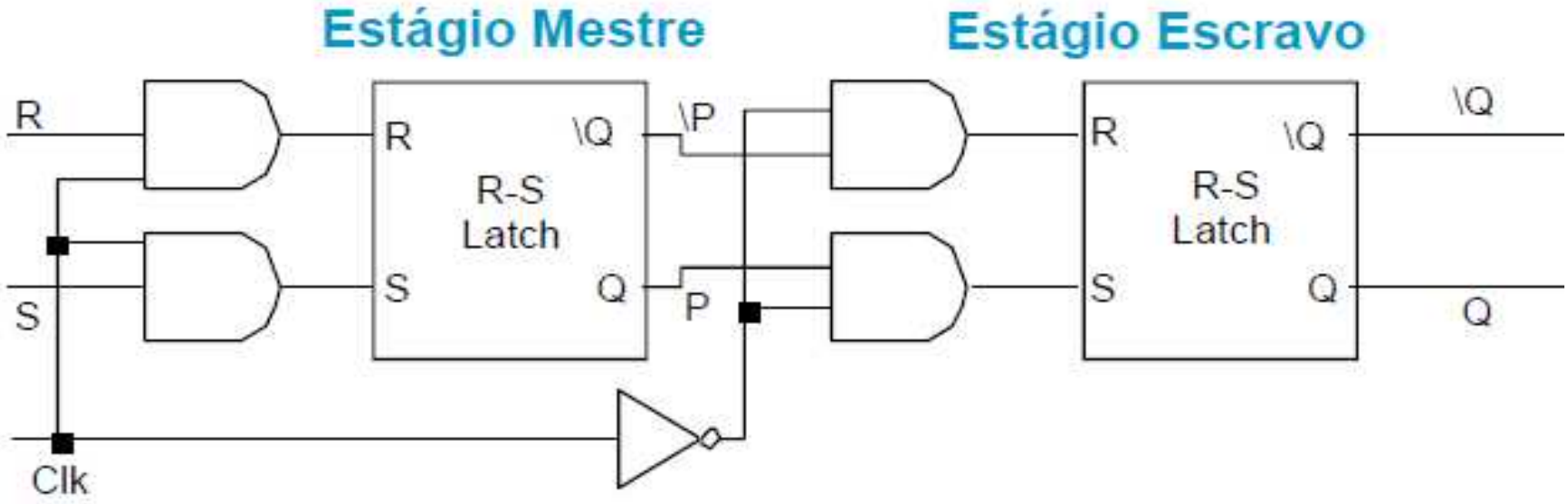
	Est. Pres.	Pró. Estado	
J(T)	K(t)	Q(t)	Q(t+Δ)	
0	0	0	0	HOLD
0	0	1	1	
0	1	0	0	RESET
0	1	1	0	
1	0	0	1	SET
1	0	1	1	
1	1	0	1	TOGGLE
1	1	1	0	

Equação de próximo estado

$$Q+ = Q\,\overline{K} + \overline{Q}\,J$$

Quando J e K são iguais a "1" a saída é invertida (Toggle)

☑ *Flip-Flop mestre-Escravo*



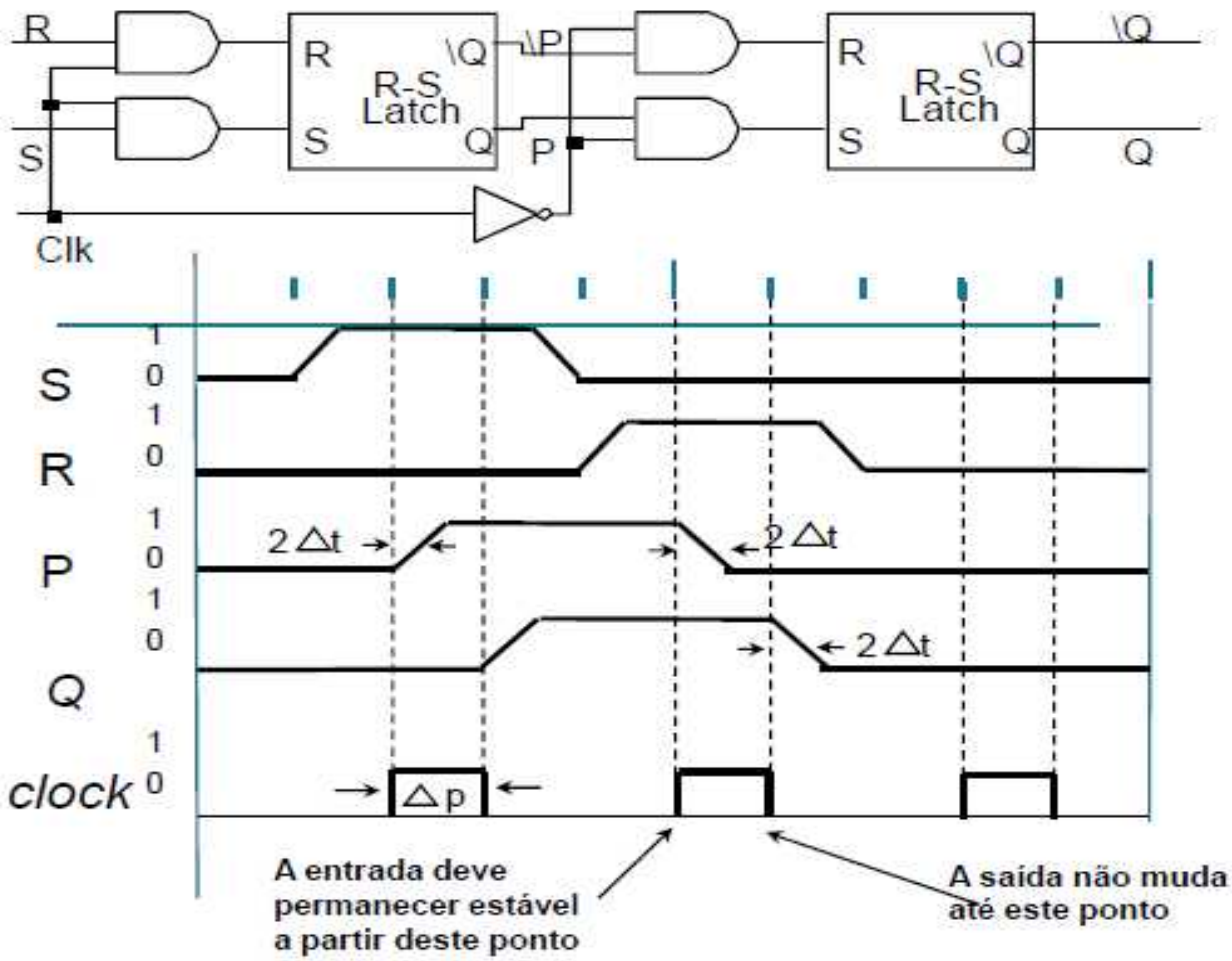
☑ Entrada disponível no latch Mestre enquanto o relógio está alto.

☑ Observe que o estágio Escravo está bloqueado (relógio está baixo).

☑ Saída disponível do latch Escravo quando o relógio for para nível lógico baixo.Relógio liberado para o estágio escravo.

☑ Observe que o estágio Mestre está bloqueado (relógio está baixo).

Flip-Flops

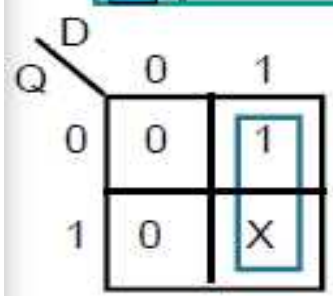

R
S
Clk
R
S
R-S
Latch
\Q
Q
\P
P
R
S
R-S
Latch
\Q
Q
\Q
Q
S
R
P
Q
clock
1
0
2 Δt
2 Δt
2 Δt
Δp
A entrada deve
permanecer estável
a partir deste ponto
A saída não muda
até este ponto

Flip-Flops Tipo D

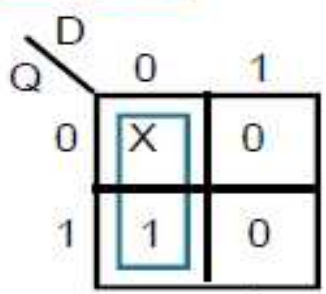
- **Características**
  - Flip-Flop tipo D construído a partir de um Flip-Flop tipo RS
  - A saída recebe a entrada
  - Equação de próximo estado:

Estado presente *Próximo estado*

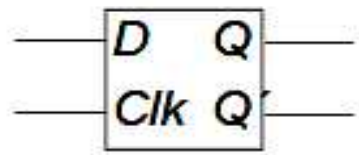
Q	Q+
0	0
0	1
1	1
1	0

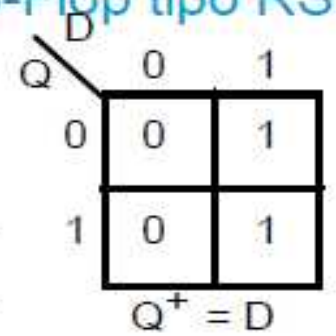


S=D



$R=\overline{D}$

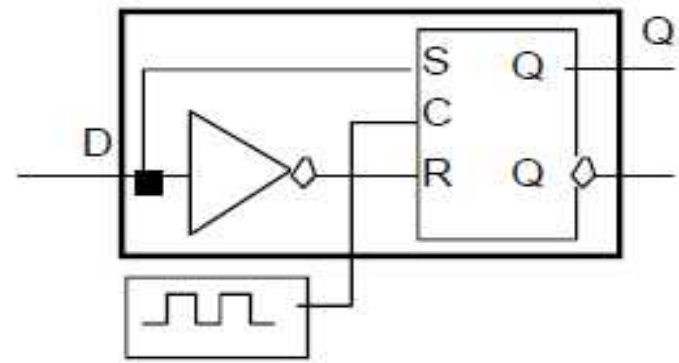




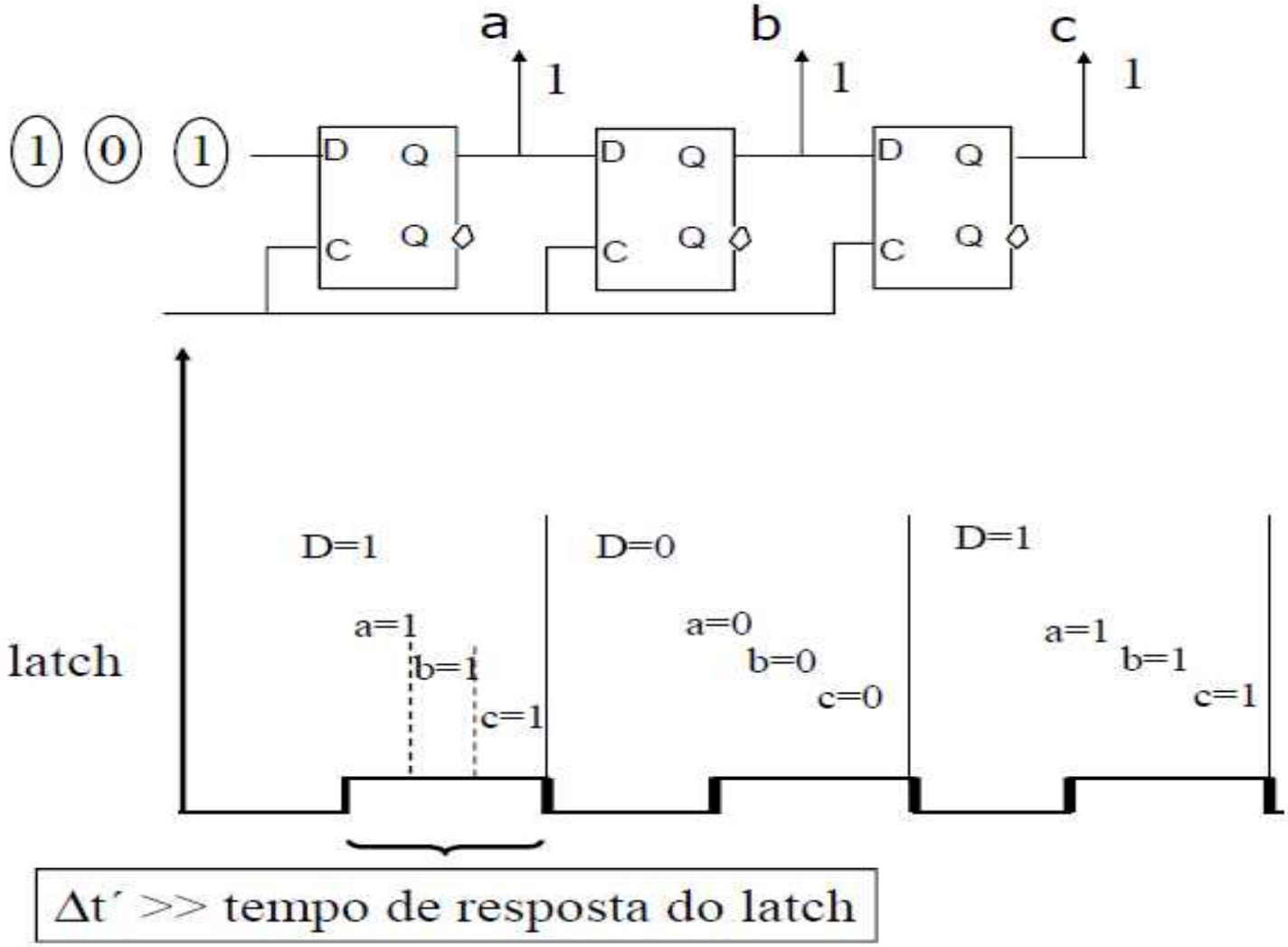
Equação de próximo estado

$Q+ = D(t)$

Flip-Flop D implementado a partir de Flip-Flop tipo RS

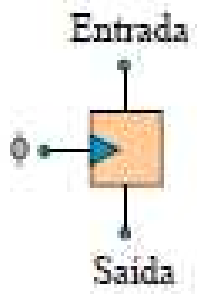


Shift Register

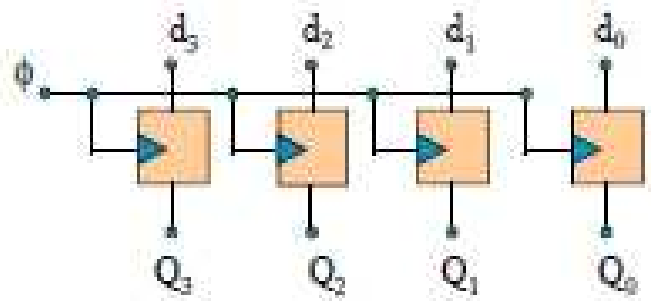


Registradores

Um registrador é um elemento lógico utilizado para armazenar uma palavra binária de n-bits.



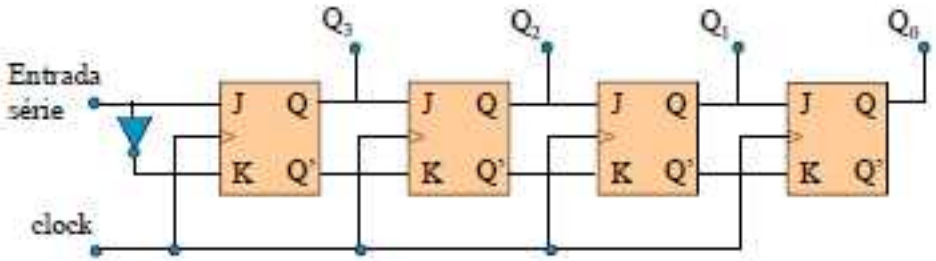
Célula individual

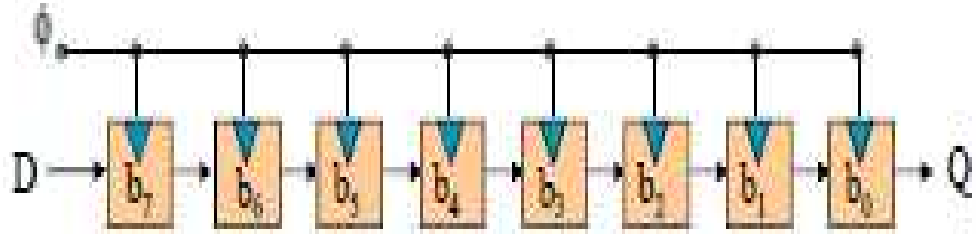


Registrador de 4 bits

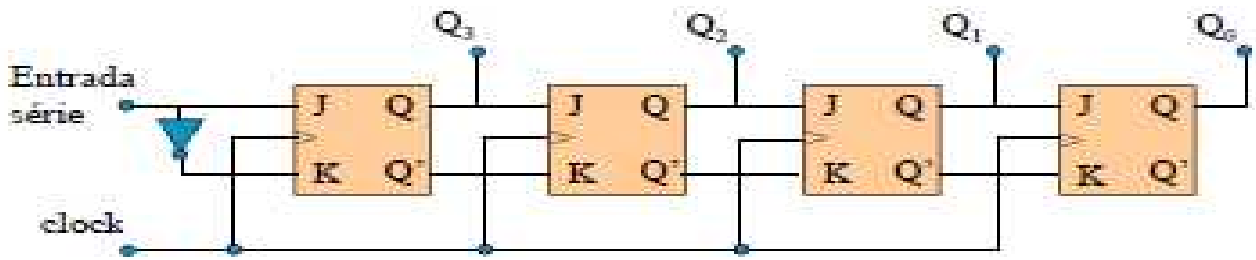
Registradores Deslocamento

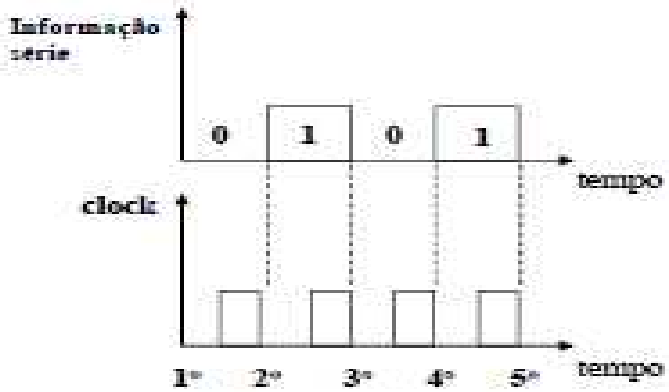
Um registrador de deslocamento é projetado para mover bits para as células vizinhas,enquanto houver pulsos de clock.
Como todas as células são controladas pelo mesmo sinal de clock f , todas elas são carregadas ao mesmo tempo





Conversor Série-Paralelo



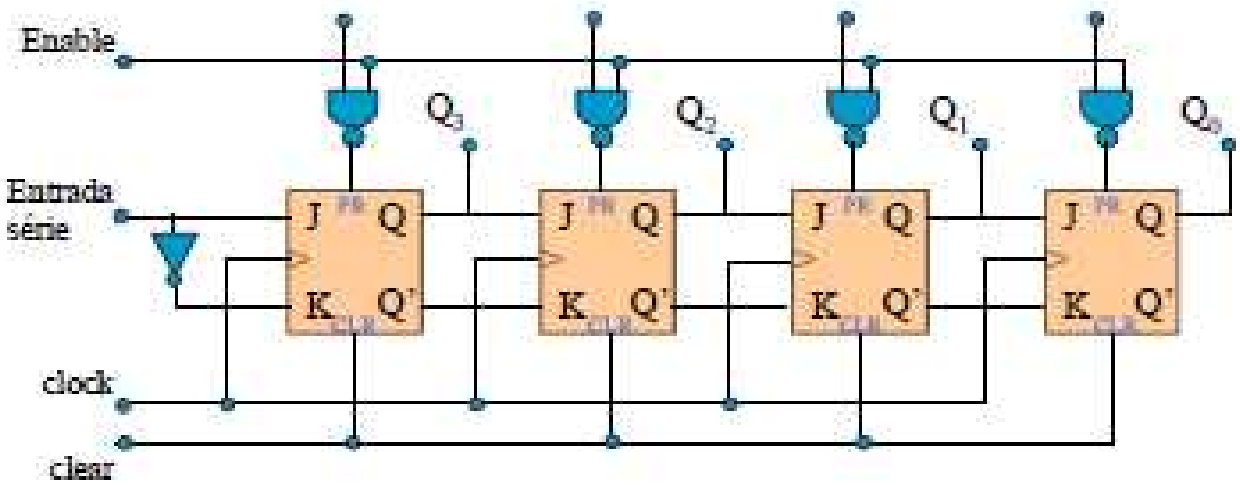


Informação	descidas do clock	Q_3	Q_2	Q_1	Q_0
$I_0 = 0$	1° pulso	0	0	0	0
$I_1 = 1$	2° pulso	1	0	0	0
$I_2 = 0$	3° pulso	0	1	0	0
$I_3 = 1$	4° pulso	1	0	1	0

Obs: A cada descida do clock as informações são deslocadas para o próximo flip-flop

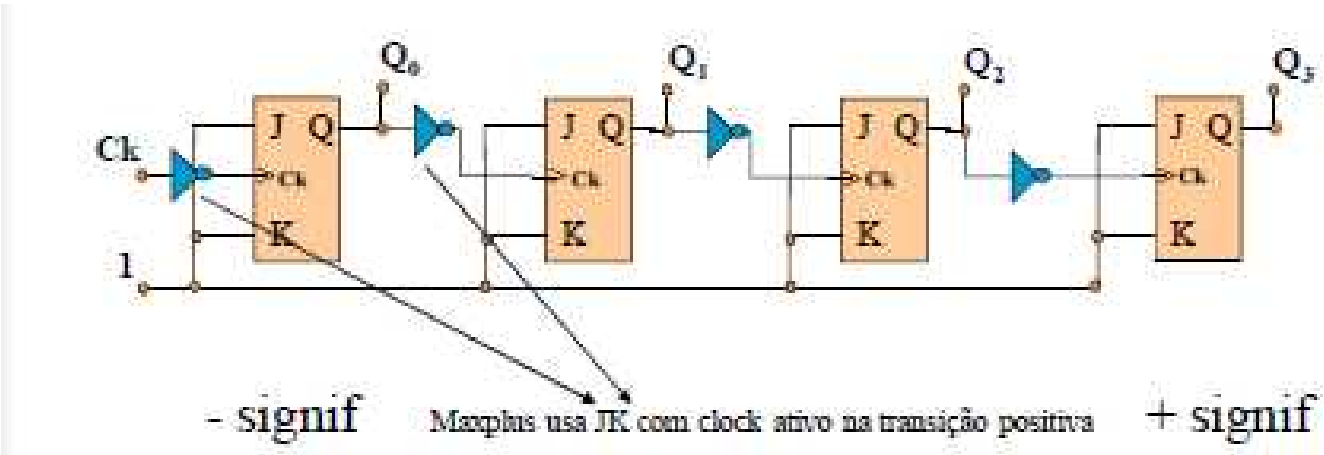
Conversor Paralelo-Série

- Podemos realizar o processo inverso, ou seja, entrar com uma palavra com n bits e “retirar” bit a bit com pulsos de clock.
- Para isto utilizamos as funções Preset e Clear dos flipflops JK mestre-escravo. O circuito utilizado está mostrado abaixo:



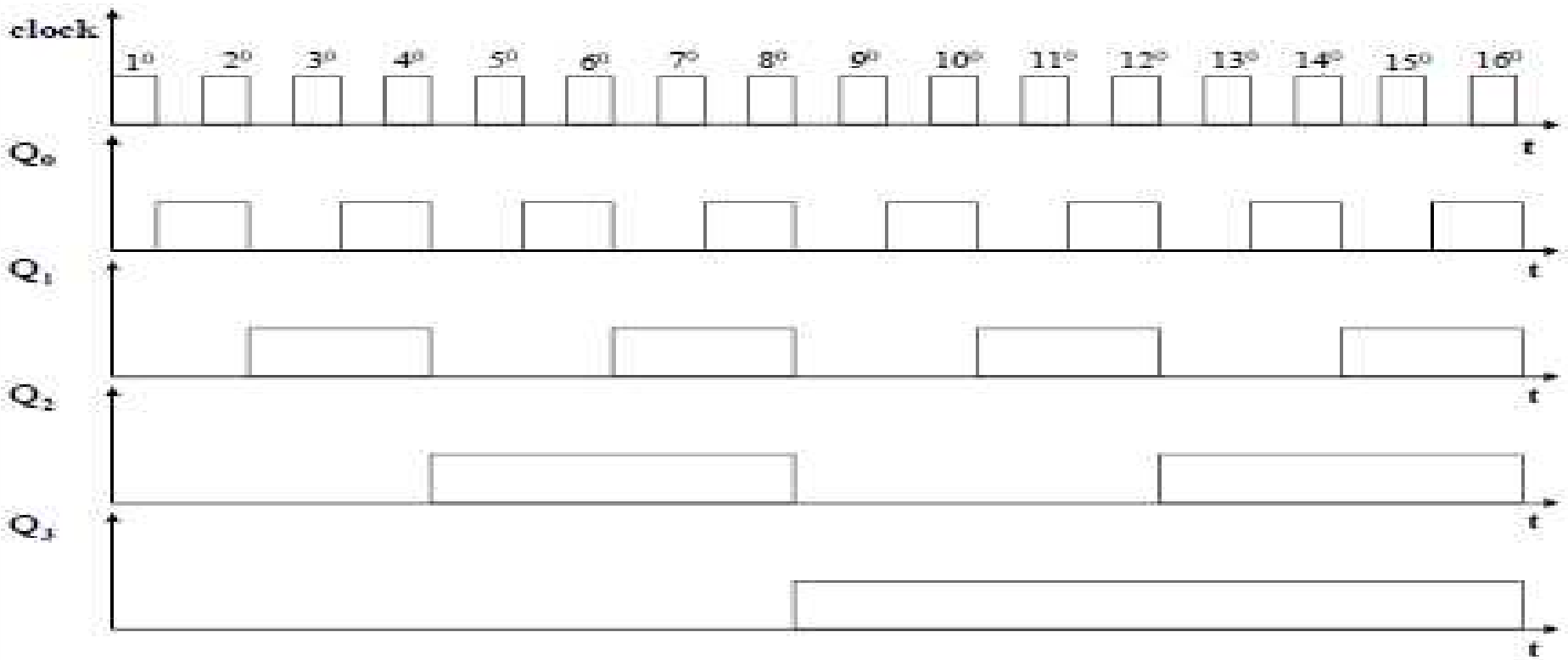
Contadores

São circuitos digitais que variam seus estados, sob comando de um clock, de acordo com uma sequência determinada. São utilizadas para:contagem,geração de palavras,divisão de frequência,medição de frequência e tempo,geração de formas de onda conversão de analógico para digital



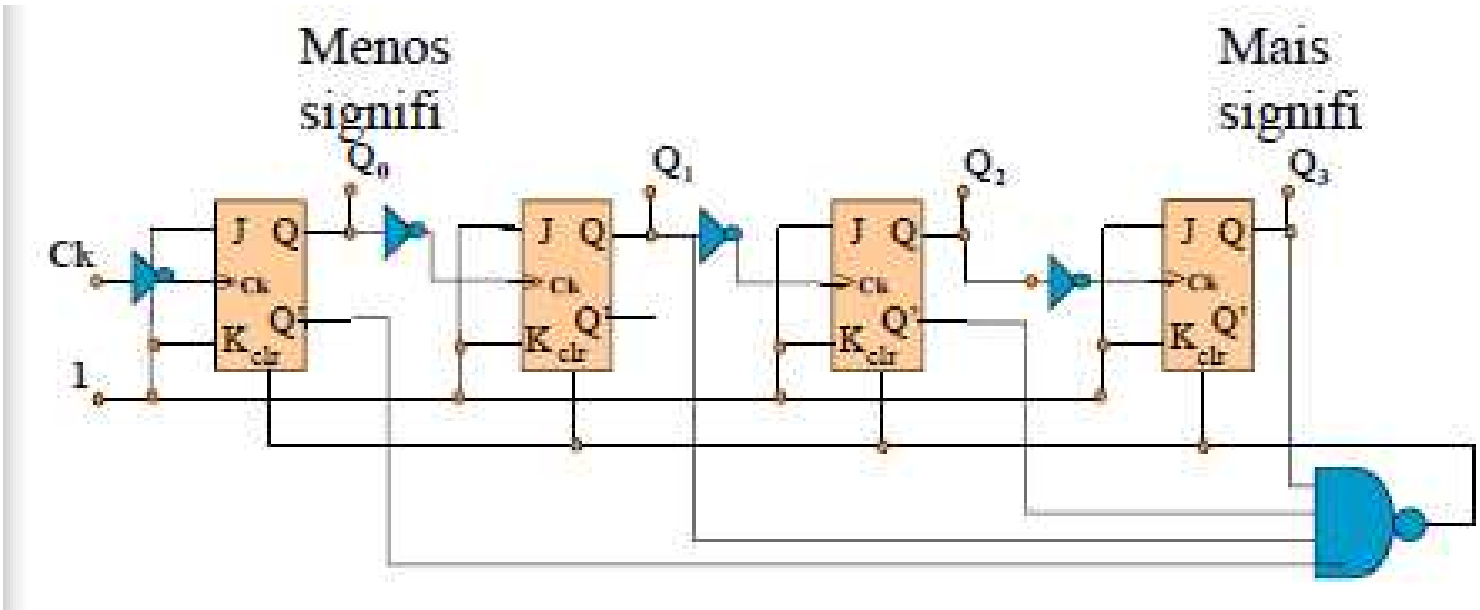
Contadores

Contadores de Pulsos



Contadores base 10

Note que o contador de década pode ser generalizado para contar até qualquer valor de n. Basta que façamos o circuito de realimentação do clear ficar ativo para o novo limite de contagem.



Referências

- http://pt.wikipedia.org/wiki/George_Boole
- http://www.cin.ufpe.br/~agsf/Sistemas_Digitais.htm