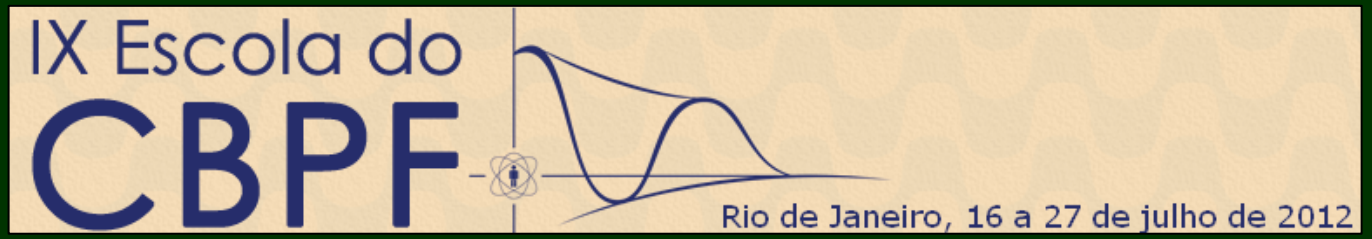


G15 Eletrônica Digital para Instrumentação

Prof.: Herman P. Lima Jr (hlima@cbpf.br)

Monitor: Rafael Gama

Centro Brasileiro de Pesquisas Físicas
Ministério da Ciência, Tecnologia e Inovação (MCTI)

Organização do curso

- ***Introdução à Eletrônica Digital***
  - ✓ analógico *vs* digital
  - ✓ representação binária
  - ✓ simplificação de circuitos
  - ✓ portas lógicas
  - ✓ flip-flops

- ***Elementos Digitais Clássicos***
  - ✓ combinacionais e sequenciais
  - ✓ somadores, contadores, codificadores e decodificadores
  - ✓ multiplexadores e demultiplexadores
  - ✓ comparadores

- ***Linguagem Descritiva de Hardware (VHDL)***

- ***Laboratório → projeto e simulação***

Analógico vs Digital

• **Circuitos digitais** utilizam variáveis digitalizadas que só podem assumir um número finito de valores distintos (ex: números binários).

Ex: computadores, câmeras digitais, CD/DVD player, DSP.

• **Circuitos analógicos** utilizam variáveis contínuas que podem assumir um número infinito de valores possíveis (ex: números reais).

Ex: amplificadores de áudio, fontes de tensão, automação industrial (PID).

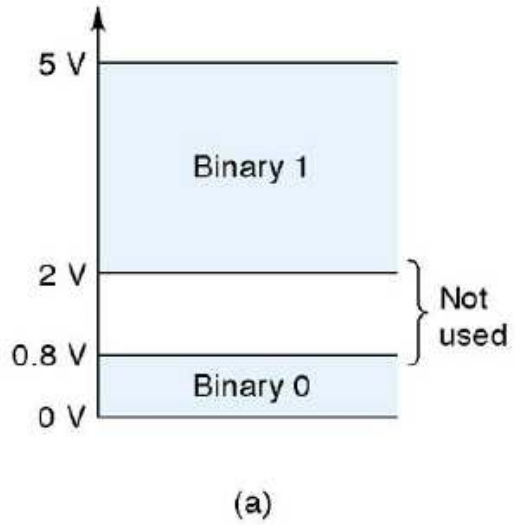
Destaques dos circuitos digitais

- Geralmente mais fáceis para projetar que os analógicos
- Armazenamento de informação mais flexível (*latch*)
- Funcionalidade programável (CPLDs, FPGAs)
- Teoria matemática bem desenvolvida
- Imunidade a ruído
- Circuitos integrados compactos
- Tecnologias avançadas de implementação e em contínuo/rápido desenvolvimento
- Confiabilidade de funcionamento

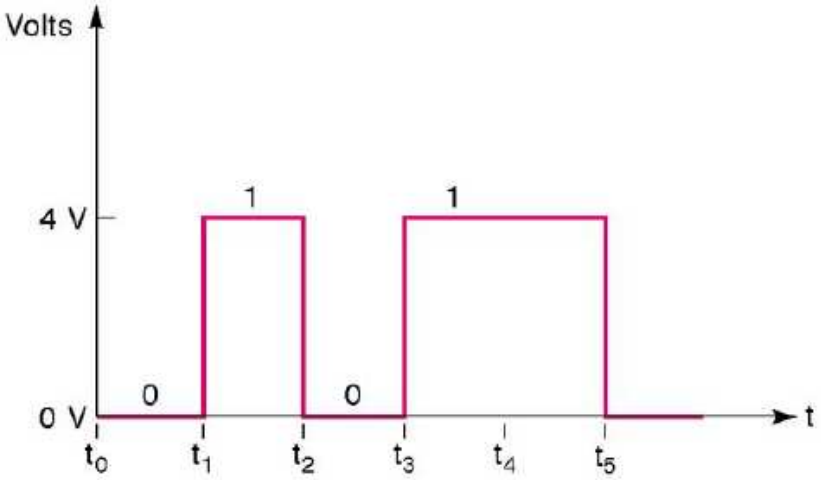
IMPORTANTE: circuitos digitais também possuem características analógicas pois são construídos a partir de componentes analógicos (transístores, diodos e resistores).

Representando quantidades binárias

- A informação binária é representada por tensões (ou correntes) em um circuito.
- O valor exato da tensão não é importante em circuitos digitais.
- A taxa do fluxo de informação digital geralmente é dada em ‘bits per second’ [bps].



(a)



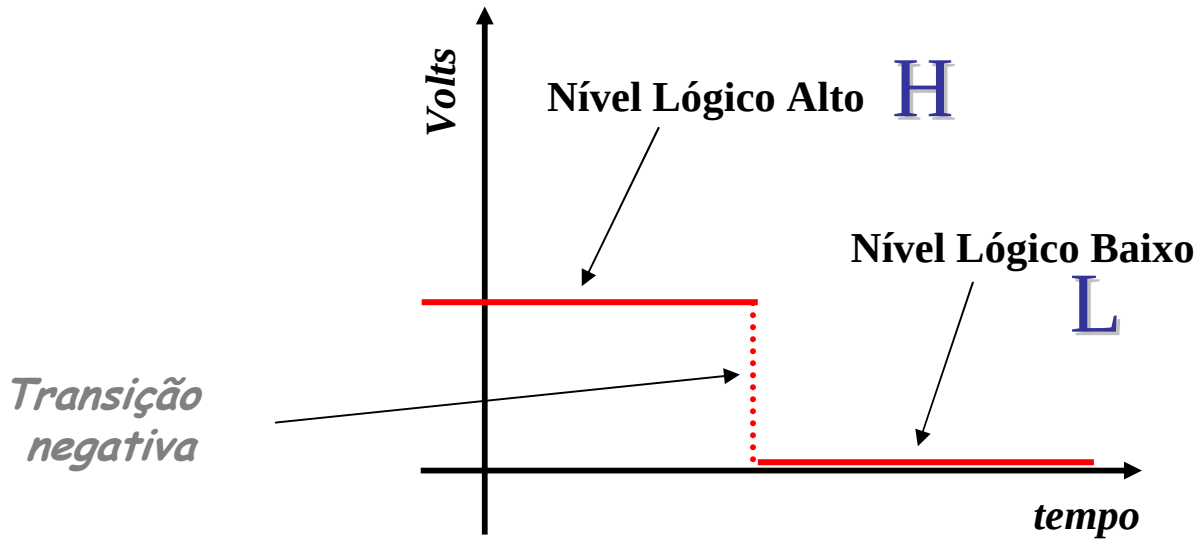
(b)

Níveis lógicos

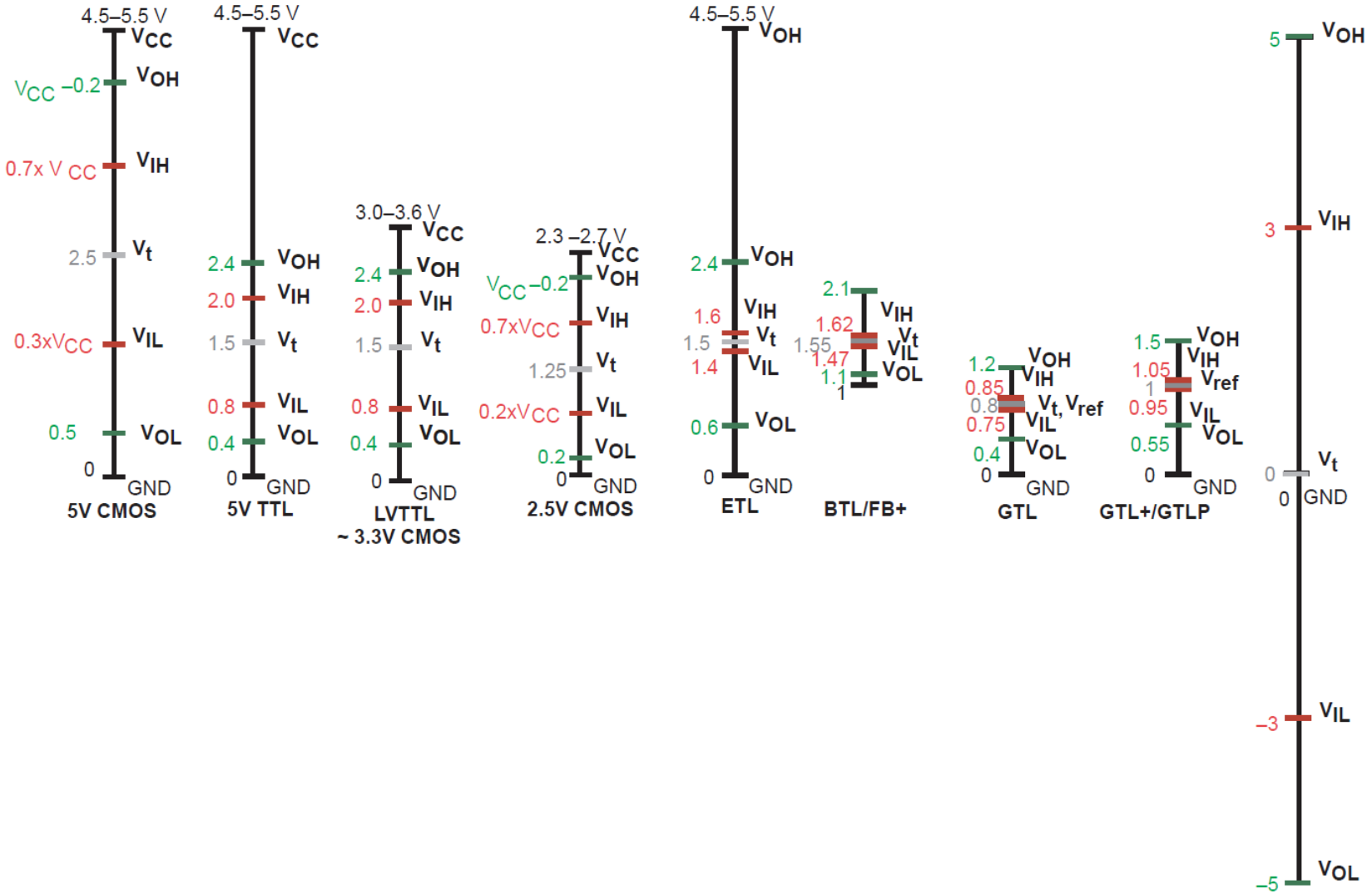
Lógica TTL (*Transistor Transistor Logic*)

Nível Lógico* 0 (False, Low) - 0 *Volts

Nível Lógico* 1 (True, High) - 5 *Volts



Níveis lógicos



Circuitos digitais

Circuitos digitais são projetados para:

- Aceitar tensões de entrada dentro das faixas 0 (*low*) e 1 (*high*)
- Processar sinais de entrada de forma previsível (definida no projeto)
- Produzir tensões de saída dentro das faixas de 0 e 1.

Sistemas numéricos e códigos

• Sistemas digitais são construídos a partir de circuitos que processam dígitos binários, entretanto dígitos binários não são objetos com os quais lidamos no mundo real.

• Como representar números do mundo real, letras, audio, video e outras coisas de interesse por 0’s e 1’s?

Sistemas numéricos posicionais

• Um número é representado por um conjunto de dígitos onde cada posição tem um peso associado.

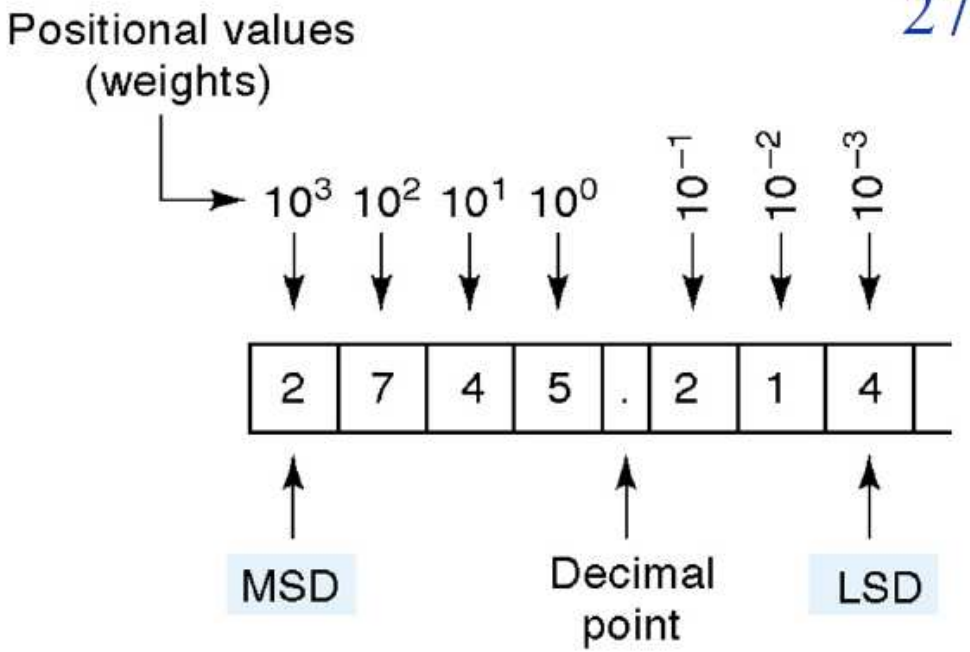
• Em um sistema base-*r*, o dígito na posição i tem peso r^i e cada dígito pode ter valor 0, 1, ..., *r*-1.

• O número base-*r* $d_{p-1}\ d_{p-2}\ ...\ d_1\ d_0\ .\ d_{-1}\ d_{-2}\ ...\ d_{-n}$ tem representação decimal (base-10):

$$D = \sum_{i=-n}^{p-1} d_i\ r^i$$

Números *decimais*

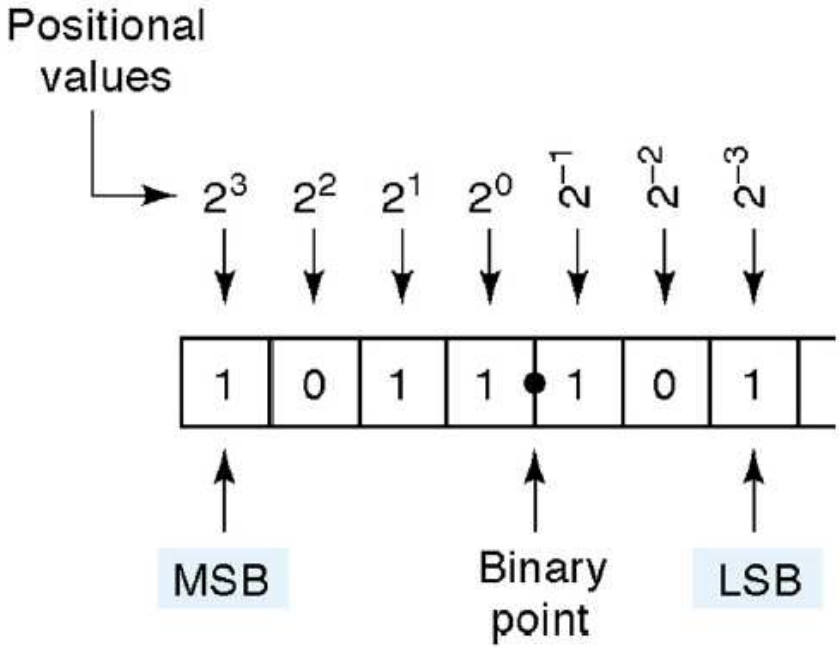
Composto de 10 símbolos (base-10): dígitos 0 a 9.



$$2745.214_{10} = \underset{d}{(2 \times 10^{+3})} + \underset{r}{(7 \times 10^{+2})} + (4 \times 10^{+1}) + (5 \times 10^{o}) + (2 \times 10^{-1}) + (1 \times 10^{-2}) + (4 \times 10^{-3})$$

Números *binários*

- Utilizam somente dois símbolos (0 e 1) (base-2).
- São os mais importantes para sistemas digitais.
- Para um número binário de N bits, temos números até $(2^N\text{-}1)_{10}$



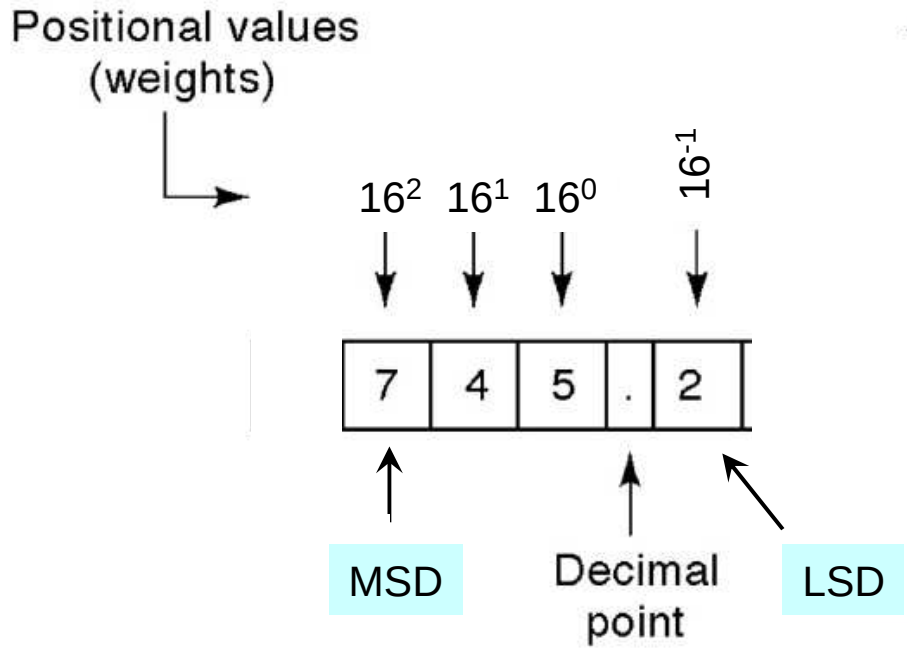
$$1011.101_2 = (1 \times 2^{+3}) + (0 \times 2^{+2}) + (1 \times 2^{+1}) + (1 \times 2^{o}) + (1 \times 2^{-1}) + (0 \times 2^{-2}) + (1 \times 2^{-3})$$

$$= 11.625$$

Números *hexadecimais*

• Compostos de 16 símbolos: os dígitos de 0 a 9 e as letras A, B, C, D, E e F (base-16).

• As posições dos dígitos recebem pesos como potências de 16, ao invés de 10, como no caso decimal.



$$745.2_{16} = (7 \times 16^2) + (4 \times 16^1) + (5 \times 16^0) + (2 \times 16^{-1})$$

$$= 1861.125_{10}$$

Código BCD (*Binary-Coded-Decimal*)

• Um código pode ser definido como um conjunto de *strings* de bits, onde cada *string* representa um número, letra ou outro símbolo qualquer.

• No código BCD, cada dígito do número decimal é codificado no binário correspondente.

• Exemplo: 943_{10} em BCD

9	4	3	
1001	0100	0011	(BCD)

Decimal	Binary	Octal	Hexadecimal	BCD
0	0	0	0	0000
1	1	1	1	0001
2	10	2	2	0010
3	11	3	3	0011
4	100	4	4	0100
5	101	5	5	0101
6	110	6	6	0110
7	111	7	7	0111
8	1000	10	8	1000
9	1001	11	9	1001
10	1010	12	A	0001 0000
11	1011	13	B	0001 0001
12	1100	14	C	0001 0010
13	1101	15	D	0001 0011
14	1110	16	E	0001 0100
15	1111	17	F	0001 0101

Códigos Alfanuméricos

• A maior parte da informação processada por computadores não é numérica (letras, sinais de pontuação e caracteres especiais).

• O código ASCII (*American Standard Code for Information Interchange*) é um código alfanumérico de 7 bits com 128 caracteres diferentes (ver tabela no próximo slide).

• Exemplo: a string de bits

1000001 1010011 1000011 1001001 1001001

é o código ASCII para “ASCII”.

Tabela ASCII

Dec	Hx	Oct	Char	Dec	Hx	Oct	Html	Chr	Dec	Hx	Oct	Html	Chr	Dec	Hx	Oct	Html	Chr
0	0	000	NUL (null)	32	20	040		Space	64	40	100	@	@	96	60	140	`	`
1	1	001	SOH (start of heading)	33	21	041	!	!	65	41	101	A	A	97	61	141	a	a
2	2	002	STX (start of text)	34	22	042	"	"	66	42	102	B	B	98	62	142	b	b
3	3	003	ETX (end of text)	35	23	043	#	#	67	43	103	C	C	99	63	143	c	c
4	4	004	EOT (end of transmission)	36	24	044	$	$	68	44	104	D	D	100	64	144	d	d
5	5	005	ENQ (enquiry)	37	25	045	%	%	69	45	105	E	E	101	65	145	e	e
6	6	006	ACK (acknowledge)	38	26	046	&	&	70	46	106	F	F	102	66	146	f	f
7	7	007	BEL (bell)	39	27	047	'	'	71	47	107	G	G	103	67	147	g	g
8	8	010	BS (backspace)	40	28	050	(	(	72	48	110	H	H	104	68	150	h	h
9	9	011	TAB (horizontal tab)	41	29	051	)	)	73	49	111	I	I	105	69	151	i	i
10	A	012	LF (NL line feed, new line)	42	2A	052	*	*	74	4A	112	J	J	106	6A	152	j	j
11	B	013	VT (vertical tab)	43	2B	053	+	+	75	4B	113	K	K	107	6B	153	k	k
12	C	014	FF (NP form feed, new page)	44	2C	054	,	,	76	4C	114	L	L	108	6C	154	l	l
13	D	015	CR (carriage return)	45	2D	055	-	-	77	4D	115	M	M	109	6D	155	m	m
14	E	016	SO (shift out)	46	2E	056	.	.	78	4E	116	N	N	110	6E	156	n	n
15	F	017	SI (shift in)	47	2F	057	/	/	79	4F	117	O	O	111	6F	157	o	o
16	10	020	DLE (data link escape)	48	30	060	0	0	80	50	120	P	P	112	70	160	p	p
17	11	021	DC1 (device control 1)	49	31	061	1	1	81	51	121	Q	Q	113	71	161	q	q
18	12	022	DC2 (device control 2)	50	32	062	2	2	82	52	122	R	R	114	72	162	r	r
19	13	023	DC3 (device control 3)	51	33	063	3	3	83	53	123	S	S	115	73	163	s	s
20	14	024	DC4 (device control 4)	52	34	064	4	4	84	54	124	T	T	116	74	164	t	t
21	15	025	NAK (negative acknowledge)	53	35	065	5	5	85	55	125	U	U	117	75	165	u	u
22	16	026	SYN (synchronous idle)	54	36	066	6	6	86	56	126	V	V	118	76	166	v	v
23	17	027	ETB (end of trans. block)	55	37	067	7	7	87	57	127	W	W	119	77	167	w	w
24	18	030	CAN (cancel)	56	38	070	8	8	88	58	130	X	X	120	78	170	x	x
25	19	031	EM (end of medium)	57	39	071	9	9	89	59	131	Y	Y	121	79	171	y	y
26	1A	032	SUB (substitute)	58	3A	072	:	:	90	5A	132	Z	Z	122	7A	172	z	z
27	1B	033	ESC (escape)	59	3B	073	;	;	91	5B	133	[	[	123	7B	173	{	{
28	1C	034	FS (file separator)	60	3C	074	<	<	92	5C	134	\	\	124	7C	174			|
29	1D	035	GS (group separator)	61	3D	075	=	=	93	5D	135	]	]	125	7D	175	}	}
30	1E	036	RS (record separator)	62	3E	076	>	>	94	5E	136	^	^	126	7E	176	~	~
31	1F	037	US (unit separator)	63	3F	077	?	?	95	5F	137	_	_	127	7F	177	&#127;	DEL

Código de verificação por paridade

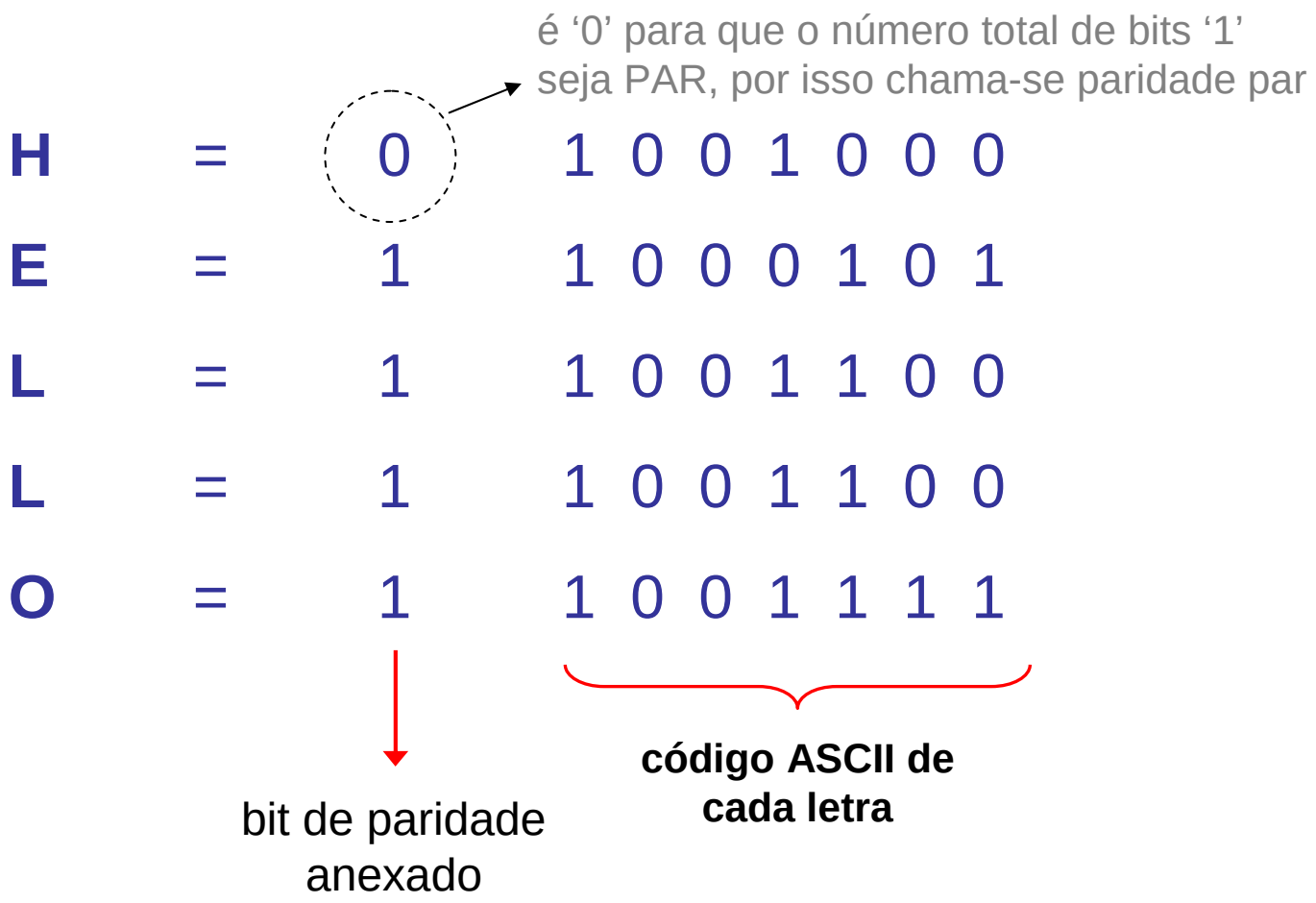
• Quando dados digitais são transmitidos de um local para outro, sempre é possível haver o recebimento de bits com erros.

• Diversos sistemas digitais utilizam códigos para detectar, e até corrigir, erros de transmissão.

• Um código muito simples para detecção de erro consiste em adicionar um bit ao caracter transmitido de tal forma que o número total de bits iguais a ‘1’ seja par (paridade par) ou ímpar (paridade ímpar).

• Não funciona para erros em dois bits no mesmo caractere, mas em geral a probabilidade desta ocorrência em sistemas digitais é nula.

Ex. paridade par



Números com sinal

Sinal-magnitude:

$$N_s = \{s, \underbrace{a_{n-1}, a_{n-2}, a_{n-3}, \ldots, a_1, a_0}_{\text{magnitude}}\}$$

Convenção do bit de sinal: $s = 0 \longrightarrow$ número POSITIVO
$s = 1 \longrightarrow$ número NEGATIVO

Faixa dinâmica para
números com **n+1** bits:

$$-(2^n - 1) < N < 2^n - 1$$

(situação com dois zeros)

Ex: n+1 = 3

```
+3 - 011
+2 - 010
+1 - 001
 0 - 000
 0 - 100
-1 - 101
-2 - 110
-3 - 111
```

Números com sinal

Complemento de 2:

- Quando o número for positivo (**MSB='0'**), funciona como no modo *sinal-magnitude*.
- Quando o número for negativo (**MSB='1'**), a magnitude do número deve ser encontrada através do complemento de 2.

```
+3 -  011
+2 -  010
+1 -  001
 0 -  000
-1 -  111
-2 -  110
-3 -  101
-4 -  100
```

Números positivos e zero

Números negativos

Ex: -1

111 → 000 (comp 1)

+1

Valor = 001

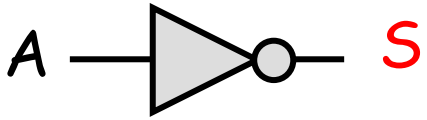
Números com sinal – 4 bits

D3 D2 D1 D0	Sinal e magnitude	Complemento a 2
0 1 1 1	+7	+7
0 1 1 0	+6	+6
0 1 0 1	+5	+5
0 1 0 0	+4	+4
0 0 1 1	+3	+3
0 0 1 0	+2	+2
0 0 0 1	+1	+1
0 0 0 0	0	0
1 0 0 0	0	-8
1 0 0 1	-1	-7
1 0 1 0	-2	-6
1 0 1 1	-3	-5
1 1 0 0	-4	-4
1 1 0 1	-5	-3
1 1 1 0	-6	-2
1 1 1 1	-7	-1

uma vantagem do comp. a 2

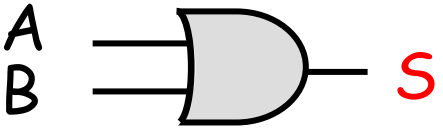
bit de sinal

Portas Lógicas



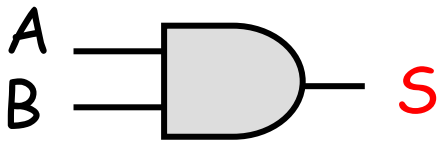
NOT

A	S
0	1
1	0



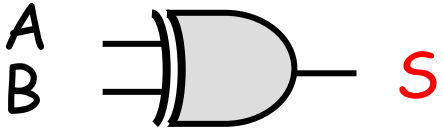
OR

A	B	S
0	0	0
0	1	1
1	0	1
1	1	1



AND

A	B	S
0	0	0
0	1	0
1	0	0
1	1	1

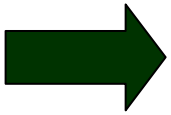


XOR

A	B	S
0	0	0
0	1	1
1	0	1
1	1	0

Projeto e Análise de Circuitos Lógicos

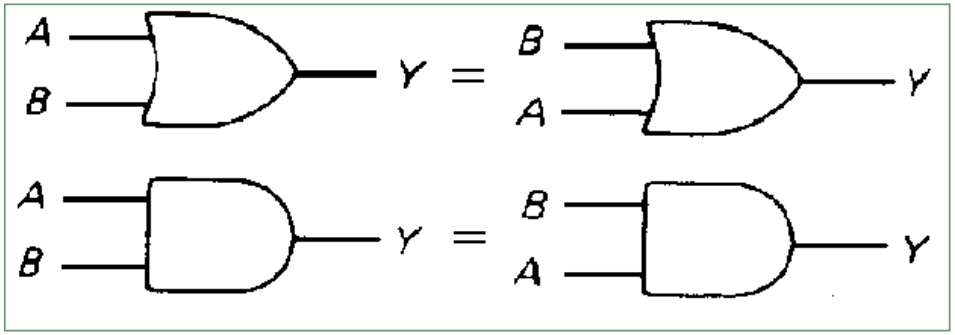
- Álgebra booleana
- Mapas de Karnaugh



simplificar circuitos lógicos

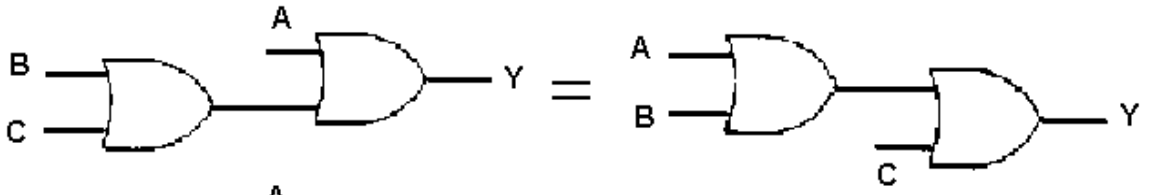
Teoremas Booleanos

Comutativa: **A+B = B+A ; AB = BA**

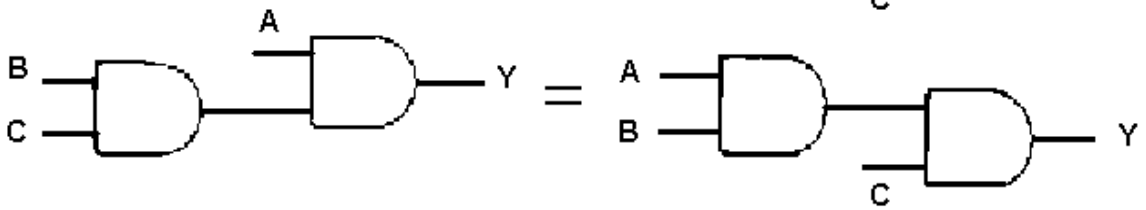


Associativa:

$A+(B+C)=(A+B)+C$ →

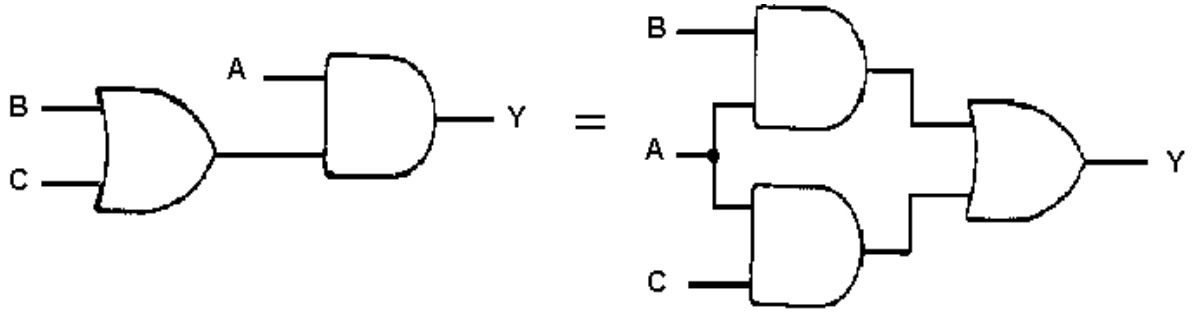


$A(BC) = (AB)C$ →



Distributiva:

$A(B+C) = AB + AC$ →



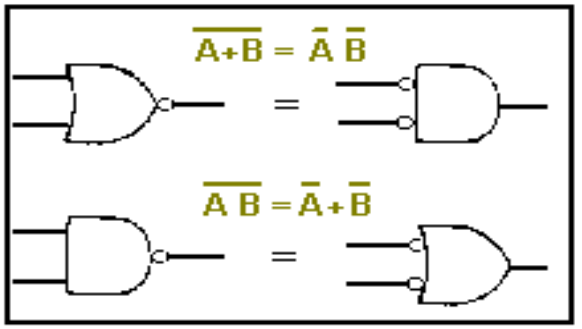
Teoremas de uma única variável

OPERAÇÕES	
OR	AND
A+0= A	A .1=A
A+A= A	A .A=A
A+1 = 1	A.0=0
$A+\bar{A}=1$	$A.\bar{A}=0$

Inversão Dupla:

$\overline{\bar{A}} = A$

Teoremas de De Morgan:



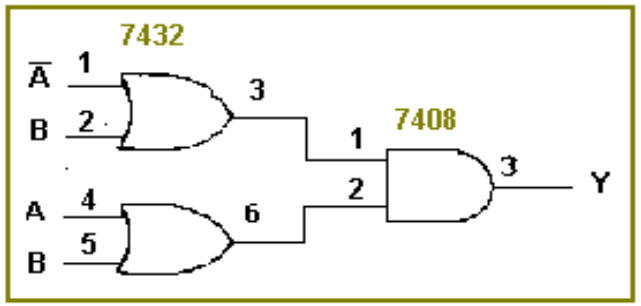
Dualidade

OR ⟷ AND; AND ⟷ OR; 0 ⟷ 1

$A + 0 = A$ ⟷ (*dual*) $A \cdot 1 = A$

Implementar circuito lógico para:

$Y = (\bar{A} + B)(A + B)$

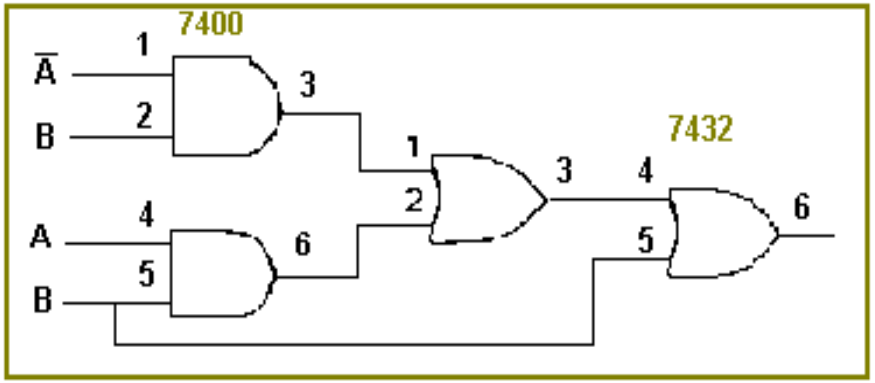


$Y = \bar{A}A + \bar{A}B + BA + BB = \bar{A}B + AB + B$

0 B

$Y = (\bar{A} + A)B + B = B$

1 B

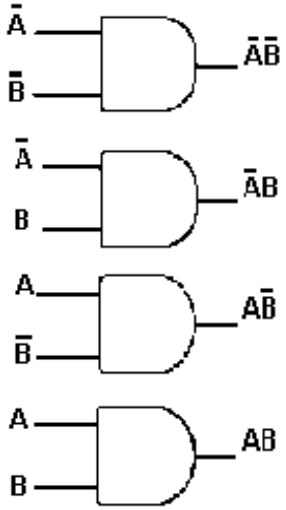


Teoremas com mais de uma variável

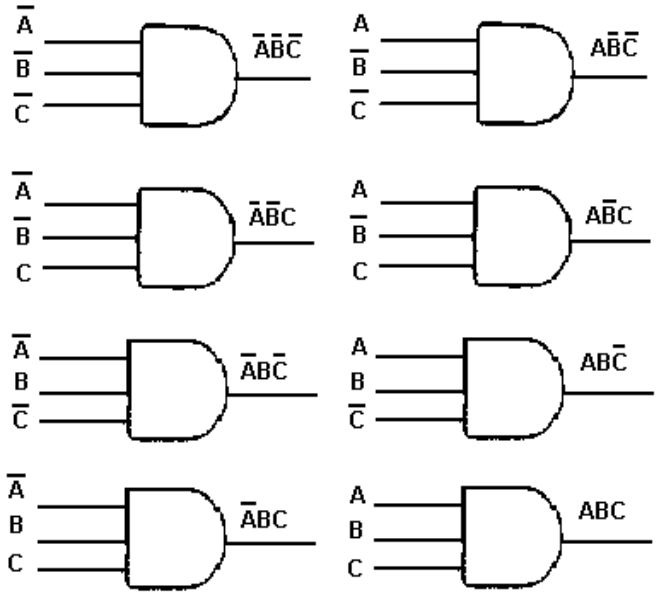
(9)	$x + y = y + x$
(10)	$x \cdot y = y \cdot x$
(11)	$x + (y + z) = (x + y) + z = x + y + z$
(12)	$x(yz) = (xy)z = xyz$
(13a)	$x(y + z) = xy + xz$
(13b)	$(w + x)(y + z) = wy + xy + wz + xz$
(14)	$x + xy = x$
(15a)	$x + \bar{x}y = x + y$
(15b)	$\bar{x} + xy = \bar{x} + y$

Método da Soma de Produtos

A	B	Produto Fundamental
0	0	$\bar{A}\bar{B}$
0	1	$\bar{A}B$
1	0	$A\bar{B}$
1	1	AB



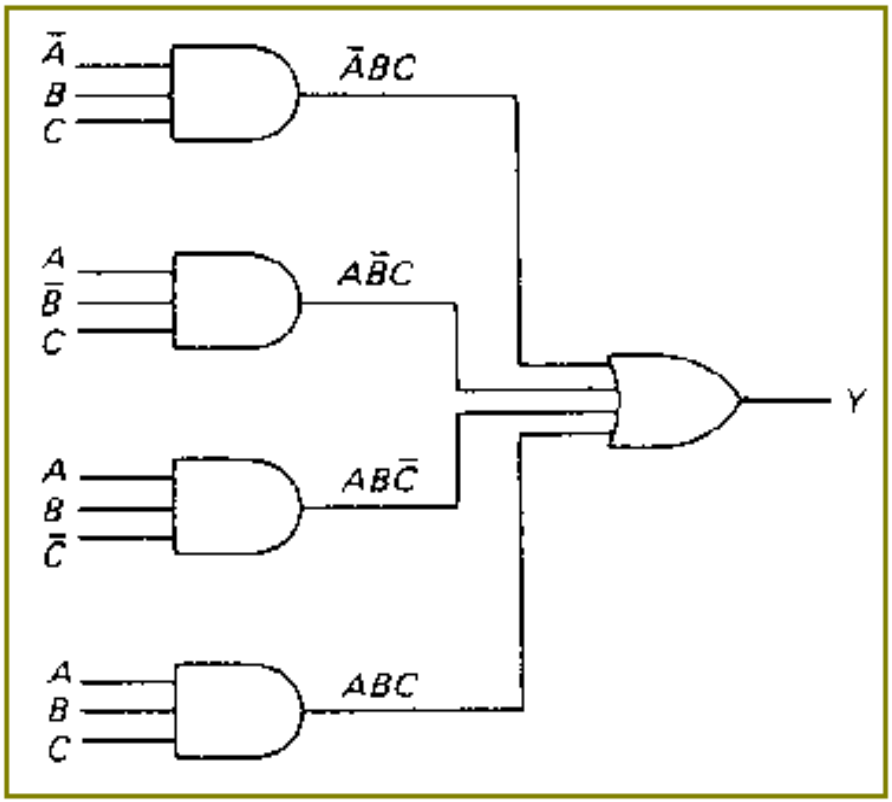
A	B	C	Produto Fundamental
0	0	0	$\bar{A}\bar{B}\bar{C}$
0	0	1	$\bar{A}\bar{B}C$
0	1	0	$\bar{A}B\bar{C}$
0	1	1	$\bar{A}BC$
1	0	0	$A\bar{B}\bar{C}$
1	0	1	$A\bar{B}C$
1	1	0	$AB\bar{C}$
1	1	1	ABC



Equação da Soma de Produtos

Ex1: dada uma tabela verdade qualquer.

A	B	C	Y	
0	0	0	0	
0	0	1	0	
0	1	0	0	
0	1	1	1	$\bar{A}BC$
1	0	0	0	
1	0	1	1	$A\bar{B}C$
1	1	0	1	$AB\bar{C}$
1	1	1	1	ABC



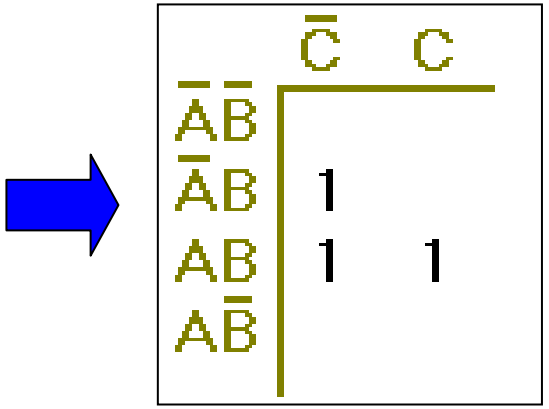
construímos a eq. da soma de produtos:

$$Y=\bar{A}BC+A\bar{B}C+AB\bar{C}+ABC$$

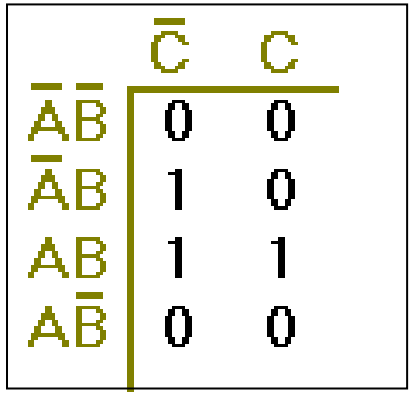
Mapa de Karnaugh

pelo método *soma de produtos*

A	B	C	Y
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	1
1	1	1	1



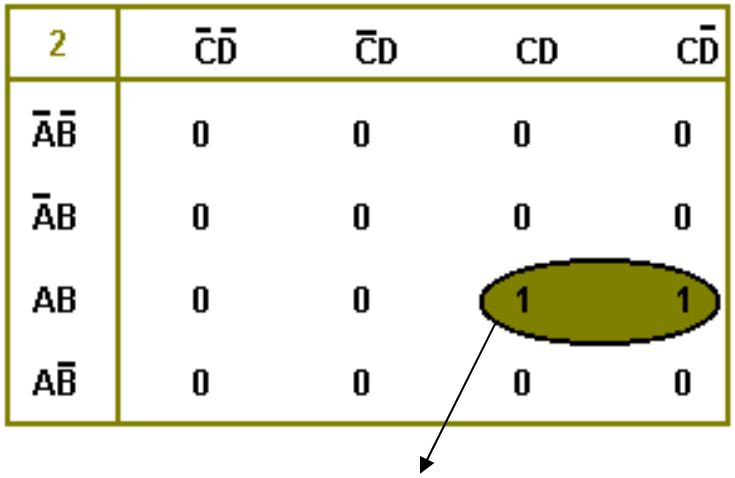
1º passo: preencher os casos de 1



2º passo: preencher o restante com 0s

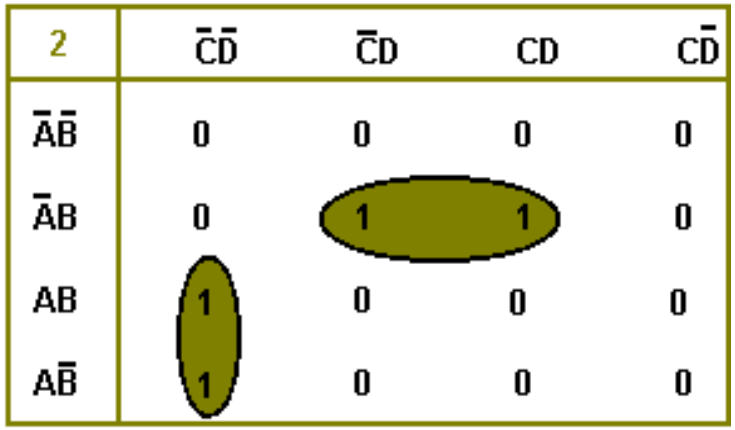
Simplificação por PARES

Elimina 1 variável



2	$\bar{C}\bar{D}$	$\bar{C}D$	CD	$C\bar{D}$
$\bar{A}\bar{B}$	0	0	0	0
$\bar{A}B$	0	0	0	0
AB	0	0	1	1
$A\bar{B}$	0	0	0	0

variável D muda de estado



2	$\bar{C}\bar{D}$	$\bar{C}D$	CD	$C\bar{D}$
$\bar{A}\bar{B}$	0	0	0	0
$\bar{A}B$	0	1	1	0
AB	1	0	0	0
$A\bar{B}$	1	0	0	0

mais de um par ⟶ op OR

$$Y=ABC\bar{D}+AB\bar{C}\bar{D}+A\bar{B}\bar{C}\bar{D}+\bar{A}BC\bar{D}+\bar{A}BCD$$

$$Y=A\bar{C}\bar{D}+\bar{A}BD$$

$$Y=ABCD + ABC\bar{D}$$

$$Y=ABC(D+\bar{D})$$

$$Y=ABC$$

Simplificação por QUADRAS

Elimina 2 variáveis

$Y=AB\bar{C}\bar{D}+AB\bar{C}D+ABCD+ABC\bar{D}$

$Y=AB\bar{C}(D+\bar{D})+ABC(D+\bar{D})$

$Y=AB(C+\bar{C})$

$Y=AB$

4	$\bar{C}\bar{D}$	$\bar{C}D$	CD	$C\bar{D}$
$\bar{A}\bar{B}$	0	0	0	0
$\bar{A}B$	0	0	0	0
AB	1	1	1	1
$A\bar{B}$	0	0	0	0

C e D mudam de estado

$Y=AC$

4	$\bar{C}\bar{D}$	$\bar{C}D$	CD	$C\bar{D}$
$\bar{A}\bar{B}$	0	0	0	0
$\bar{A}B$	0	0	0	0
AB	0	0	1	1
$A\bar{B}$	0	0	1	1

B e D mudam de estado

Simplificação por OCTETOS

Elimina 3 variáveis

$$Y=A\bar{C}+AC$$

$$Y=A(\bar{C}+C)$$

$$Y=A$$

8	$\bar{C}\bar{D}$	$\bar{C}D$	CD	$C\bar{D}$
$\bar{A}\bar{B}$	0	0	0	0
$\bar{A}B$	0	0	0	0
AB	1	1	1	1
$A\bar{B}$	1	1	1	1

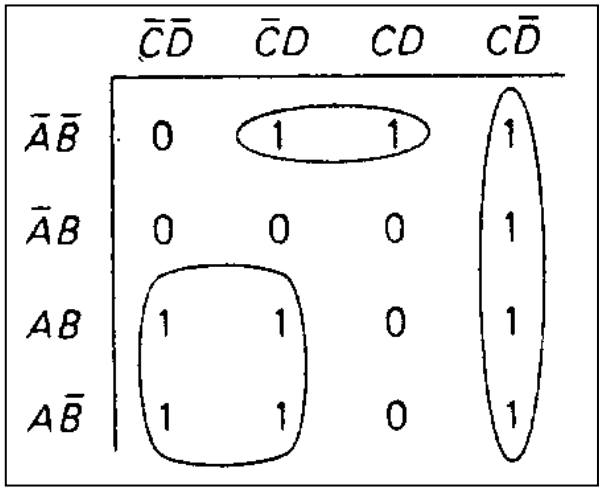
8	$\bar{C}\bar{D}$	$\bar{C}D$	CD	$C\bar{D}$
$\bar{A}\bar{B}$	0	0	0	0
$\bar{A}B$	0	0	0	0
AB	1	1	1	1
$A\bar{B}$	1	1	1	1

Resumo das simplificações por Karnaugh

- Um par elimina uma variável e seu complemento.

- Uma quadra elimina duas variáveis e seus complementos.

- Um octeto elimina três variáveis e seus complementos.

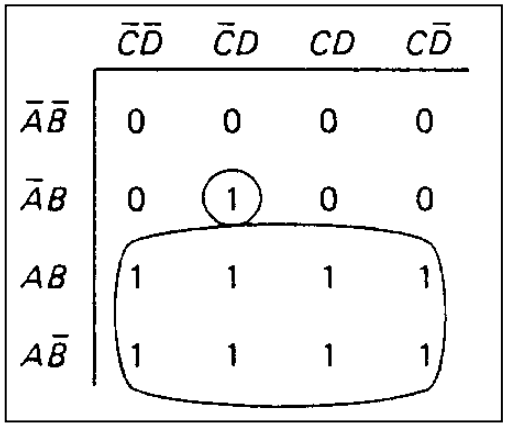


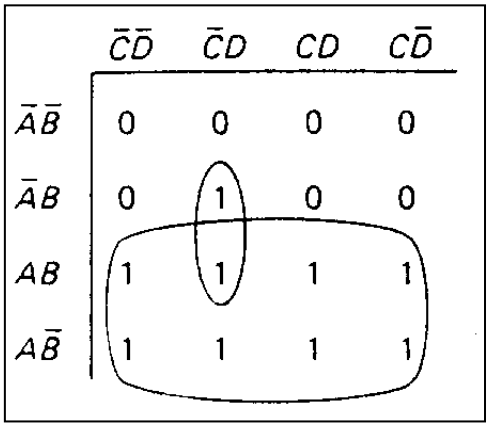
$$Y = \bar{A}\bar{B}D + A\bar{C} + C\bar{D}$$

Sobrepondo grupos

Pode-se usar o mesmo 1 mais de uma vez

$Y = A + \bar{A}B\bar{C}D$



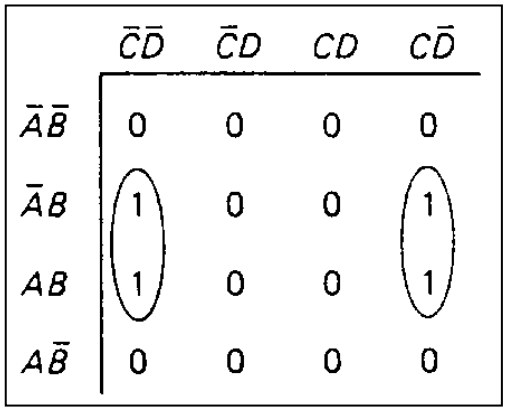


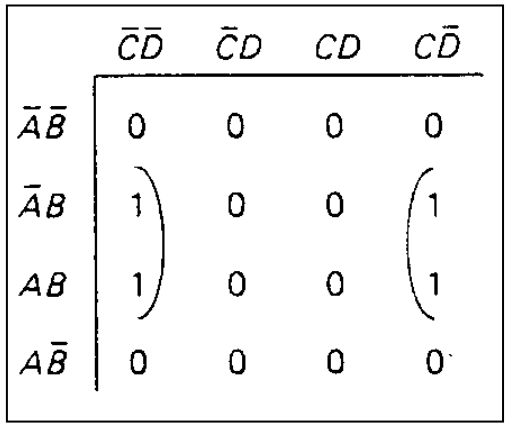
$Y = A + B\bar{C}D$

mais simplificado

Pode-se usar o mesmo 1 mais de uma vez

$Y = B\bar{C}\bar{D} + BC\bar{D}$

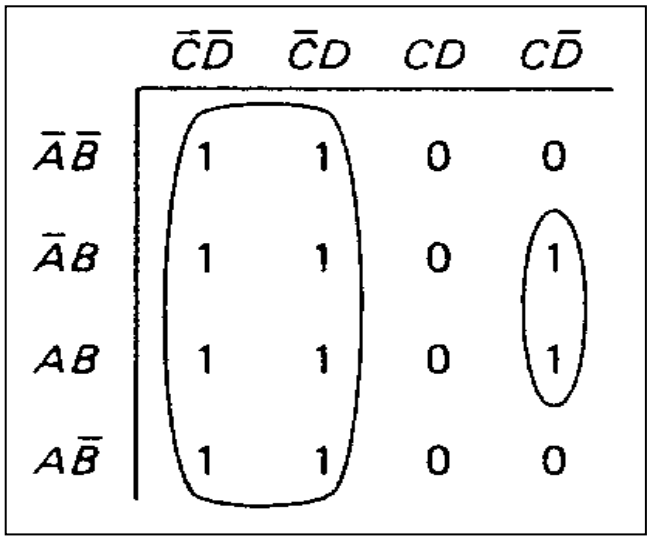




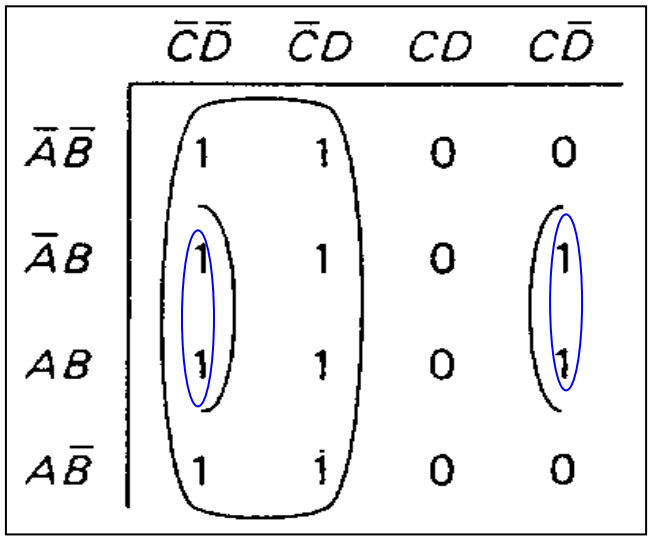
$Y = B\bar{D}$

mais simplificado

Ex. 1

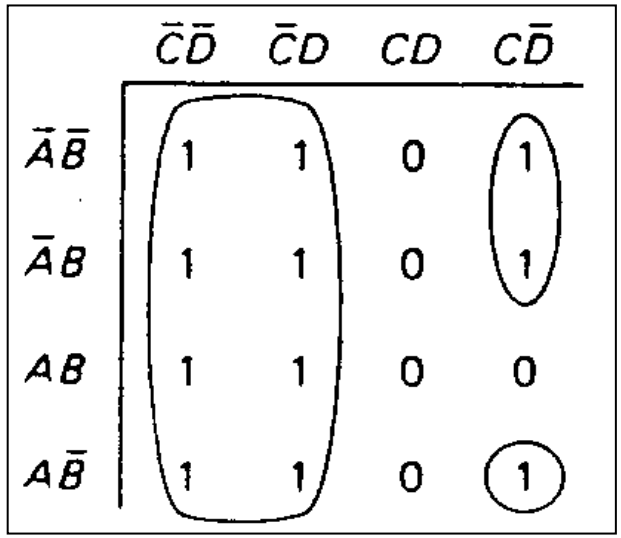


$$Y = \bar{C} + BC\bar{D}$$

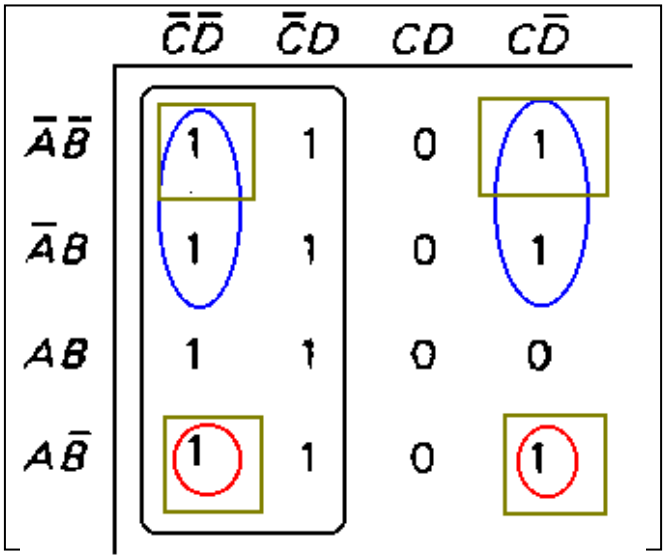


$$Y = \bar{C} + B\bar{D}$$

Ex. 2

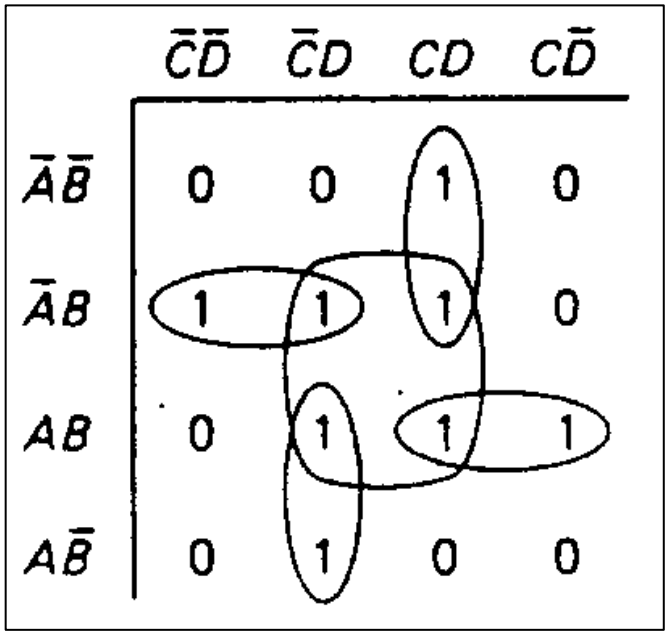


$$Y = \bar{C} + \bar{A}C\bar{D} + A\bar{B}C\bar{D}$$



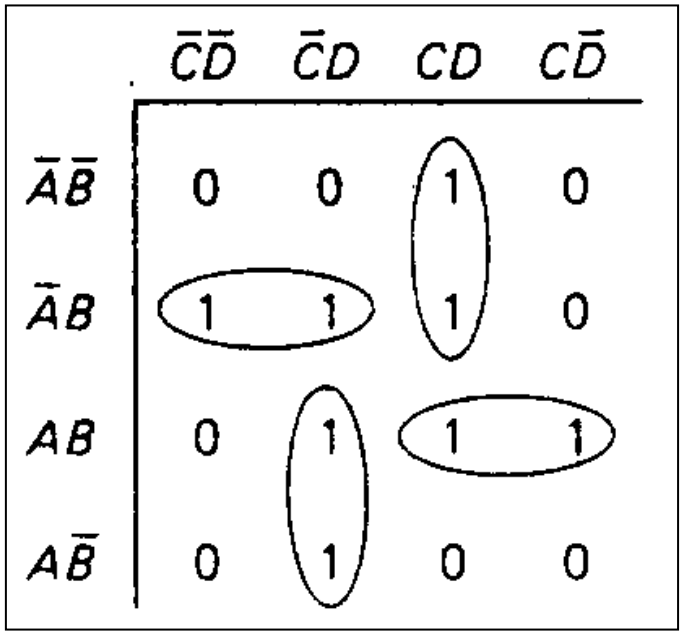
$$Y = \bar{C} + \bar{A}\bar{D} + \bar{B}\bar{D}$$

Eliminando grupos redundantes



$$Y = BD + \bar{A}B\bar{C} + ABC + A\bar{C}D + \bar{A}CD$$

$$Y = \bar{A}B\bar{C} + ABC + A\bar{C}D + \bar{A}CD$$



mais simplificado

Resumindo – passo a passo

1. Insira 1 no mapa de Karnaugh para cada produto fundamental com saída 1 na tabela-verdade. Insira 0s nos espaços restantes.

2. Circunde os octetos, quadras e pares. Lembre-se de sobrepor para obter os maiores grupos possíveis.

3. Se restar qualquer 1 isolado, circule cada um.

4. Elimine qualquer grupo redundante.

5. Escreva a equação booleana fazendo a operação OR dos produtos correspondentes aos grupos definidos.

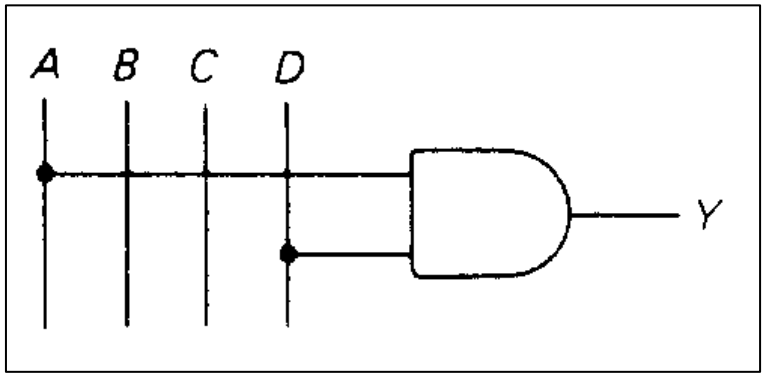
Condições irrelevantes (*don´t care*)

- Condições de entrada que nunca ocorrem durante o funcionamento normal; portanto a correspondente saída nunca aparece (X).

- A condição que não importa pode ser deixada igual a 1 ou 0, devendo-se usar o valor que produza um circuito lógico mais simples.

Ex. 1

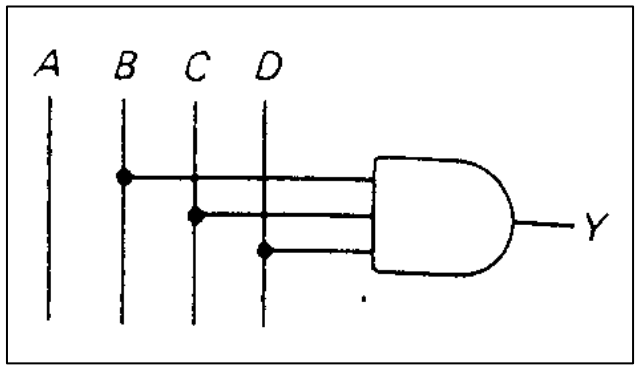
	$\bar{C}\bar{D}$	$\bar{C}D$	CD	$C\bar{D}$
$\bar{A}\bar{B}$	0	0	0	0
$\bar{A}B$	0	0	0	0
AB	x	x	x	x
$A\bar{B}$	0	1	x	x



$$Y = AD$$

Ex. 2

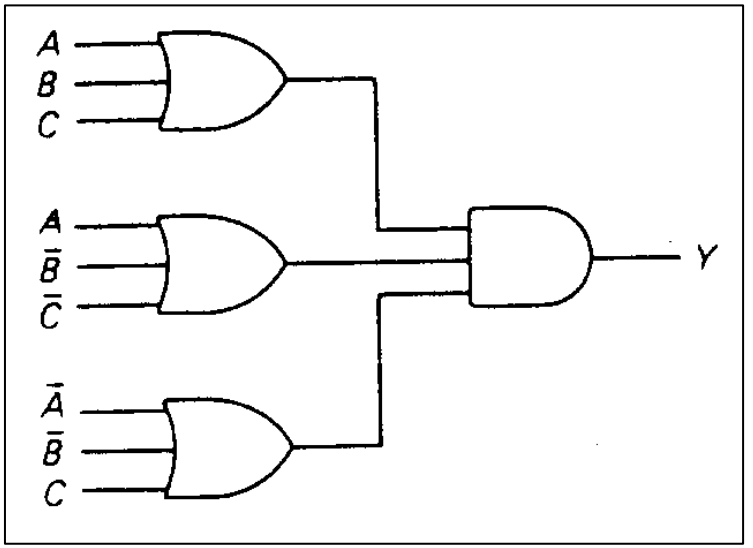
	$\bar{C}\bar{D}$	$\bar{C}D$	CD	$C\bar{D}$
$\bar{A}\bar{B}$	0	0	0	0
$\bar{A}B$	0	0	1	0
AB	x	x	x	x
$A\bar{B}$	0	0	x	x



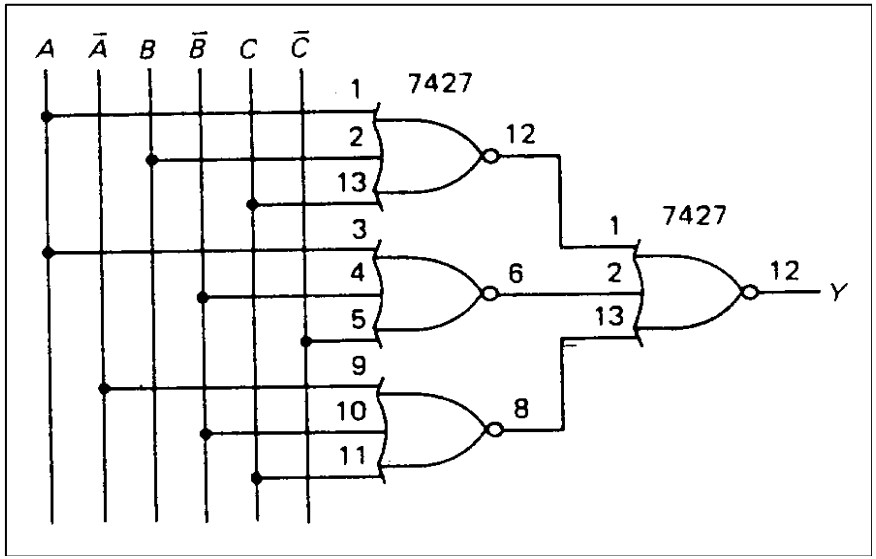
$$Y = BCD$$

Método do Produto de Somas

A	B	C	Y	
0	0	0	0	→ A+B+C
0	0	1	1	
0	1	0	1	
0	1	1	0	→ A+$\overline{B}$+$\overline{C}$
1	0	0	1	
1	0	1	1	
1	1	0	0	→ $\overline{A}$+$\overline{B}$+C
1	1	1	1	

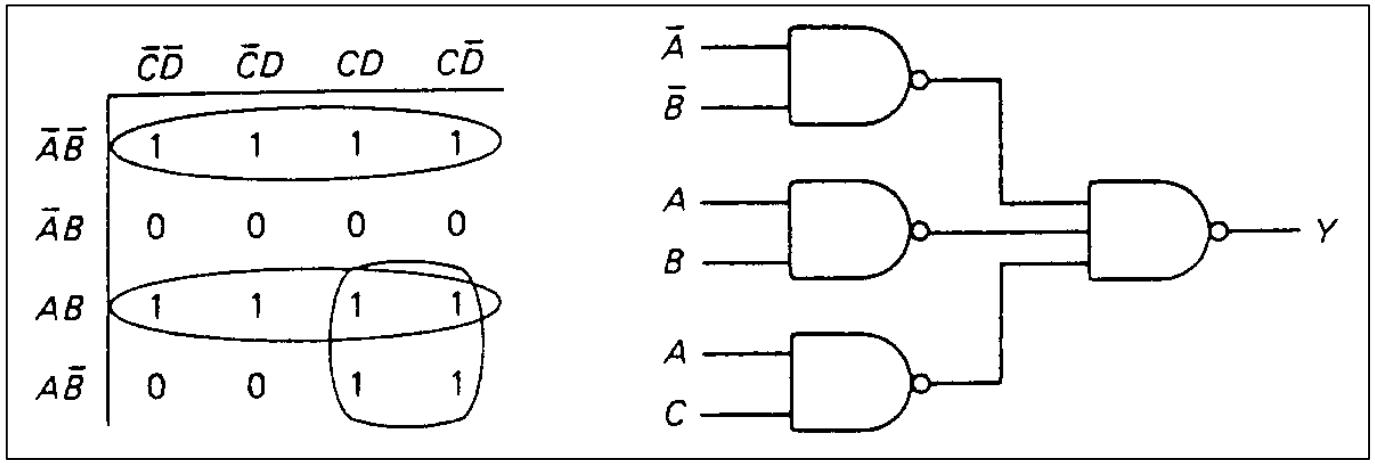


$$Y = (A+B+C)(A+\overline{B}+\overline{C})(\overline{A}+\overline{B}+C)$$

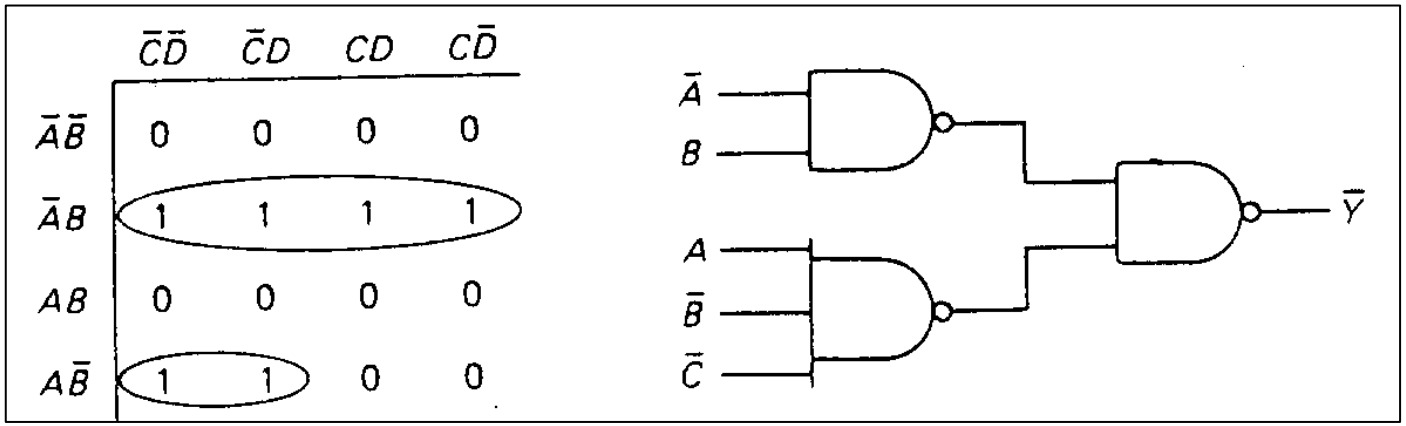


Simplificação do Produto de Somas

Soma de produtos



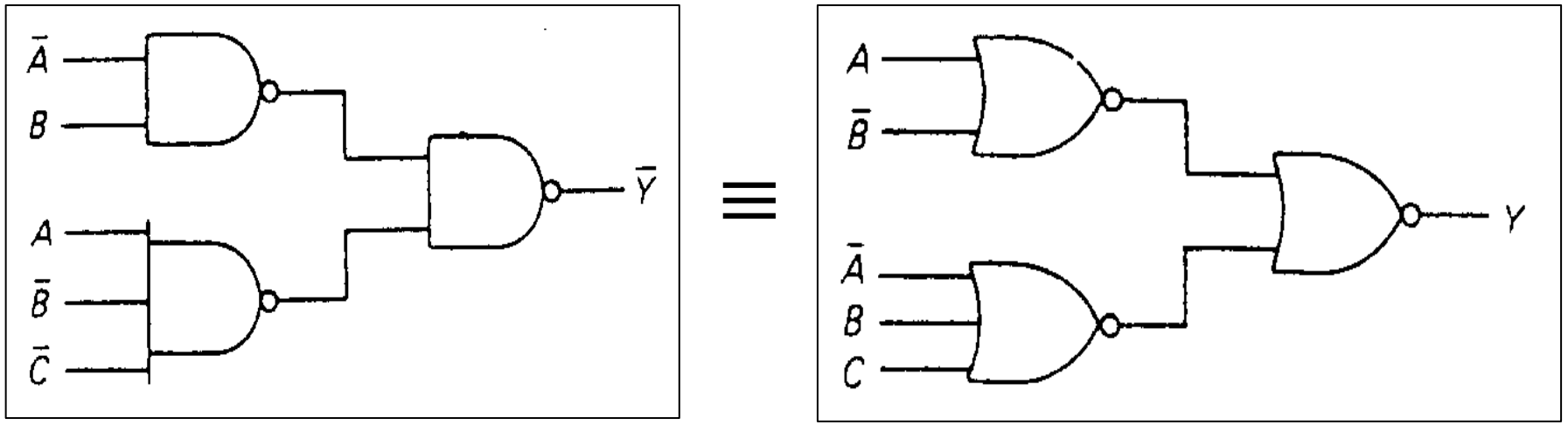
Produto de somas



Dualidade de Portas

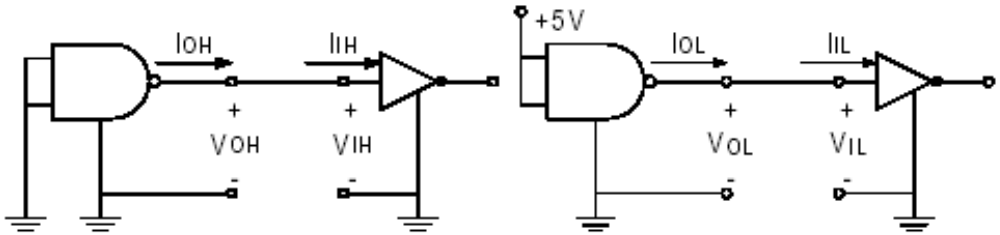
- Transforme cada porta AND em uma porta OR, transforme cada porta OR em uma porta AND e complemente todos os sinais de entrada e saída.

- Transforme cada porta NAND em uma porta NOR, transforme cada porta NOR em uma porta NAND e complemente todos os sinais de entrada e saída.



Circuitos integrados lógicos

Current and Voltage Parameters



V_{IH}(min) – High-Level Input Voltage
- The minimum voltage level required for a logical 1 at an input.
- Any voltage below this level will not be accepted as a HIGH by the logic circuit

V_{IL}(max) – Low-Level Input Voltage
- The maximum voltage level required for a logical 0 at an input.
- Any voltage above this level will not be accepted as a LOW by the logic circuit

V_{OH}(min) – High-Level Output Voltage
- The minimum voltage level at a logic circuit output in the logical 1 stage under defined load conditions.

V_{OL} – Low-Level Output Voltage
- The maximum voltage level at a logic circuit output in the logical 0 stage under defined load conditions.

I_{IH} – High-Level Input Current
- The current that flows into an input when a specified high-level voltage is applied to that input

I_{IL} – Low-Level Input Current
- The current that flows into an input when a specified low-level voltage is applied to that input

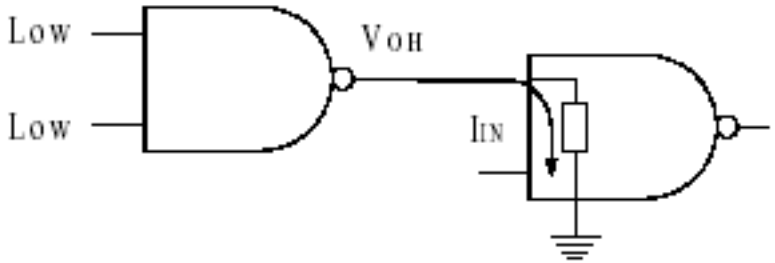
I_{OH} – High-Level Output Current
- The current that flows from an output in the logical 1 state under specified load condition.

I_{OL} – Low-Level Output Current
- The current that flows from an output in the logical 0 state under specified load condition.

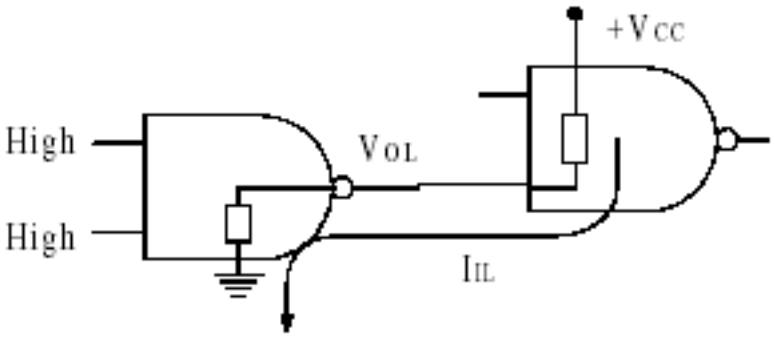
Circuitos integrados lógicos

Current Sourcing and Current Sinking Logic

- Logic families can be categorized according to how current flows from the output of on logic circuit to another. The figures below illustrate the difference between the two types.



Current Sourcing
Supplies current to load gate in HIGH state

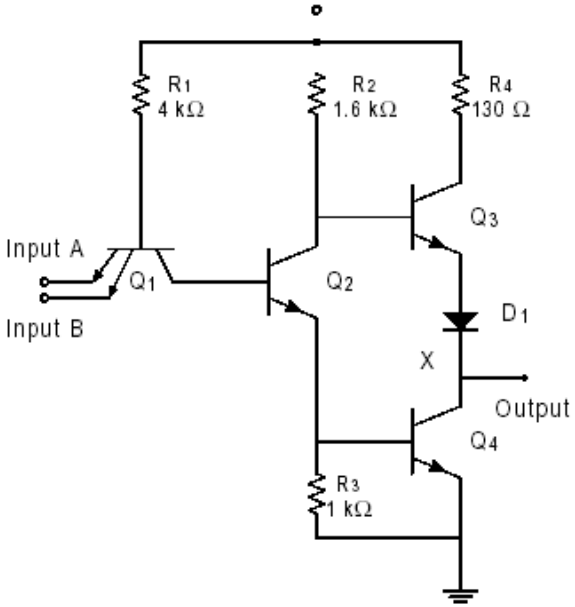


Current Sinking
receives current from load gate in LOW state

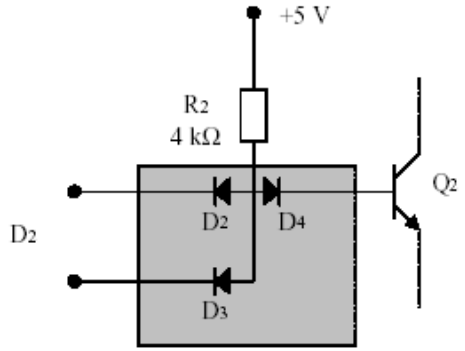
TTL Logic

2.1.2.1 Operation of TTL Logic Devices(Totem-pole Output Circuits)

- Since the transistor-transistor logic (TTL) family is widely used, we will examine the operation of a typical TTL NAND gate. Because NAND gate can be used to generate many types of logic functions, an understanding of the operation of this simple gate provides a good insight into the operation of most TTL devices.

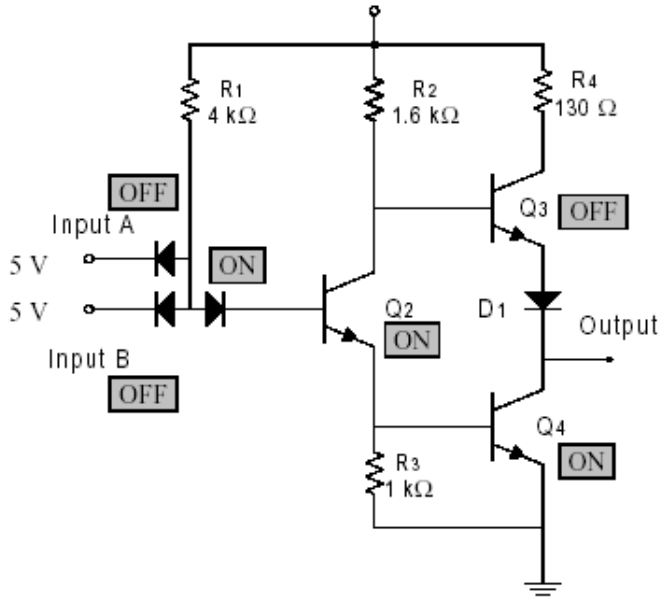


- The circuit above shows a basic two input TTL NAND gates.
- Q1 is a multiple-transmitter transistor. It can be seen as

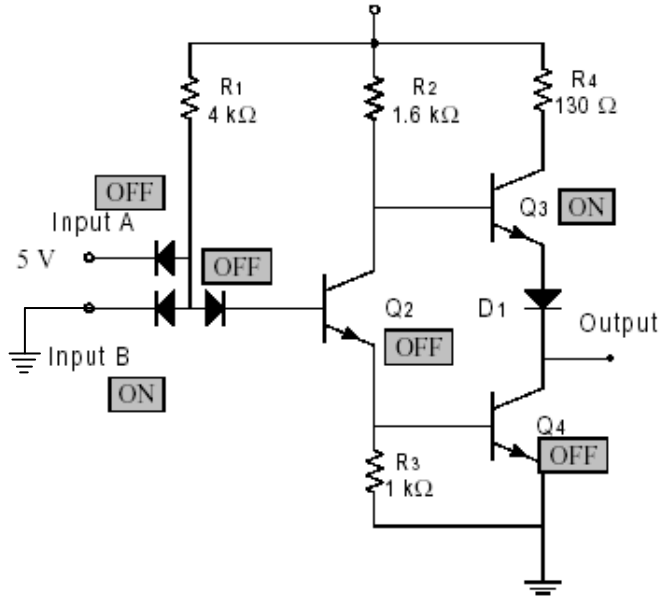


Equivalent Q1

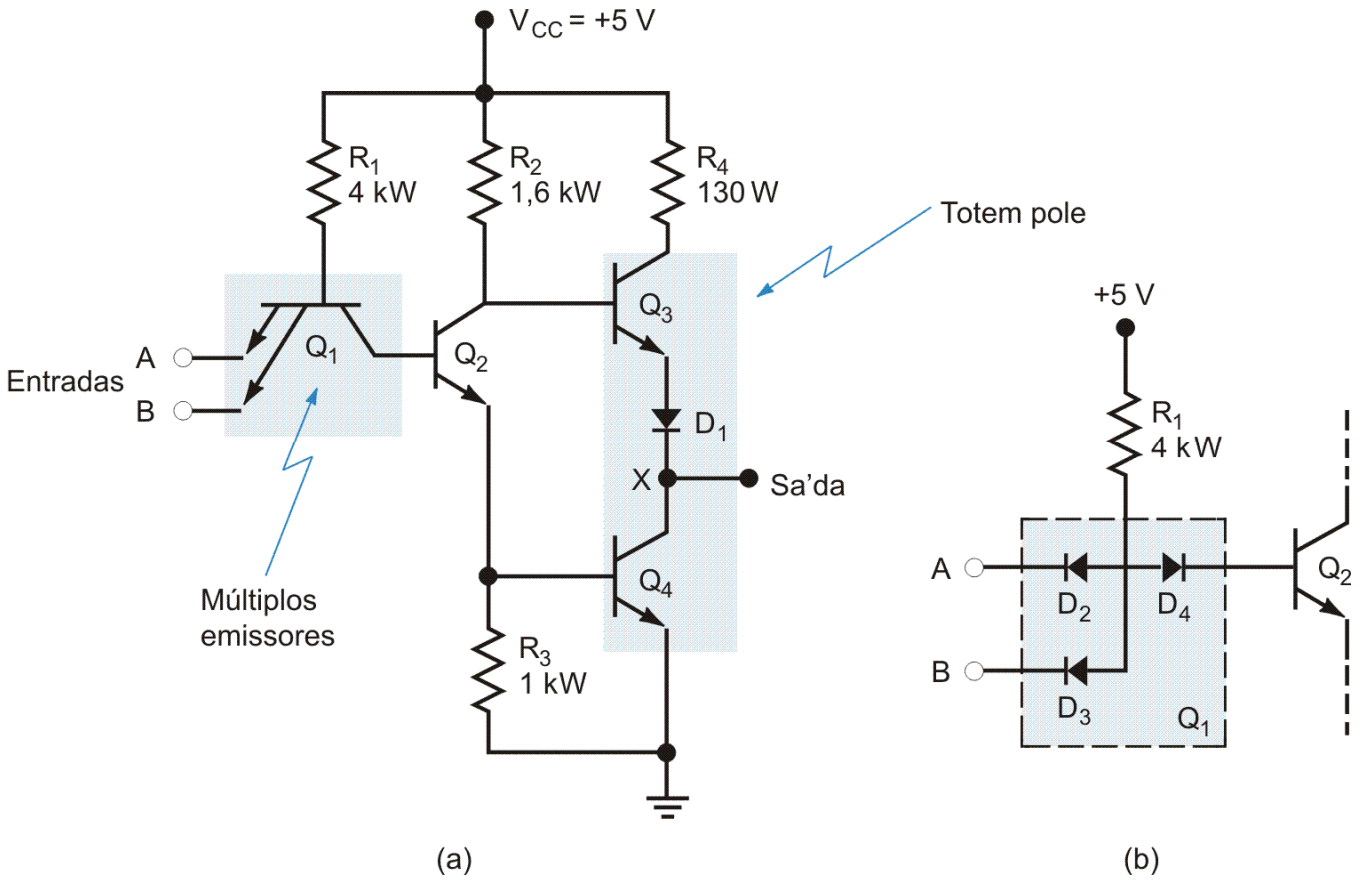
- Output LOW case



- Output HIGH case



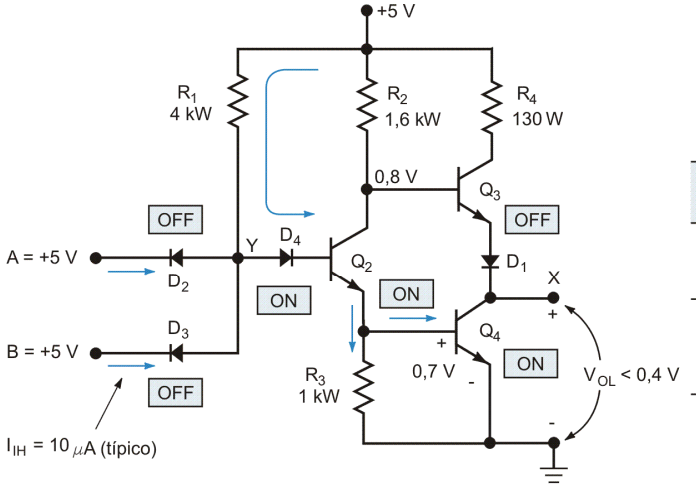
Circuitos integrados lógicos



(a) Porta NAND TTL básica. (b) Equivalente com diodo para Q_1.

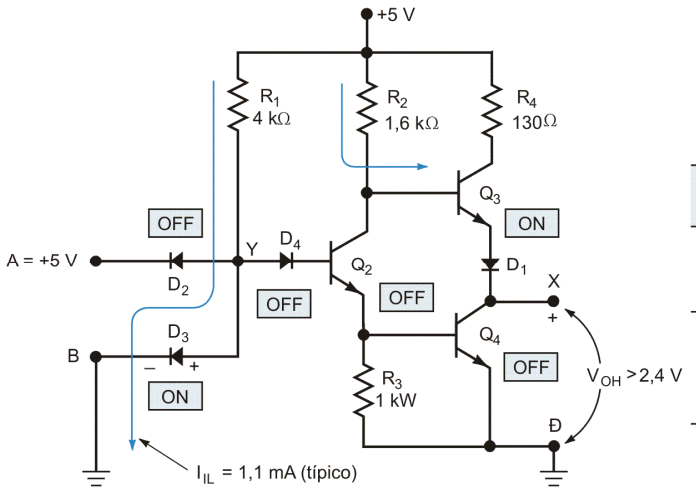
Circuitos integrados lógicos

Porta NAND TTL em seus dois estados de saída.



(a) Saída em nível BAIXO

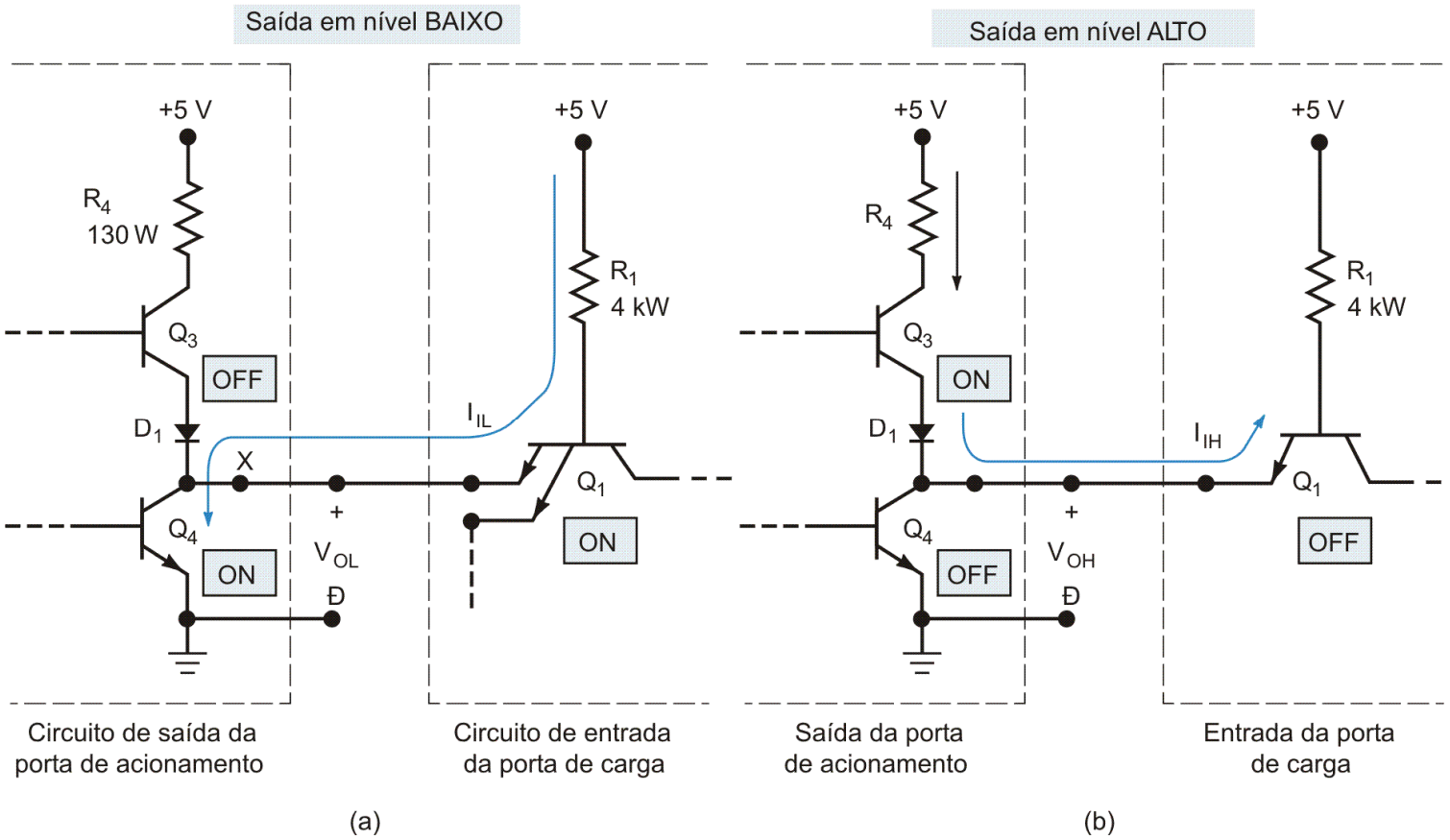
Condições de entrada	Condições de saída
A e B estão ambas em nível ALTO (≥2 V)	Q_3 OFF
As correntes de entrada são muito baixas I_{IH} = 10 μA	Q_4 ON, logo, V_X está em nível baixo (≤ 0,4 V)



(b) Saída em nível ALTO

Condições de entrada	Condições de saída
A e B estão ambas em nível ALTO (≤ 0,8 V)	Q_4 OFF
A corrente flui para GND através do terminal de entrada em nível baixo. I_{IL} = 1,1 mA	Q_3 atua como um seguidor de emissor e V_{OH} ≥ 2,4 V, geralmente 3,6 V

Circuitos integrados lógicos



(a) Quando a saída TTL está em nível BAIXO, Q_4 atua drenando corrente da carga.
(b) Com a saída em nível ALTO, Q_3 atua fornecendo corrente para a carga.

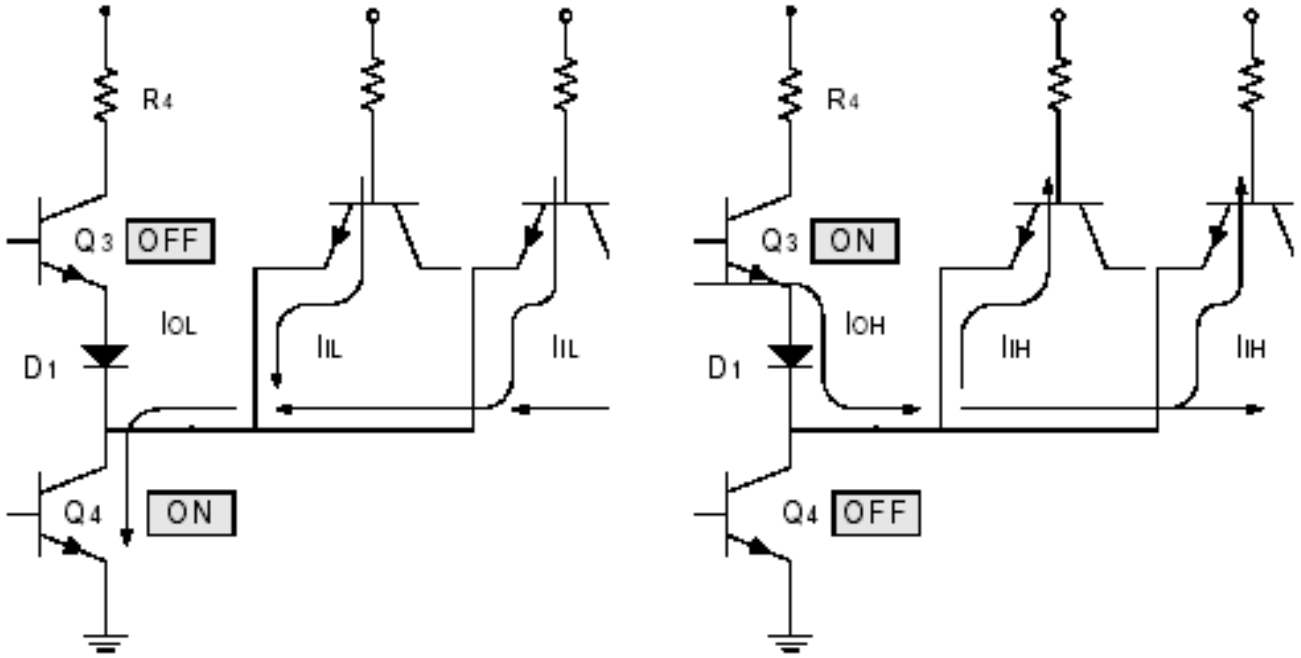
Circuitos integrados lógicos

Fan-out

- Sometime referred to as ***loading factor***.
- It is defined as the maximum number of standard logic inputs that an output can drive reliably.
- Say, if a logic gate has a fan-out of 10, then it can drive 10 standard inputs. If this number is exceeded, the output logic-level voltages can no longer be guaranteed.

Circuitos integrados lógicos

TTL loading and Fan-out



- If a high state output of a driving chip can source X amount of current and the input of the driven chip can sink Y amount of current, then the ***high state fan-out*** or the number of inputs the output can drive is simply the integer value of X/Y. i.e.

$$\text{Fan-out(High)} = \frac{I_{OH}(\max)}{I_{IH}(\max)}$$

- Similarly, the ***low state fan-out*** is defined as,

$$\text{Fan-out(Low)} = \frac{I_{OL}(\max)}{I_{IL}(\max)}$$

Circuitos integrados lógicos

- Example:
  How many 7400 NAND gate inputs can be driven by a 7400 NAND output?
  Solution:
  From the data sheet, we can see that
  $I_{OL}(max) = 16\ mA$, $I_{IL}(max) = 1.6\ mA$
  $I_{OH}(max) = 400\ \mu A$, $I_{IH}(max) = 40\ \mu A$

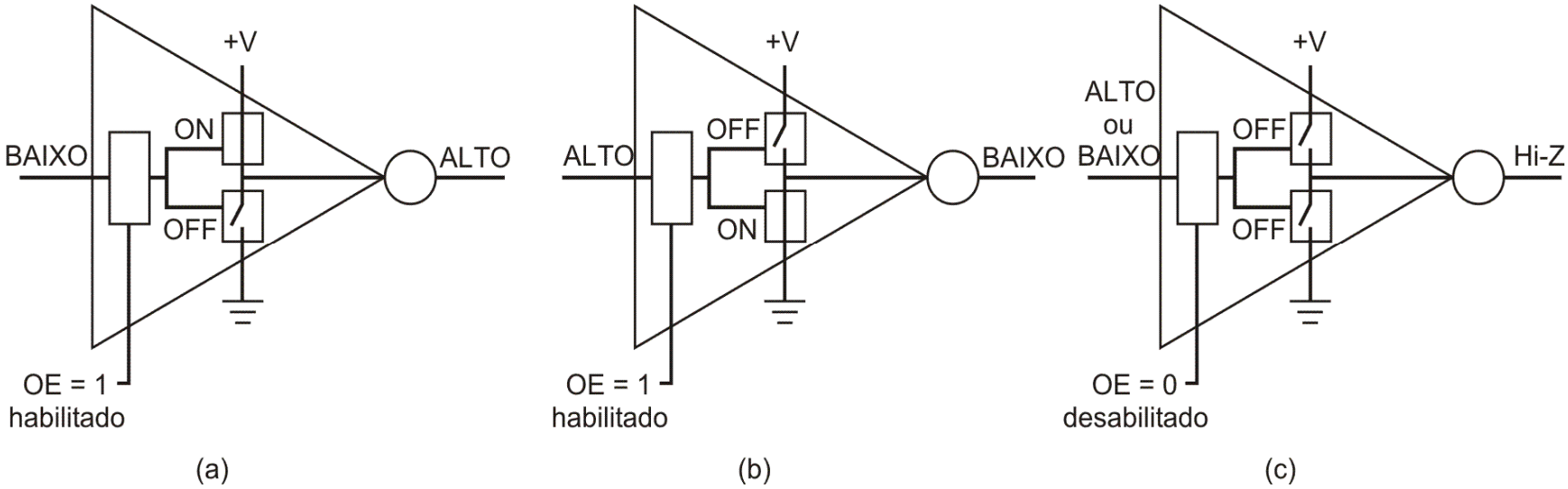
  Therefore,

$$\text{Fan-out(Low)} = \frac{I_{OL}(max)}{I_{IL}(max)} = \frac{16\ mA}{1.6\ mA} = 10$$

$$\text{Fan-out(High)} = \frac{I_{OH}(max)}{I_{IH}(max)} = \frac{400\ \mu A}{40\ \mu A} = 10$$

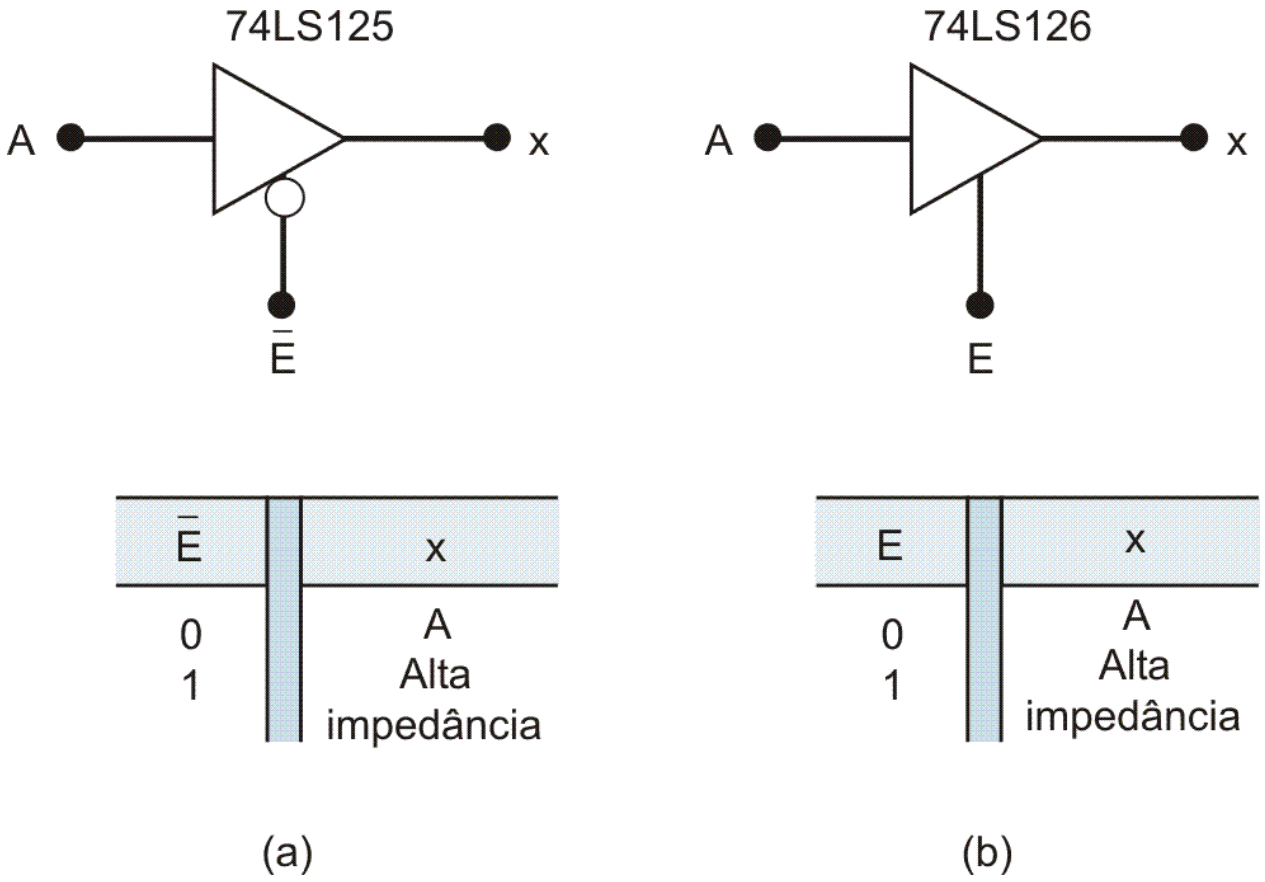
  i.e. the fan-out is 10

Circuitos integrados lógicos



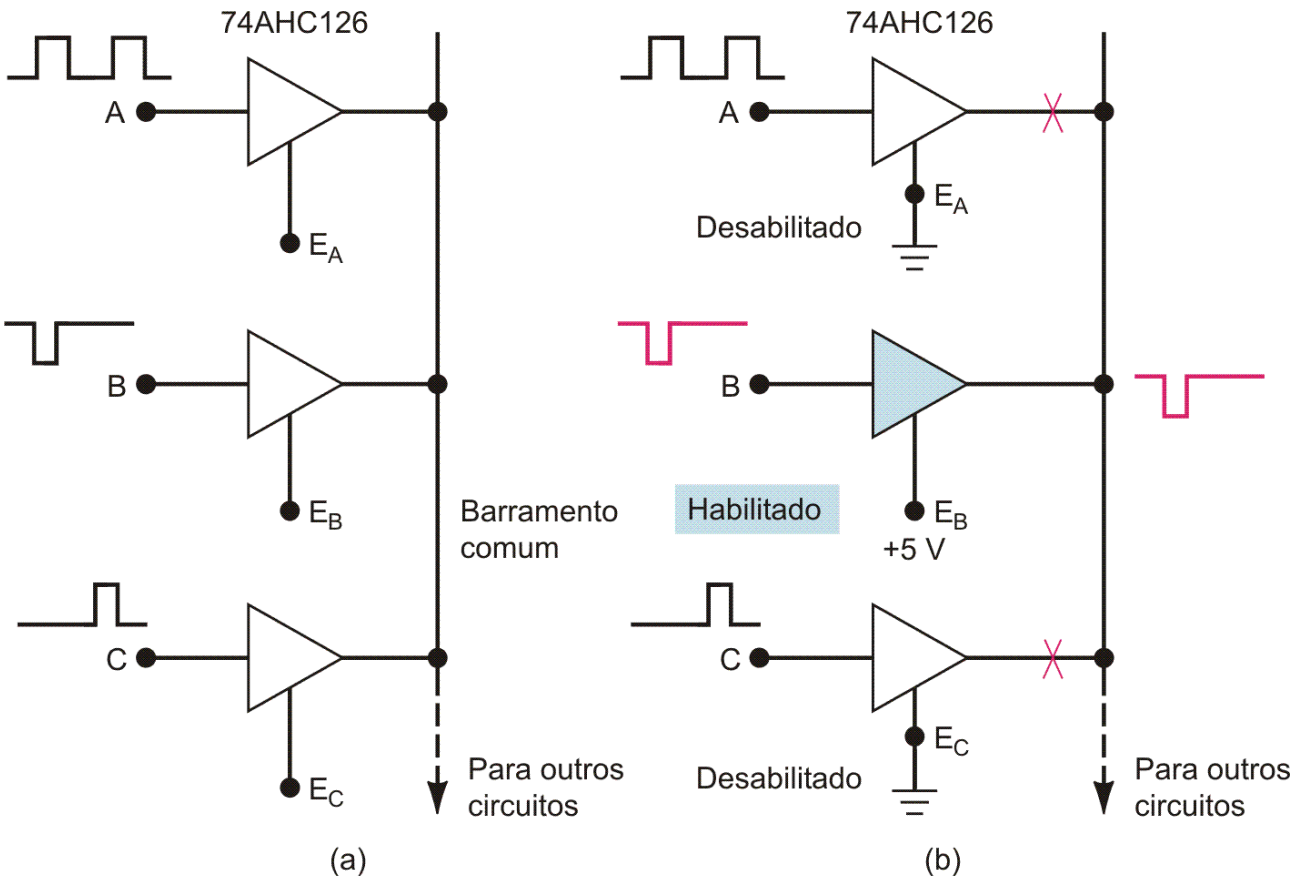
Três condições de saída: 1, 0 e *tri-state* (Z)

Circuitos integrados lógicos

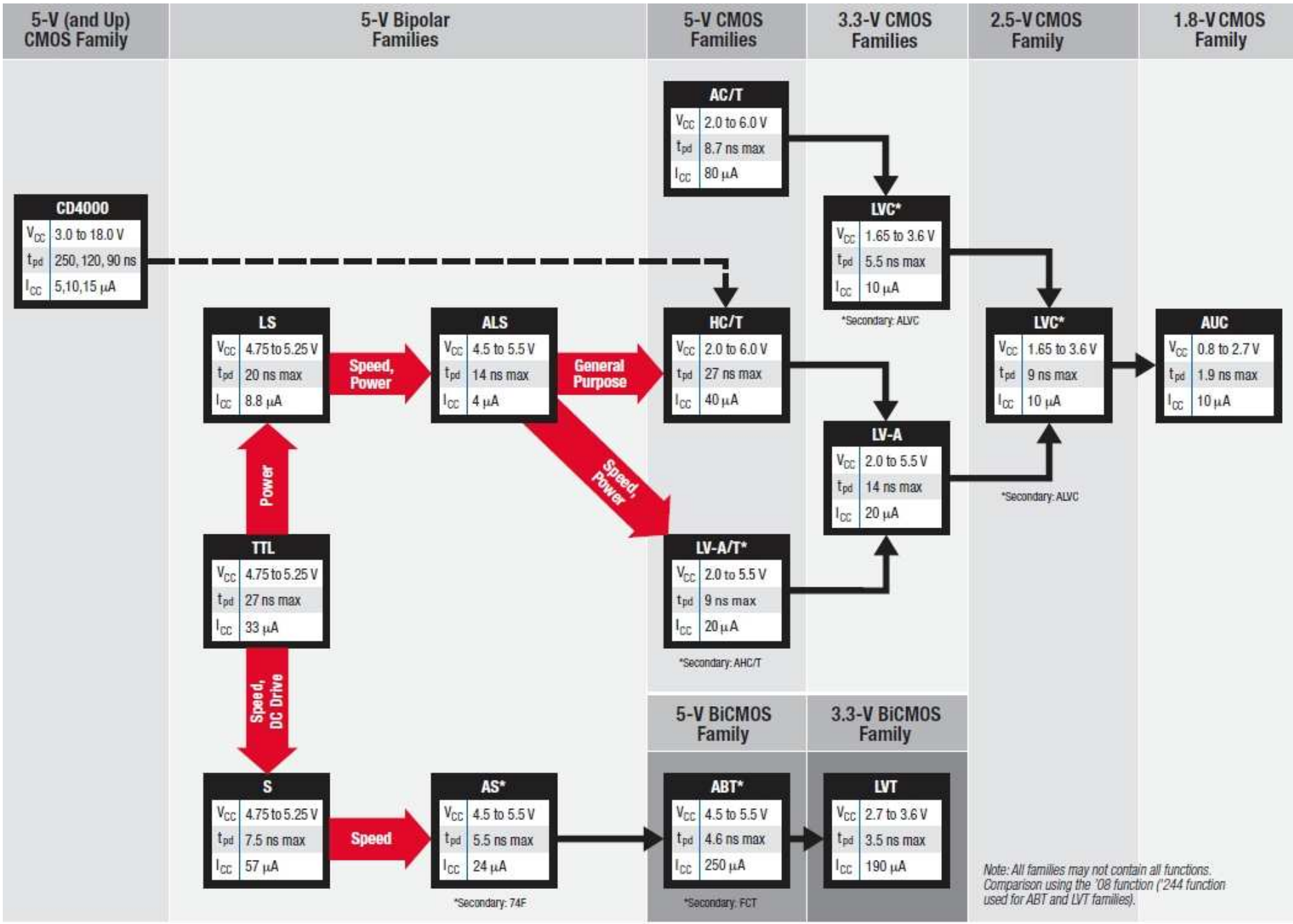


Buffers *tri-state* não inversores

Circuitos integrados lógicos



(a) Buffers tri-state usados para conectar sinais a um barramento comum.
(b) Condições para transmitir o sinal B para o barramento.


5-V (and Up) CMOS Family
5-V Bipolar Families
5-V CMOS Families
3.3-V CMOS Families
2.5-V CMOS Family
1.8-V CMOS Family
CD4000
VCC 3.0 to 18.0 V
tpd 250, 120, 90 ns
ICC 5,10,15 µA
AC/T
VCC 2.0 to 6.0 V
tpd 8.7 ns max
ICC 80 µA
LVC*
VCC 1.65 to 3.6 V
tpd 5.5 ns max
ICC 10 µA
*Secondary: ALVC
LS
VCC 4.75 to 5.25 V
tpd 20 ns max
ICC 8.8 µA
Speed, Power
ALS
VCC 4.5 to 5.5 V
tpd 14 ns max
ICC 4 µA
General Purpose
HC/T
VCC 2.0 to 6.0 V
tpd 27 ns max
ICC 40 µA
LVC*
VCC 1.65 to 3.6 V
tpd 9 ns max
ICC 10 µA
AUC
VCC 0.8 to 2.7 V
tpd 1.9 ns max
ICC 10 µA
Power
Speed, Power
LV-A
VCC 2.0 to 5.5 V
tpd 14 ns max
ICC 20 µA
*Secondary: ALVC
TTL
VCC 4.75 to 5.25 V
tpd 27 ns max
ICC 33 µA
LV-A/T*
VCC 2.0 to 5.5 V
tpd 9 ns max
ICC 20 µA
*Secondary: AHC/T
Speed, DC Drive
5-V BiCMOS Family
3.3-V BiCMOS Family
S
VCC 4.75 to 5.25 V
tpd 7.5 ns max
ICC 57 µA
Speed
AS*
VCC 4.5 to 5.5 V
tpd 5.5 ns max
ICC 24 µA
*Secondary: 74F
ABT*
VCC 4.5 to 5.5 V
tpd 4.6 ns max
ICC 250 µA
*Secondary: FCT
LVT
VCC 2.7 to 3.6 V
tpd 3.5 ns max
ICC 190 µA
Note: All families may not contain all functions. Comparison using the '08 function ('244 function used for ABT and LVT families).

TTL Families

1 Standard TTL (74 series)

- Draws an average power of 10 mW
- Typical propagation delays of t_{PLH} = 11ns and t_{PHL} = 7ns
- Offers a wide variety of
  - gates, flip-flops and one-shots in SSI line
  - counter, registers, decoders/encoders, arithmetic circuits in MSI line
- Gradually replaced by better improve version of TTL Families

2 74L and 74H Series

- Developed to provide low power and high-speed version of TTL respectively.
- Low power version has low power consumption but slower speed
- High speed version has fast propagation delays but consume more power

3 Schottky TTL, T4S Series

- Improve on the speed by avoid the transistor to go into saturation.
- Accomplished by using a Schottky barrier diode connected between base and collector.

4 Low power Schottky TTL, 74LS Series

- Slow-speed version of Schottky TTL
- Uses larger resistor value to reduce power consumption

5 Advanced Schottky TTL, 74AS Series

- Improvement to Schottky to have faster speed and lower power consumption

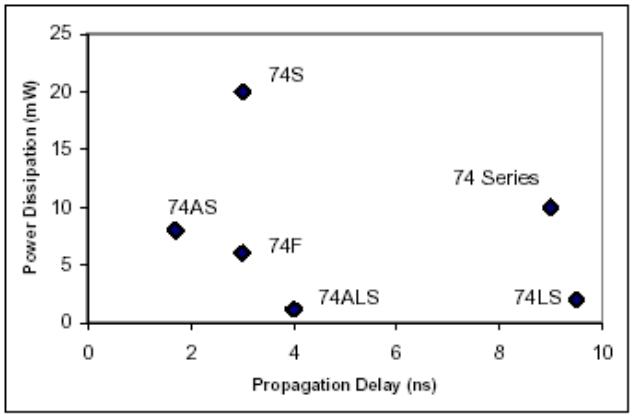
6 Advanced Low power Schottky TTL, 74ALS Series

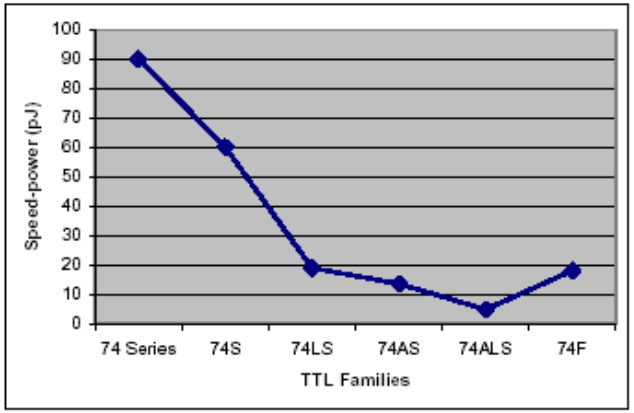
- Improvement to Low power Schottky to have faster speed and lower power consumption

7 74F-Fast TT

- Latest technology.
- Uses new integrated circuit fabrication technique to reduce inter-device capacitance to achieve reduced propagation delays.

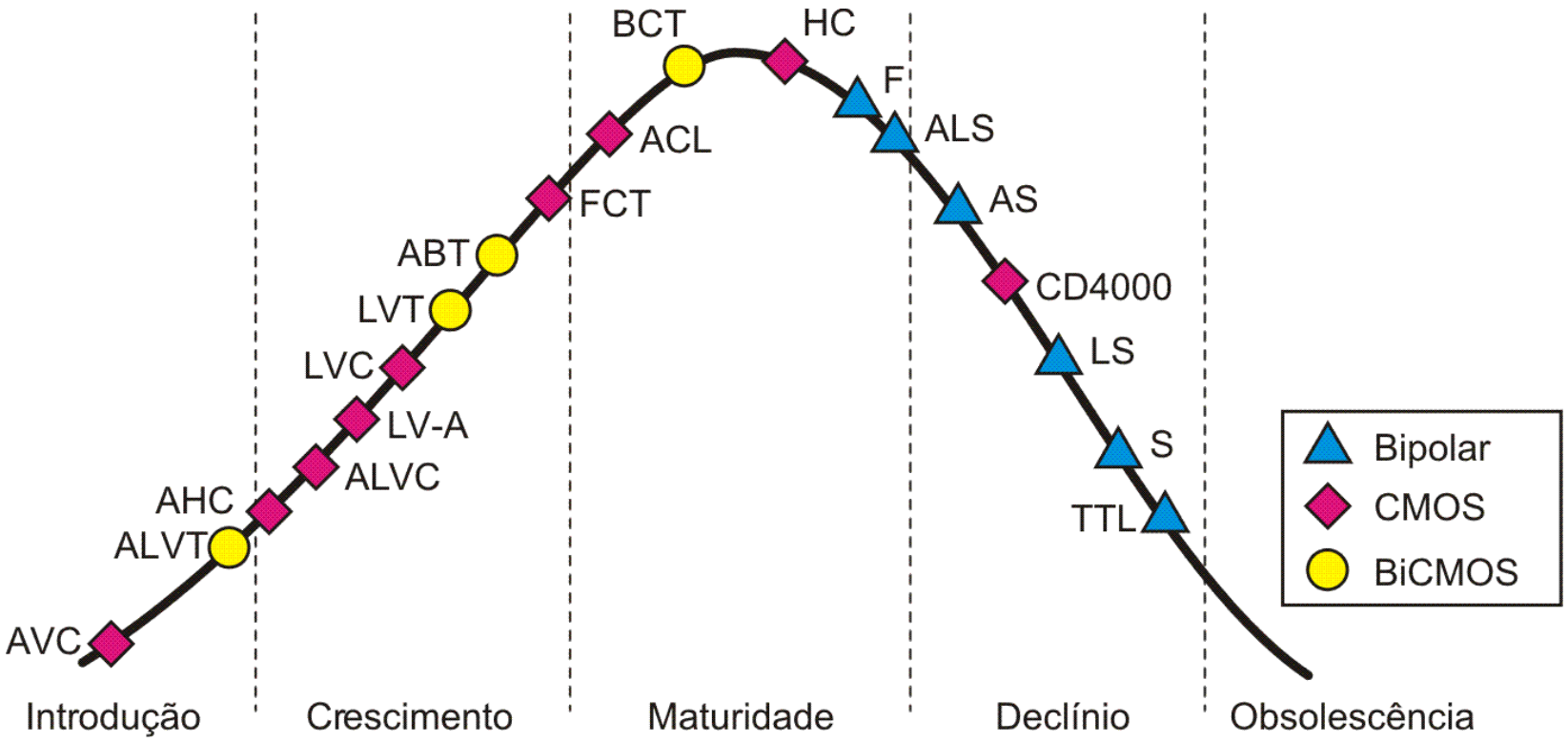
	74	74S	74LS	74AS	74ALS	74F
Performance ratings						
Propagation delay (ns)	9	3	9.5	1.7	4	3
Power dissipation (mW)	10	20	2	8	1.2	6
Speed-power product (pJ)	90	60	19	13.6	4.8	18
Max. clock rate (MHz)	35	125	45	200	70	100
Fan-out (same series)	10	20	20	40	20	33
Voltage parameters						
V_{OH}(min)	2.4	2.7	2.7	2.5	2.5	2.5
V_{OL}(max)	0.4	0.5	0.5	0.5	0.4	0.5
V_{IH}(min)	2.0	2.0	2.0	2.0	2.0	2.0
V_{IL}(max)	0.8	0.8	0.8	0.8	0.8	0.8





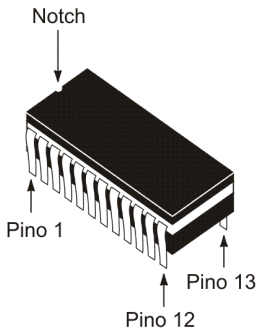
Ciclo de vida das famílias lógicas

(*Texas Instruments*)



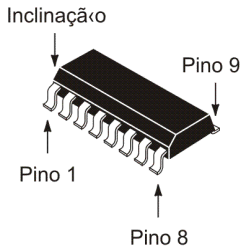
Circuitos integrados lógicos

Encapsulamentos típicos de CIs (*Texas Instruments*)



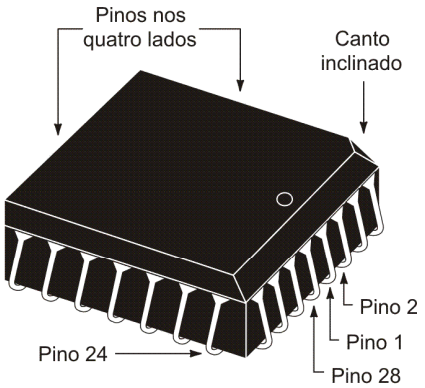
DIP de 24 pinos

(a)



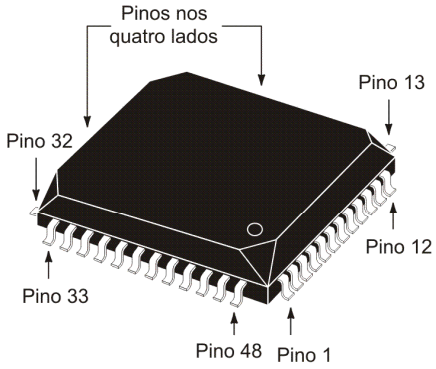
SOIC de 16 pinos
(asa de gaivota)
para montagem em superfície

(b)



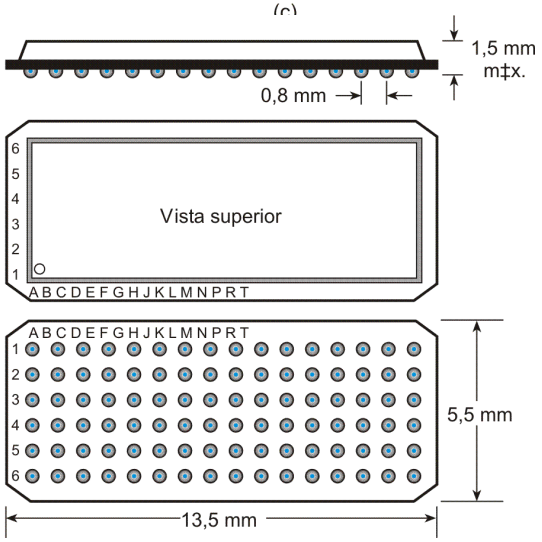
PLCC de 28 pinos (pino J)
para soquete ou montagem
em superfície

(c)



QFP de 48 pinos
(asa de gaivota)
para montagem em superfície

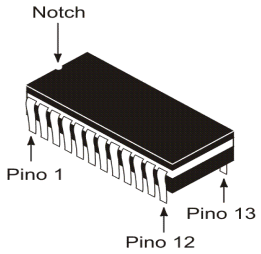
(d)



LFBGA de 96 pinos
para montagem em superfície

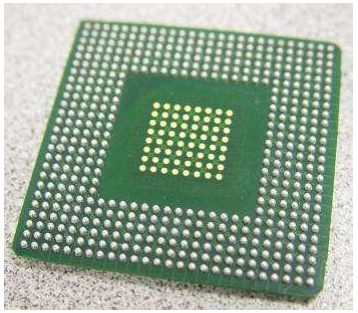
(e)

Evolução ...



DIP de 24 pinos

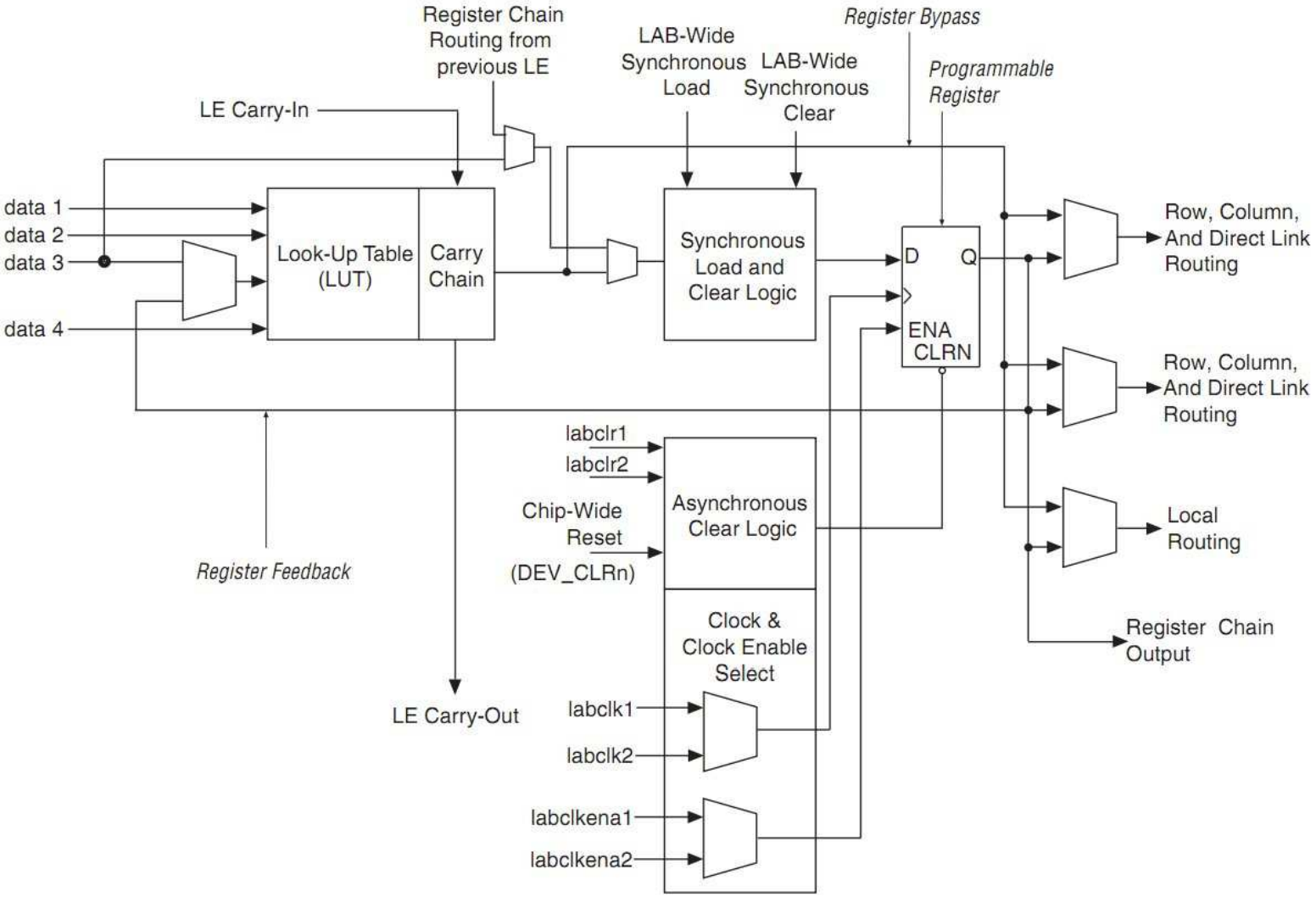
(a)



	74LS08
Tecnologia	L=? µm (5,0V)
Lógica não-programável	4 portas lógicas AND
Velocidade	45MHz
Memória	-
Multiplicadores	-
Pinos de I/O	12
I/O programável	NÃO
Preço unitário (US$)	US$0,37 (1k)

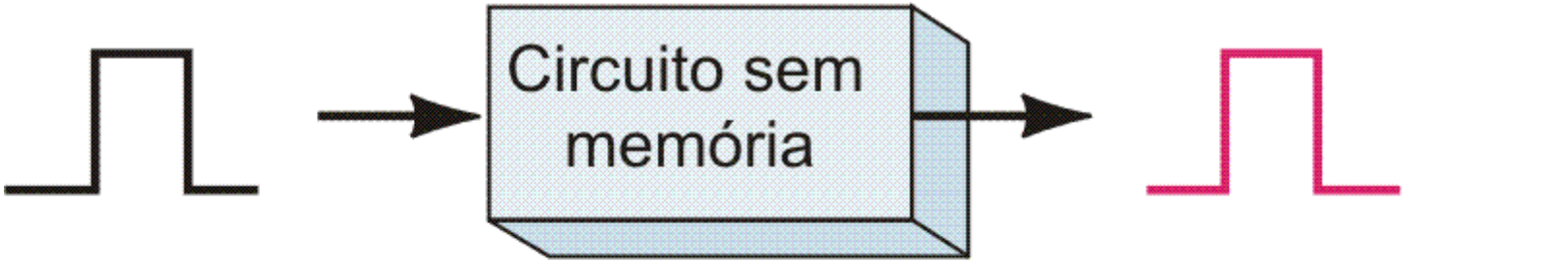
	Família Cyclone IV
Tecnologia	L=60 nm (1,2V)
Lógica programável	6.272 - 114.480 LEs (>240k portas)
Velocidade	> 300MHz
Memória	270 – 3.888 kbits
Multiplicadores	15 - 266 (18 bits X 18 bits)
Pinos de I/O	179 - 528
I/O programável	SIM
Preço unitário (US$)	US$ 11,95 (menor dispositivo)

Lógica Programável

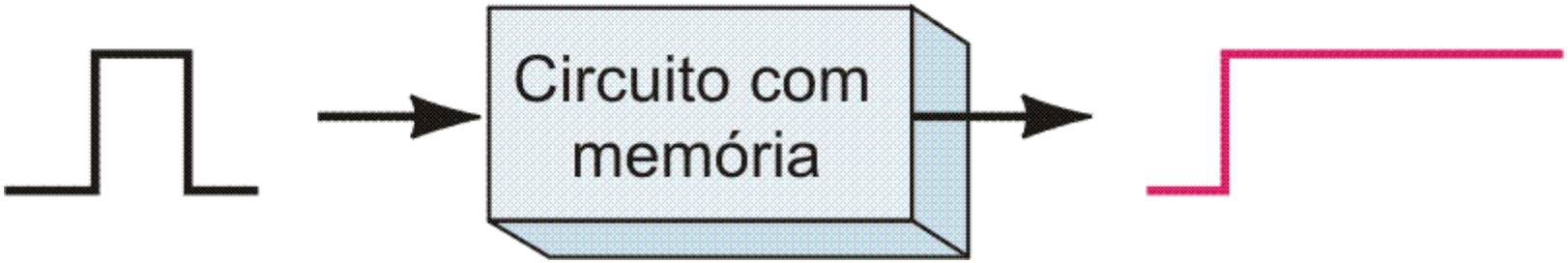


Célula lógica típica de uma FPGA (*Logic Element*)

Circuitos sem memória e com memória

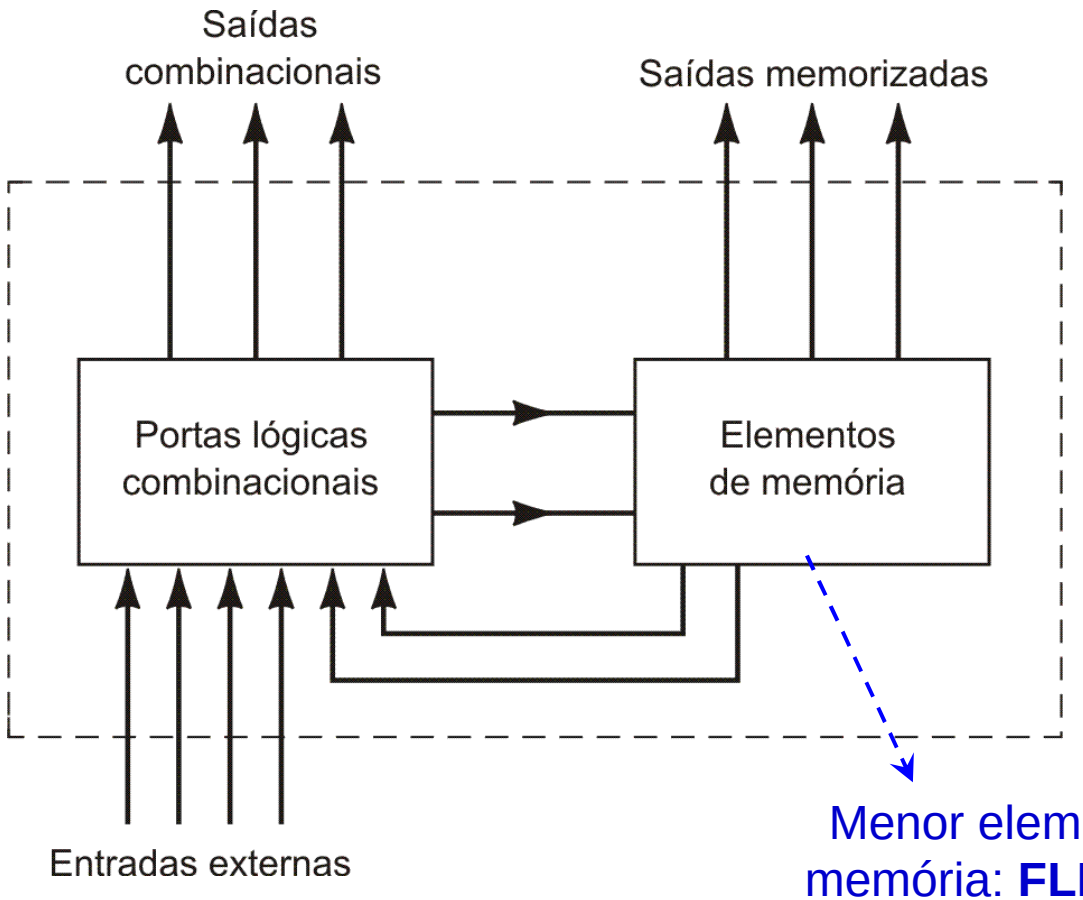


combinacional

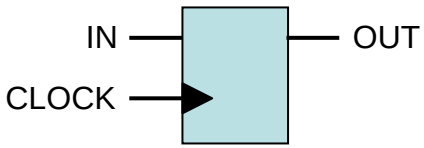


sequencial

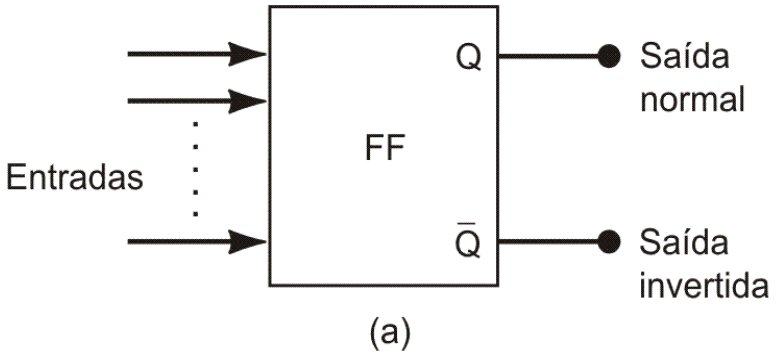
Diagrama genérico de um circuito digital



Menor elemento de memória: **FLIP-FLOP**



Símbolo de um Flip-Flop e os dois estados de saída possíveis



(a)

Estados de saída

$Q = 1, \bar{Q} = 0$: denominado estado ALTO ou 1; também chamado de estado SET

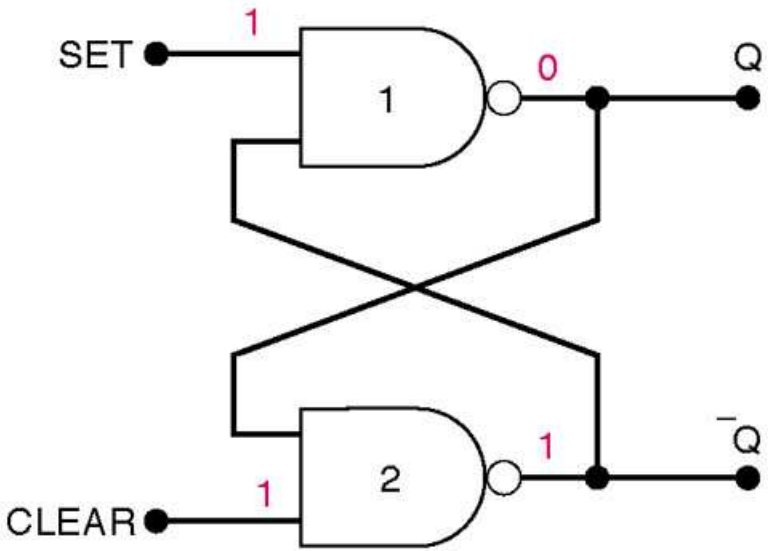
$Q = 0, \bar{Q} = 1$: denominado estado BAIXO ou 0; também chamado de estado CLEAR ou RESET

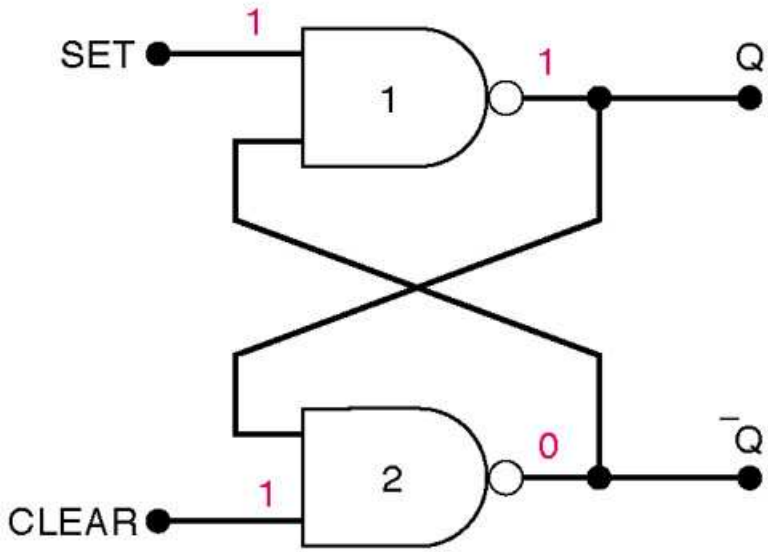
(b)

* O termo **estado** do flip-flop sempre faz referência à saída NORMAL (Q).

FF S-C com portas NAND

dois estados estáveis possíveis quando SET=CLEAR=1





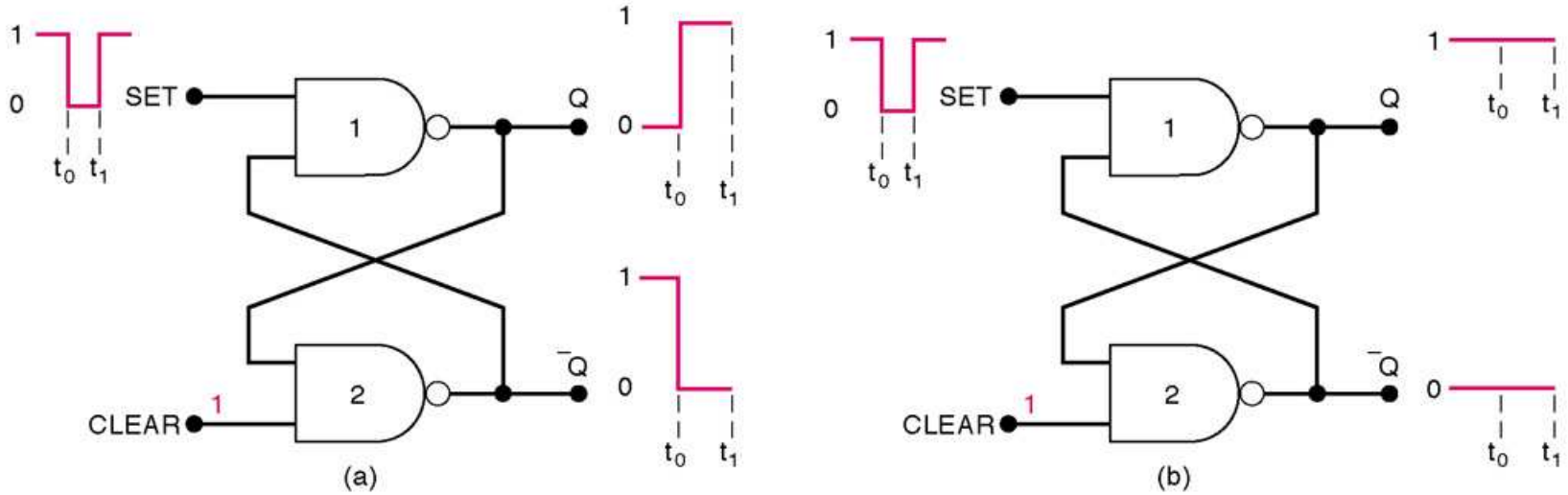
O **FF S-C NAND** opera com pulsos ativos em nível baixo nas entradas SET e CLEAR.

A	B	NAND
0	0	1
0	1	1
1	0	1
1	1	0

O estado atual das saídas depende do que ocorreu anteriormente nas entradas SET e CLEAR.

Pulsando a entrada SET para o nível baixo

(a) Q=0 antes do pulso na entrada SET;
(b) Q=1 antes do pulso na entrada SET.

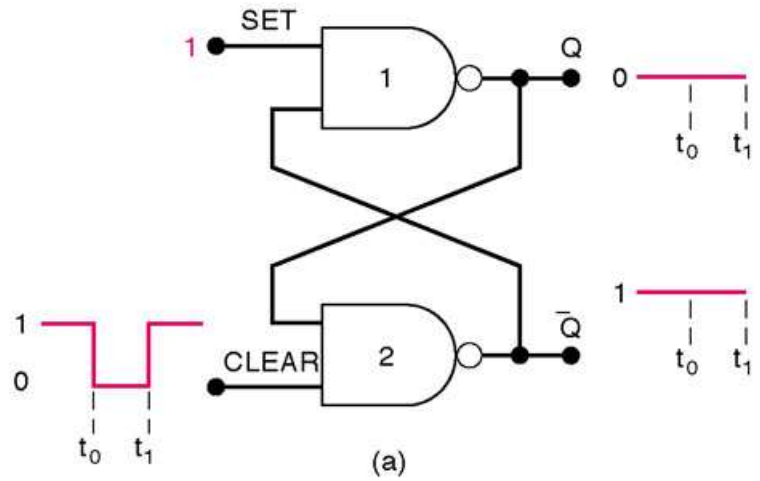


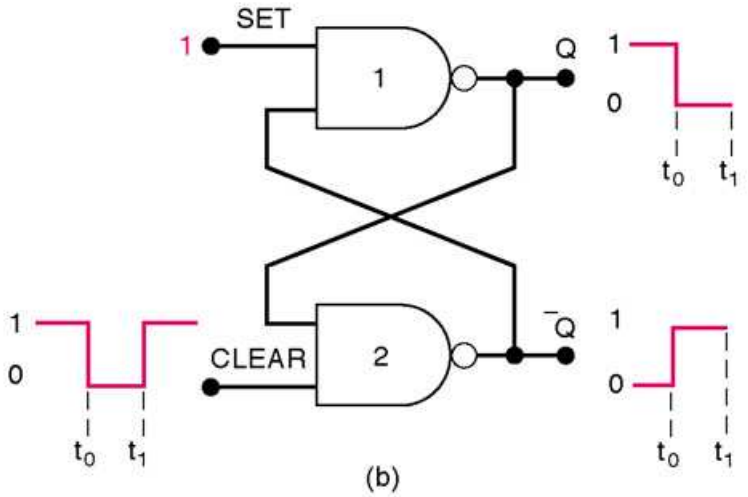
A	B	NAND
0	0	1
0	1	1
1	0	1
1	1	0

Nos dois casos a saída Q termina em nível ALTO.

Pulsando a entrada CLEAR para o nível baixo

(a) Q=0 antes do pulso na entrada CLEAR;
(b) Q=1 antes do pulso na entrada CLEAR.

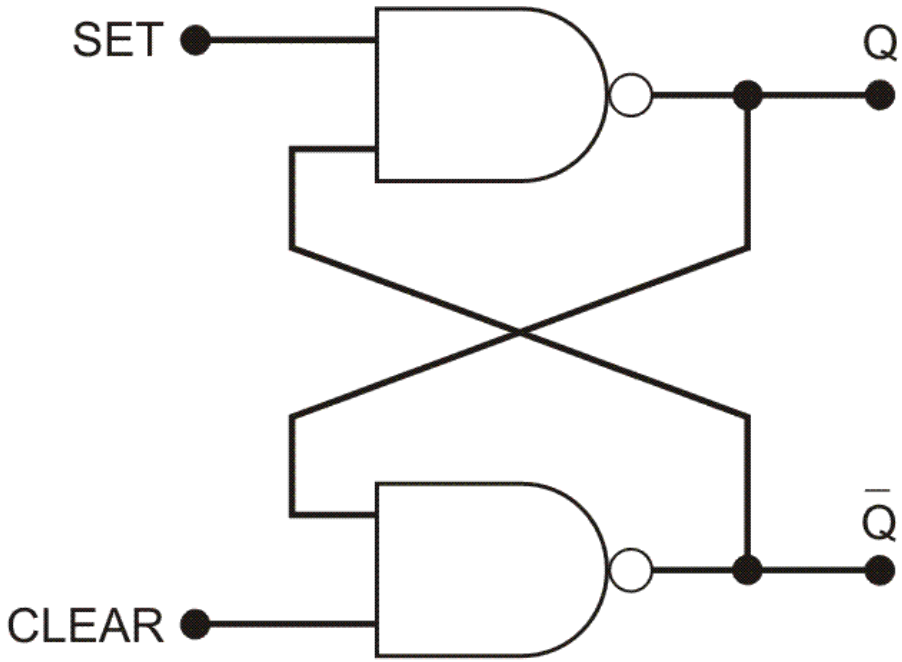




A	B	NAND
0	0	1
0	1	1
1	0	1
1	1	0

Nos dois casos a saída Q termina em nível BAIXO.

Tabela-verdade do FF S-C com portas NAND



Q_0 é o estado anterior

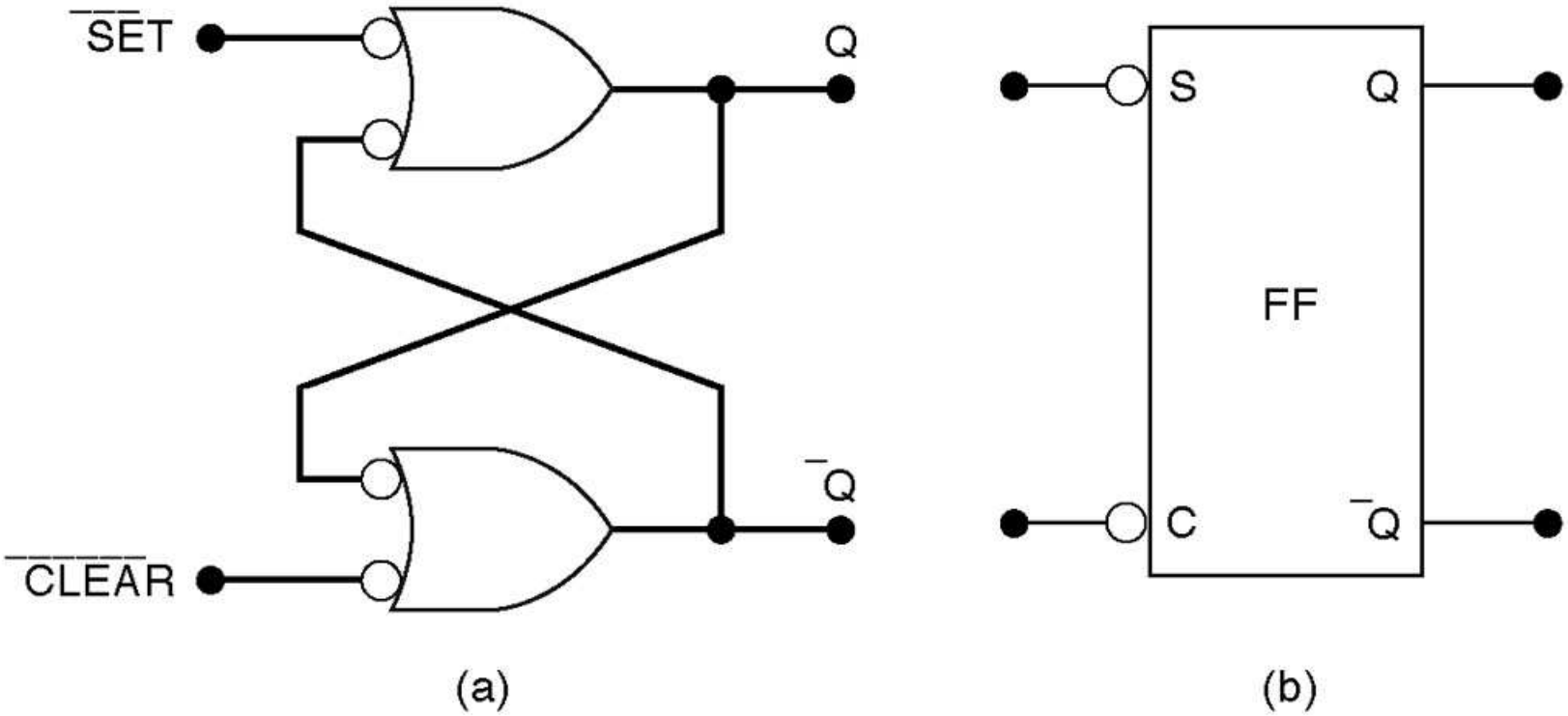
Set	Clear	Saída
1	1	Q_0
0	1	Q = 1
1	0	Q = 0
0	0	Inválida*

*Produz $Q = \overline{Q} = 1$

A	B	NAND
0	0	1
0	1	1
1	0	1
1	1	0

O caso em que SET=CLEAR=0 produz resultados imprevisíveis, uma vez que as duas saídas serão forçadas para nível alto.
NAO SE UTILIZA O LATCH NESTA CONDIÇÃO.

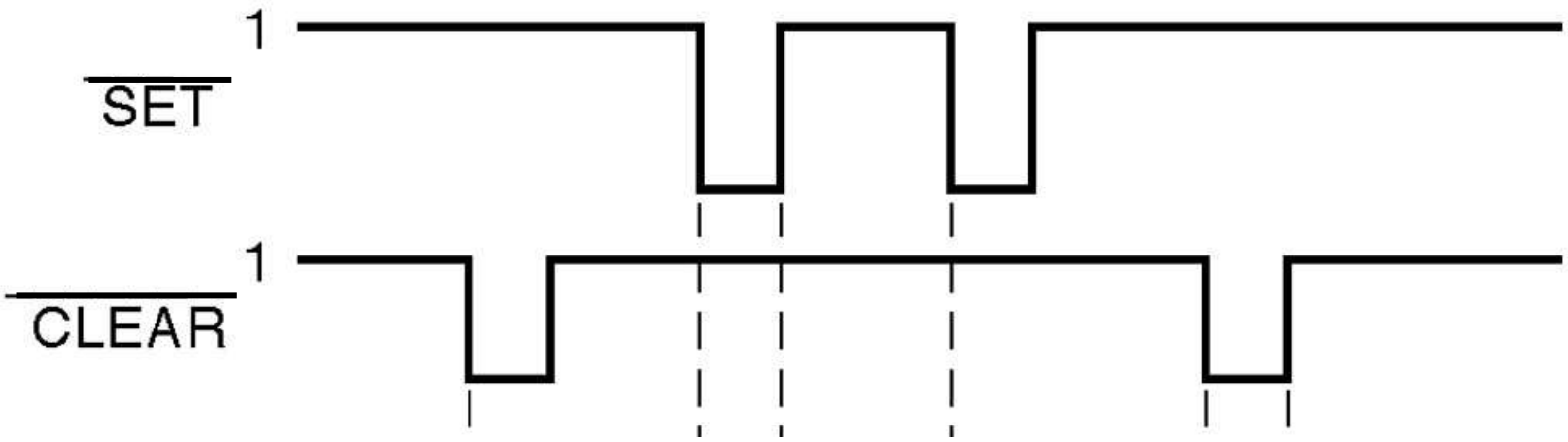
Implementação alternativa de um latch NAND e símbolo simplificado



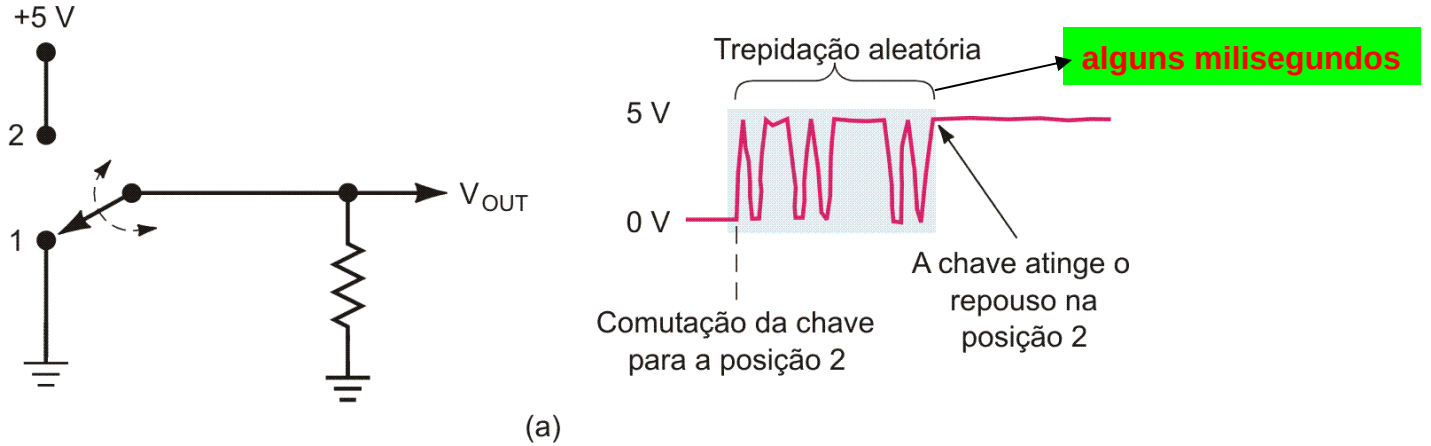
Ao se ligar um FF, existem chances iguais do estado inicial ser baixo e alto.

Fatores como **atrasos internos de propagação**, **capacitâncias parasitas** e **carga externa** definem o estado inicial.

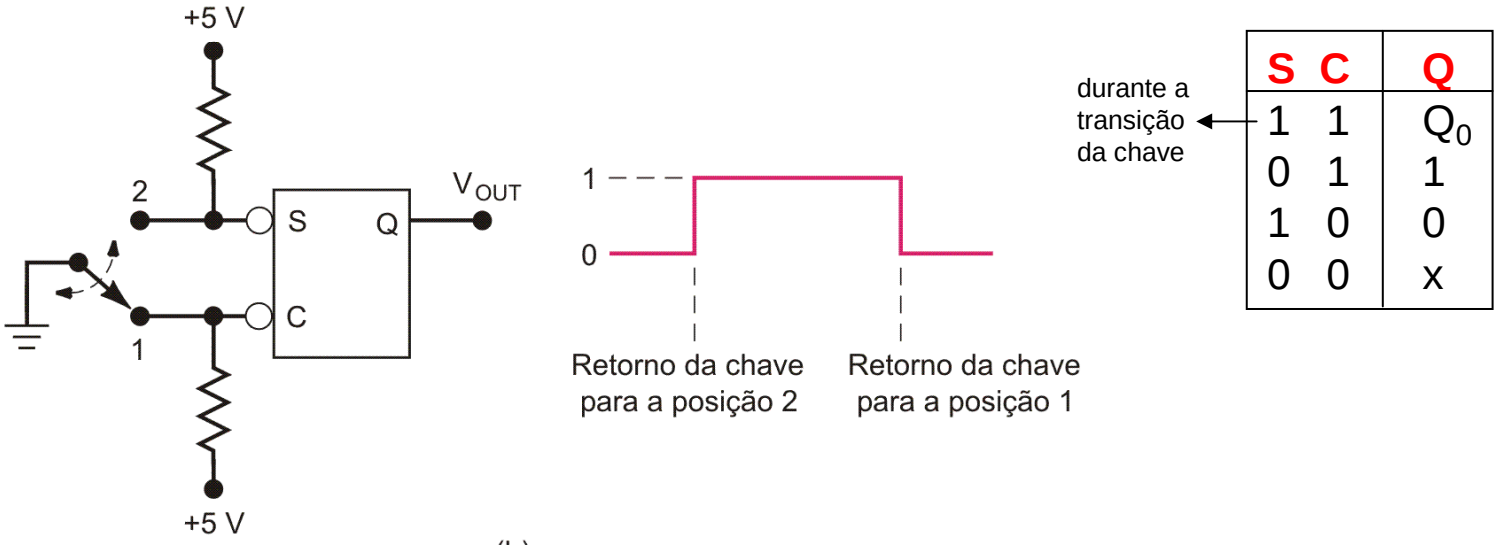
Exercício: determinar a forma de onda na saída Q do FF S-C



Exemplo de aplicação: chave sem trepidação



(a)

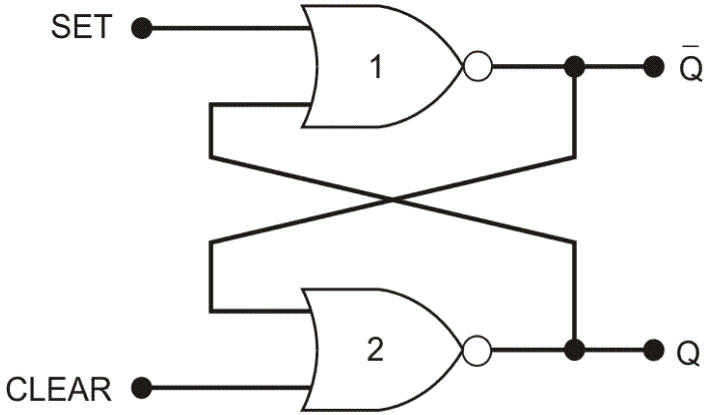


S	C	Q
1	1	Q_0
0	1	1
1	0	0
0	0	x

(b)

(a) A trepidação do contato mecânico gera múltiplas transições na tensão Vout.
(b) FF S-C NAND usado para eliminar as múltiplas transições.

FF S-C com portas NOR, tabela-verdade, símbolo simplificado

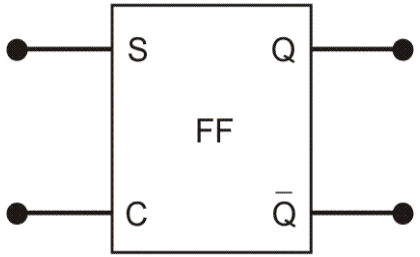


(a)

Set	Clear	Saída
0	0	Não muda
1	0	Q = 1
0	1	Q = 0
1	1	Inválida*

*produz $Q = \bar{Q} = 0$

(b)

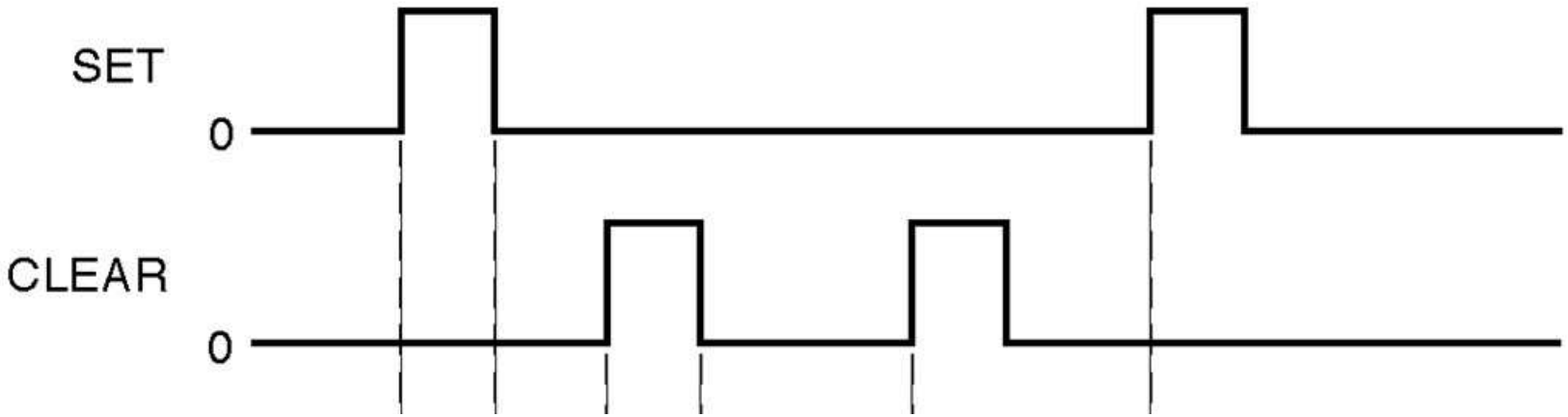


(c)

A	B	NOR
0	0	1
0	1	0
1	0	0
1	1	0

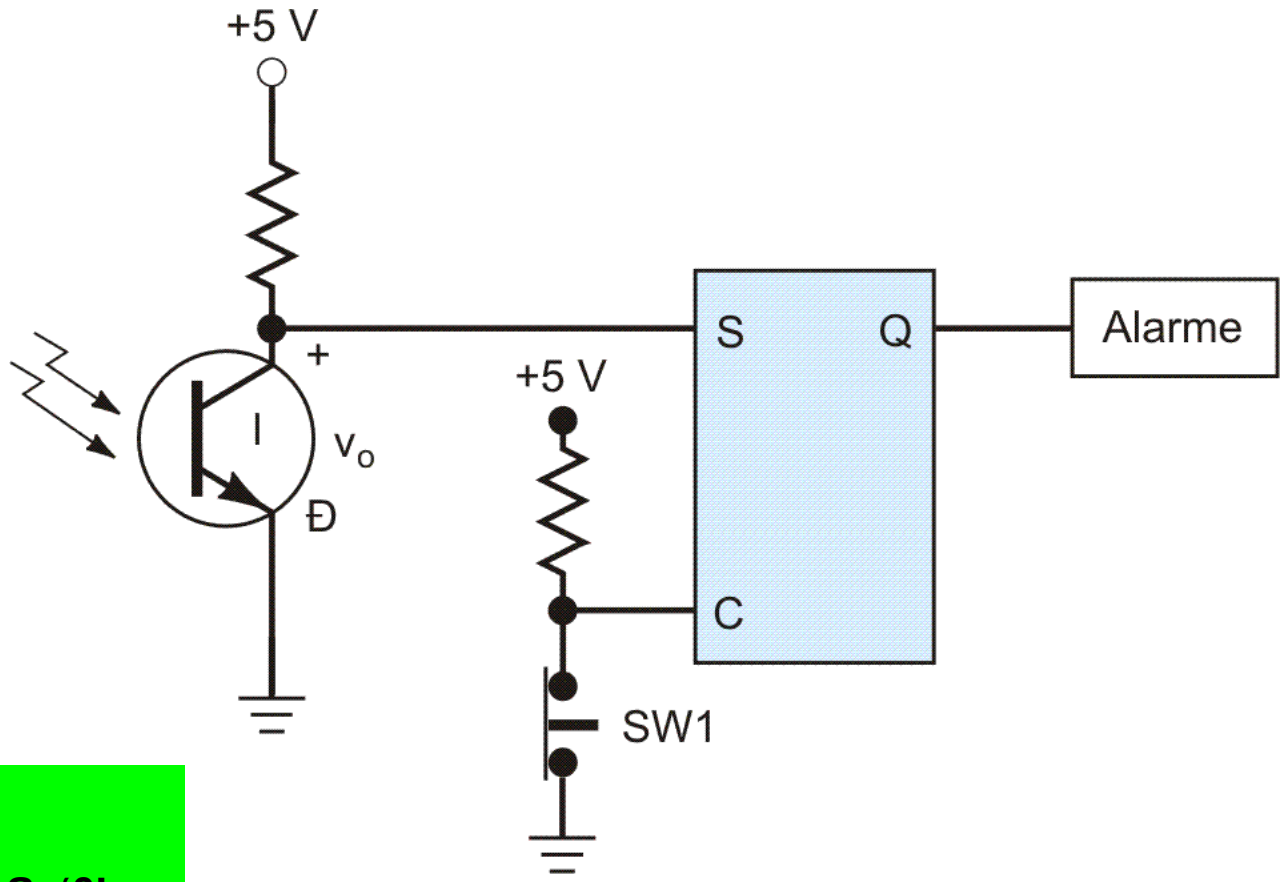
O **FF S-C NOR** opera com pulsos ativos em nível alto nas entradas SET e CLEAR.

Exercício: determinar a forma de onda na saída Q



Exemplo de aplicação do FF S-C NOR

disparo de alarme pela interrupção de um feixe de luz



Fototransistor:

com luz ⇒ saturado ⇒ S=‘0’

sem luz ⇒ cortado ⇒ S=‘1’ ⇒ ***LIGA ALARME***

→ até aqui

circuitos **Assíncronos**

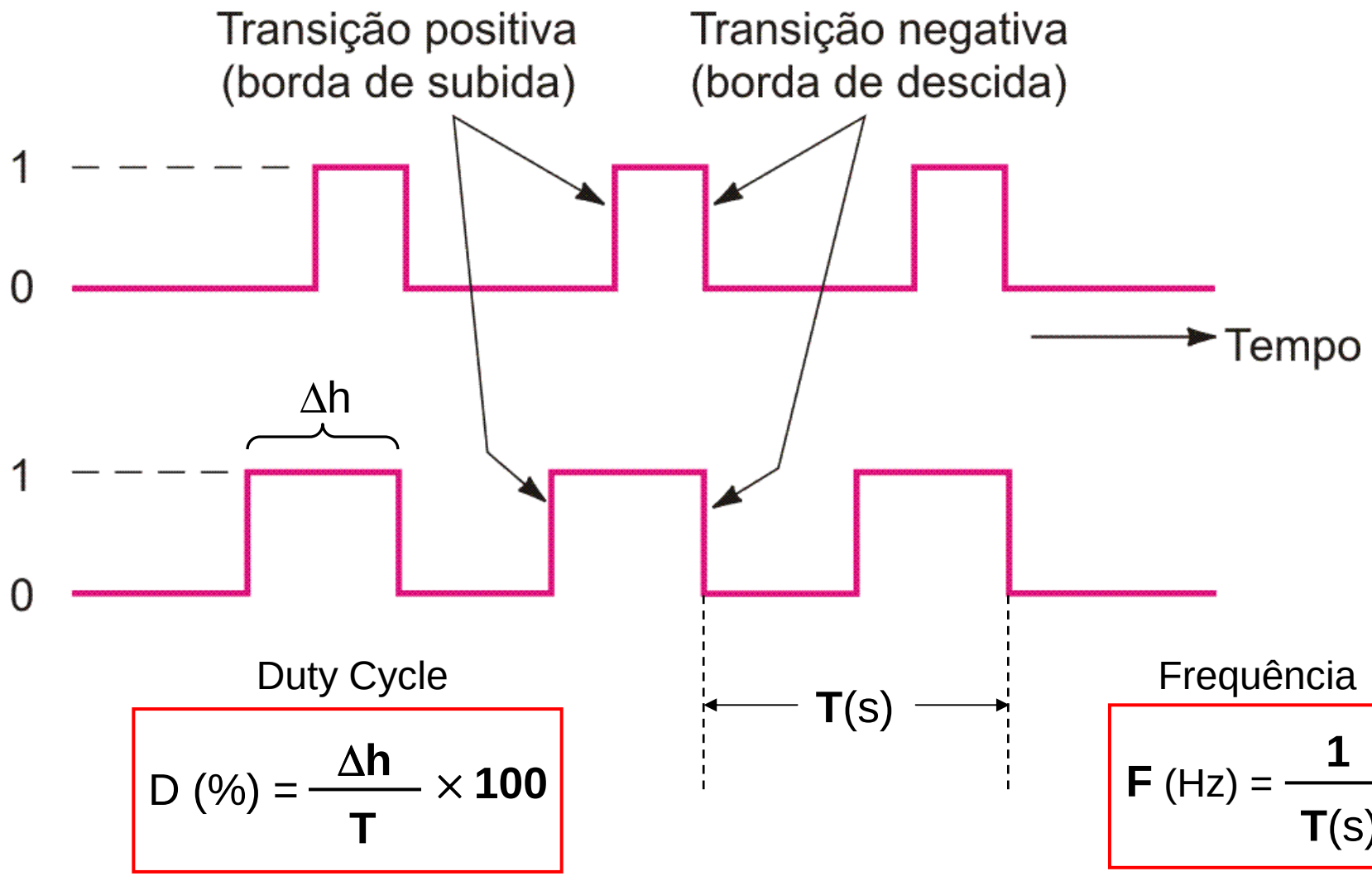
(sem clock)

a partir daqui →

circuitos SÍNCRONOS

(com clock)

Sinais de clock – circuitos síncronos



Duty Cycle

$$D\ (\%) = \frac{\Delta h}{T} \times 100$$

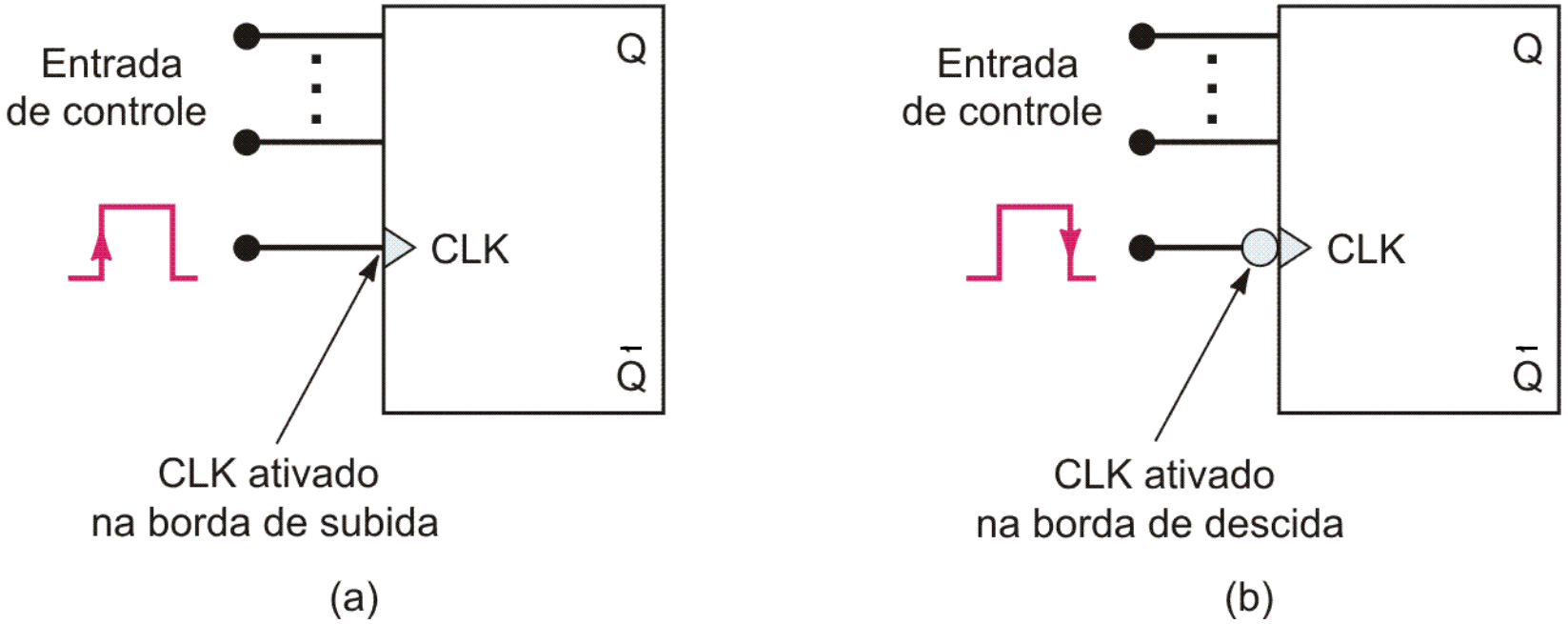
Frequência

$$F\ (Hz) = \frac{1}{T(s)}$$

Flip-flop síncrono com entrada de clock (CLK)

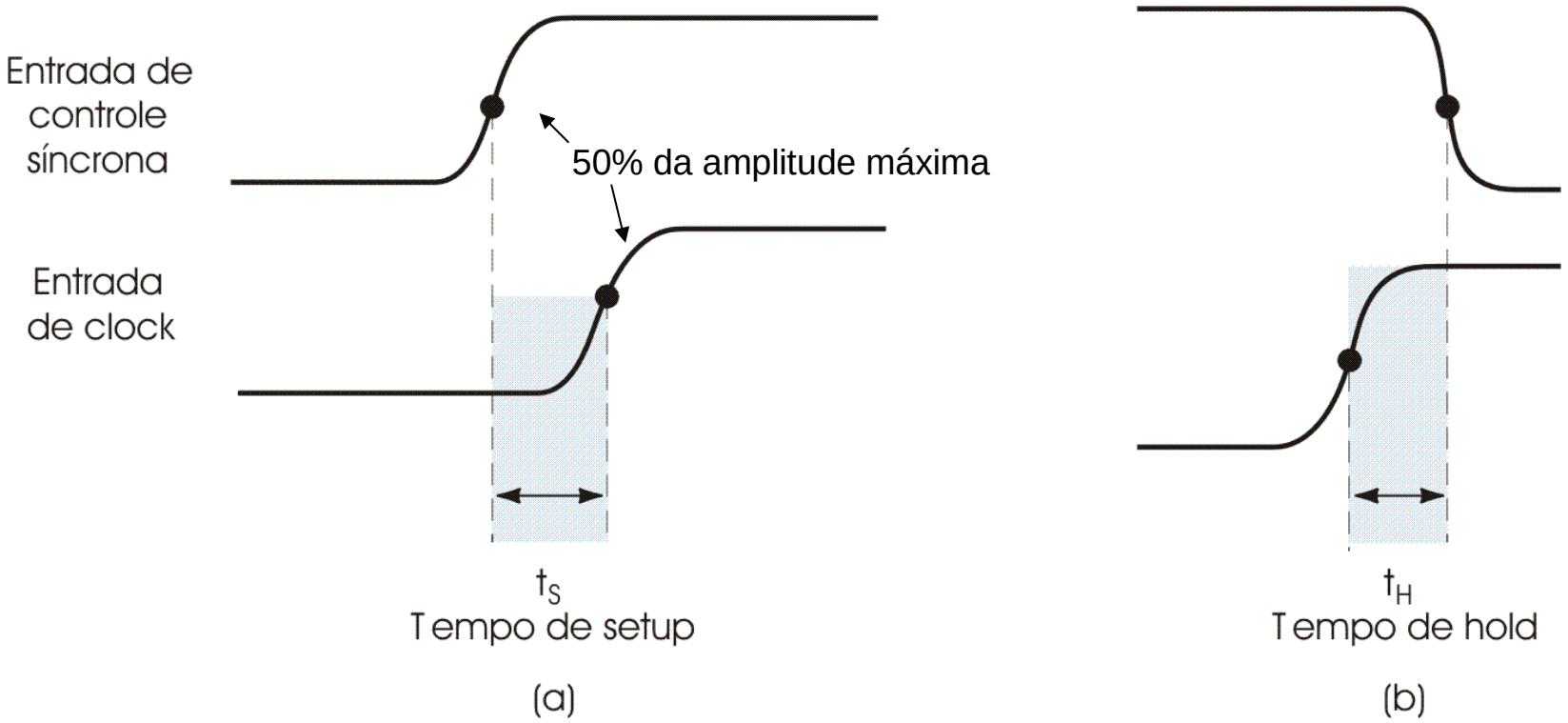
(a) por borda de subida do clock

(b) por borda de descida do clock

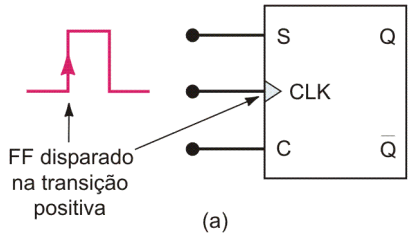


As entradas de controle determinam o efeito da transição ativa do ***clock***.

Setup Time e Hold Time



(a) Flip-flop SC síncrono com a borda positiva do pulso de clock; (b) Tabela-verdade; (c) Forma de onda típica.

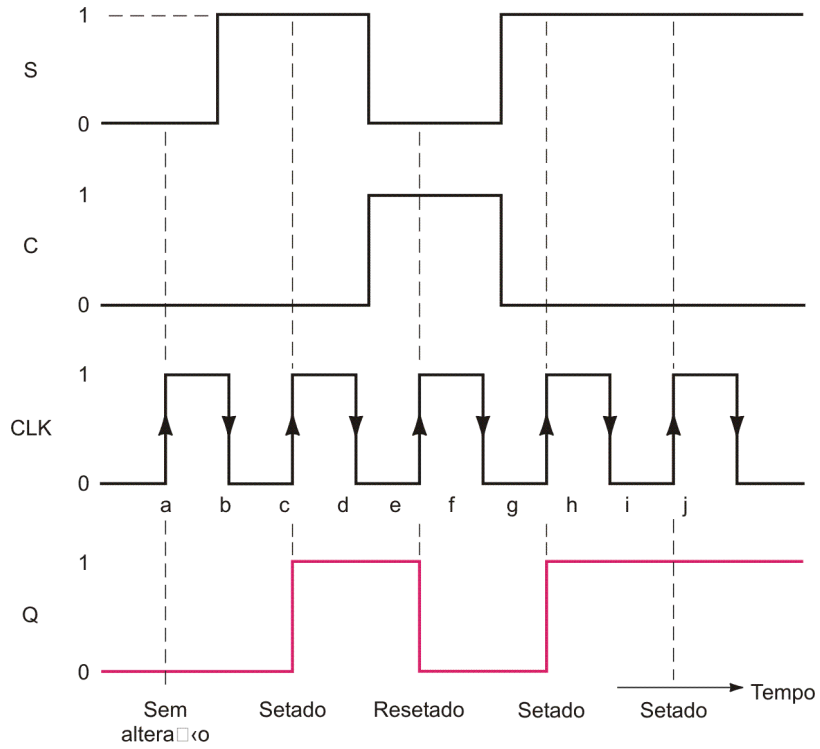


(a)

Entradas			Saídas
S	C	CLK	Q
0	0	_	Q_0 (não muda)
1	0	_	1
0	1	_	0
1	1	_	Ambígua

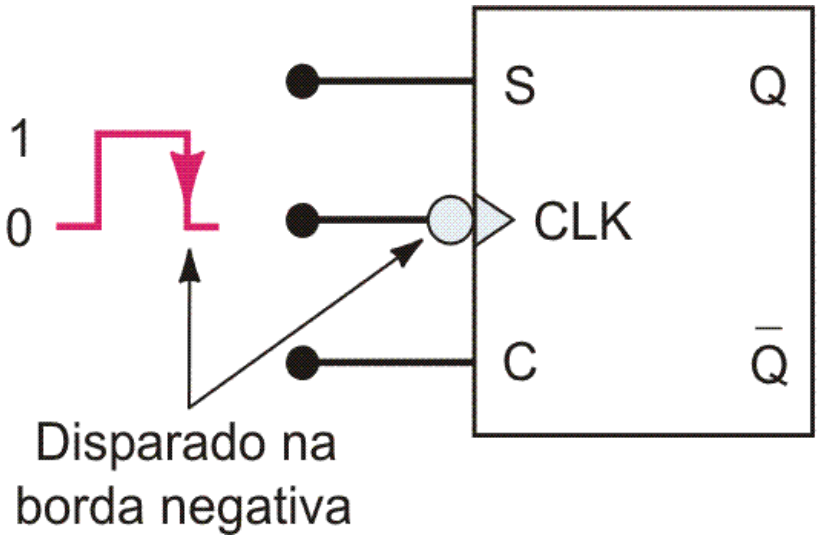
Q_0 é o nível lógico da saída antes da borda de subida (↑) do CLK.
A borda de descida (↓) do CLK não gera mudança na saída Q.

(b)



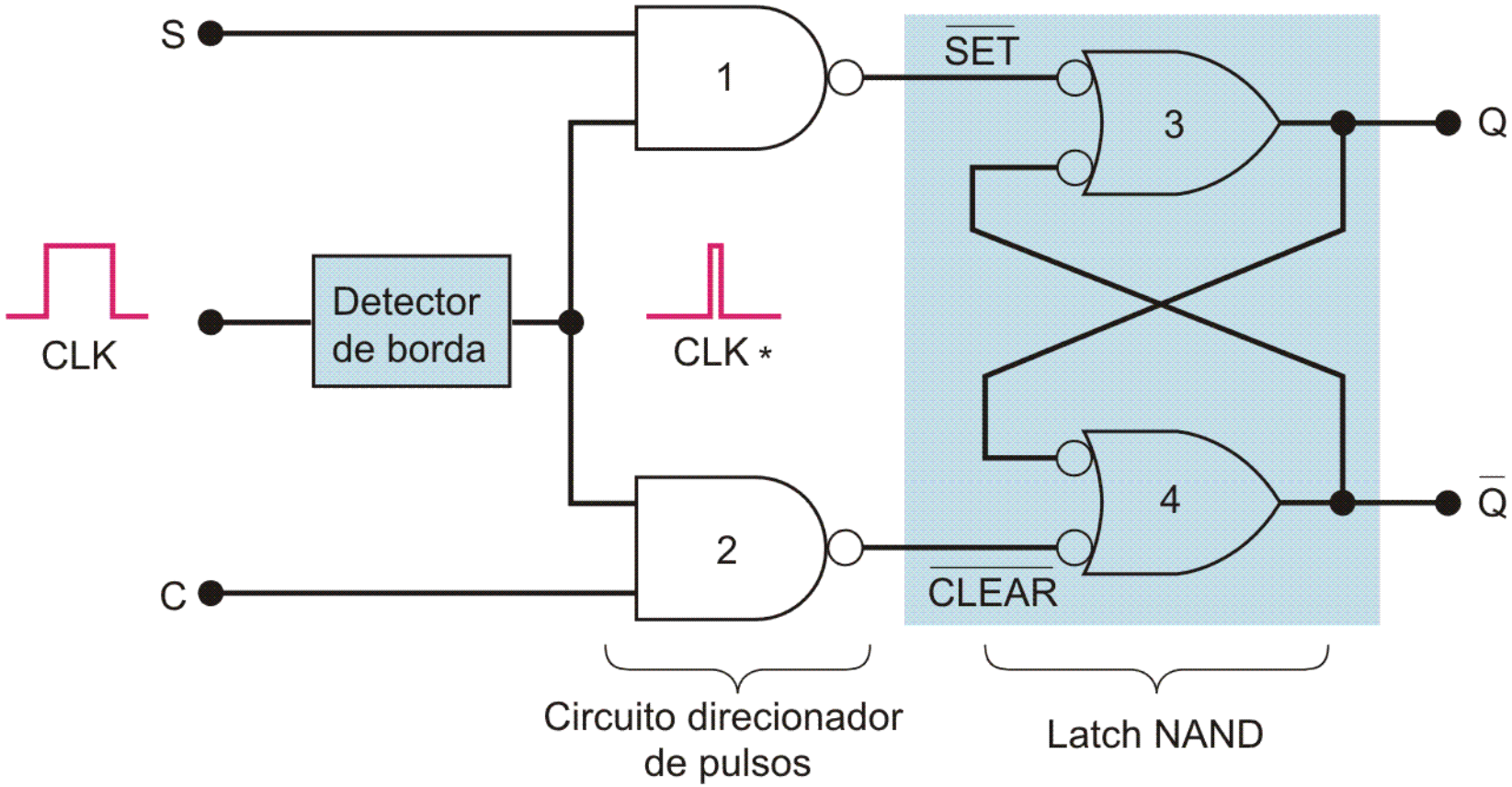
(c)

Flip-flop SC síncrono com a borda negativa do pulso de clock e Tabela-Verdade.



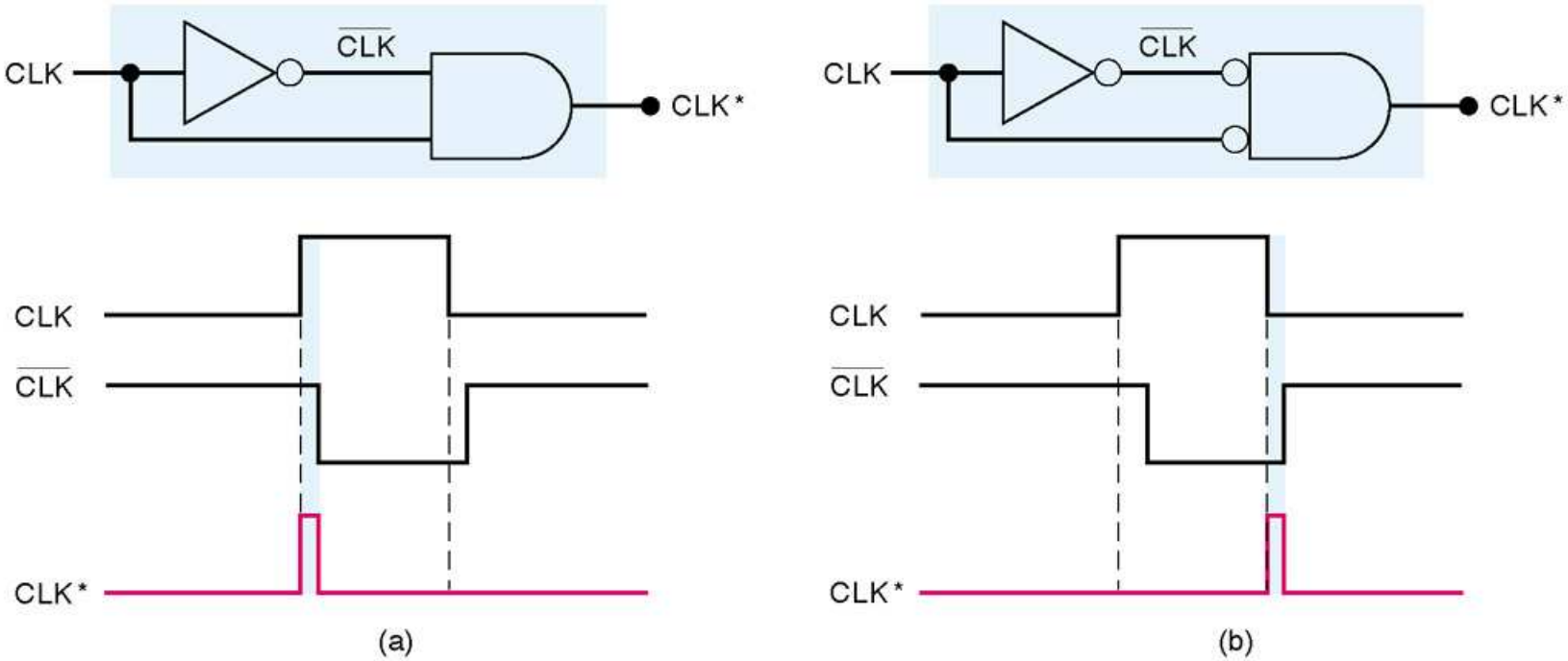
Entradas			Saídas
S	C	CLK	Q
0	0	–	Q_0 (não muda)
1	0	–	1
0	1	–	0
1	1	–	Ambígua

Versão simplificada do circuito interno de um flip-flop SC síncrono.



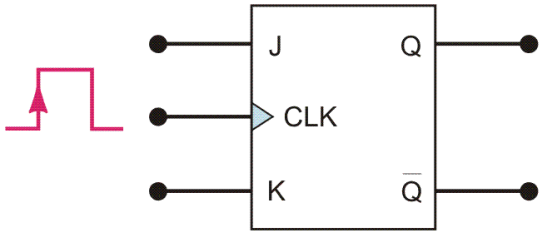
Implementação do circuito detector de borda

(a) borda positiva. (b) borda negativa.



A duração dos pulsos CLK* é normalmente de 2 a 5 nano-segundos e corresponde ao atraso da porta inversora.

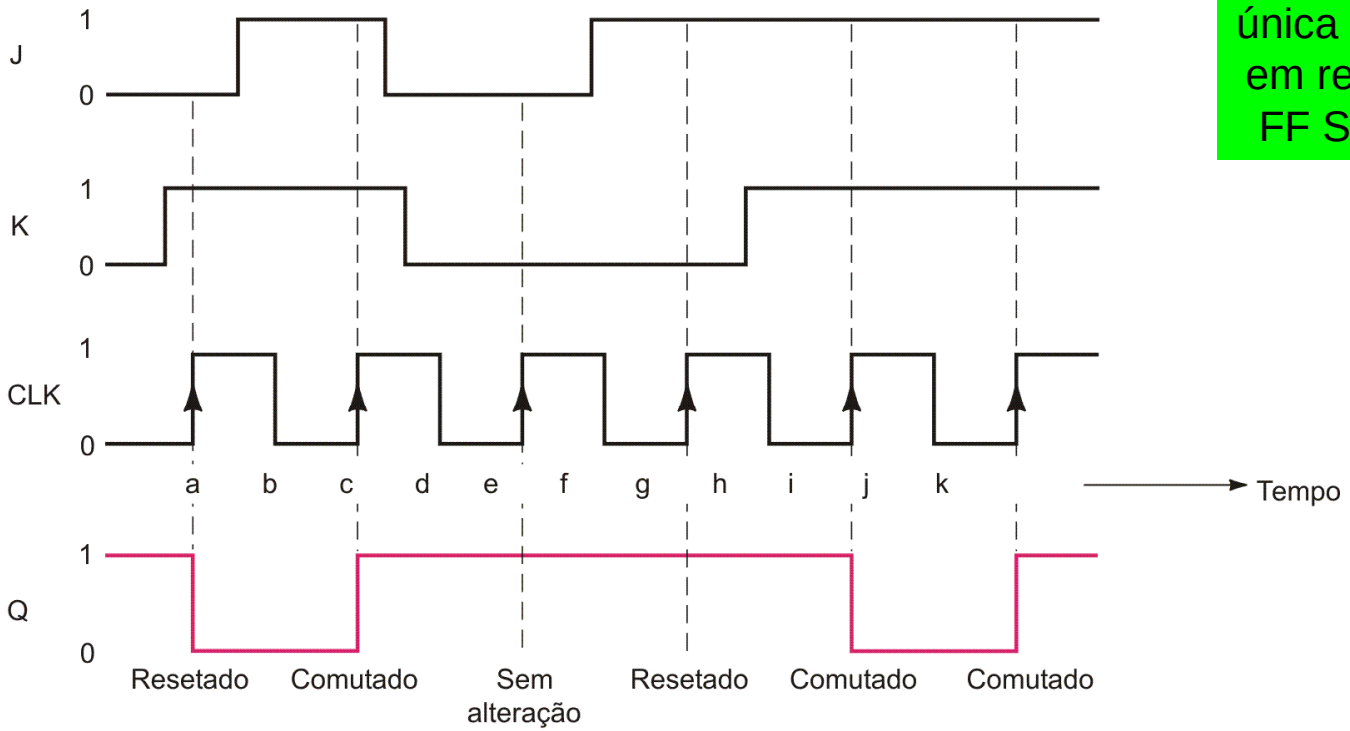
Flip-flop JK síncrono com a borda positiva do clock



(a)

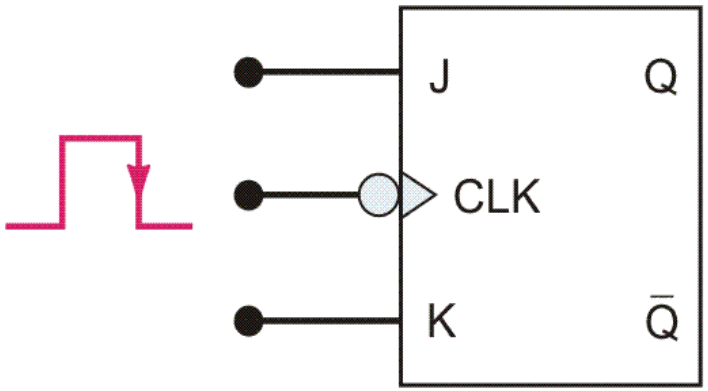
J	K	CLK	Q
0	0	_	Q_0 (não muda)
1	0	_	1
0	1	_	0
1	1	_	$\overline{Q_0}$ (comuta)

única diferença em relação ao FF S-C NOR



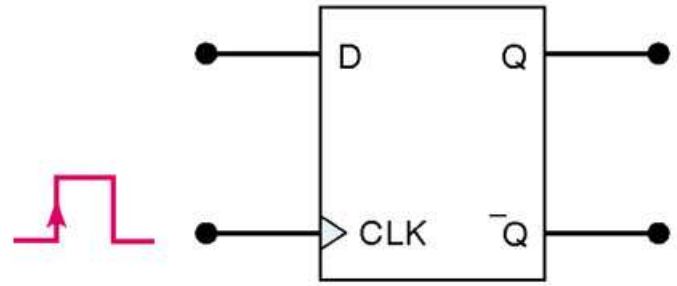
(b)

Flip-flop JK síncrono com a transição negativa do clock



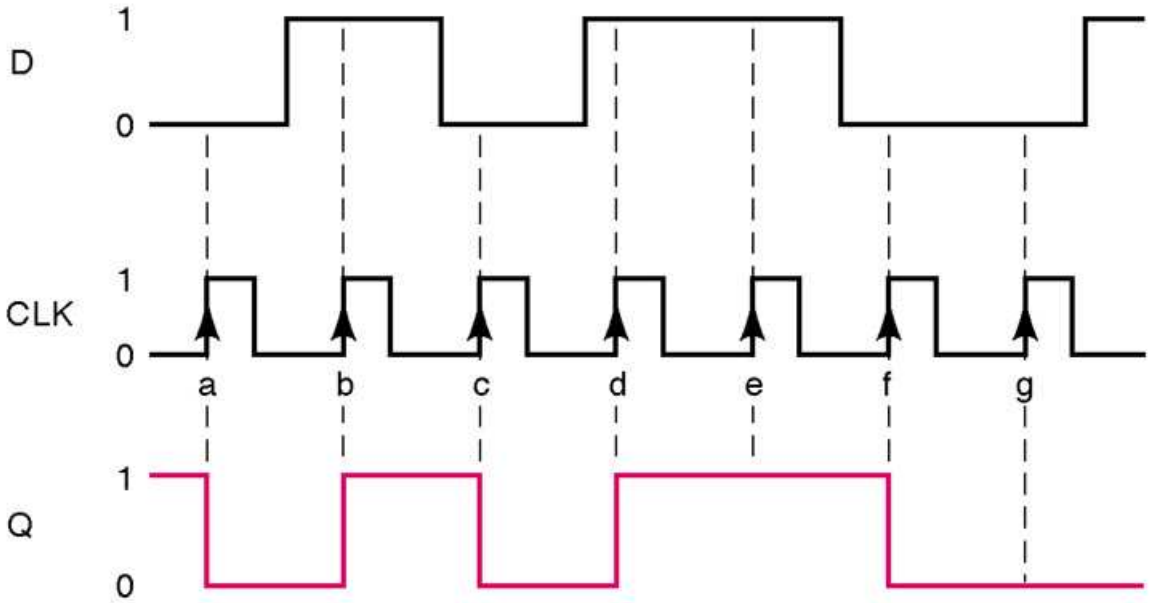
J	K	CLK	Q
0	0	–	Q_0 (não muda)
1	0	–	1
0	1	–	0
1	1	–	$\overline{Q_0}$ (comuta)

Flip-flop D síncrono com a transição positiva do clock



D	CLK	Q
0	↑	0
1	↑	1

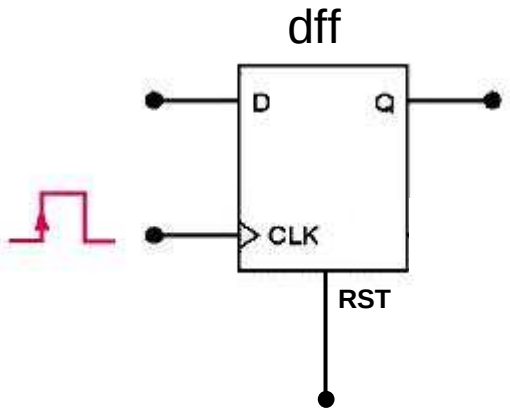
(a)



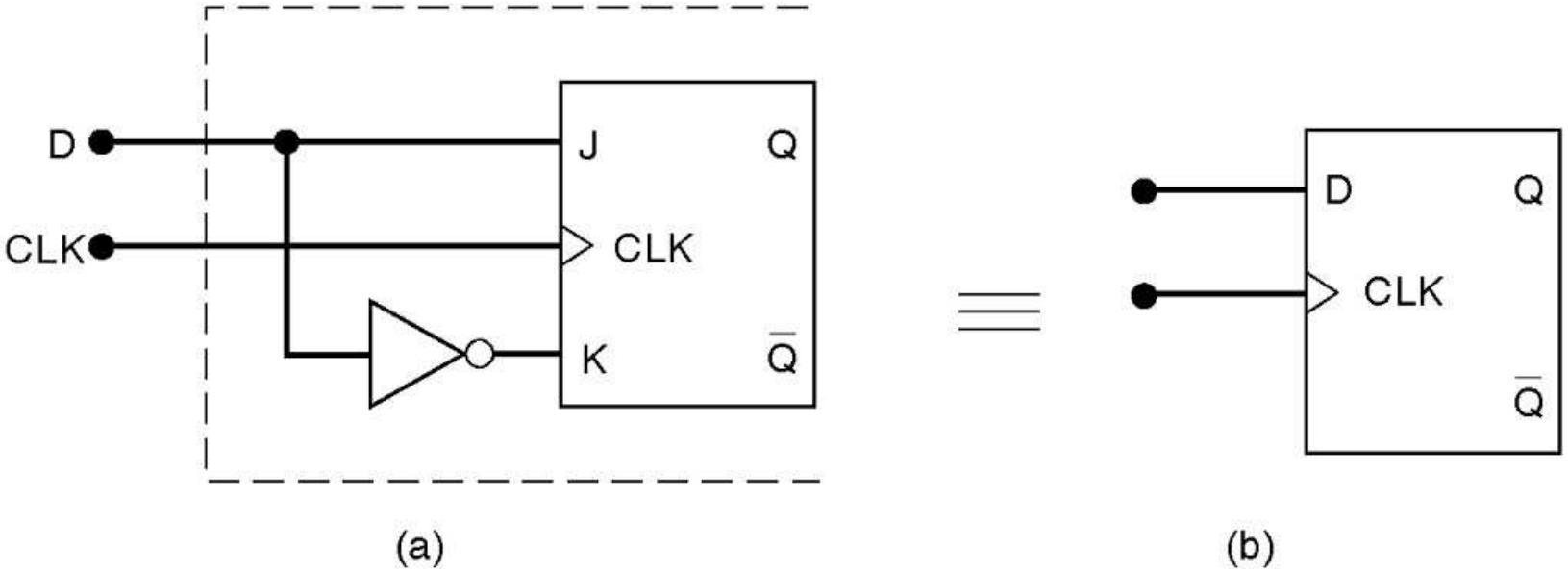
(b)

Código VHDL para um flip-flop D síncrono com a borda positiva do clock

```
1  ------------------------------------
2  LIBRARY ieee;
3  USE ieee.std_logic_1164.all;
4  ------------------------------------
5  ENTITY dff IS
6     PORT (d, clk, rst: IN STD_LOGIC;
7           q: OUT STD_LOGIC);
8  END dff;
9  ------------------------------------
10 ARCHITECTURE behavior OF dff IS
11 BEGIN
12    PROCESS (clk, rst)
13    BEGIN
14       IF (rst='1') THEN
15          q <= '0';
16       ELSIF (clk'EVENT AND clk='1') THEN
17          q <= d;
18    END IF;
19    END PROCESS;
20 END behavior;
21 ------------------------------------
```

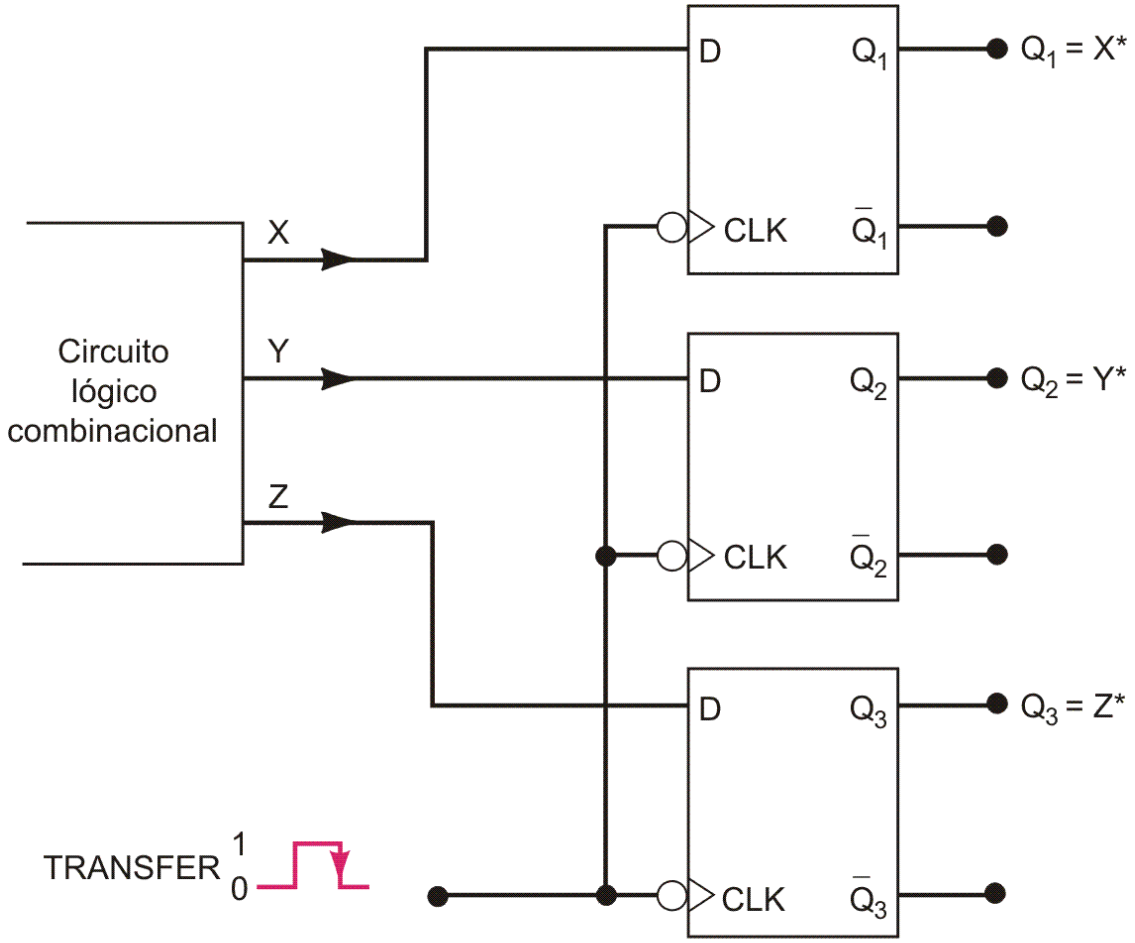


Implementação de um flip-flop D a partir de um flip-flop JK



J	K	CLK	Q
0	0	⎺	Q_0 (não muda)
1	0	⎺	1
0	1	⎺	0
1	1	⎺	$\overline{Q_0}$ (comuta)

Transferência de dados em paralelo utilizando flip-flop D

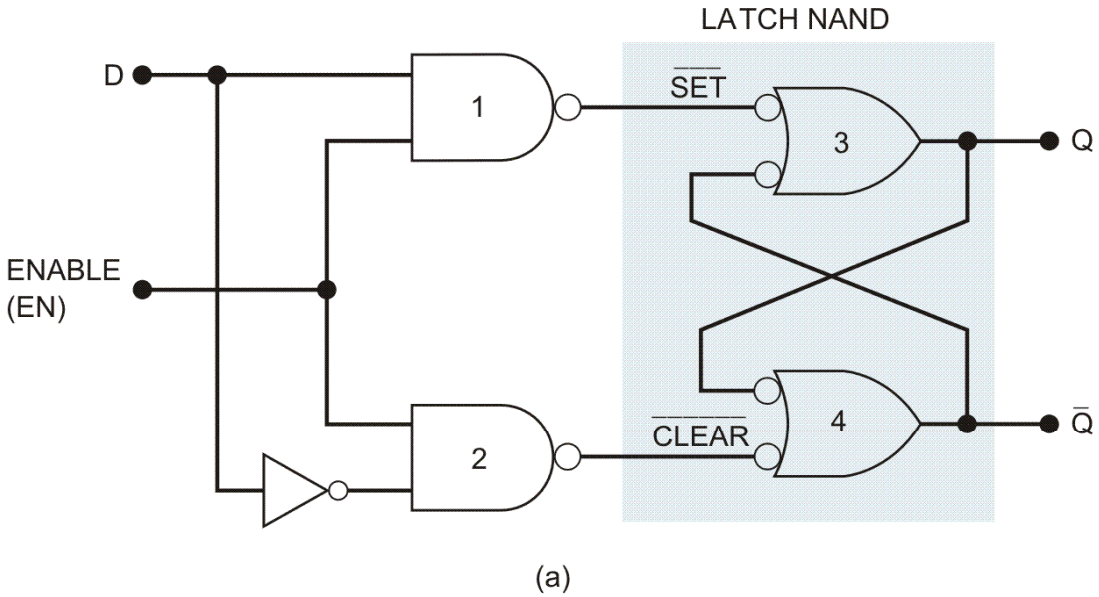


*Após a ocorrência da borda de descida.

princípio de circuitos **SÍNCRONOS**

Latch D transparente

(a) estrutura, (b) tabela-verdade, (c) símbolo lógico

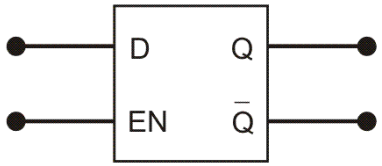


(a)

Entradas		Saída
EN	D	Q
0	X	Q_0 (não muda)
1	0	0
1	1	1

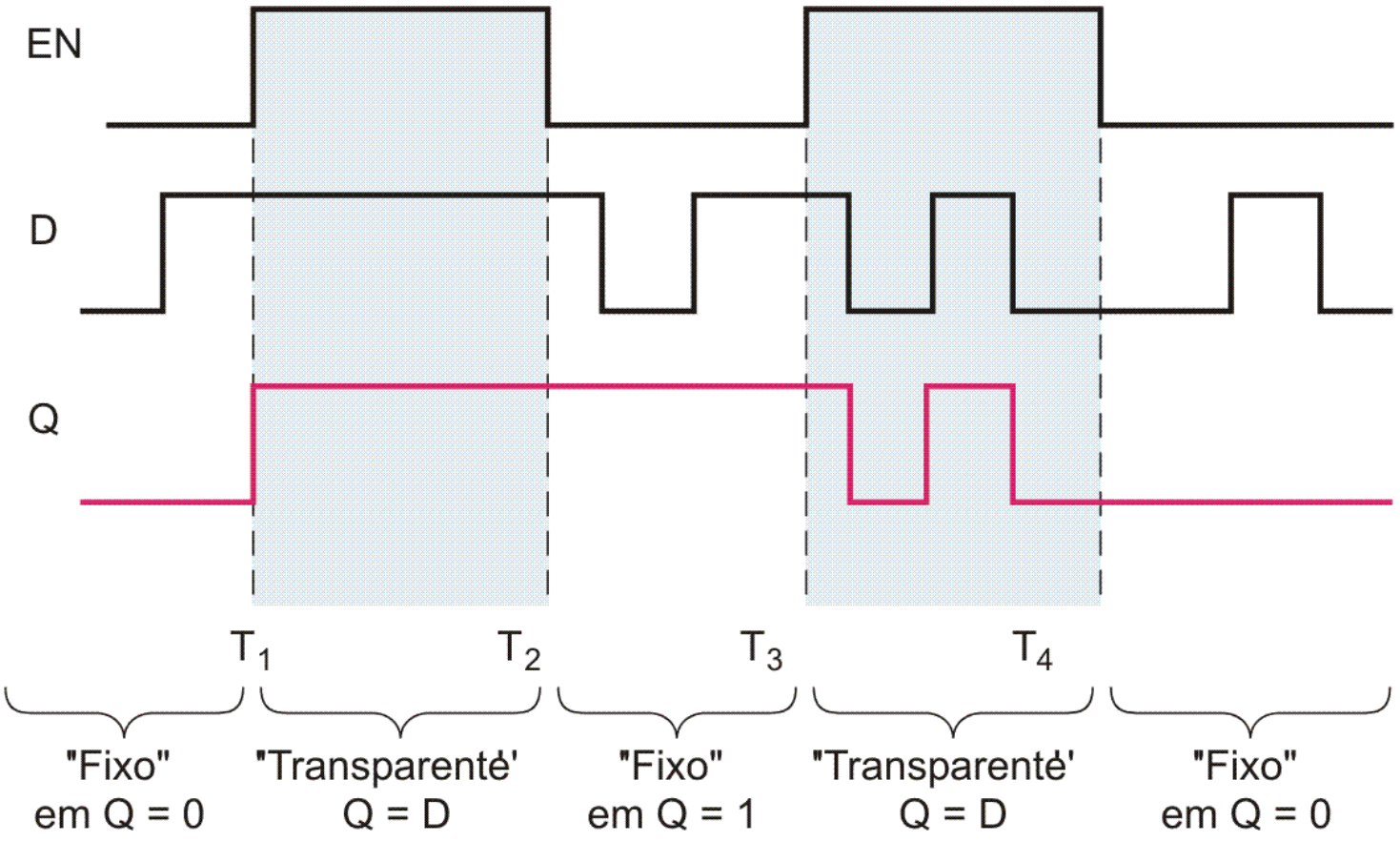
"X" indica a condição "don't care"
Q_0 é o estado de Q imediatamente antes de EN ir para o nível BAIXO

(b)



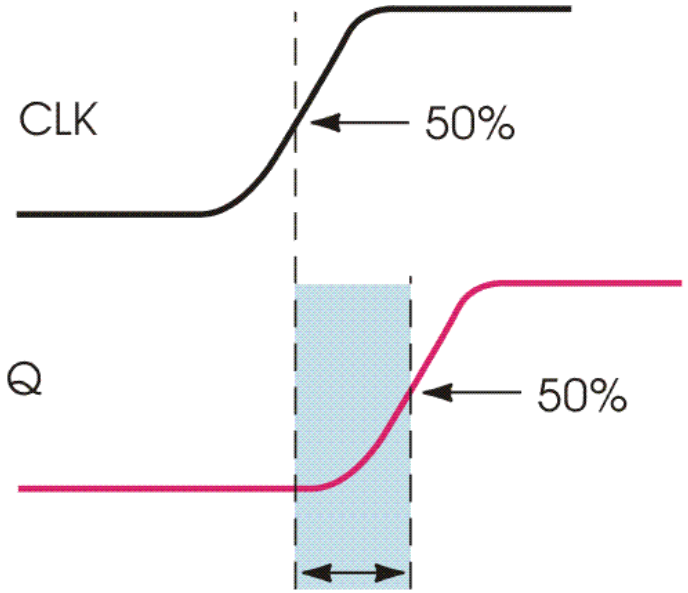
(c)

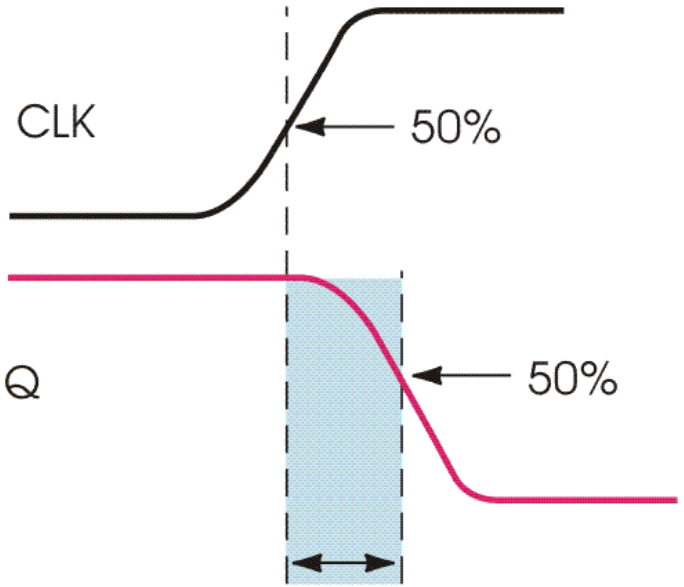
Formas de onda mostrando os dois modos de operação de um latch D transparente



Atraso de propagação em FFs síncronos

Atraso de tempo entre a transição ativa do *clock* e o instante em que a saída comuta.

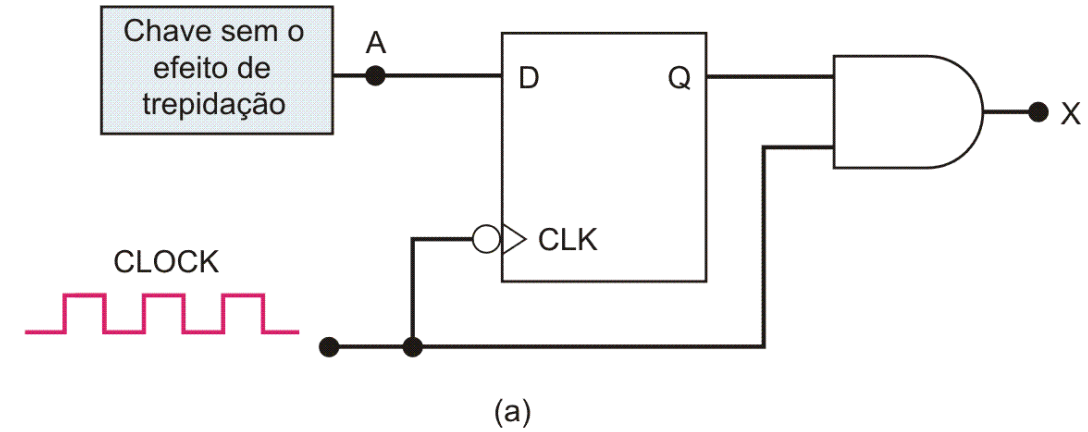




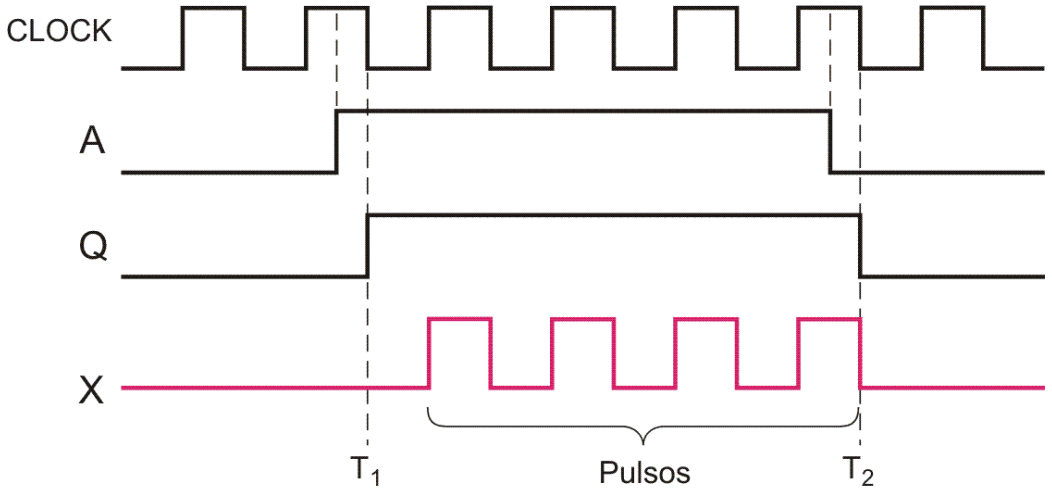
Outros parâmetros em FFs síncronos

- Frequência máxima de clock (F_{max})
- Tempos de duração do pulso de *clock* (níveis alto e baixo)
- Largura de pulsos assíncronos (PRESET, CLEAR)
- Tempos de transição do *clock*

APLICAÇÃO: Flip-flop D sincronizando a habilitação de uma porta AND para o sinal de clock

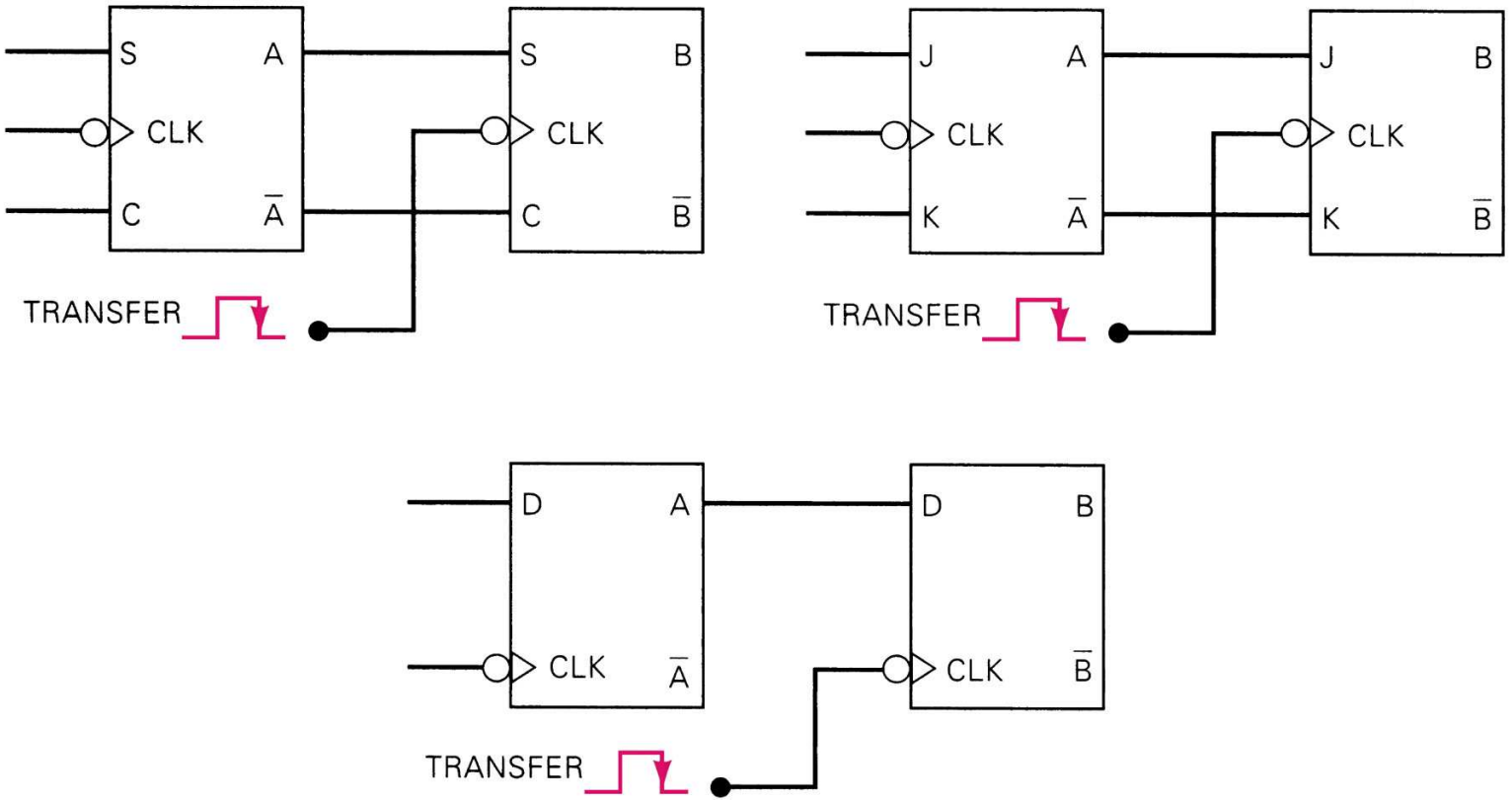


(a)



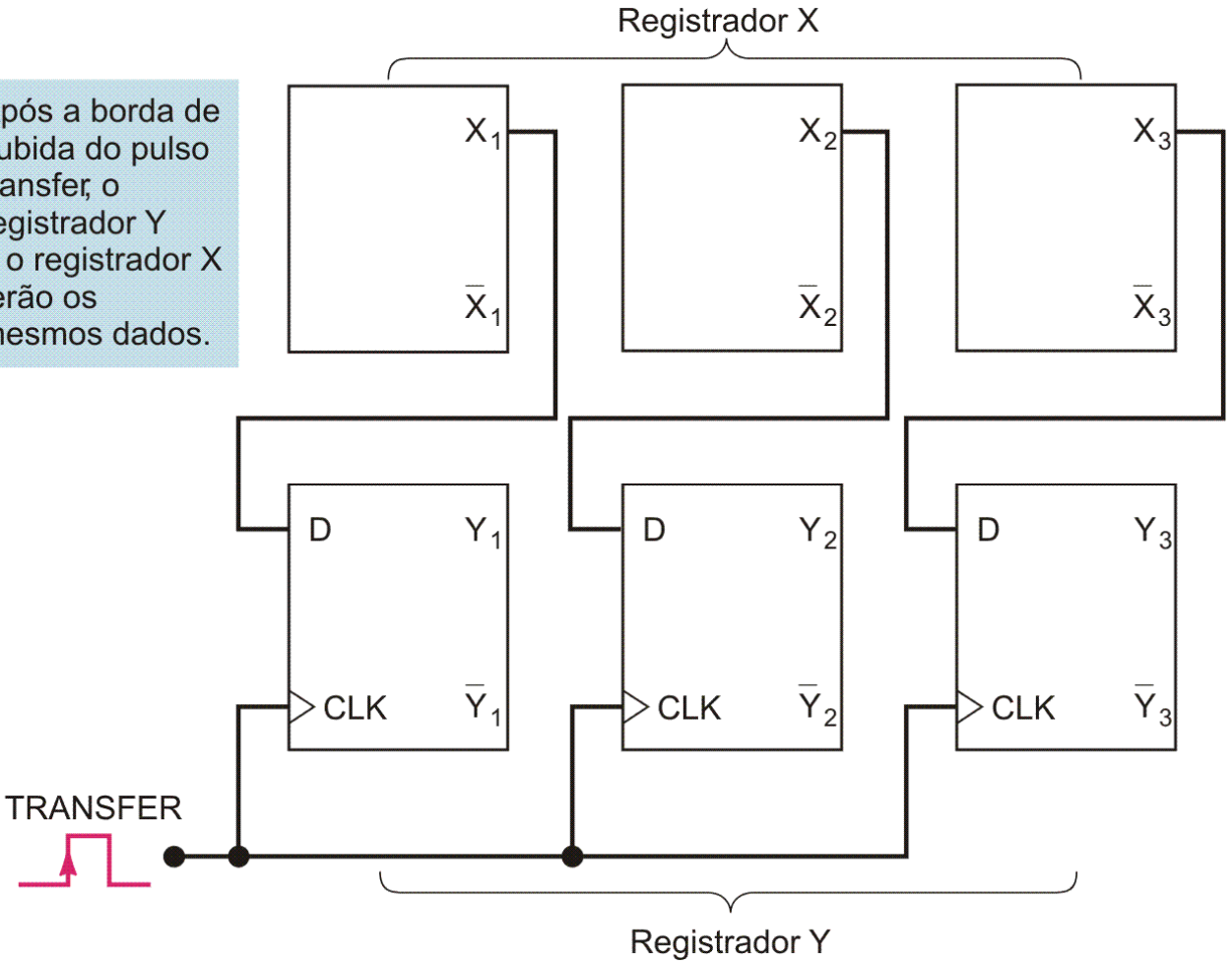
(b)

Transferência síncrona de dados realizada por diversos tipos de FFs

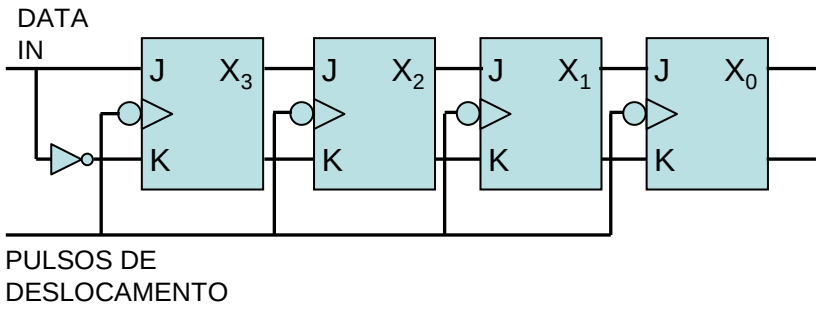


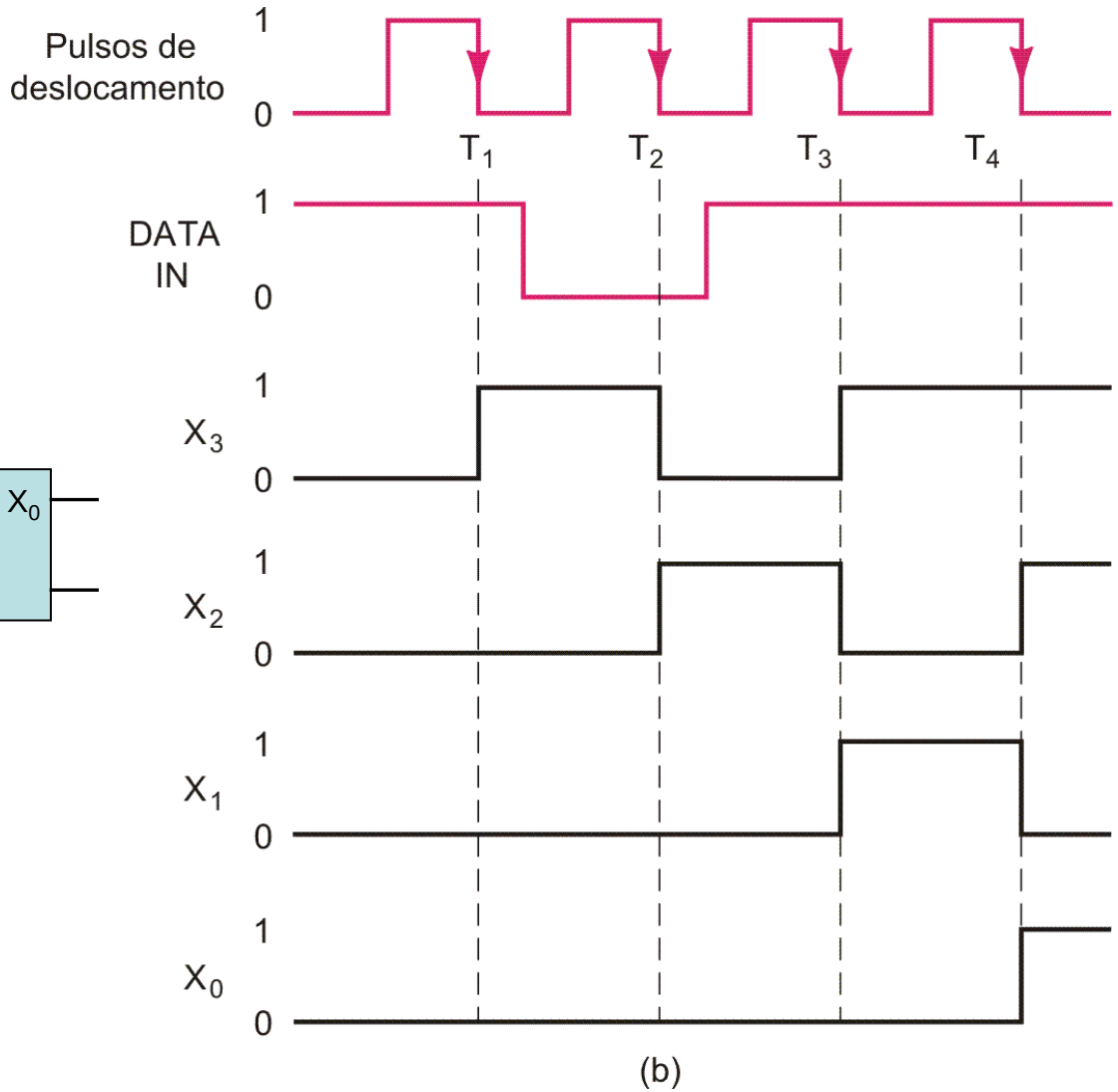
Transferência paralela do conteúdo do registrador *X* para o registrador *Y*

Nota: Após a borda de subida do pulso transfer, o registrador Y e o registrador X terão os mesmos dados.

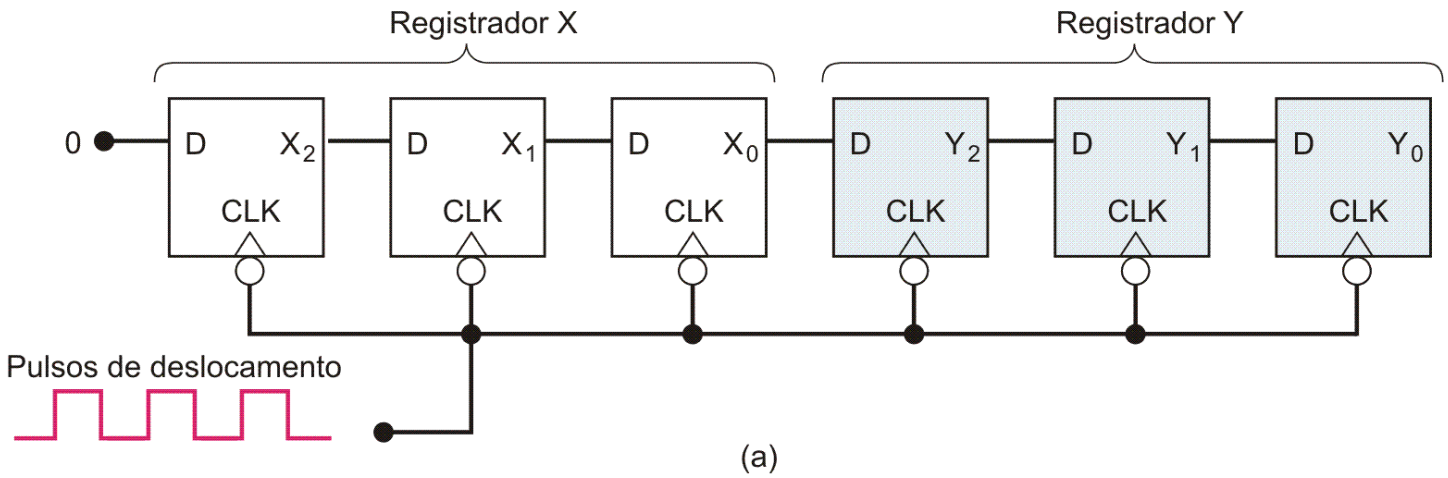


Registrador de deslocamento de quatro bits





Transferência serial de dados de um registrador *X* para um registrador *Y*

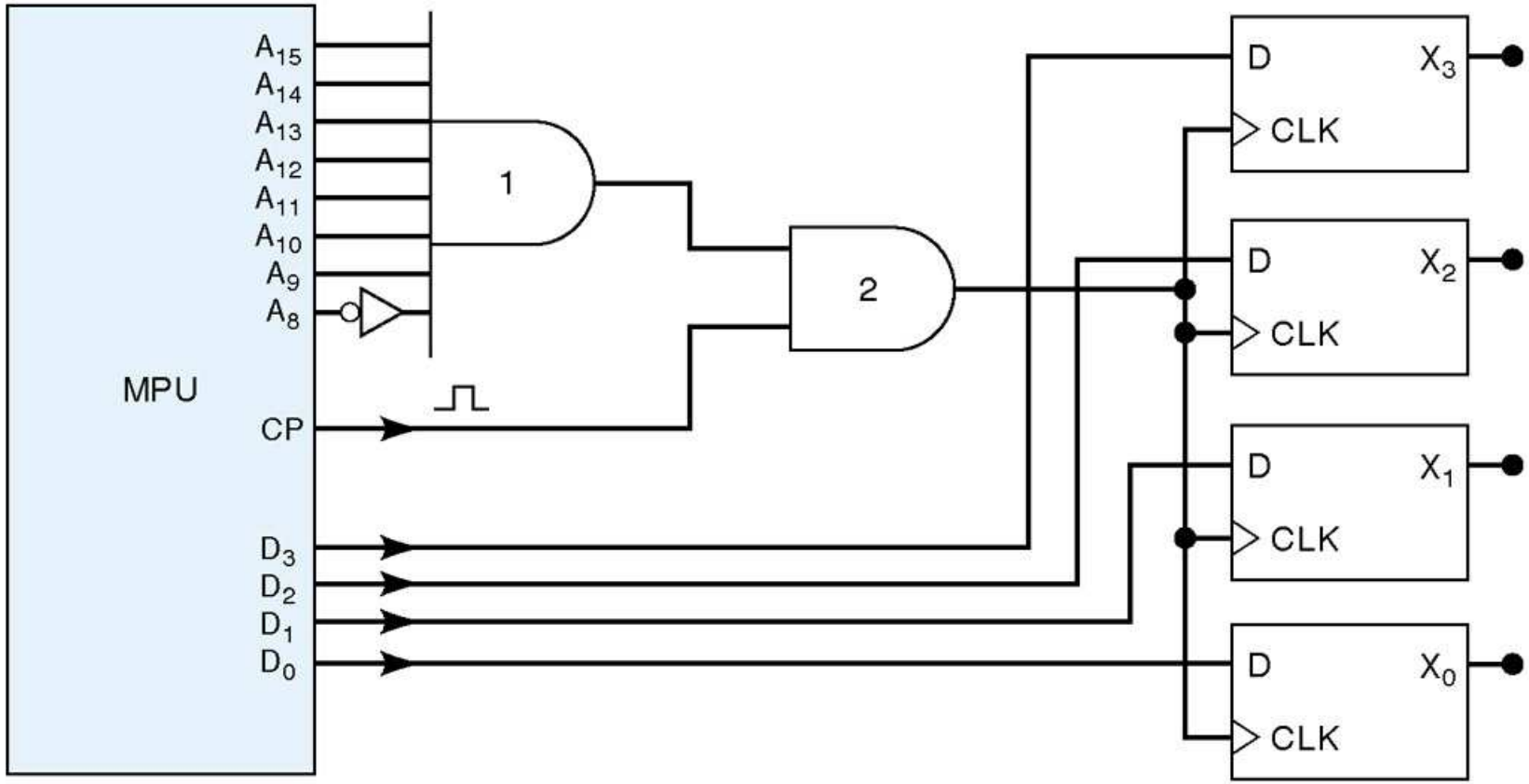


(a)

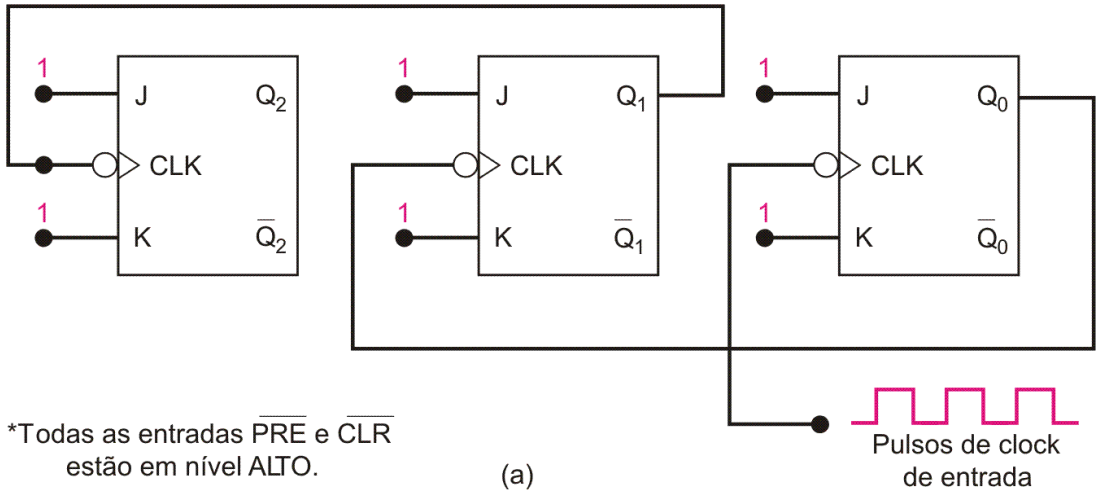
X_2	X_1	X_0	Y_2	Y_1	Y_0	
1	0	1	0	0	0	← Antes dos pulsos serem aplicados
0	1	0	1	0	0	← Após o primeiro pulso
0	0	1	0	1	0	← Após o segundo pulso
0	0	0	1	0	1	← Após o terceiro pulso

(b)

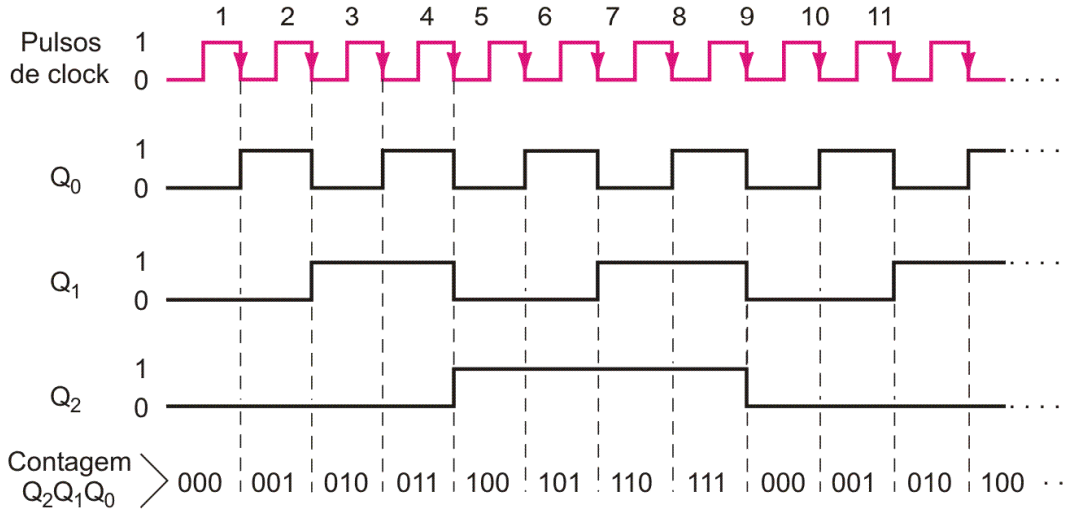
Exemplo: microprocessador transferindo dados para um registrador externo



Flip-flops JK conectados para formar um contador binário de três bits (módulo 8)

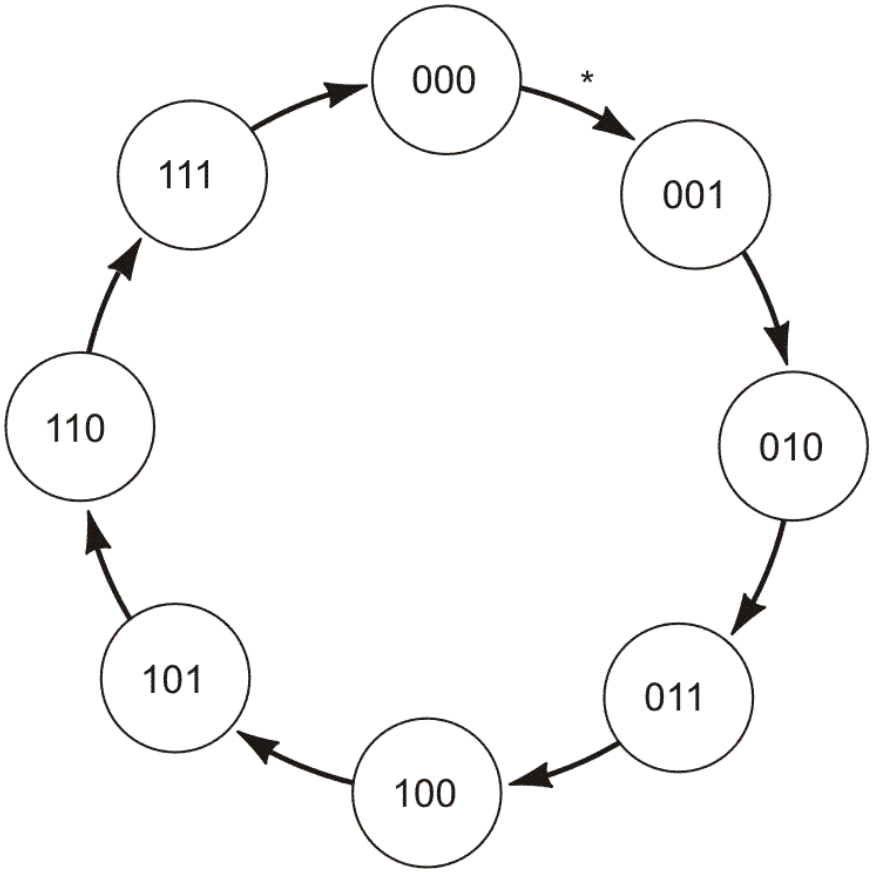


(a)



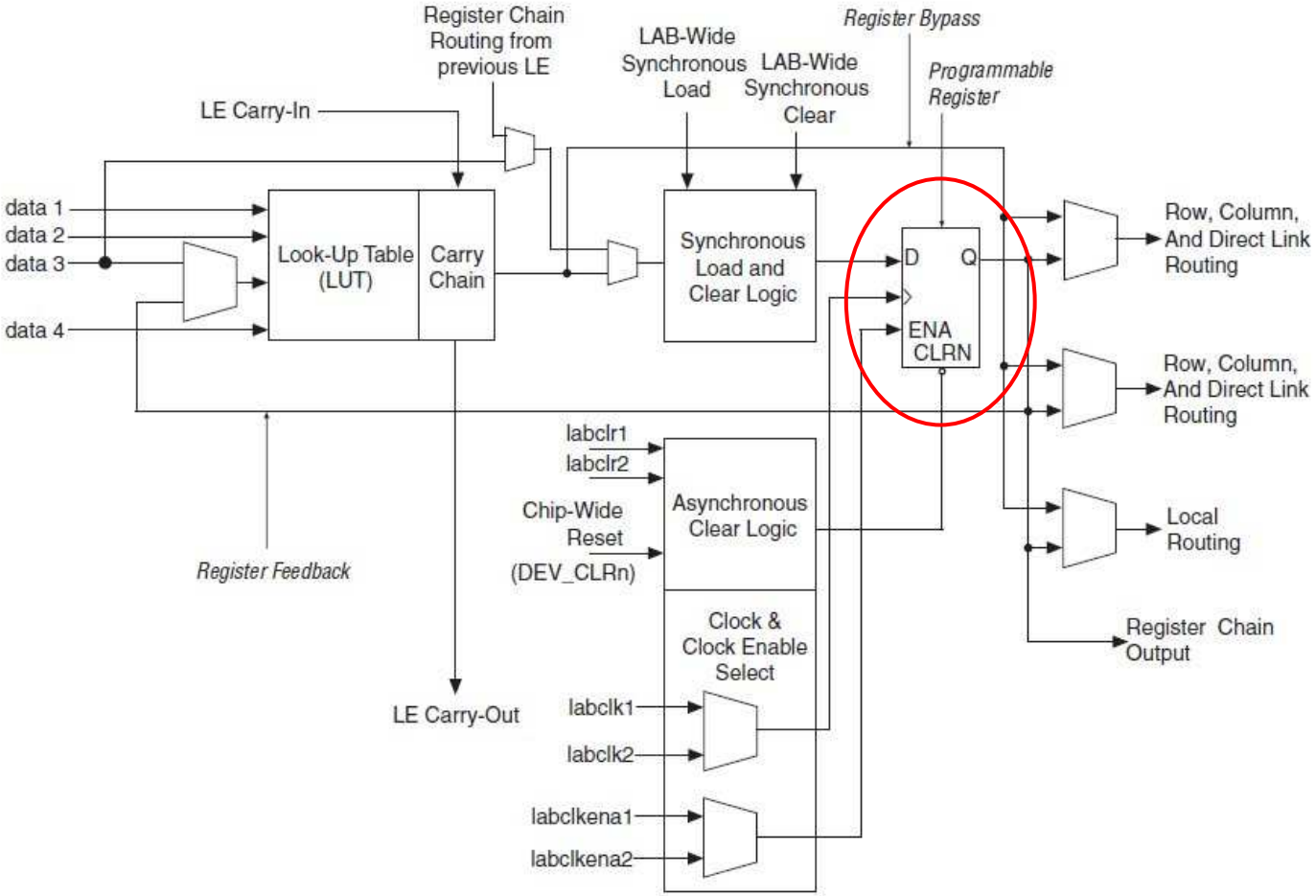
(b)

Diagrama de estados de um contador síncrono



* Nota: cada seta representa a ocorrência de um pulso de clock

Flip-Flops em FPGAs



Bloco Lógico (LE) da família Cyclone IV (Altera).

Adição binária

Existem somente quatro casos possíveis na adição binária:

0 + 0 = 0

1 + 0 = 1

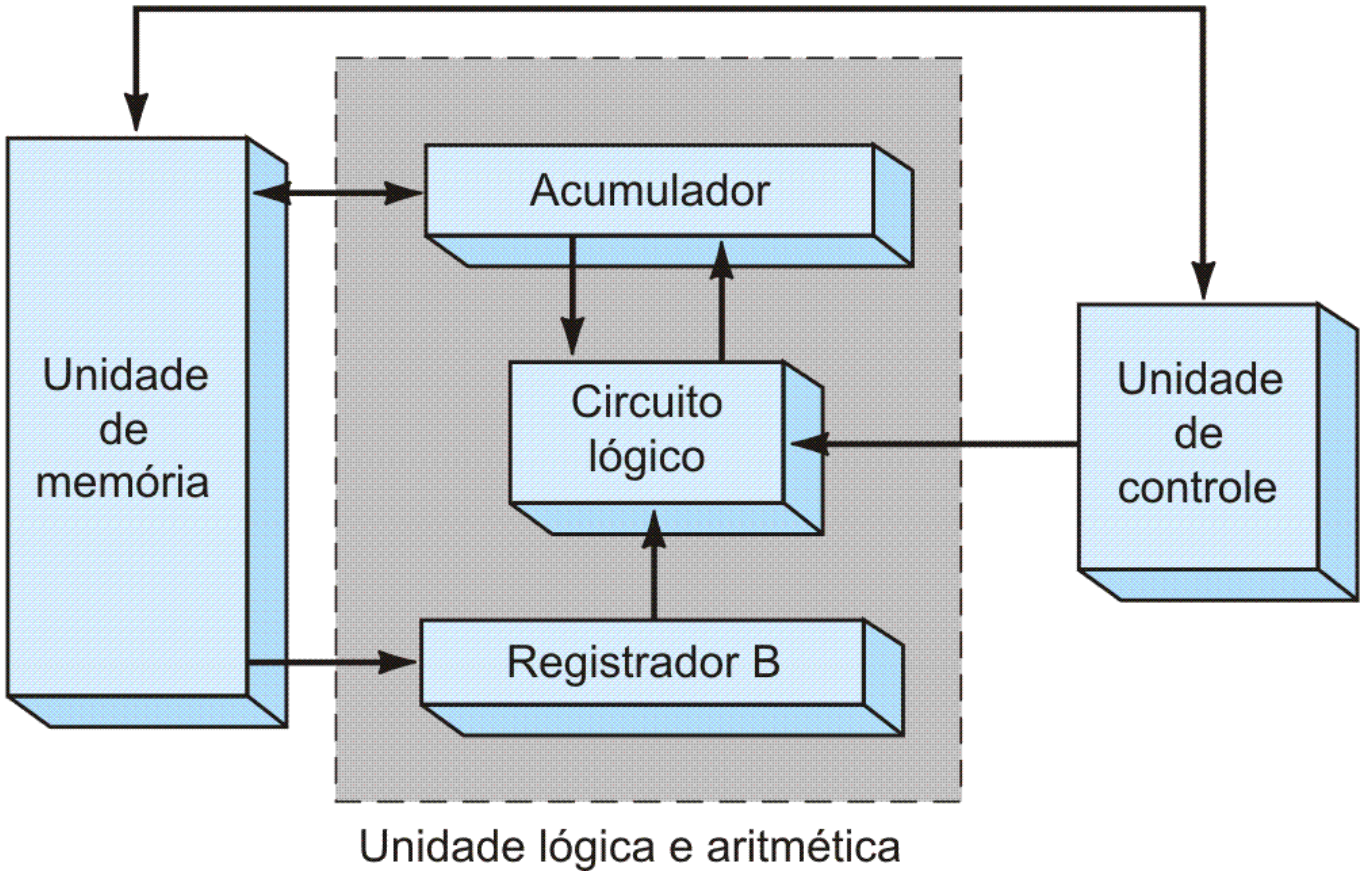
1 + 1 = 10 = 0 + carry de 1 para a próxima posição

1 + 1 + 1 = 11 = 1 + carry de 1 para a próxima posição

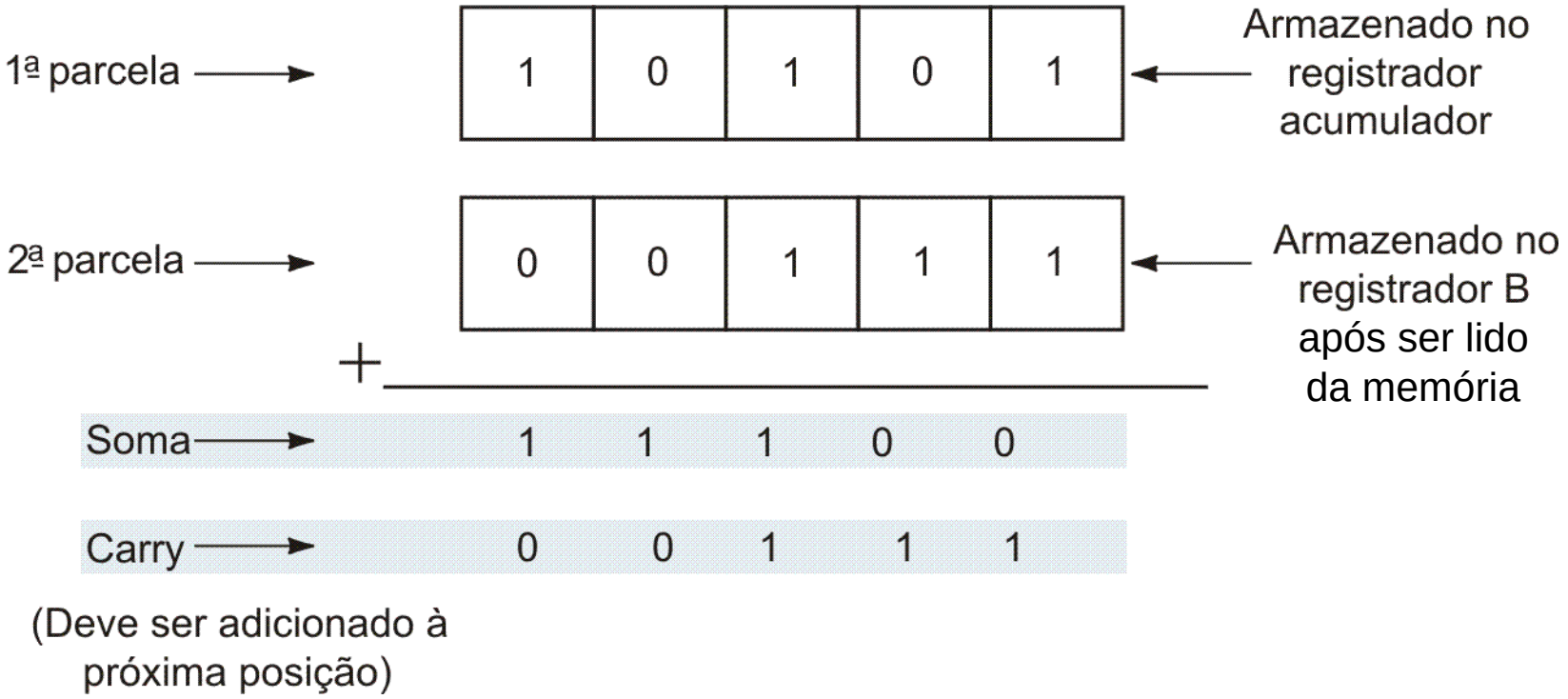
carry

A operação de adição em computadores ocorre somente entre dois números de cada vez. Isto não representa uma limitação na velocidade, visto que os processadores modernos realizam uma soma em nano-segundos.
(Ex: *clock > 2,5GHz*).

Blocos funcionais de uma ALU



Processo típico de uma adição binária

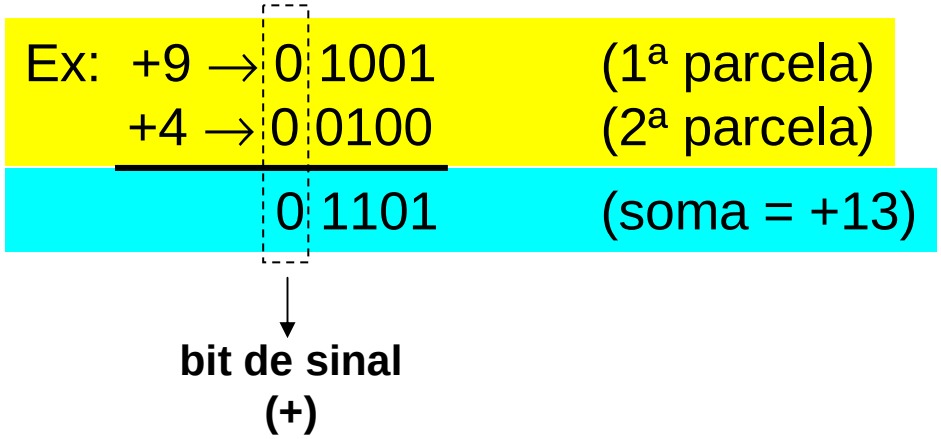


Adição binária no sistema complemento a 2

➢ Calculadoras e processadores utilizam normalmente o sistema em comp. a 2 para somar e subtrair binários. O *hardware* é mais simples.

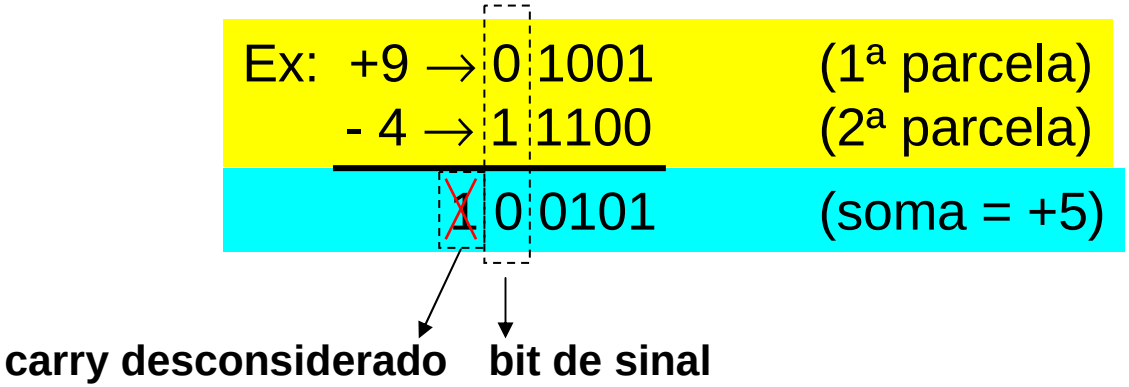
➢ As operações adição/subtração são realizadas também sobre o **bit de sinal**.

CASO 1: adição de 2 números positivos

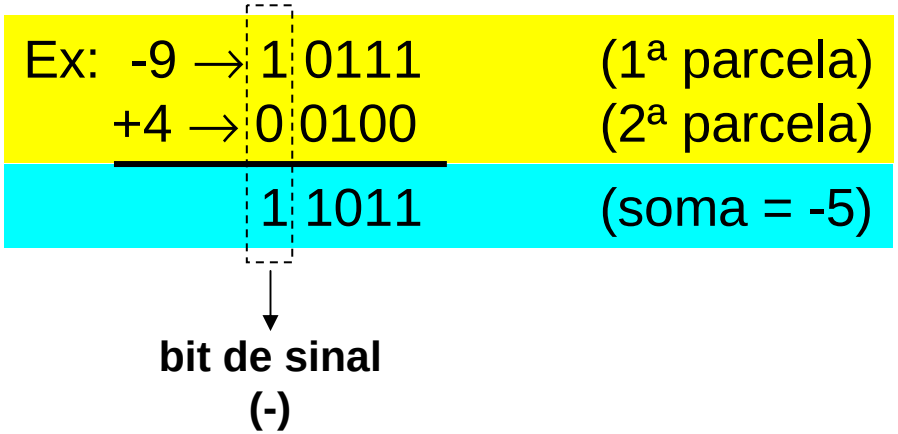


Adição binária no sistema complemento a 2

CASO 2: adição de um nº positivo e outro menor e negativo



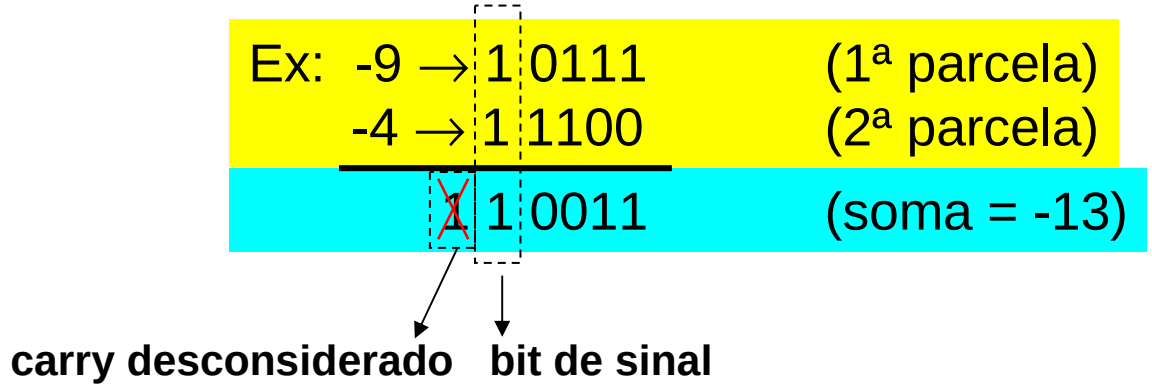
CASO 3: adição de um nº positivo e outro maior e negativo



D[5..0]	Comp. a 2
00000	0
10000	-16
10001	-15
10010	-14
10011	-13
10100	-12
10101	-11
10110	-10
10111	-9
11000	-8
11001	-7
11010	-6
11011	-5
11100	-4
11101	-3
11110	-2
11111	-1

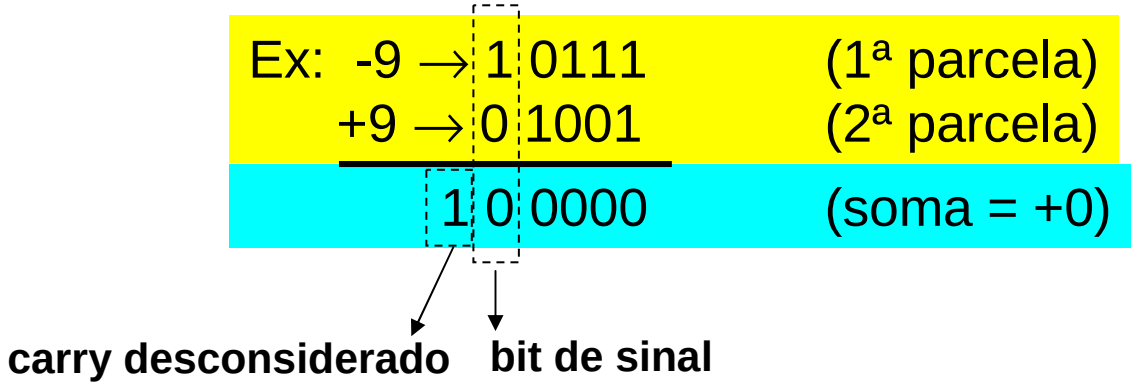
Adição binária no sistema complemento a 2

CASO 4: adição de dois números negativos



D[5..0]	Comp. a 2
00000	0
10000	-16
10001	-15
10010	-14
10011	-13
10100	-12
10101	-11
10110	-10
10111	-9
11000	-8
11001	-7
11010	-6
11011	-5
11100	-4
11101	-3
11110	-2
11111	-1

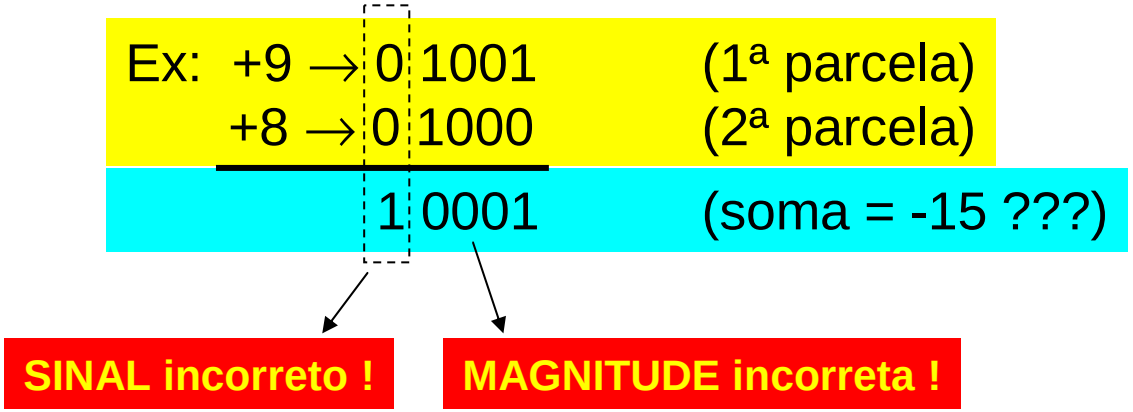
CASO 5: números iguais de sinais opostos



Adição binária no sistema complemento a 2
- observações -

✓ **Subtração:** na operação de subtração, basta converter o subtraendo para seu equivalente negativo em complemento a 2, e realizar a operação de adição.

✓ **Overflow:** se a soma produzir um resultado cuja magnitude não pode ser representada pelos bits disponíveis, o resultado estará INCORRETO.

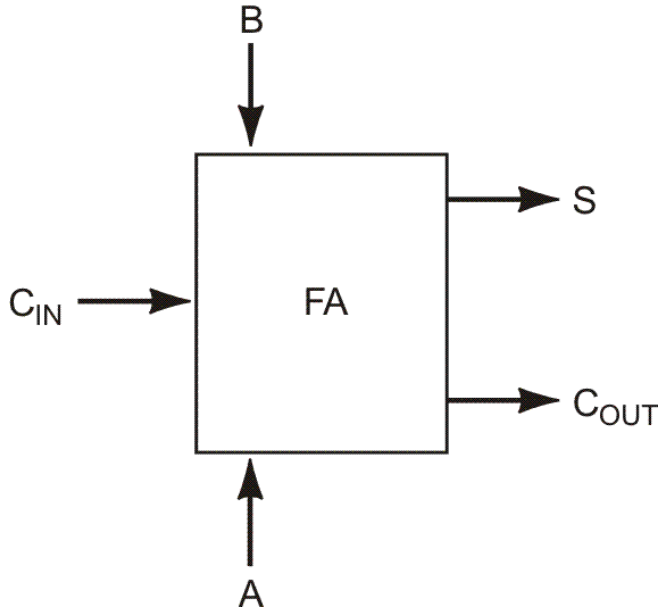


D[5..0]	Comp. a 2
00000	0
10000	-16
10001	-15
10010	-14
10011	-13
10100	-12
10101	-11
10110	-10
10111	-9
...	

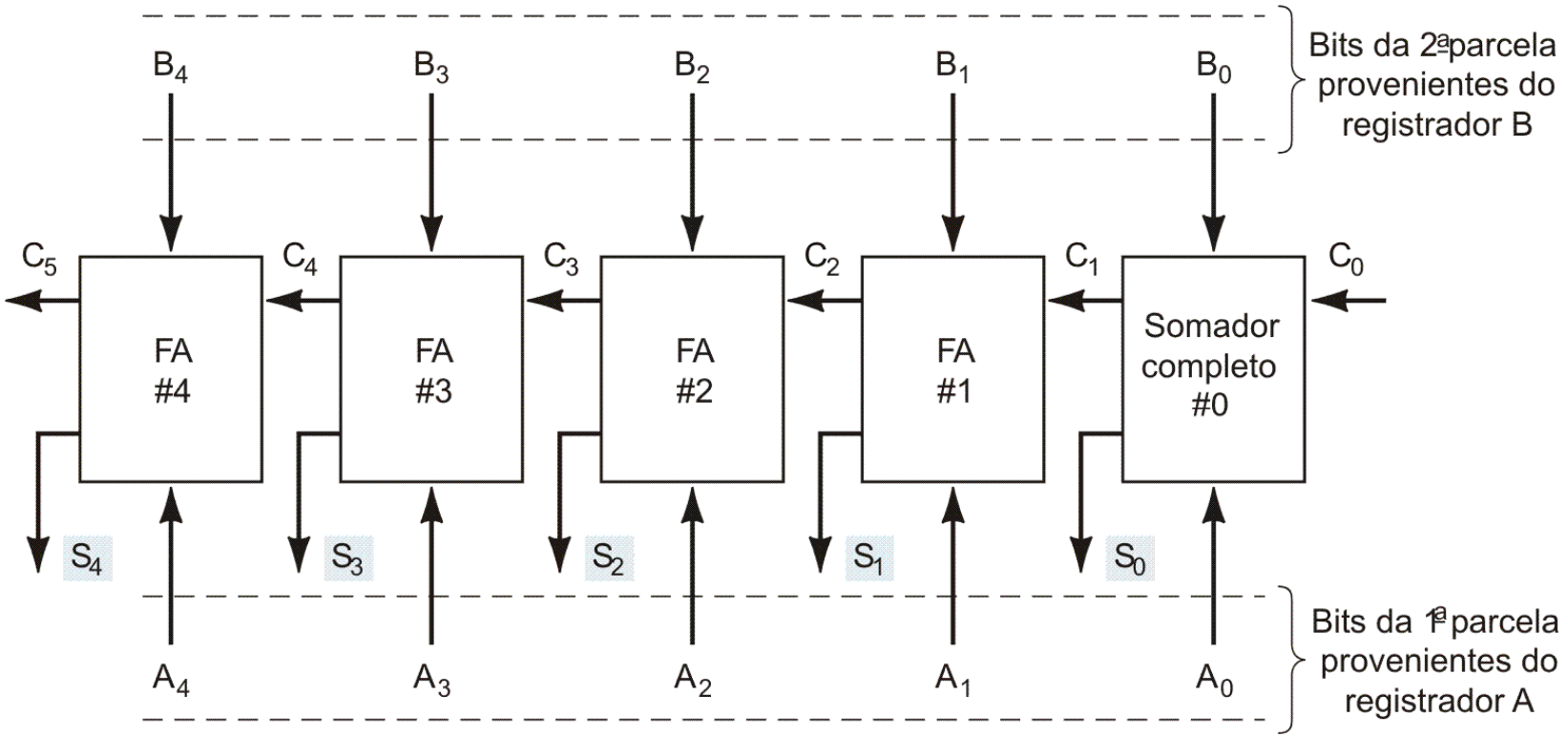
O overflow pode ser detectado verificando se o bit de sinal resultante tem o mesmo valor dos bits de sinal dos números originais → **tem que ter**.

Tabela-verdade de um somador completo

Bit de entrada da 1ª parcela	Bit de entrada da 2ª parcela	Bit de entrada do carry		Bit de saída da soma	Bit de saída do carry
A	B	C_{IN}		S	C_{OUT}
0	0	0		0	0
0	0	1		1	0
0	1	0		1	0
0	1	1		0	1
1	0	0		1	0
1	0	1		0	1
1	1	0		0	1
1	1	1		1	1

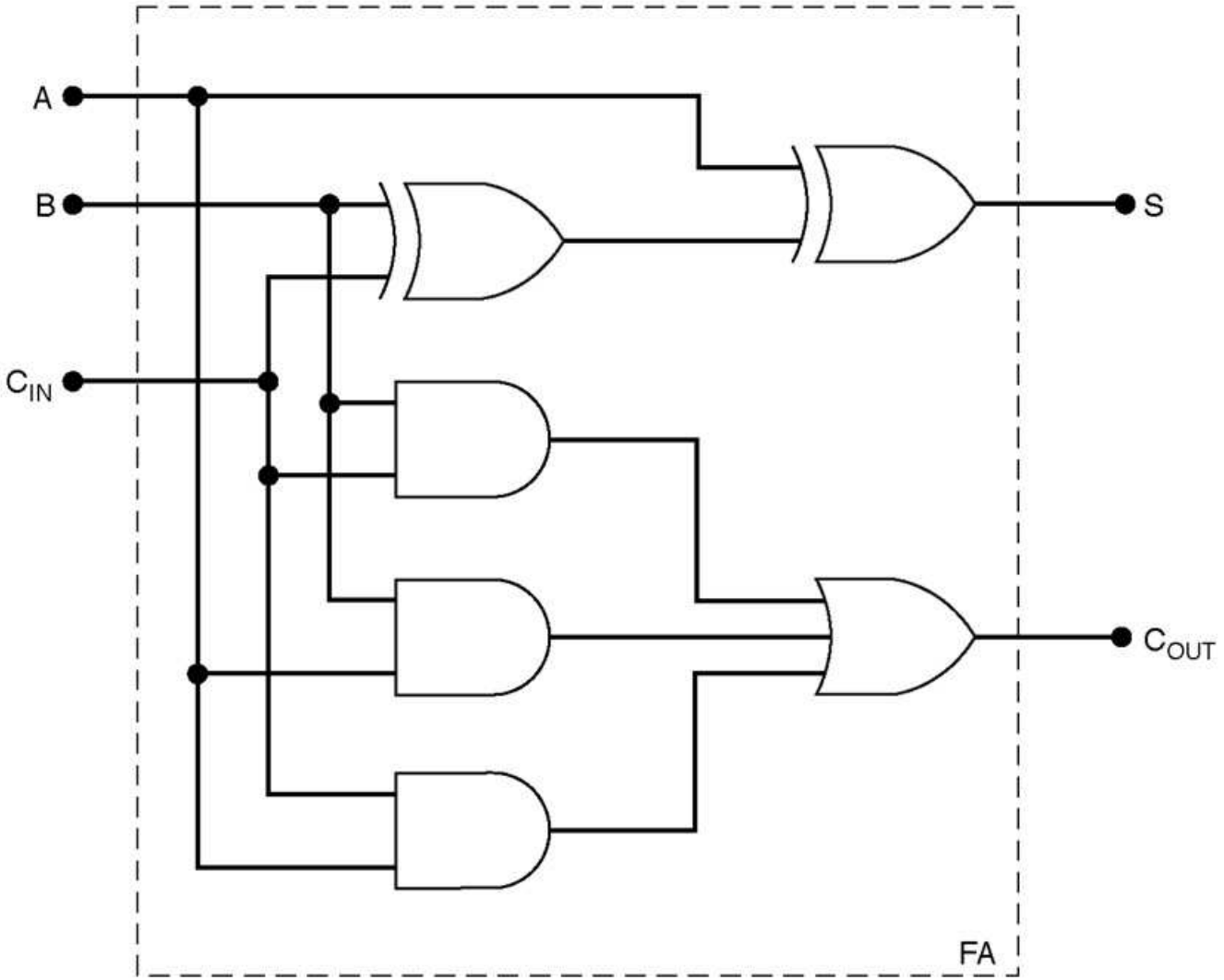


Circuito somador paralelo usando somadores completos



A soma aparece nas saídas S_4, S_3, S_2, S_1, S_0.

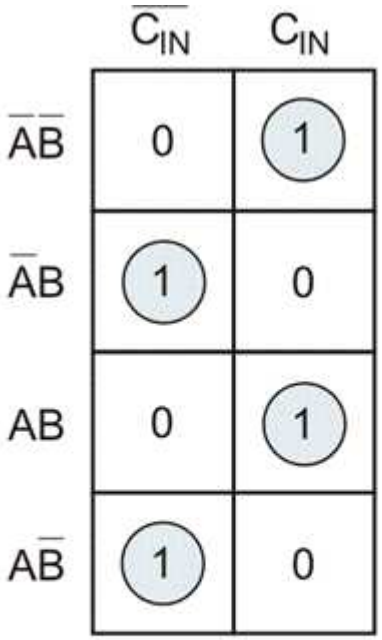
Circuito de um somador completo



A	B	XOR
0	0	0
0	1	1
1	0	1
1	1	0

Mapas de Karnaugh para as saídas do somador completo

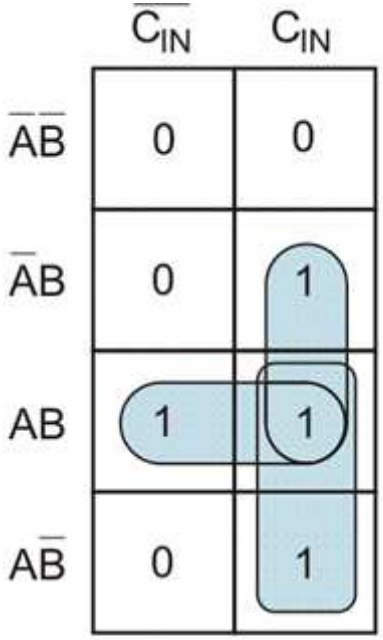
Bit de entrada da 1ª parcela	Bit de entrada da 2ª parcela	Bit de entrada do carry		Bit de saída da soma	Bit de saída do carr
A	B	C_{IN}		S	C_{OUT}
0	0	0		0	0
0	0	1		1	0
0	1	0		1	0
0	1	1		0	1
1	0	0		1	0
1	0	1		0	1
1	1	0		0	1
1	1	1		1	1



Mapa K para S

$$S = \overline{A}\,\overline{B}C_{IN} + \overline{A}B\overline{C}_{IN} + ABC_{IN} + A\overline{B}\overline{C}_{IN}$$

(a)

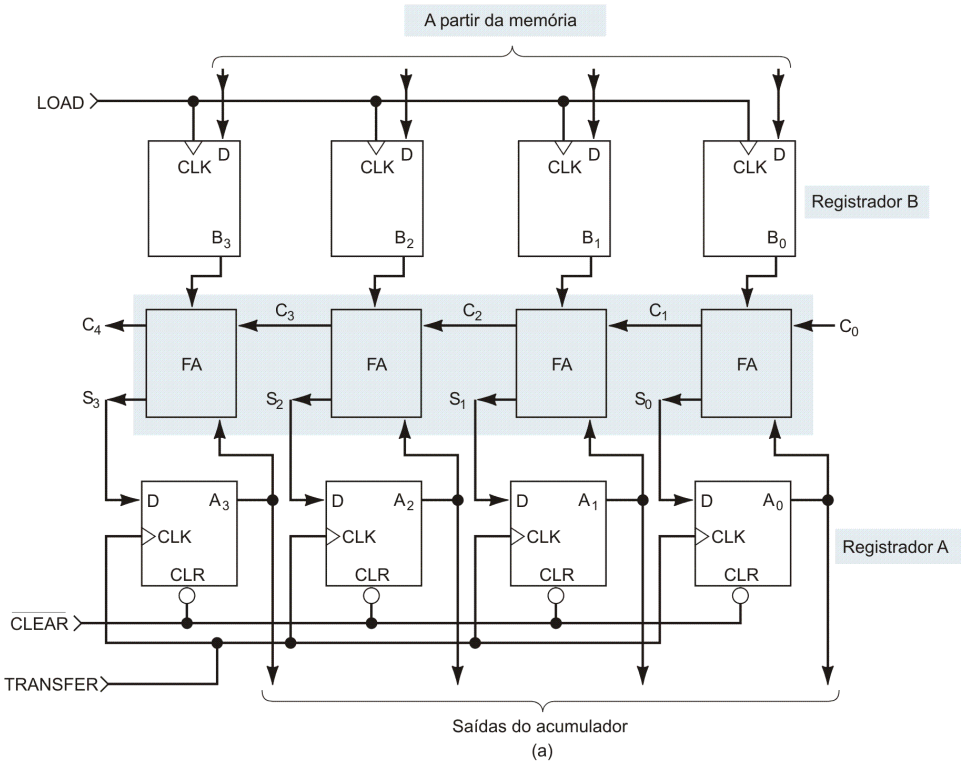


Mapa K para C_{OUT}

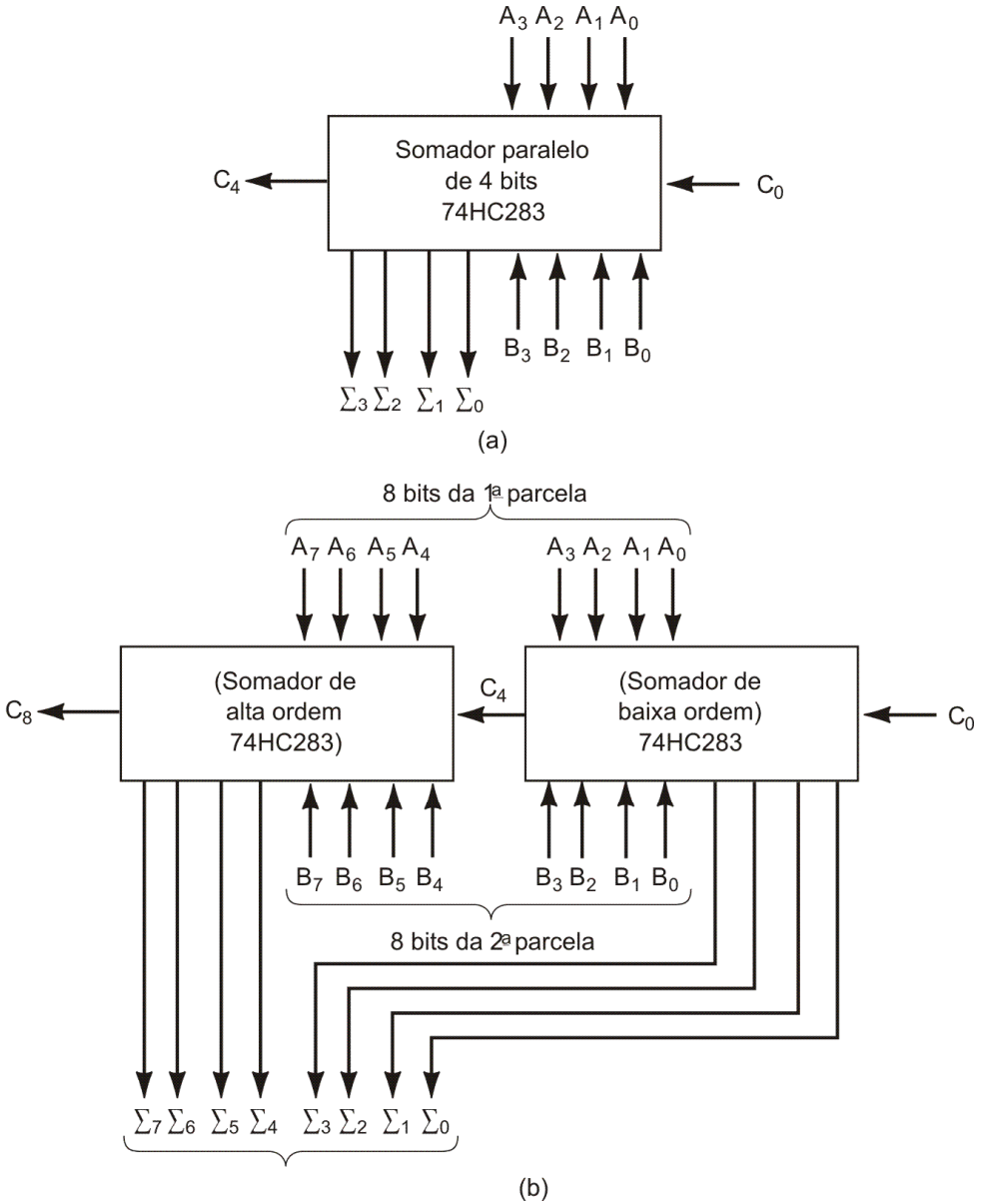
$$C_{OUT} = BC_{IN} + AC_{IN} + AB$$

(b)

Somador completo com registradores e sinais para somar e armazenar em memória



Somador paralelo de 4 bits – 74HC283 e conexão em cascata formando um somador de 8 bits



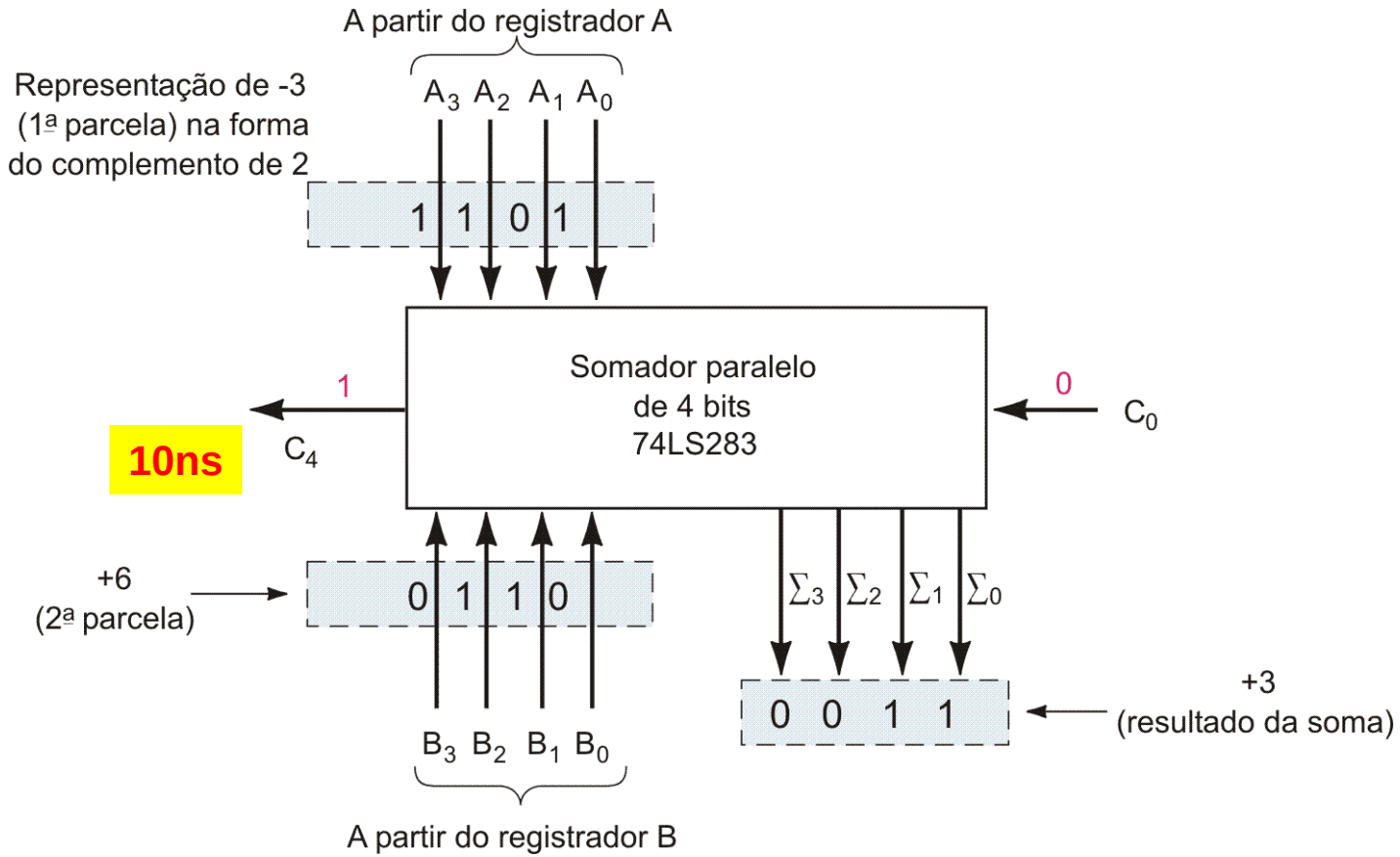
PARALELO

os bits são somados simultaneamente

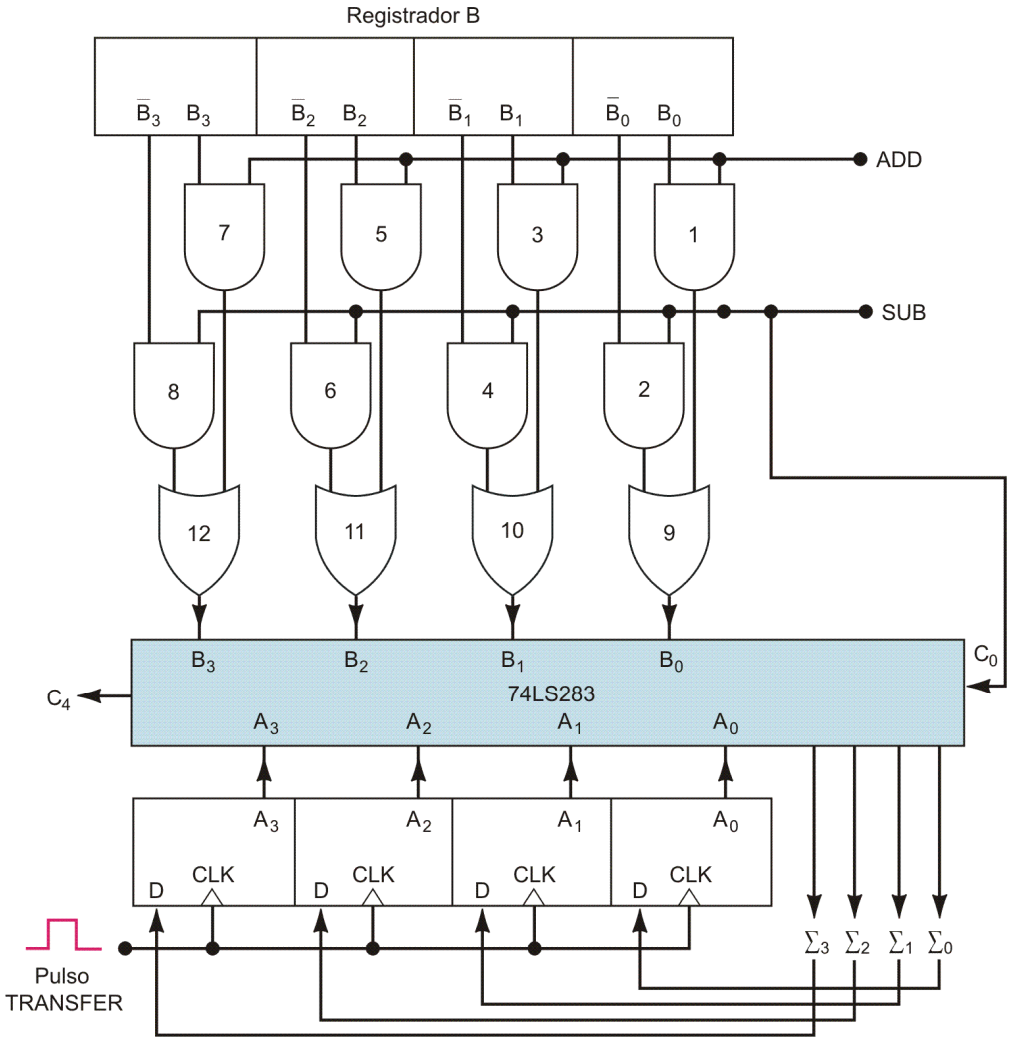
velocidade limitada pela propagação do carry

look-ahead carry
(carry antecipado)

Somador paralelo para somar número positivo com número negativo em complemento a 2



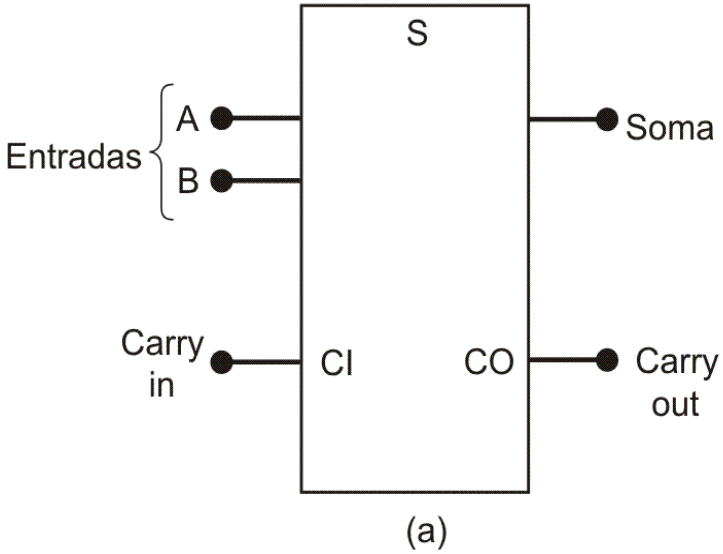
Somador/Subtrator paralelo usando complemento a 2



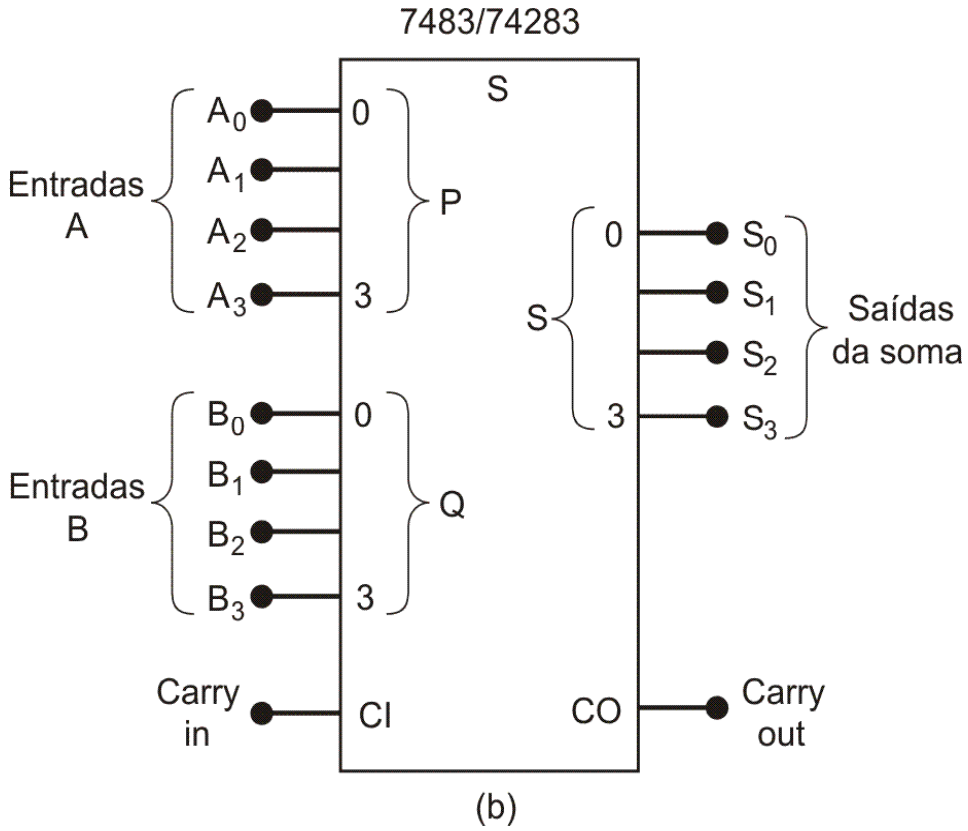
Operação	
ADD	SUB
1	0
0	1

Neste caso, o CARRY = 1, produzindo o comp. a 2 de B.

Símbolos IEEE/ANSI para somador completo e CI somador paralelo de 4 bits

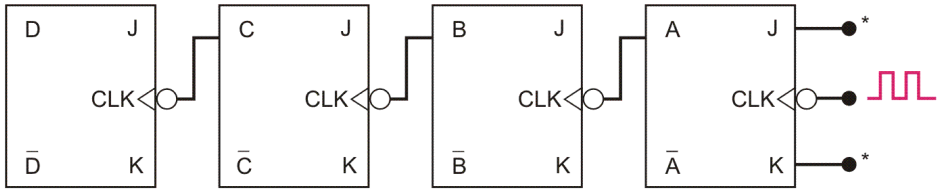


(a)

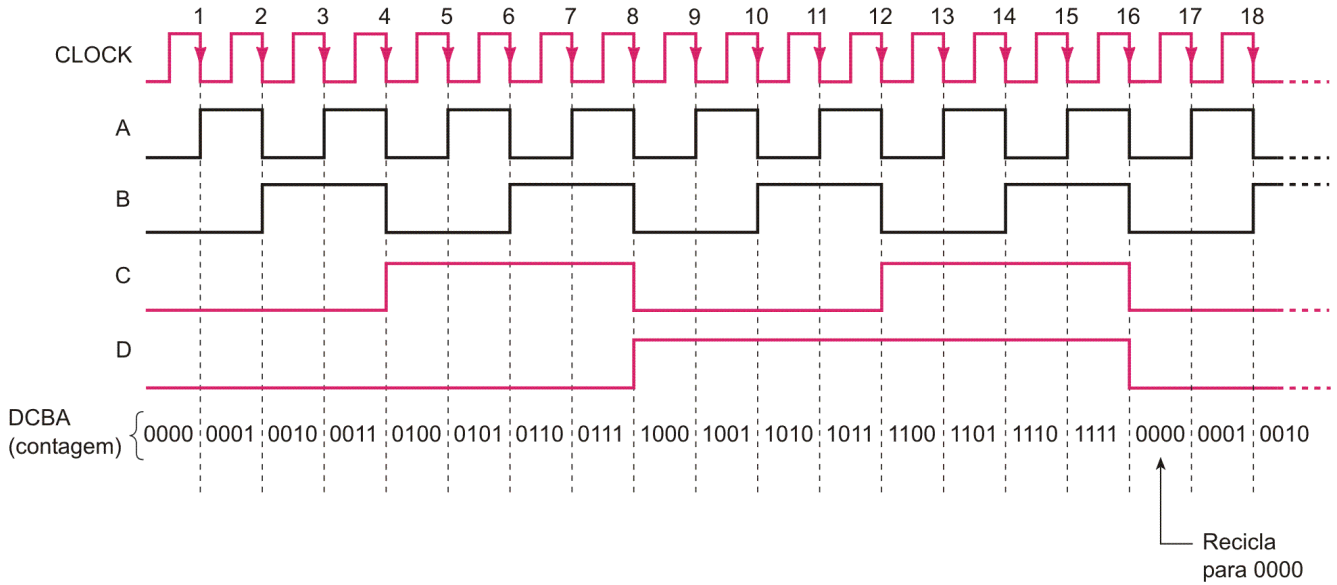


(b)

Contador assíncrono de 4 bits (*ripple counter*)

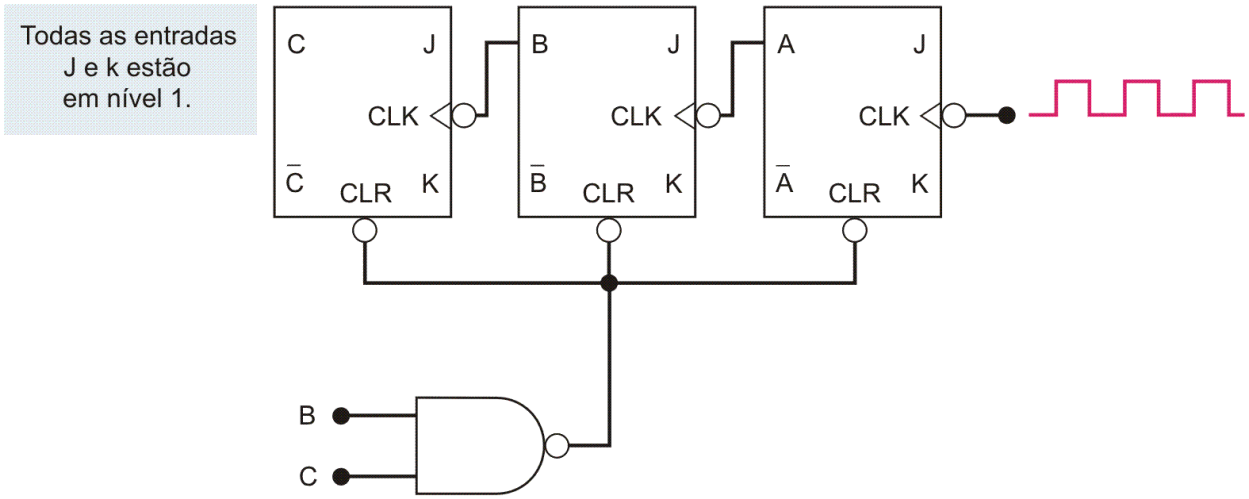


*Todas as entradas J e K
estão em nível 1



J	K	CLK	Q
0	0	–	Q_0 (não muda)
1	0	–	1
0	1	–	0
1	1	–	$\overline{Q_0}$ (comuta)

Contador módulo 6 através do *reset* de um contador módulo 8 na contagem 6



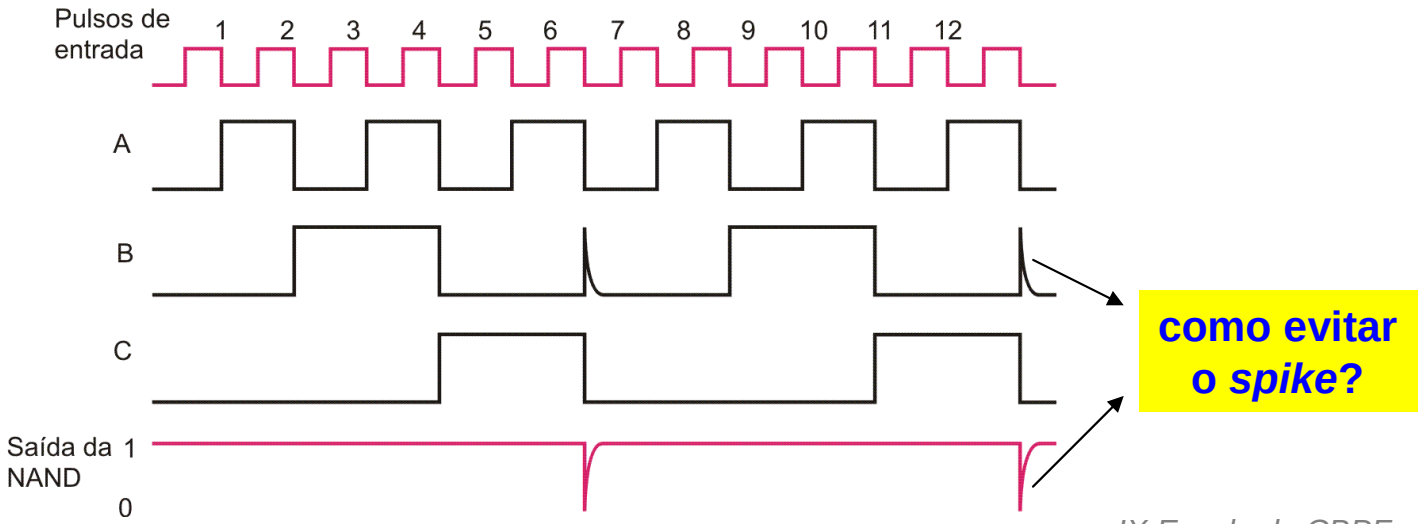
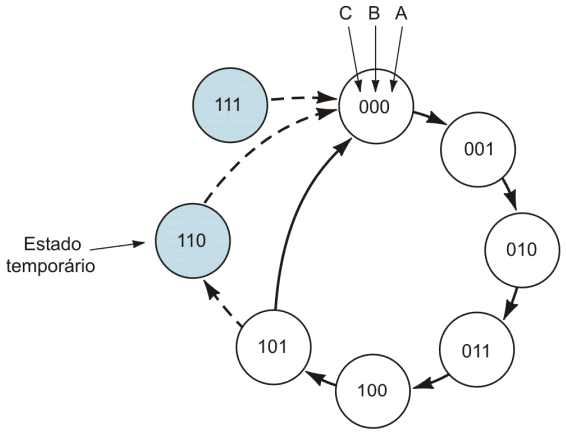
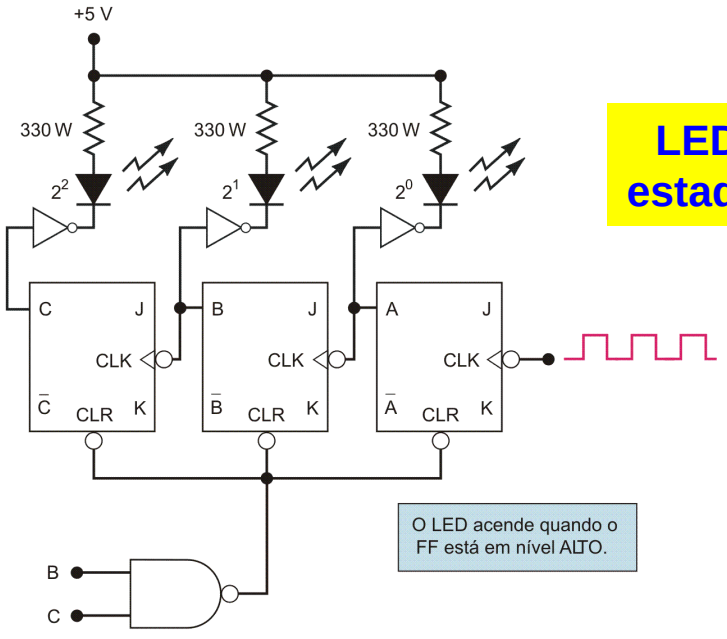


Diagrama de estados para o contador módulo 6



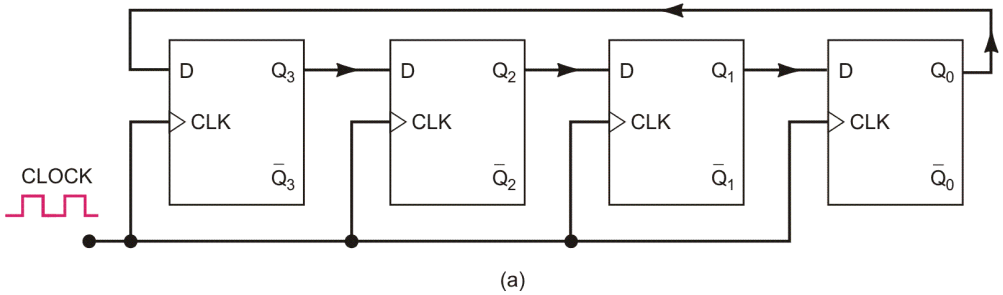
(a)



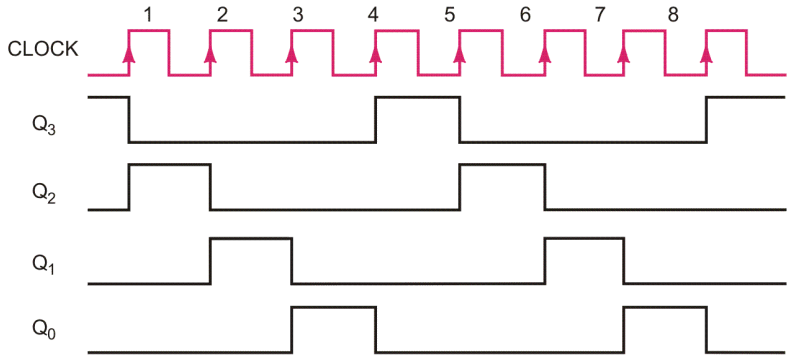
(b)

LEDs mostram o estado do contador

Shift-register em anel de 4 bits



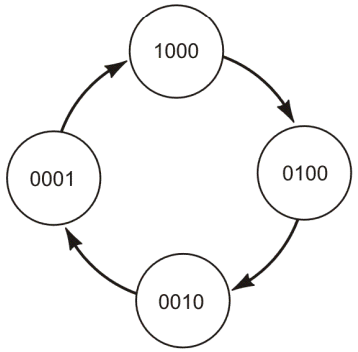
(a)



(b)

Q_3	Q_2	Q_1	Q_0	Pulso de CLOCK
1	0	0	0	0
0	1	0	0	1
0	0	1	0	2
0	0	0	1	3
1	0	0	0	4
0	1	0	0	5
0	0	1	0	6
0	0	0	1	7
.	.	.	.	.
.	.	.	.	.

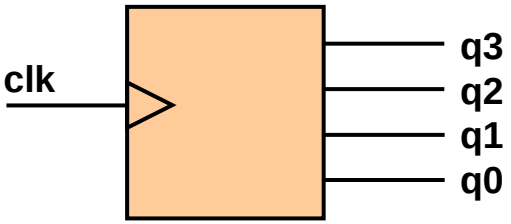
(c)



(d)

Shift-register em anel - VHDL

```
entity shift_reg is
port(clk      : in bit;
     q        : out bit_vector(3 downto 0));

architecture vhdl of shift_reg is
  signal   ser_in : bit;
begin
  process(clk)
    variable ff : bit_vector(3 downto 0);
  begin

    if (ff(3 downto 1) = “000”) then
      ser_in <= ‘1’;  -- auto início
    else
      ser_in <= ‘0’;
    end if;

    if (clk’event and clk=‘1’) then
      ff := (ser_in & ff(3 downto 1));  -- deslocamento p/direita
    end if;

    q <= ff;

  end process;
end vhdl;
```



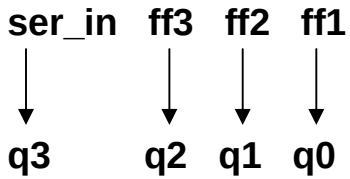
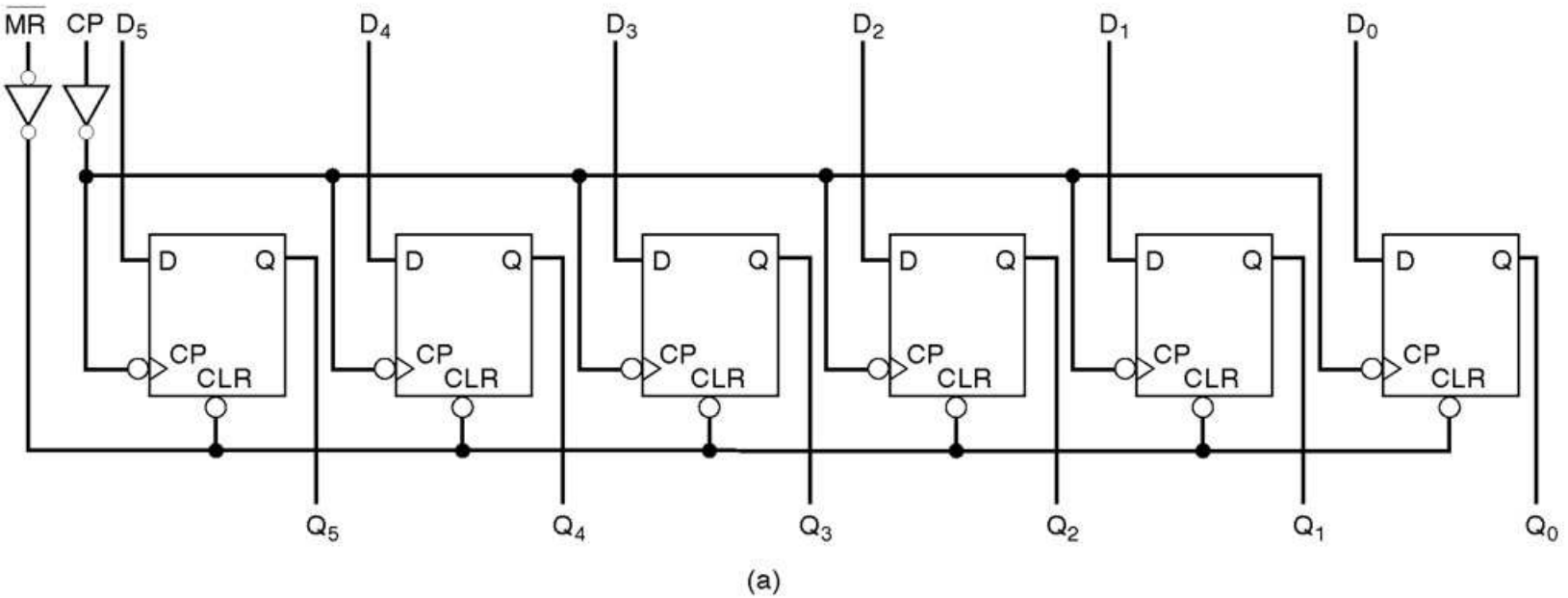
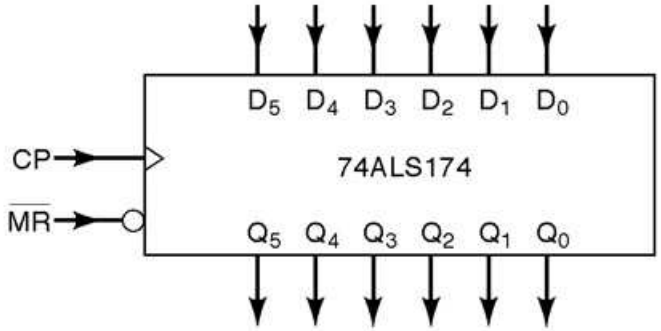


Diagrama do CI 74ALS174 e símbolo lógico

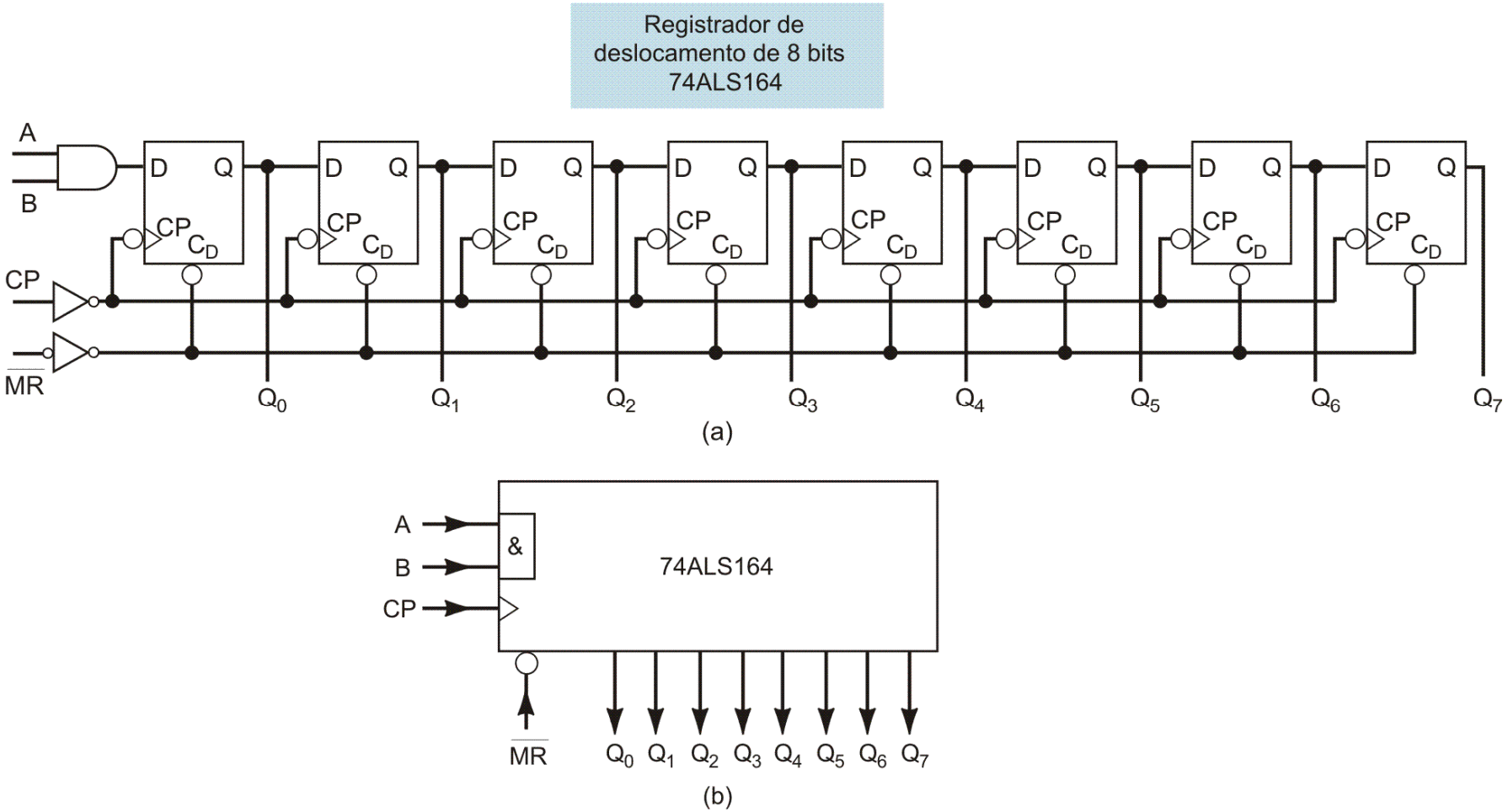


(a)

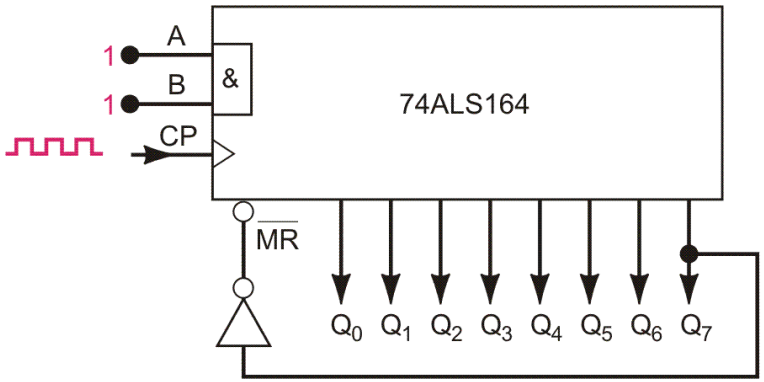


(b)

Entrada Serial / Saída Paralela – 74ALS164



Funcionamento do 74ALS164



(a)

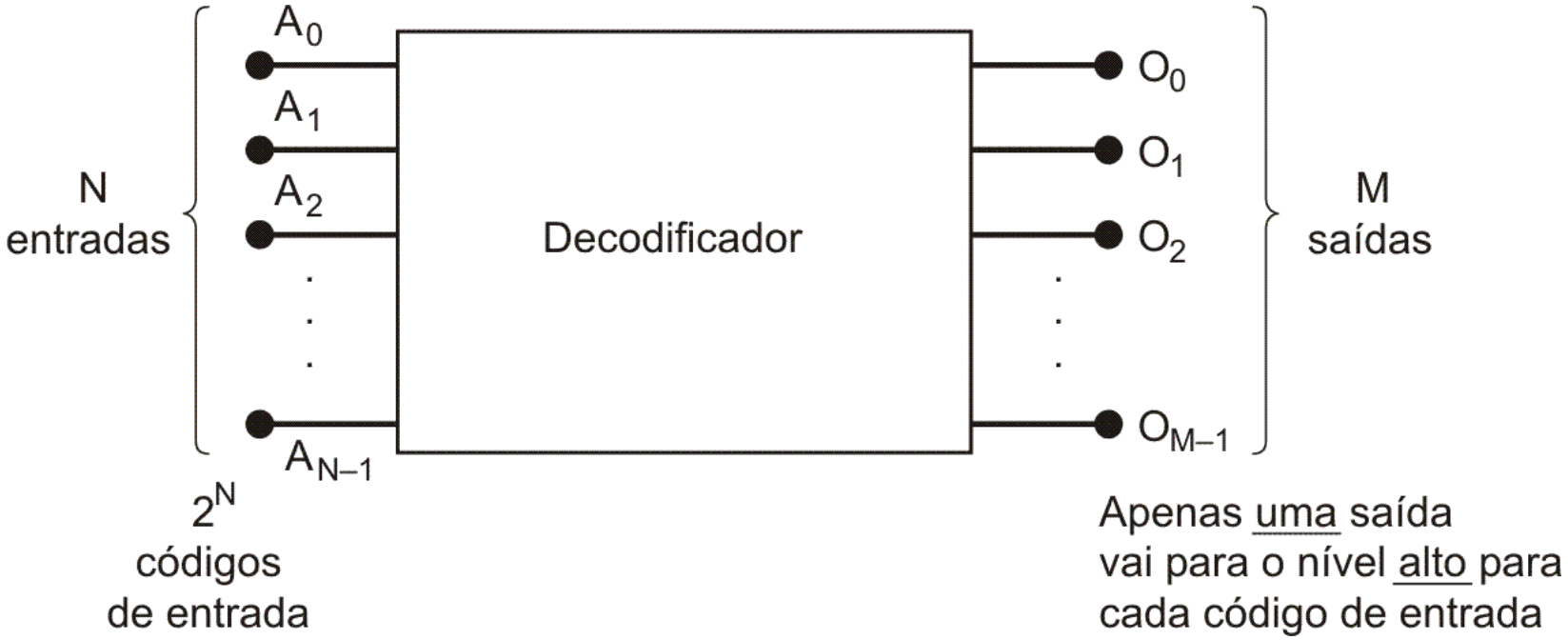
Número de pulsos de entrada	Q_0	Q_1	Q_2	Q_3	Q_4	Q_5	Q_6	Q_7
0	0	0	0	0	0	0	0	0
1	1	0	0	0	0	0	0	0
2	1	1	0	0	0	0	0	0
3	1	1	1	0	0	0	0	0
4	1	1	1	1	0	0	0	0
5	1	1	1	1	1	0	0	0
6	1	1	1	1	1	1	0	0
7	1	1	1	1	1	1	1	0
8	1	1	1	1	1	1	1	1

Recicla

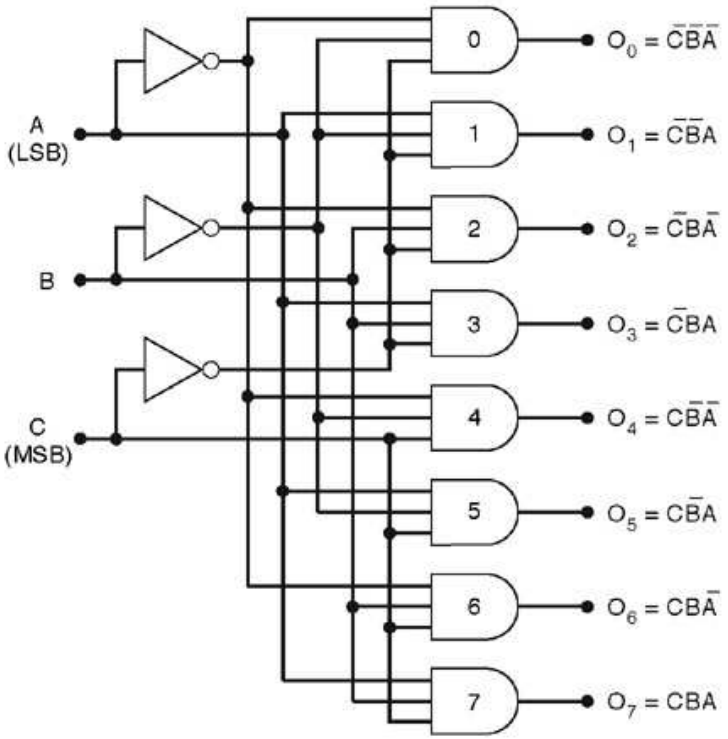
Estado temporário

(b)

Diagrama de um decodificador

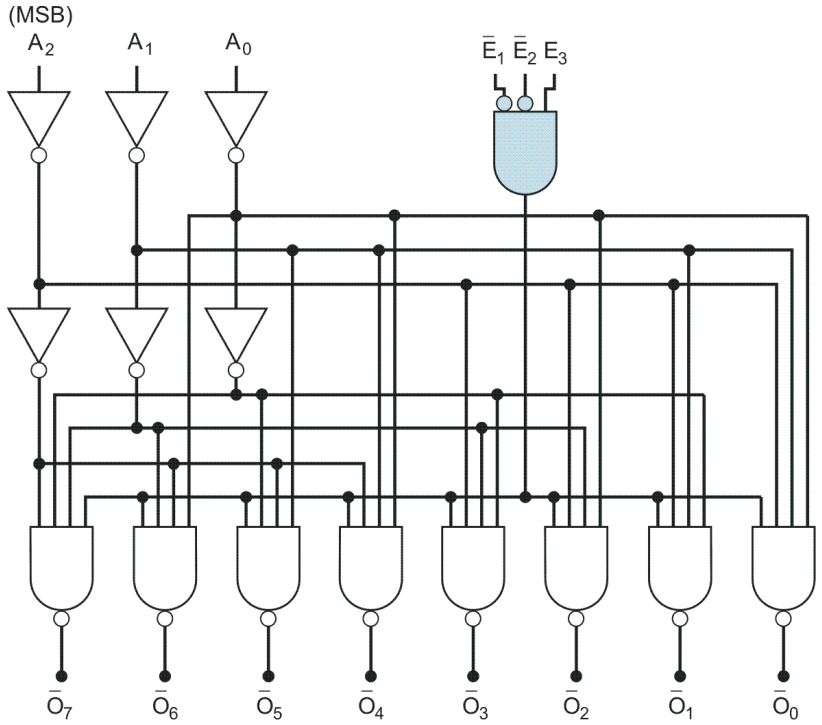


Decodificador 3 pra 8



C	B	A	O_7	O_6	O_5	O_4	O_3	O_2	O_1	O_0
0	0	0	0	0	0	0	0	0	0	1
0	0	1	0	0	0	0	0	0	1	0
0	1	0	0	0	0	0	0	1	0	0
0	1	1	0	0	0	0	1	0	0	0
1	0	0	0	0	0	1	0	0	0	0
1	0	1	0	0	1	0	0	0	0	0
1	1	0	0	1	0	0	0	0	0	0
1	1	1	1	0	0	0	0	0	0	0

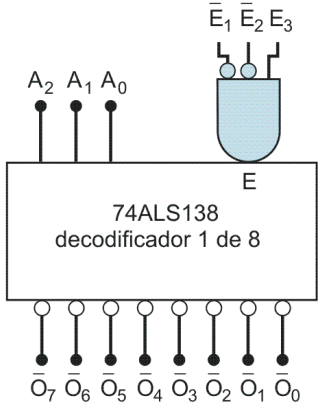
Decodificador 74LS138



(a)

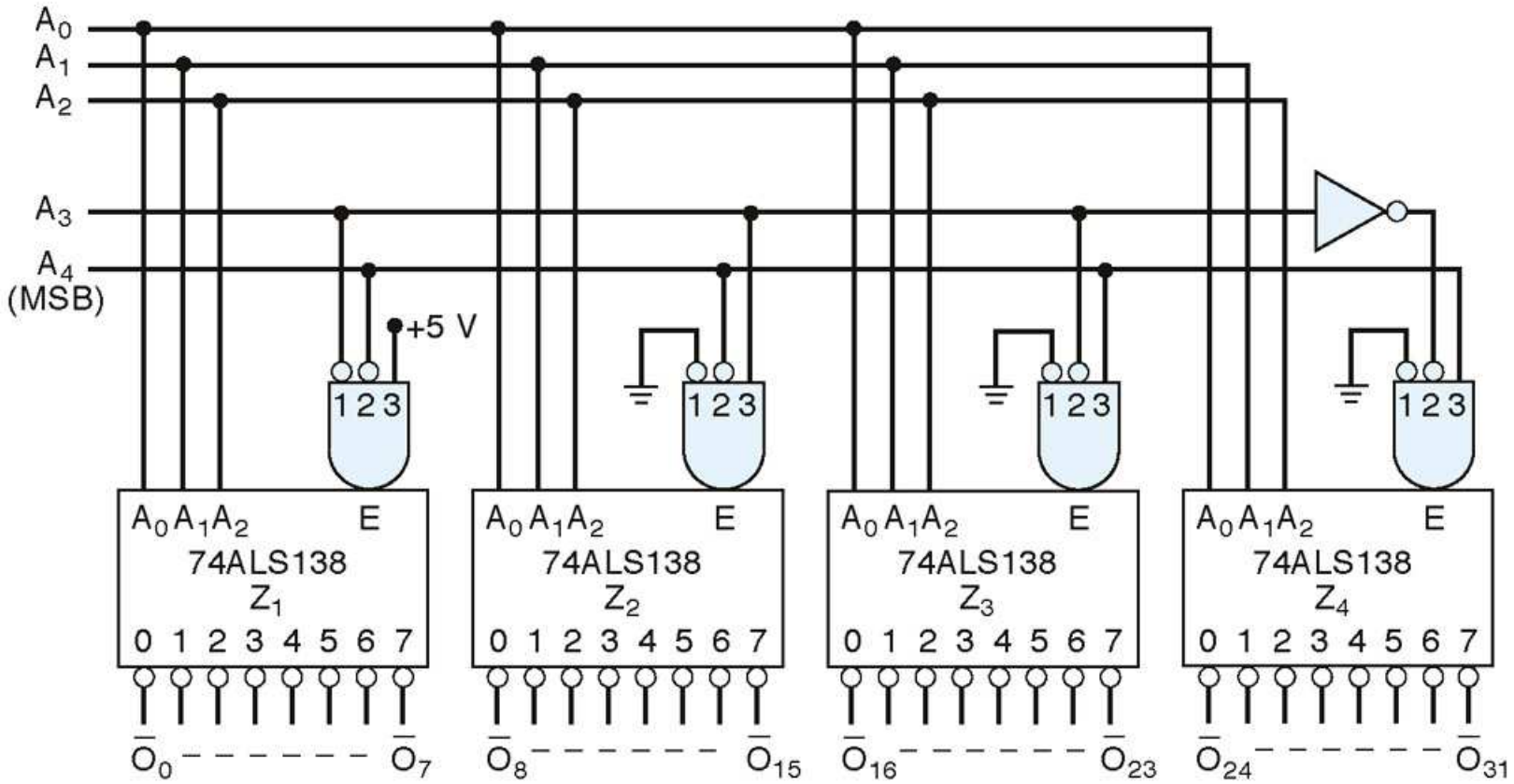
$\bar{E}_1$	$\bar{E}_2$	E_3	Saídas
0	0	1	Responde ao código de entrada $A_2A_1A_0$
1	X	X	Desabilitadas – todas em nível ALTO
X	1	X	Desabilitadas – todas em nível ALTO
X	X	0	Desabilitadas – todas em nível ALTO

(b)



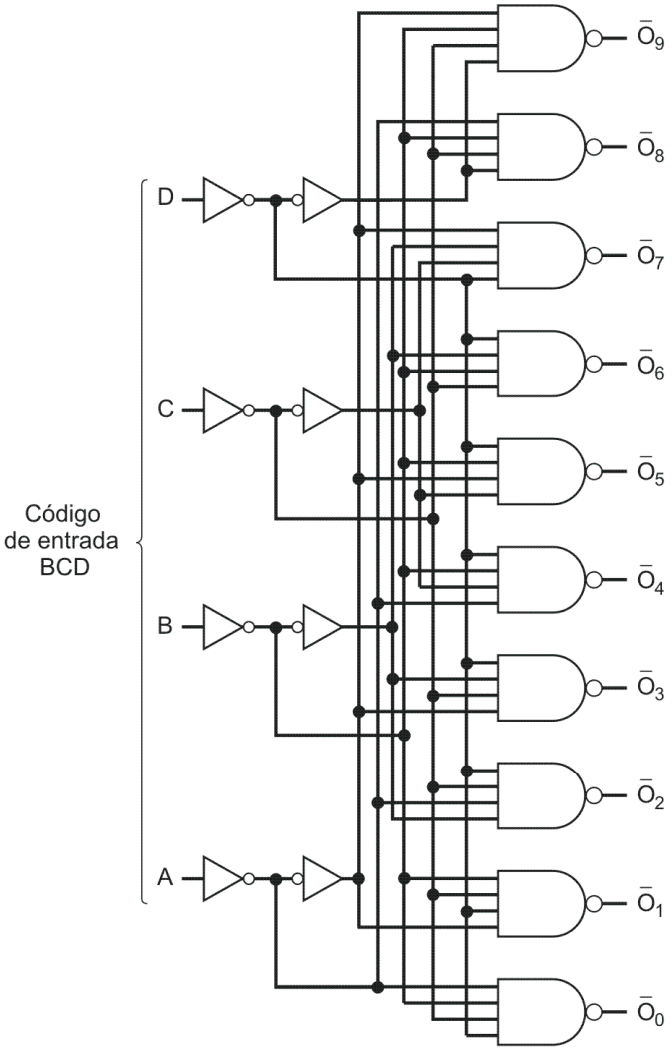
(c)

Decodificador 5 pra 32 a partir de quatro 3 pra 8

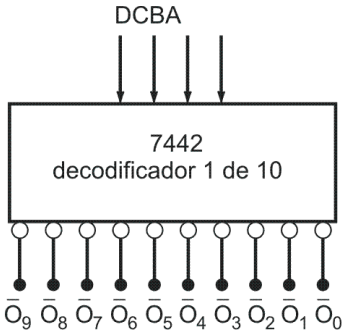


A_4	A_3		
0	0	→	habilita Z_1
0	1	→	habilita Z_2
1	0	→	habilita Z_3
1	1	→	habilita Z_4

Decodificador BCD para Decimal



(a)



(b)

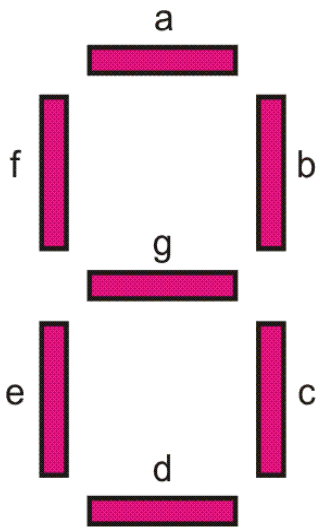
Entradas

D	C	B	A	Saída Ativa
L	L	L	L	$\bar{O}_0$
L	L	L	H	$\bar{O}_1$
L	L	H	L	$\bar{O}_2$
L	L	H	H	$\bar{O}_3$
L	H	L	L	$\bar{O}_4$
L	H	L	H	$\bar{O}_5$
L	H	H	L	$\bar{O}_6$
L	H	H	H	$\bar{O}_7$
H	L	L	L	$\bar{O}_8$
H	L	L	H	$\bar{O}_9$
H	L	H	L	Nenhuma
H	L	H	H	Nenhuma
H	H	L	L	Nenhuma
H	H	L	H	Nenhuma
H	H	H	L	Nenhuma
H	H	H	H	Nenhuma

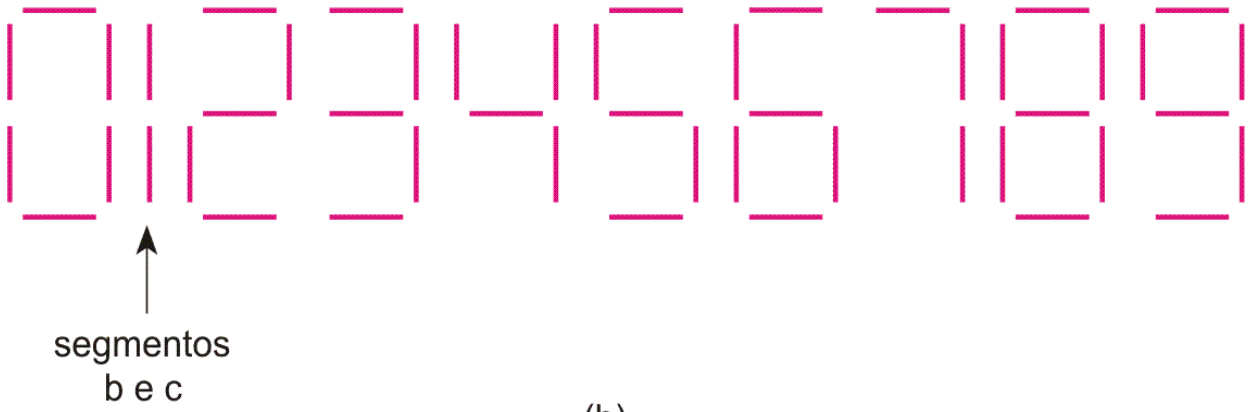
H = Nível de tensão ALTO
L = Nível de tensão BAIXO

(c)

Display de 7 segmentos



(a)



(b)

Decodificador BCD para display de 7 segmentos

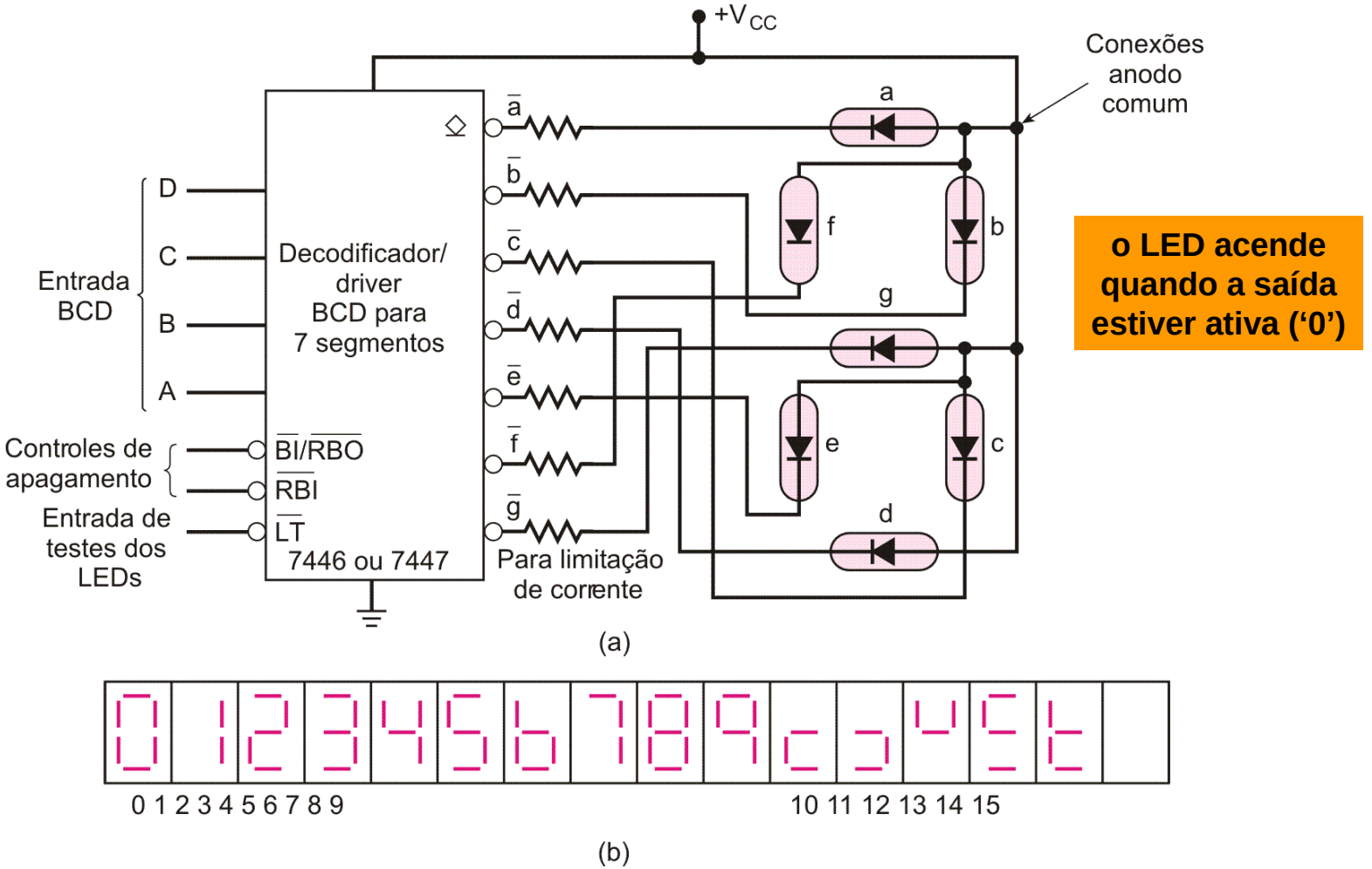
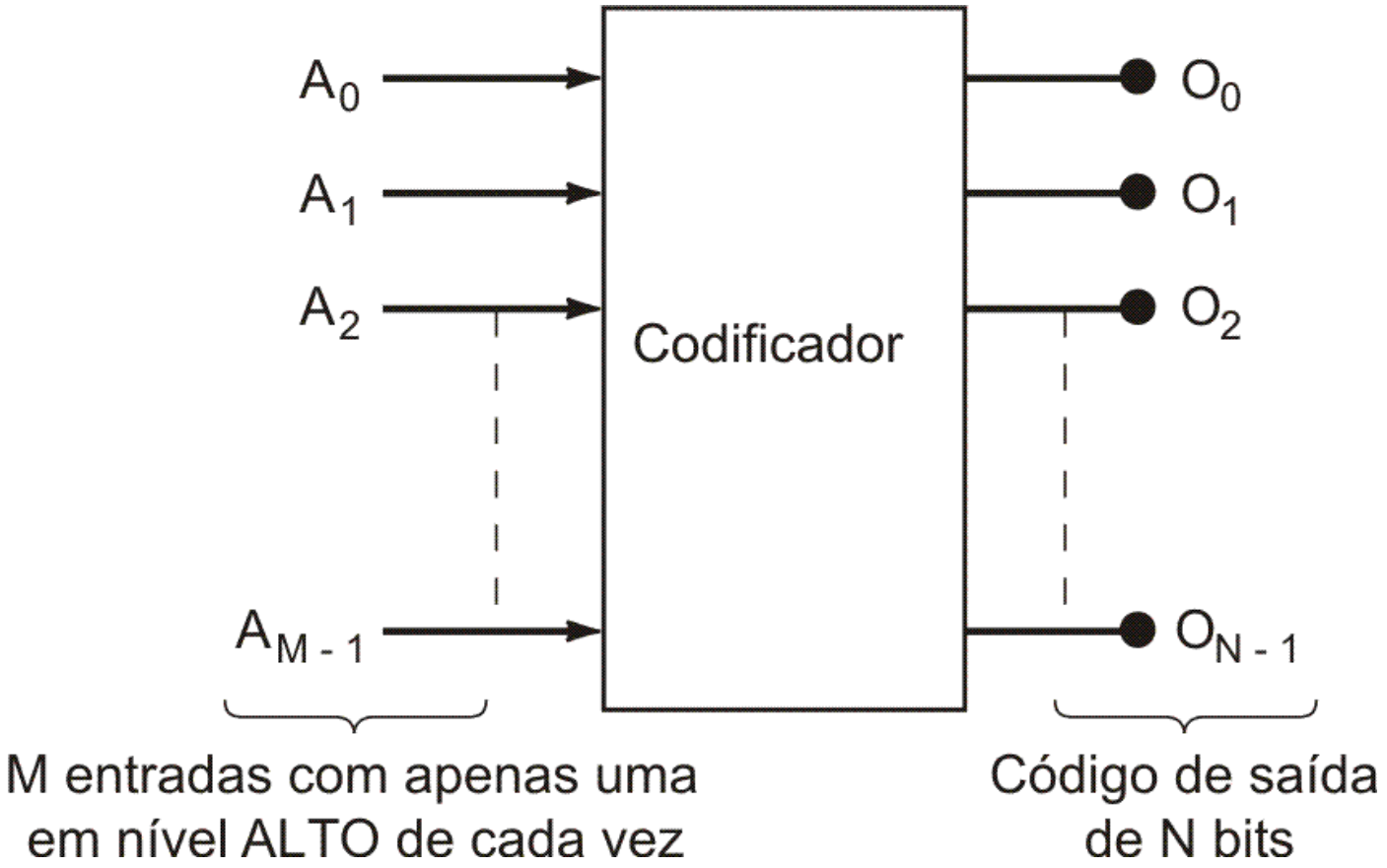
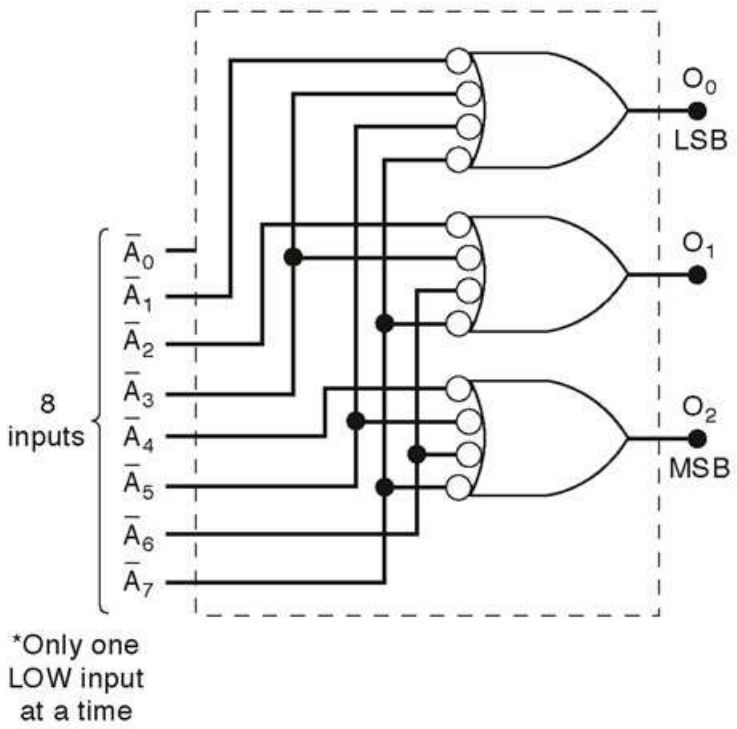


Diagrama de um codificador



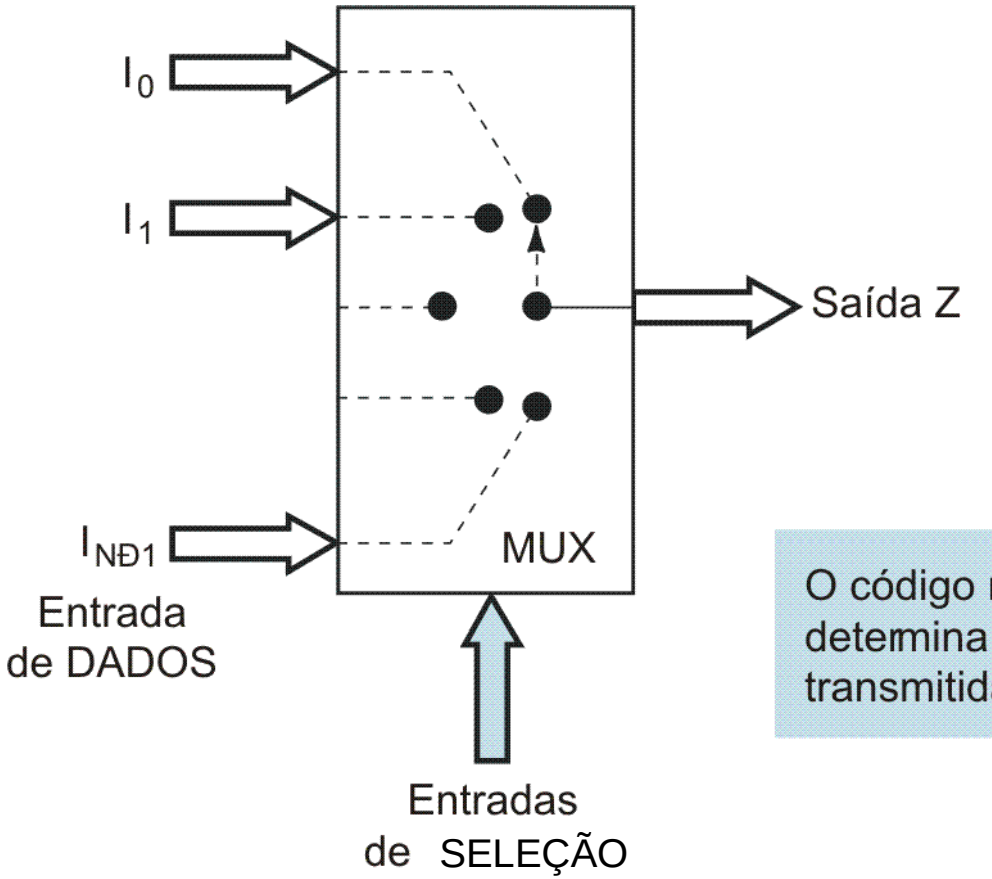
Codificador Octal para Binário (8 pra 3)



Inputs								Outputs		
$\bar{A}_0$	$\bar{A}_1$	$\bar{A}_2$	$\bar{A}_3$	$\bar{A}_4$	$\bar{A}_5$	$\bar{A}_6$	$\bar{A}_7$	O_2	O_1	O_0
X	1	1	1	1	1	1	1	0	0	0
X	0	1	1	1	1	1	1	0	0	1
X	1	0	1	1	1	1	1	0	1	0
X	1	1	0	1	1	1	1	0	1	1
X	1	1	1	0	1	1	1	1	0	0
X	1	1	1	1	0	1	1	1	0	1
X	1	1	1	1	1	0	1	1	1	0
X	1	1	1	1	1	1	0	1	1	1

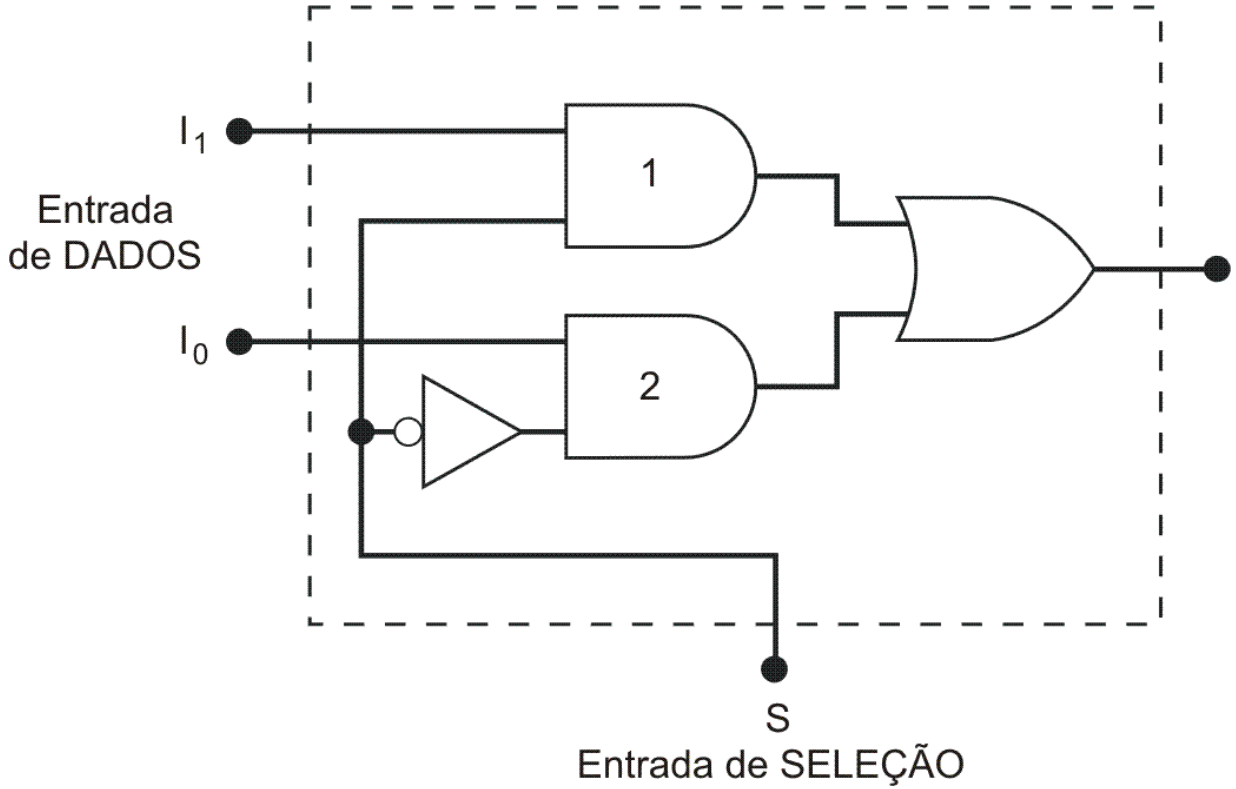
Para uma operação adequada, apenas uma entrada deve ser ativada de cada vez.

Diagrama funcional de um multiplexador



O código na entrada de SELEÇÃO determina a entrada que é transmitida para a saída Z

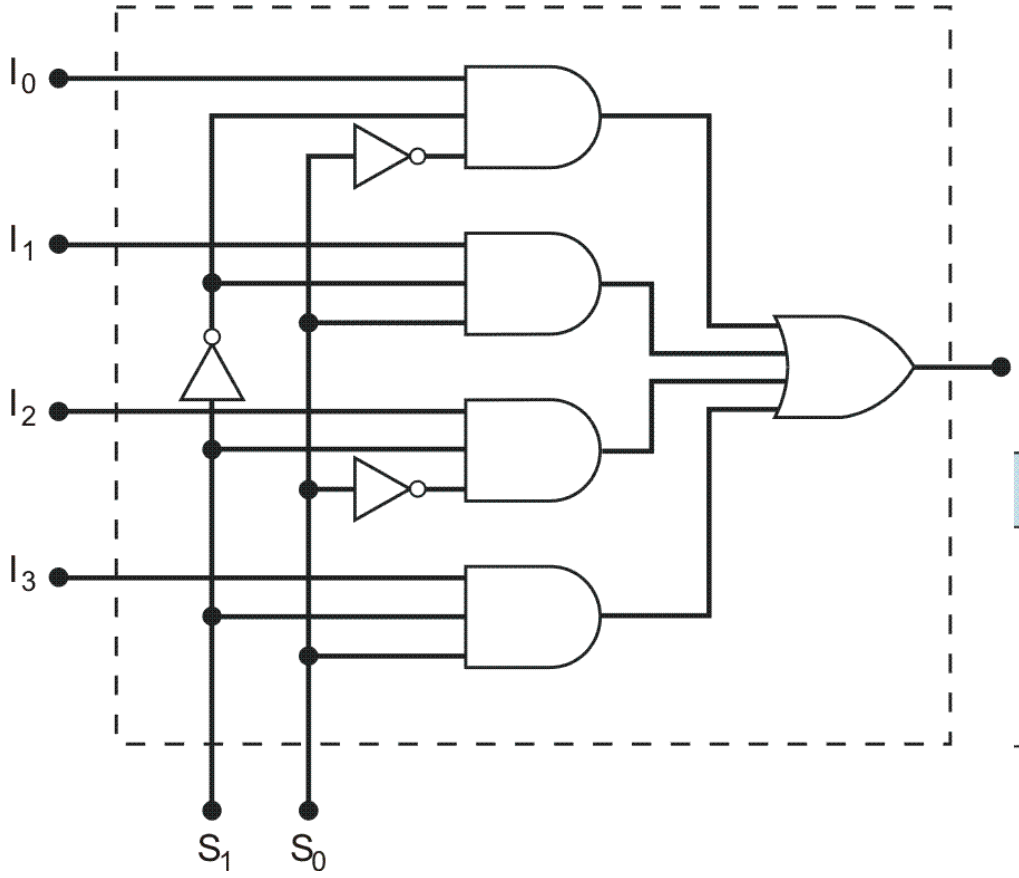
Implementação do multiplexador de 2 entradas



$$Z = I_0 \cdot \bar{S} + I_1 \cdot S$$

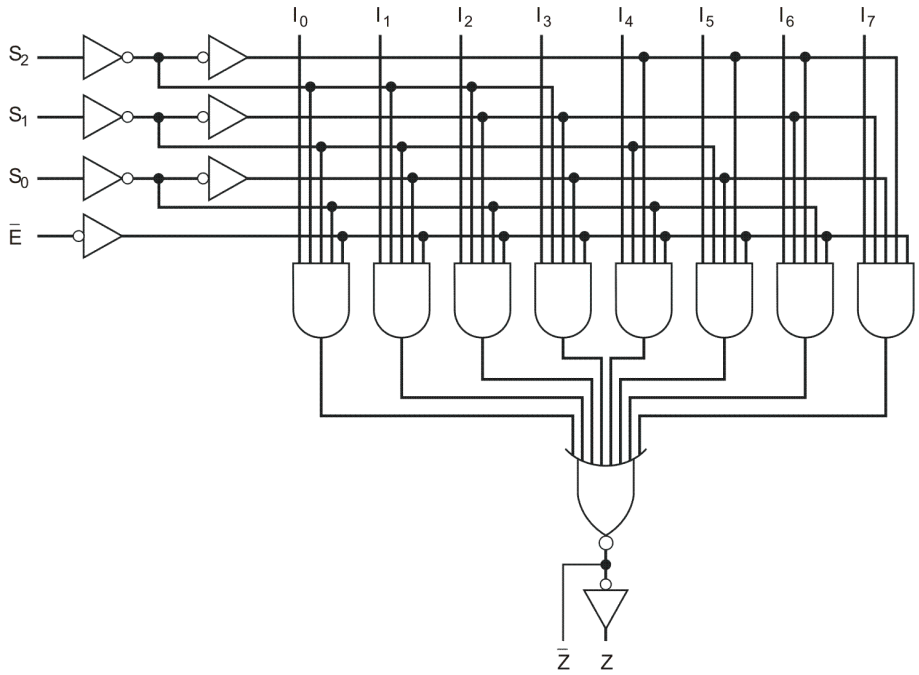
S	Saída
0	$Z = I_0$
1	$Z = I_1$

Implementação do multiplexador de 4 entradas



S_1	S_0	Saída
0	0	$Z = I_0$
0	1	$Z = I_1$
1	0	$Z = I_2$
1	1	$Z = I_3$

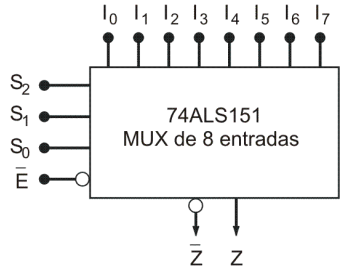
Multiplexador 74ALS151: 3 entradas + EN



(a)

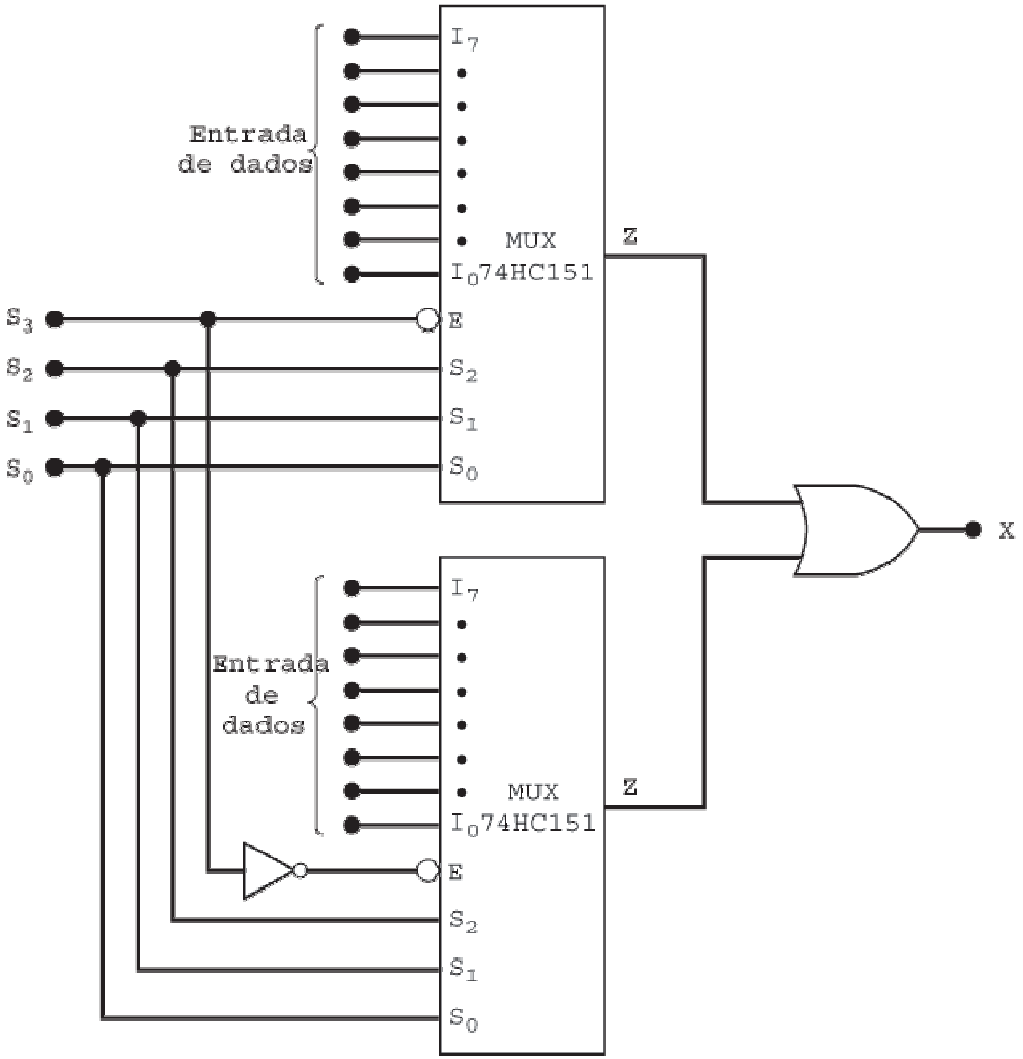
Inputs				Saída	
$\bar{E}$	S_2	S_1	S_0	$\bar{Z}$	Z
H	X	X	X	H	L
L	L	L	L	$\bar{I}_0$	I_0
L	L	L	H	$\bar{I}_1$	I_1
L	L	H	L	$\bar{I}_2$	I_2
L	L	H	H	$\bar{I}_3$	I_3
L	H	L	L	$\bar{I}_4$	I_4
L	H	L	H	$\bar{I}_5$	I_5
L	H	H	L	$\bar{I}_6$	I_6
L	H	H	H	$\bar{I}_7$	I_7

(b)

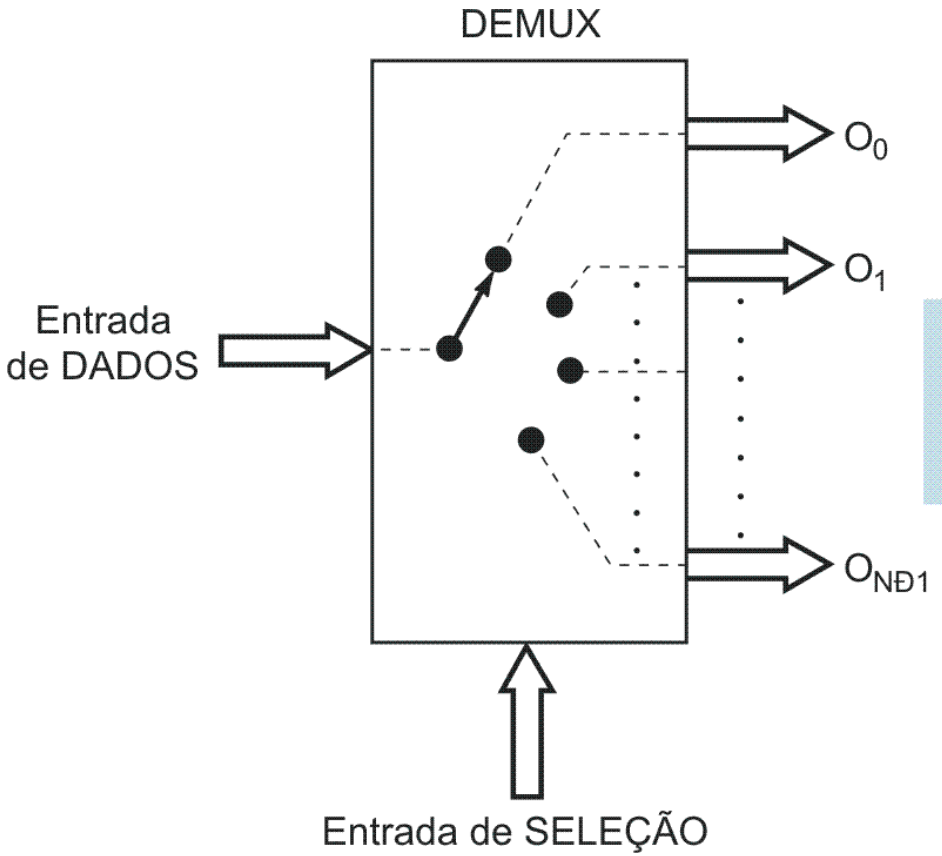


(c)

Multiplexador de 16 entradas a partir de dois CIs 74HC151

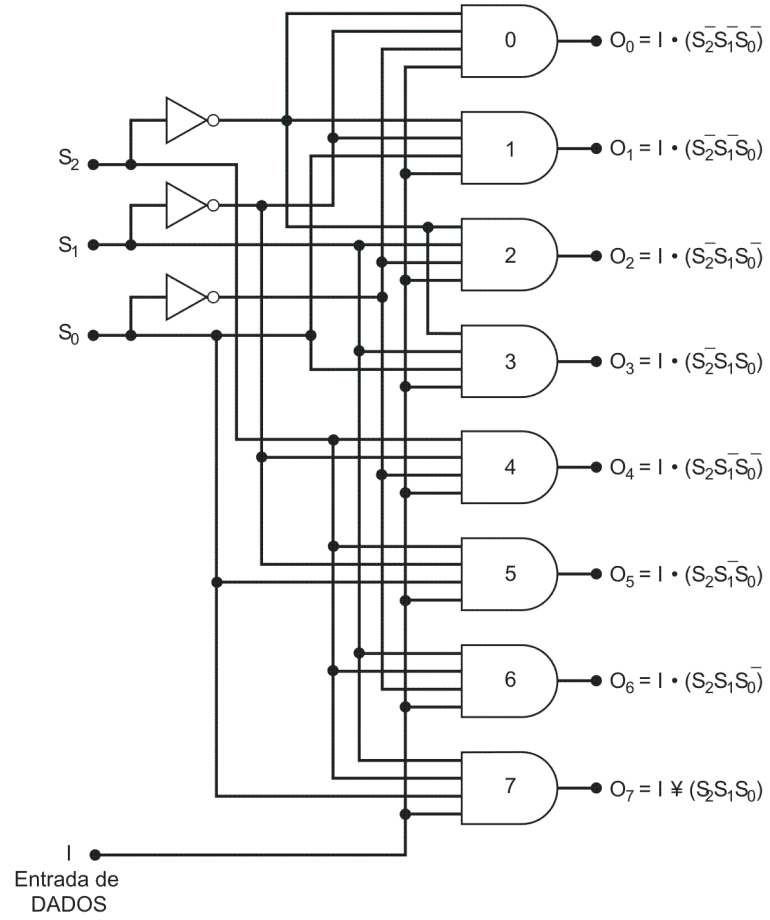


Demultiplexador



A entrada de DADOS é transmitida apenas para uma das saídas, conforme determinado pelo código de seleção de entrada

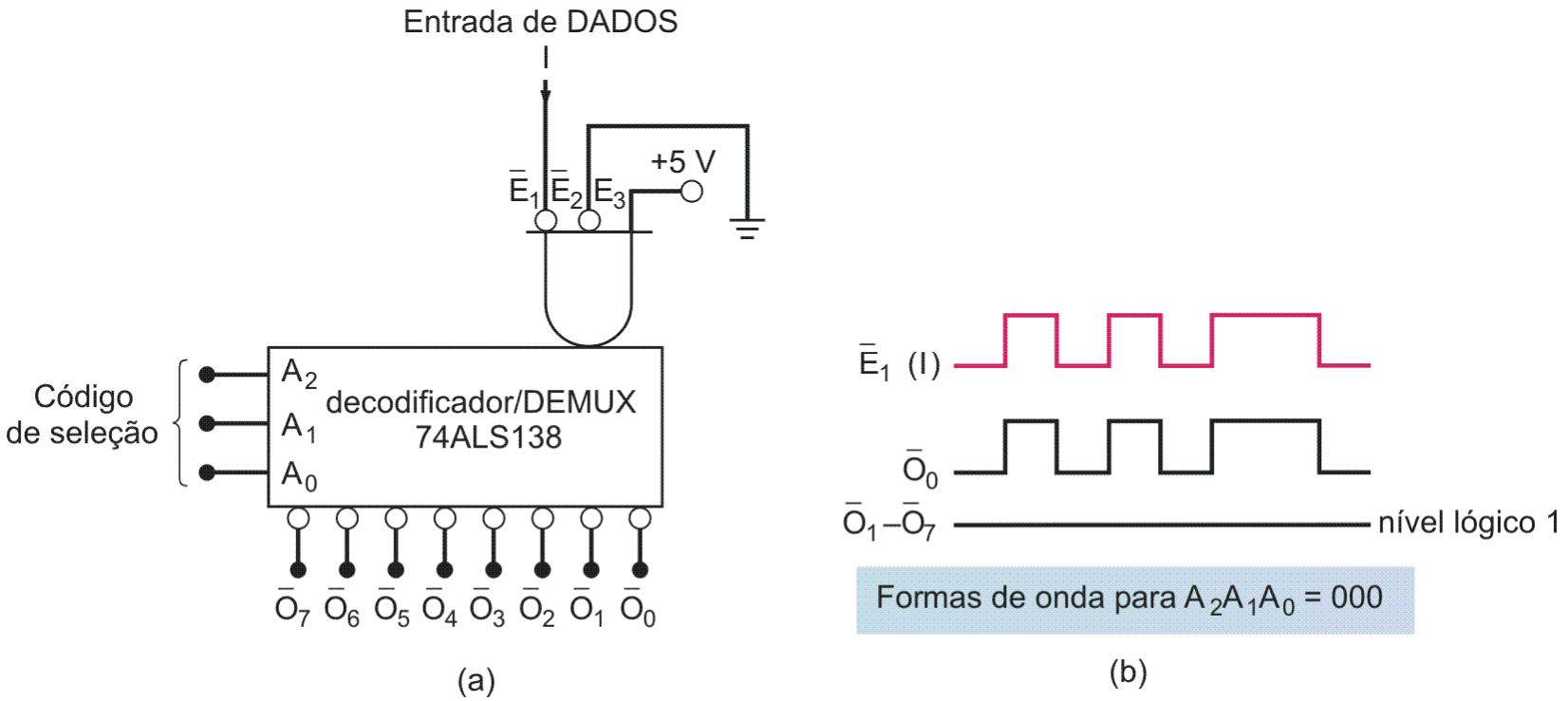
Demultiplexador de 1 pra 8



Código de SELEÇÃO			SAÍDAS							
S_2	S_1	S_0	O_7	O_6	O_5	O_4	O_3	O_2	O_1	O_0
0	0	0	0	0	0	0	0	0	0	I
0	0	1	0	0	0	0	0	0	I	0
0	1	0	0	0	0	0	0	I	0	0
0	1	1	0	0	0	0	I	0	0	0
1	0	0	0	0	0	I	0	0	0	0
1	0	1	0	0	I	0	0	0	0	0
1	1	0	0	I	0	0	0	0	0	0
1	1	1	I	0	0	0	0	0	0	0

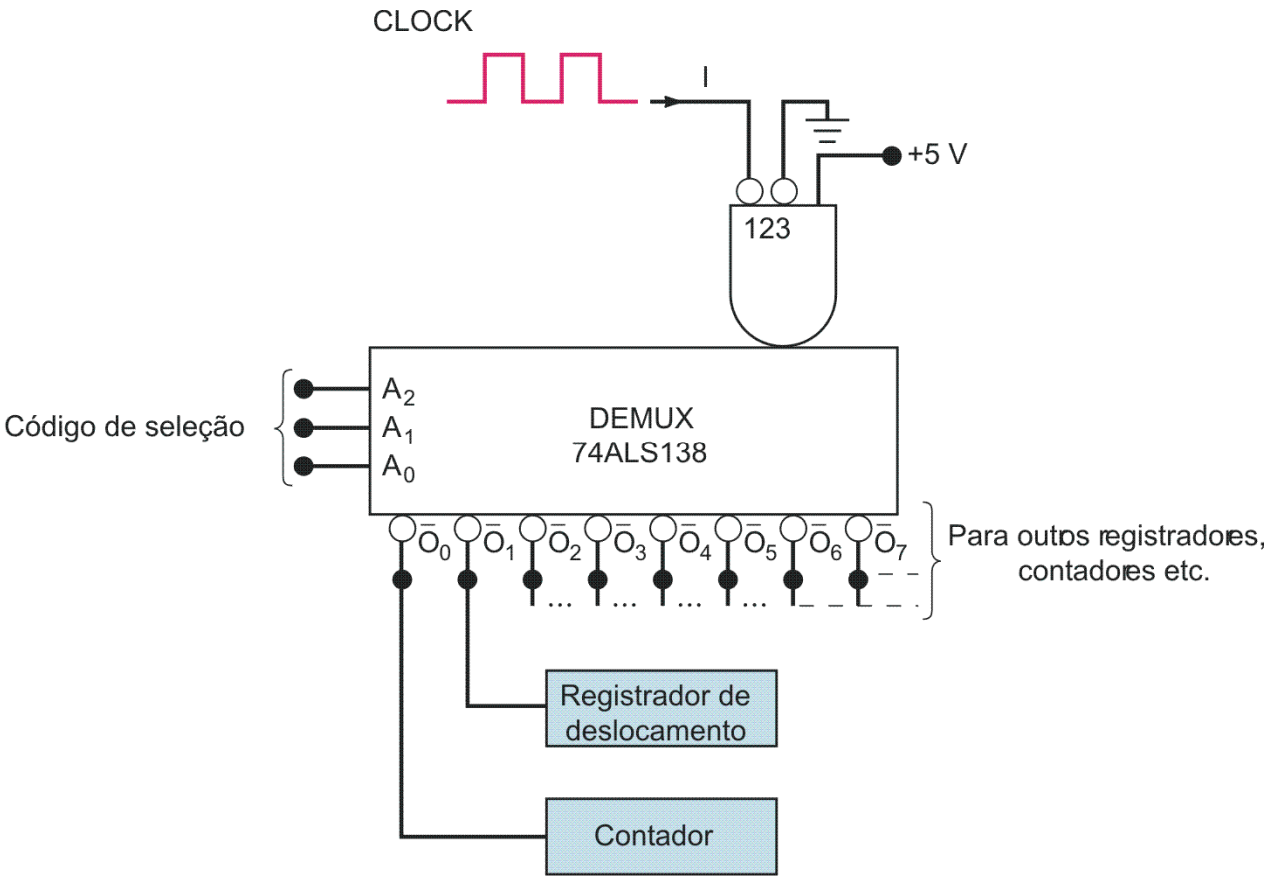
Nota: I é a entrada de dados

Decodificador 138 funcionando como demultiplexador



(a) E_1 usada como entrada de dado. (b) Formas de onda típicas para o código de seleção A_2 A_1 A_0 = 000 mostram que O_0 é idêntica a entrada de dados E_1.

Demultiplexador de clock



Transmite o sinal de *clock* para um destino determinado pelas entradas de seleção.

Comparador de magnitude – 4 bits

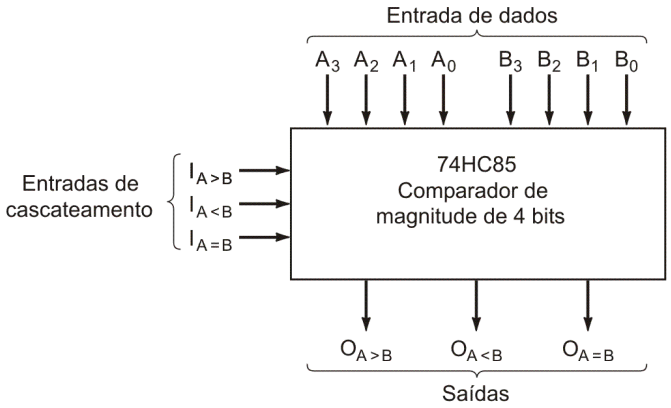
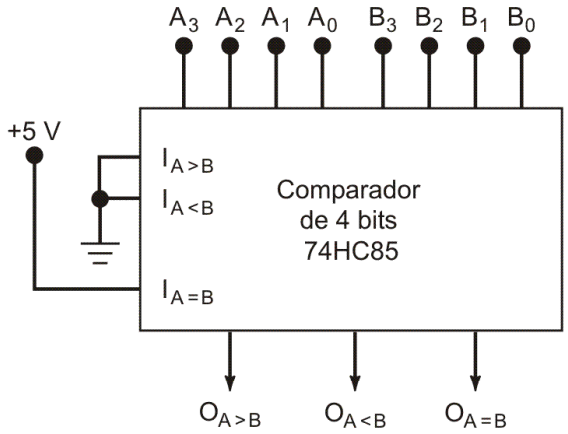


TABELA-VERDADE

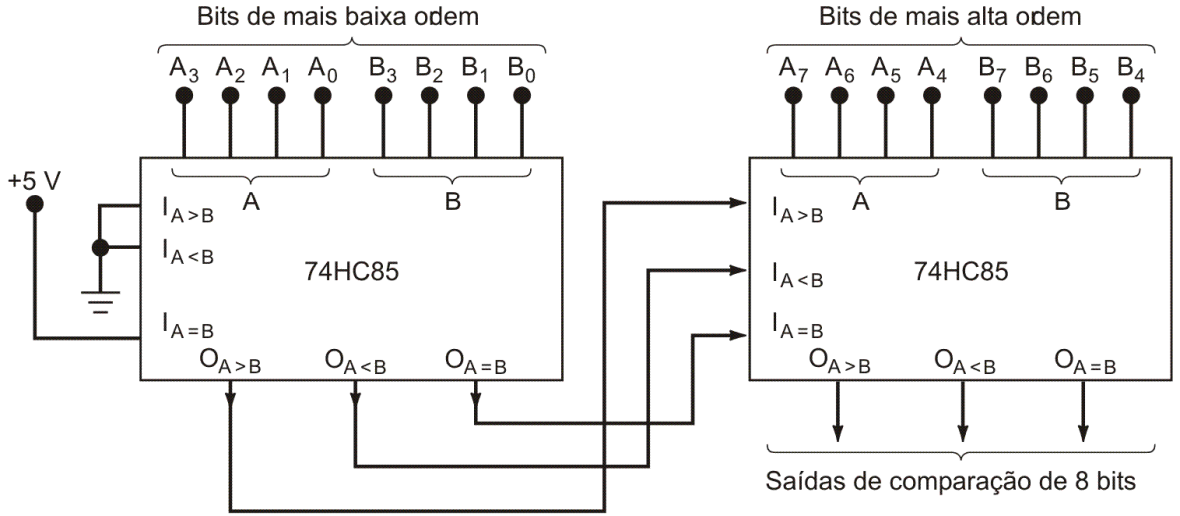
ENTRADAS DE COMPARAÇÃO				ENTRADAS DE CASCATEAMENTO			SAÍDAS		
A_3, B_3	A_2, B_2	A_1, B_1	A_0, B_0	$I_{A>B}$	$I_{A<B}$	$I_{A=B}$	$O_{A>B}$	$O_{A<B}$	$O_{A=B}$
$A_3>B_3$	X	X	X	X	X	X	H	L	L
$A_3<B_3$	X	X	X	X	X	X	L	H	L
$A_3=B_3$	$A_2>B_2$	X	X	X	X	X	H	L	L
$A_3=B_3$	$A_2<B_2$	X	X	X	X	X	L	H	L
$A_3=B_3$	$A_2=B_2$	$A_1>B_1$	X	X	X	X	H	L	L
$A_3=B_3$	$A_2=B_2$	$A_1<B_1$	X	X	X	X	L	H	L
$A_3=B_3$	$A_2=B_2$	$A_1=B_1$	$A_0>B_0$	X	X	X	H	L	L
$A_3=B_3$	$A_2=B_2$	$A_1=B_1$	$A_0<B_0$	X	X	X	L	H	L
$A_3=B_3$	$A_2=B_2$	$A_1=B_1$	$A_0=B_0$	H	L	L	H	L	L
$A_3=B_3$	$A_2=B_2$	$A_1=B_1$	$A_0=B_0$	L	H	L	L	H	L
$A_3=B_3$	$A_2=B_2$	$A_1=B_1$	$A_0=B_0$	X	X	H	L	L	H
$A_3=B_3$	$A_2=B_2$	$A_1=B_1$	$A_0=B_0$	L	L	L	H	H	L
$A_3=B_3$	$A_2=B_2$	$A_1=B_1$	$A_0=B_0$	H	H	L	L	L	L

H = Nível de tensão ALTO
L = Nível de tensão BAIXO
X = Irr elevante

Comparador de 8 bits a partir de dois de 4 bits

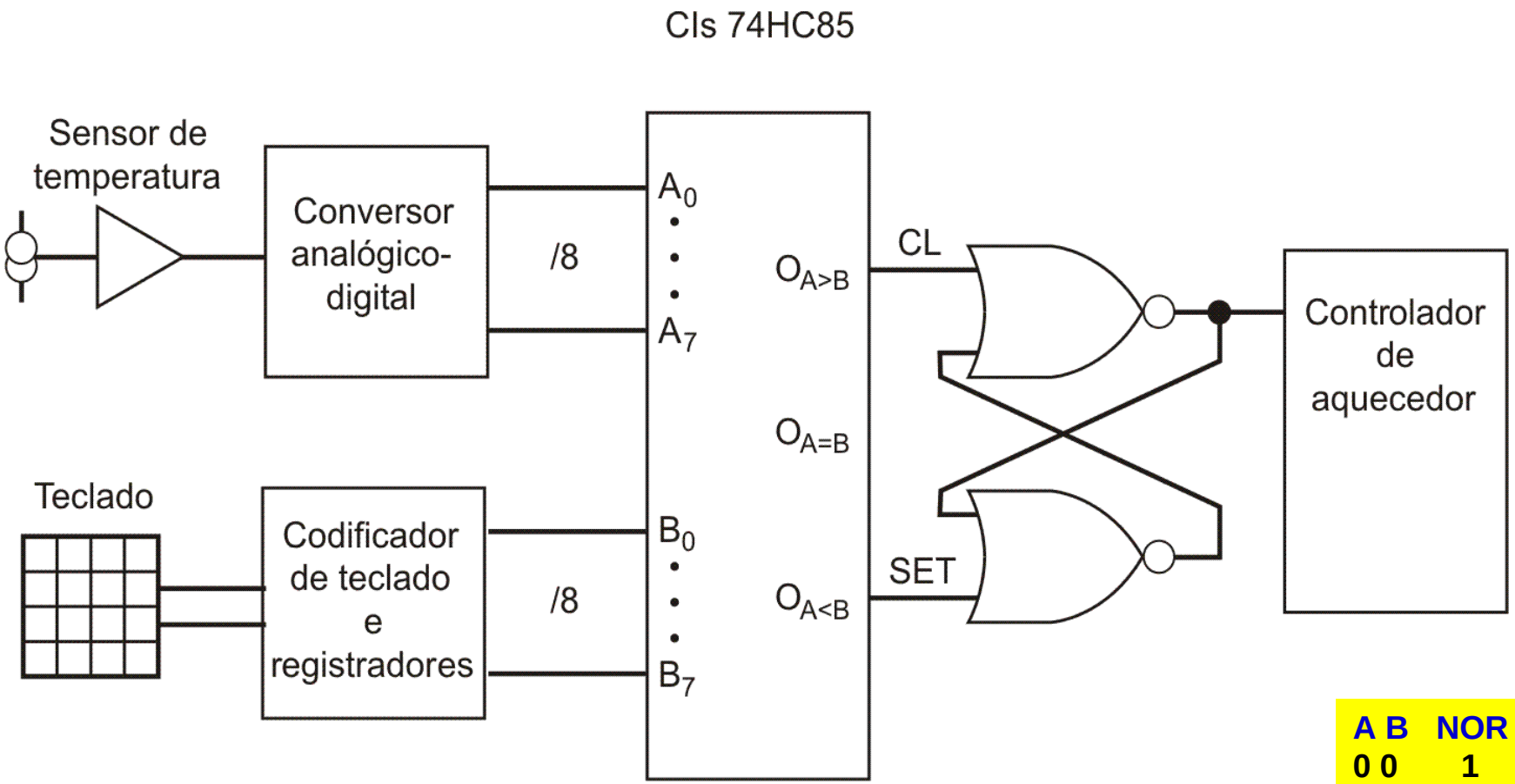


(a)



(b)

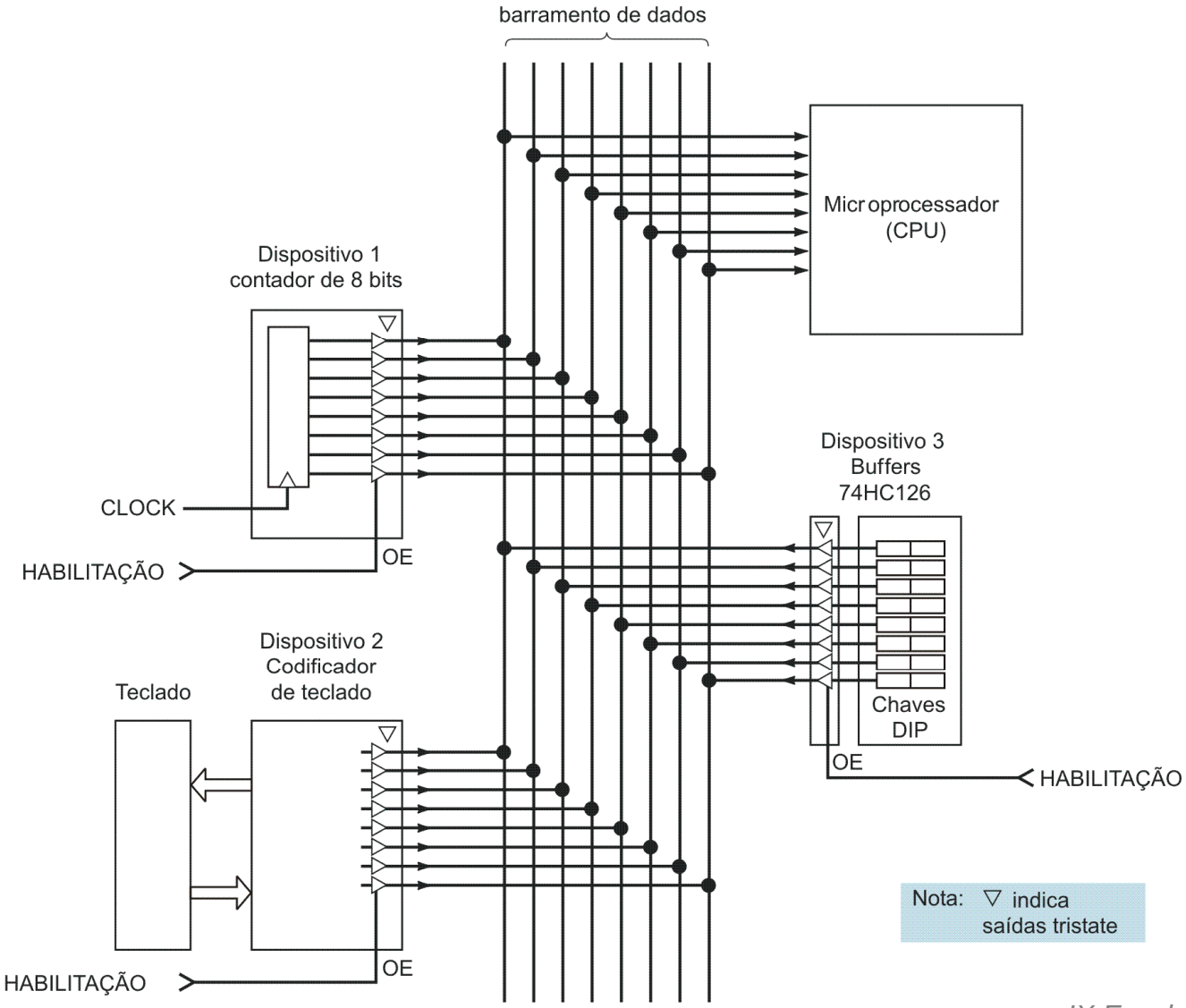
Comparador de magnitude usado em termostato digital



A	B	NOR
0	0	1
0	1	0
1	0	0
1	1	0

Três dispositivos transmitindo 1 byte para um microprocessador

(apenas um dispositivo é habilitado por vez)



Registrador tri-state 74ALS173

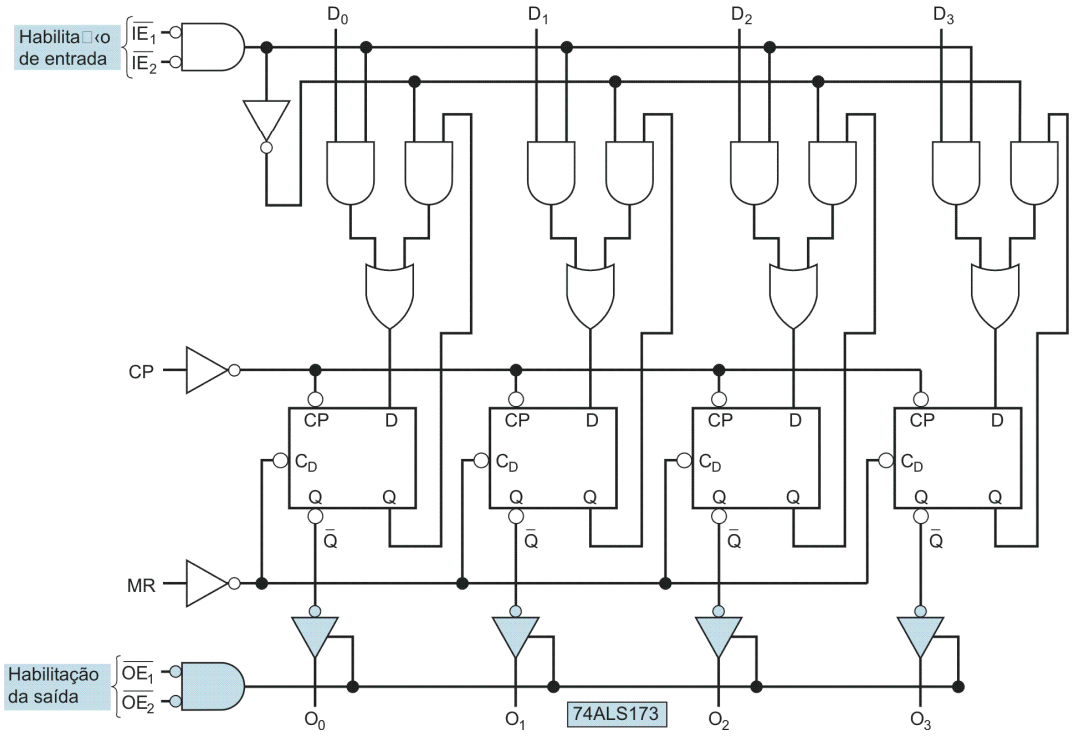
Entradas					Saídas dos FFs
MR	C	$\overline{IE_1}$	$\overline{IE_2}$	D_n	Q
H	X	X	X	X	L
L	L	X	X	X	Q_0
L	⎍	H	X	X	Q_0
L	⎍	X	H	X	Q_0
L	⎍	L	L	L	L
L	⎍	L	L	H	H

Quando $\overline{OE_1}$ ou $\overline{OE_2}$ estão em nível ALTO, a saída está em OFF (alta impedância); entretanto, isso não afeta o conteúdo ou a operação seqüencial do registrador

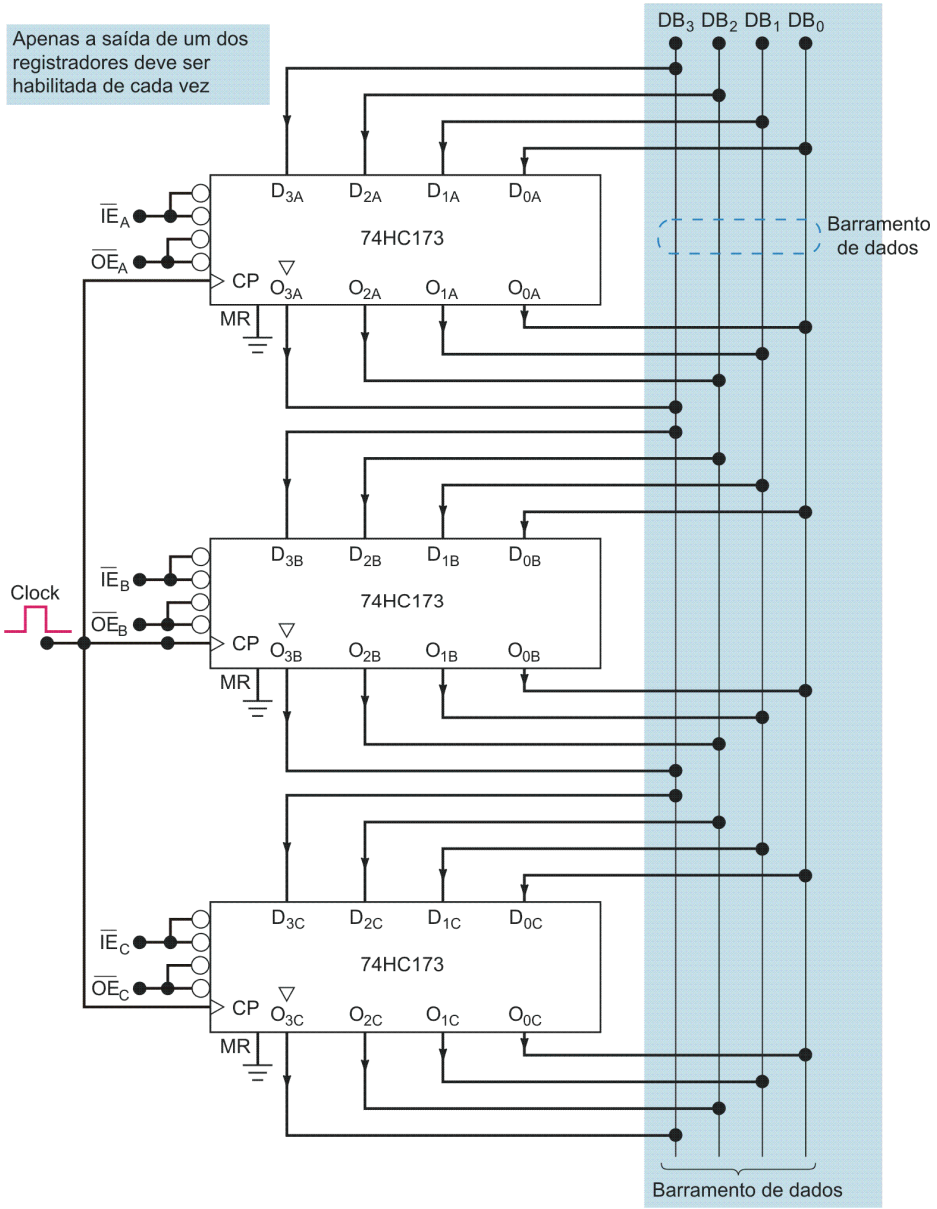
H = Nível de tensão ALTO
L = Nível de tensão BAIXO
X = irrelevante

Q_0 = saída antes da transição positiva

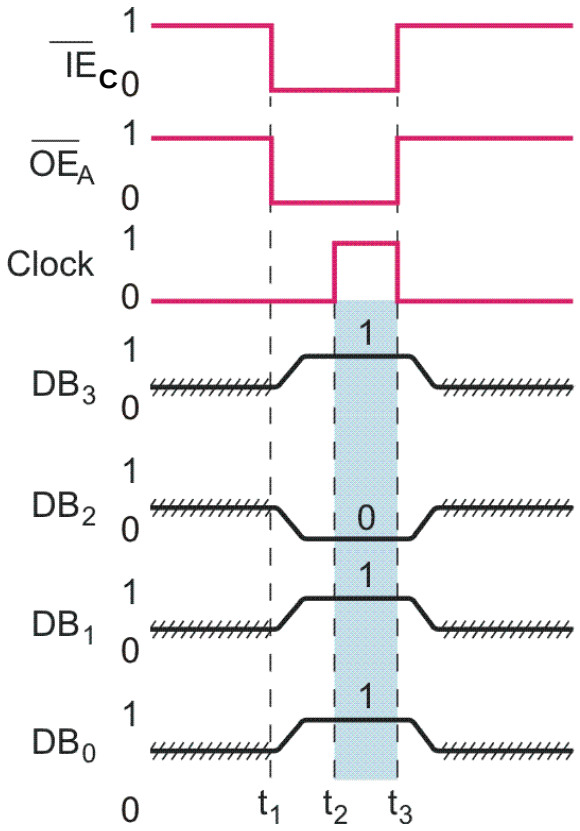
Diagrama lógico



Registradores tri-state conectados a um barramento de dados



Transferência do dado 1011 do registrador A para o registrador C



NOTAS:

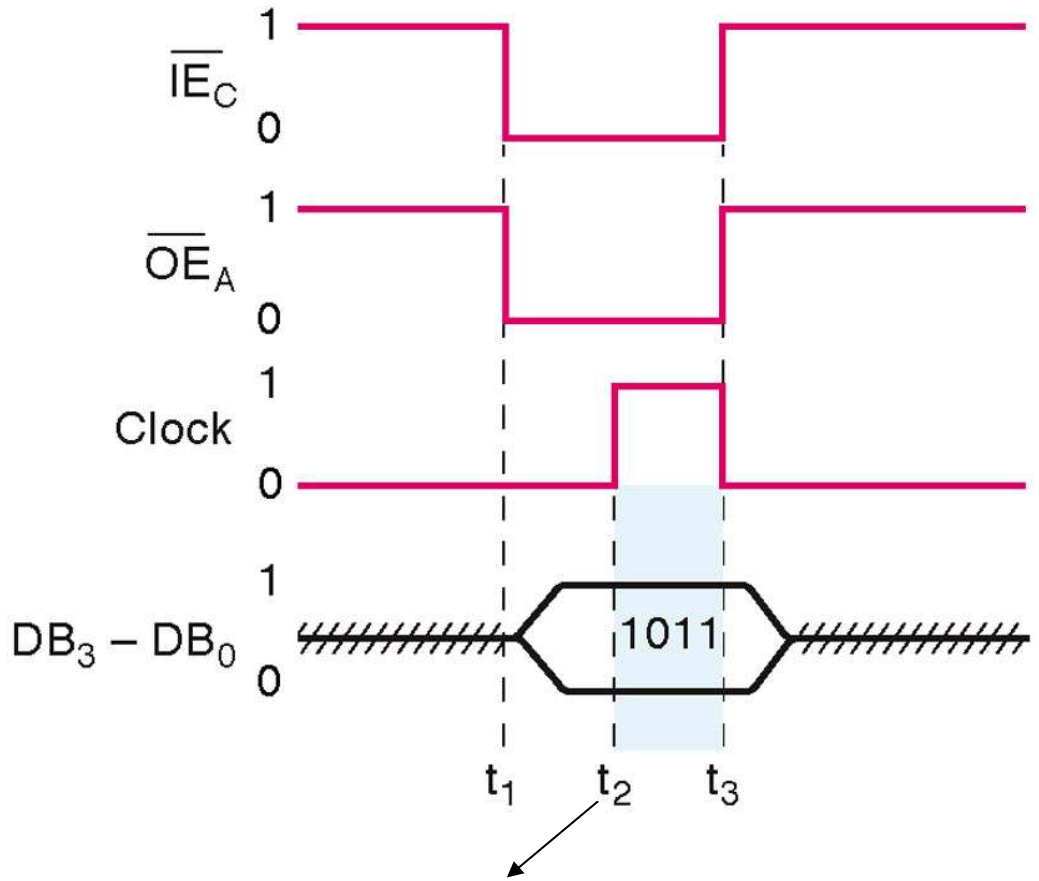
= Flutuação (alta impedância)

t_1: As saídas do registrador A são habilitadas. Seus dados são colocados nas linhas do barramento de dados.

t_2: A transiçãoo positiva do clock transfere os dados válidos do barramento de dados para o registrador C.

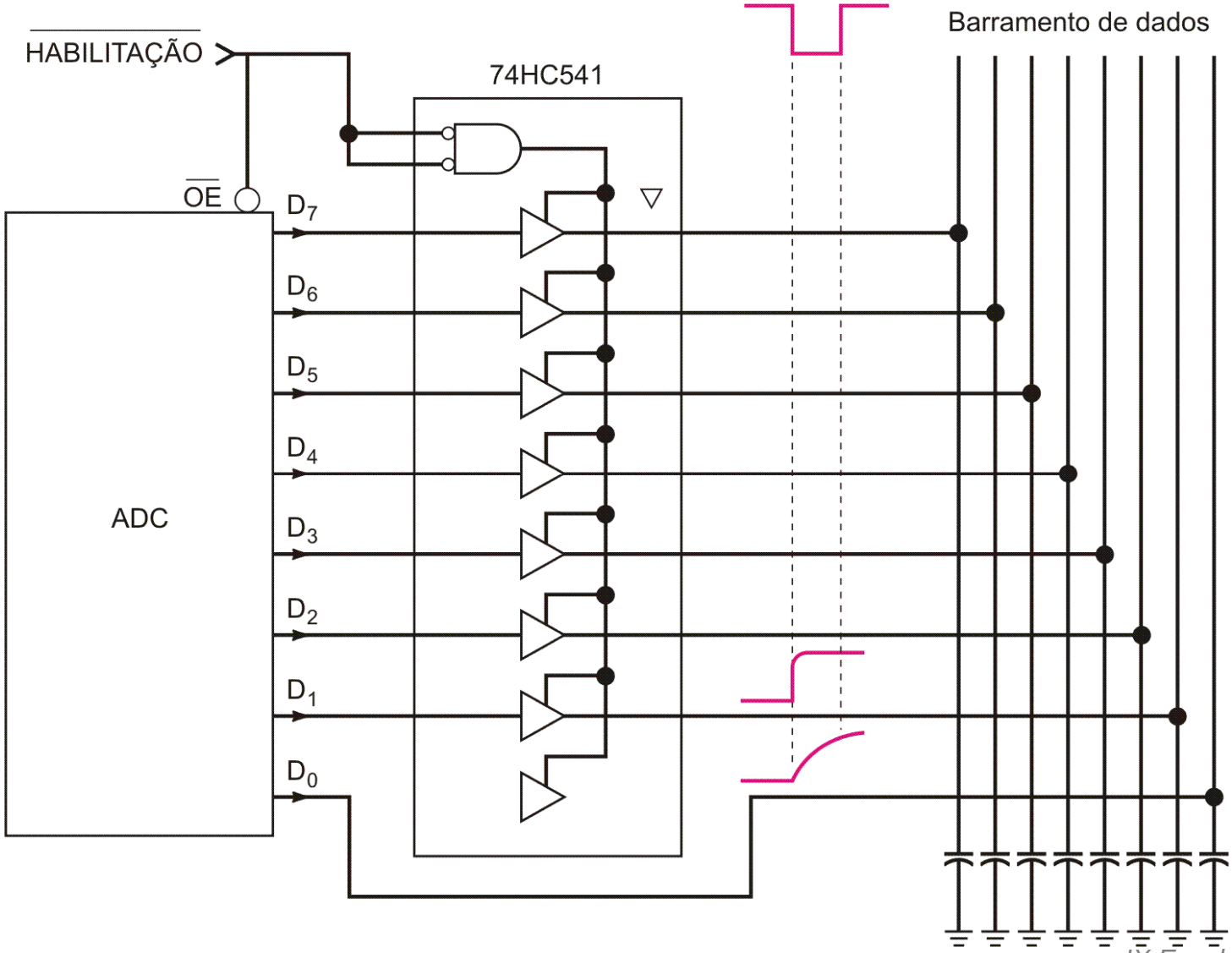
t_3: As saídas do registrador A são desabilitadas e as linhas do barramento de dados retornam para o estado de alta impedância.

Forma simplificada de mostrar a ativação de sinais nas linhas do barramento de dados

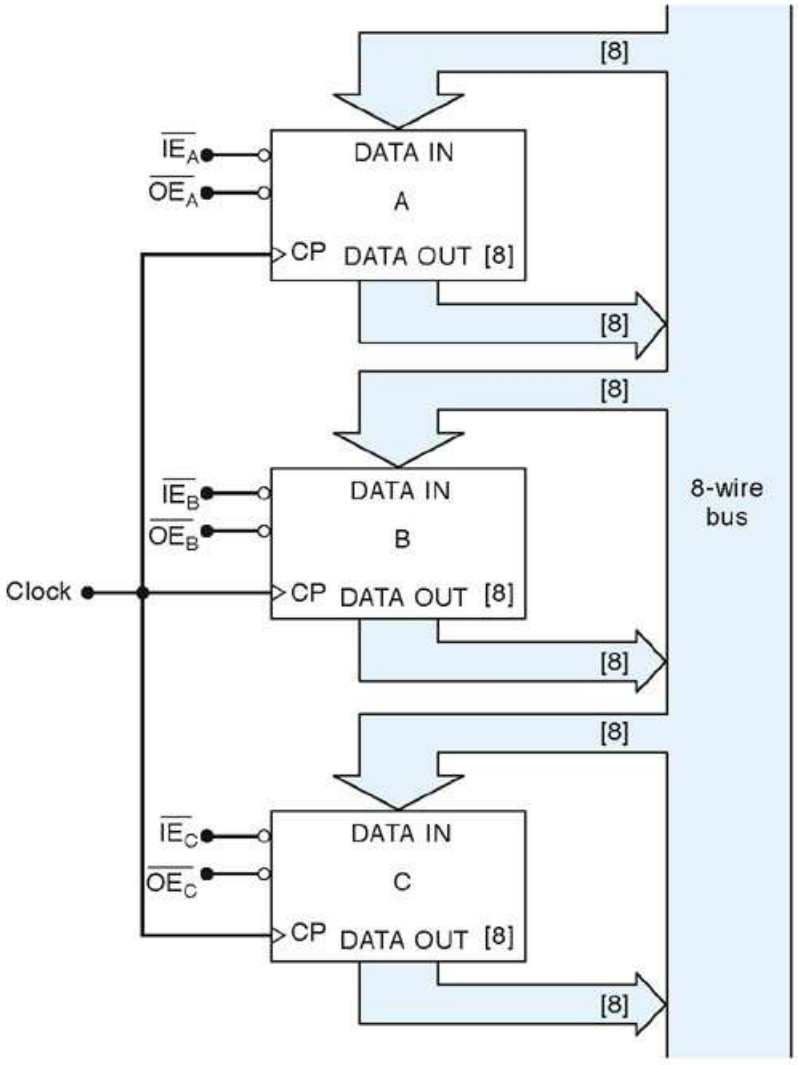


Observar o *SETUP TIME* e o *HOLD TIME* !

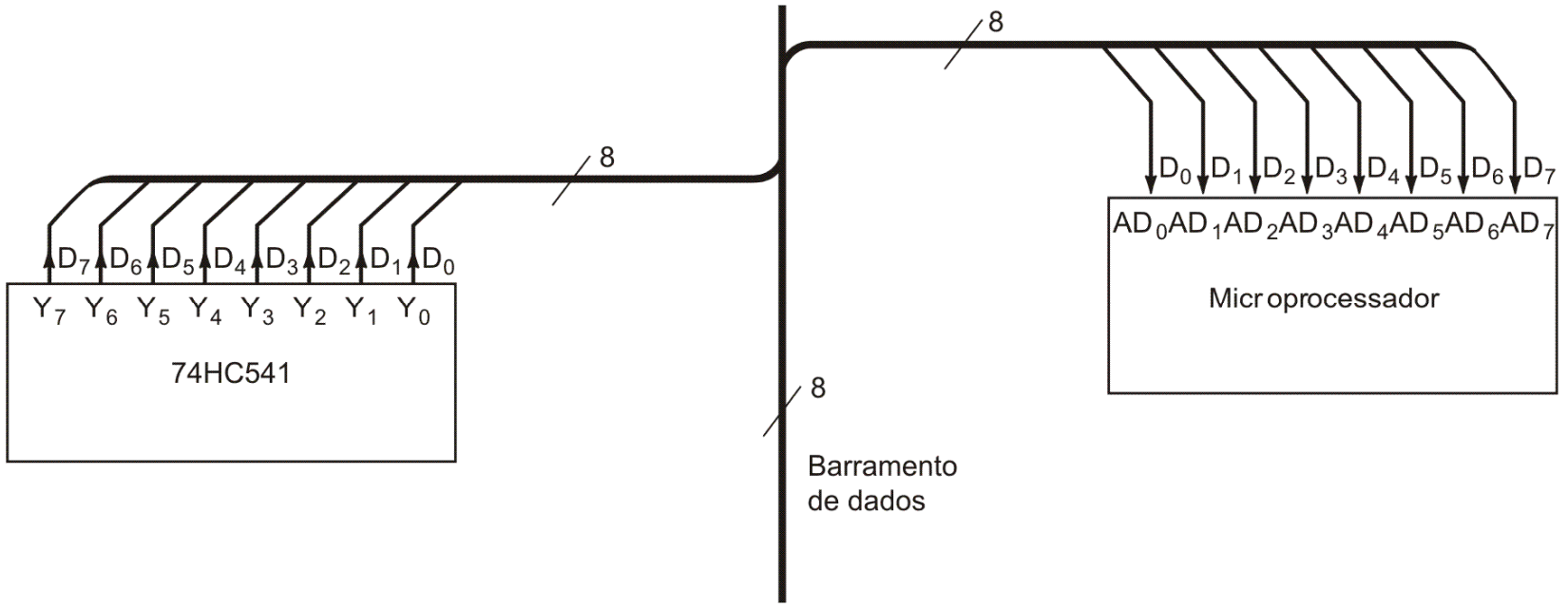
Um driver de barramento octal 74HC541 conecta as saídas de um conversor analógico-digital (ADC) a um barramento de dados. A saída D_0 está conectada diretamente no barramento, mostrando o efeito das capacitâncias parasitas.



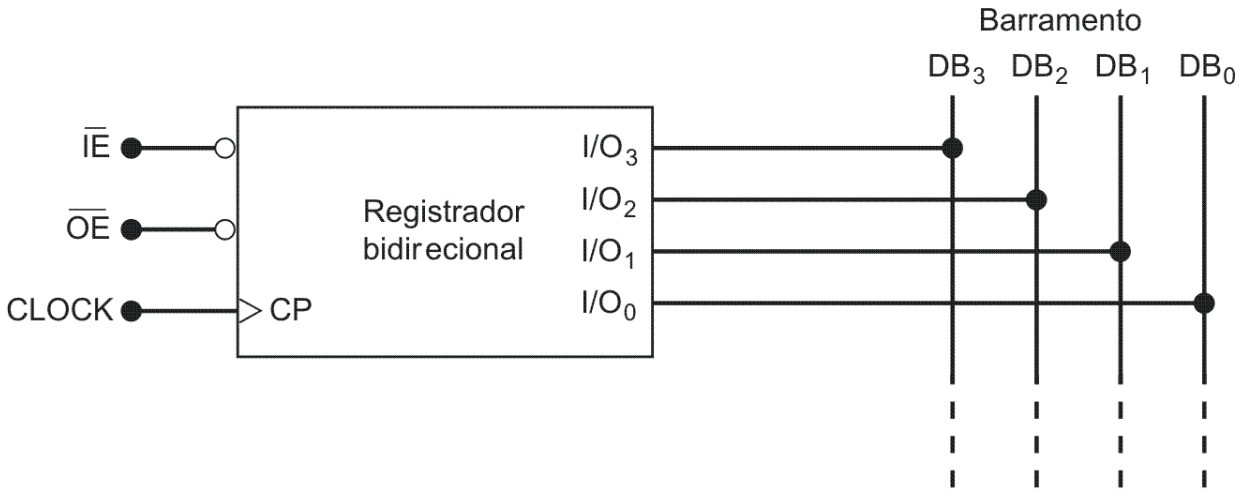
Representação simplificada de barramento



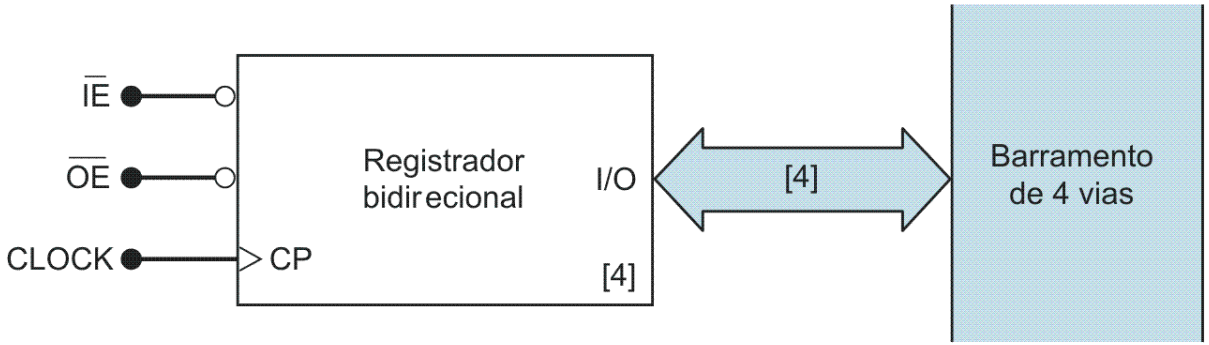
Representação simplificada de barramento



Registrador bidirecional conectado no barramento de dados

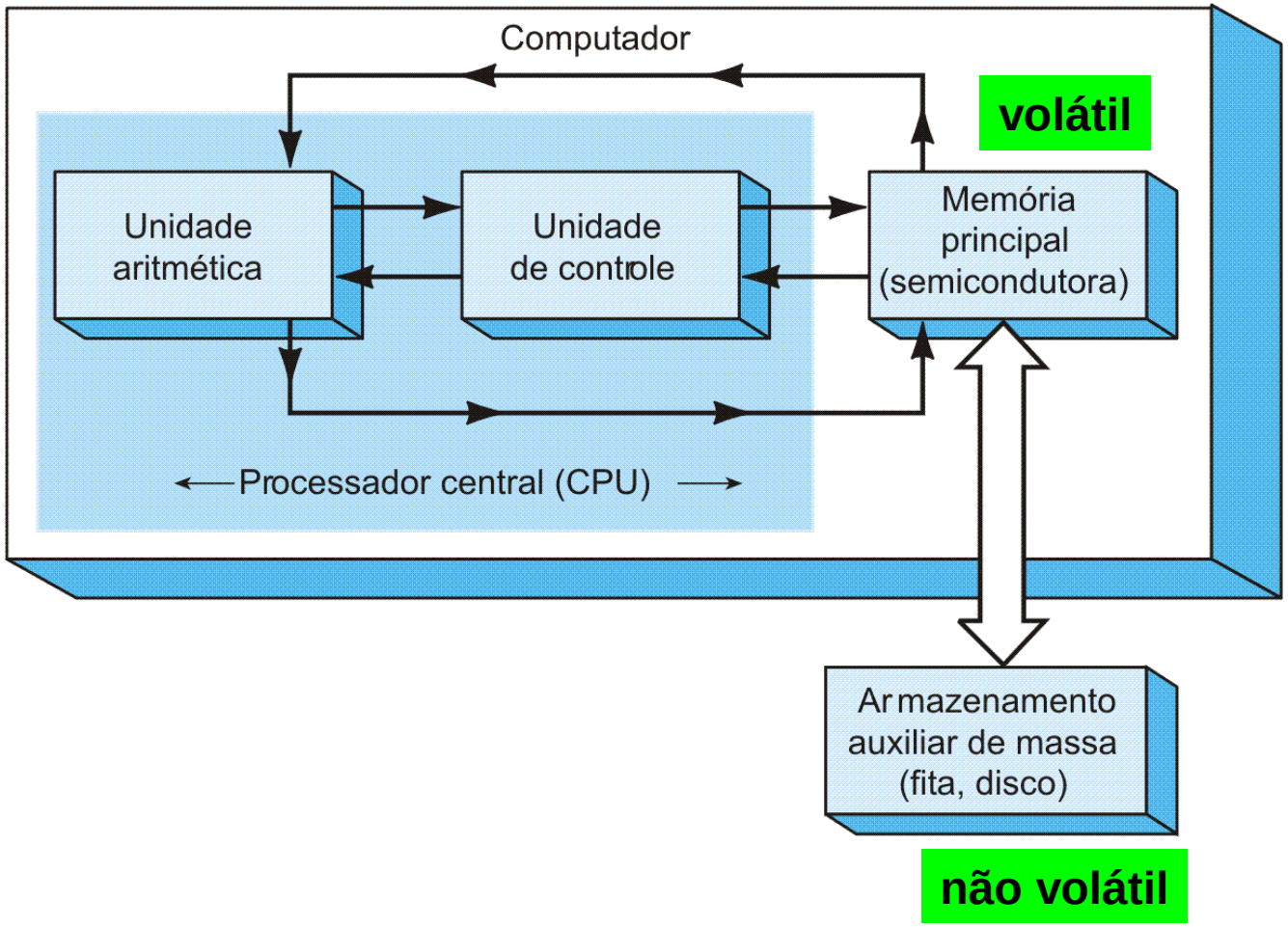


(a)

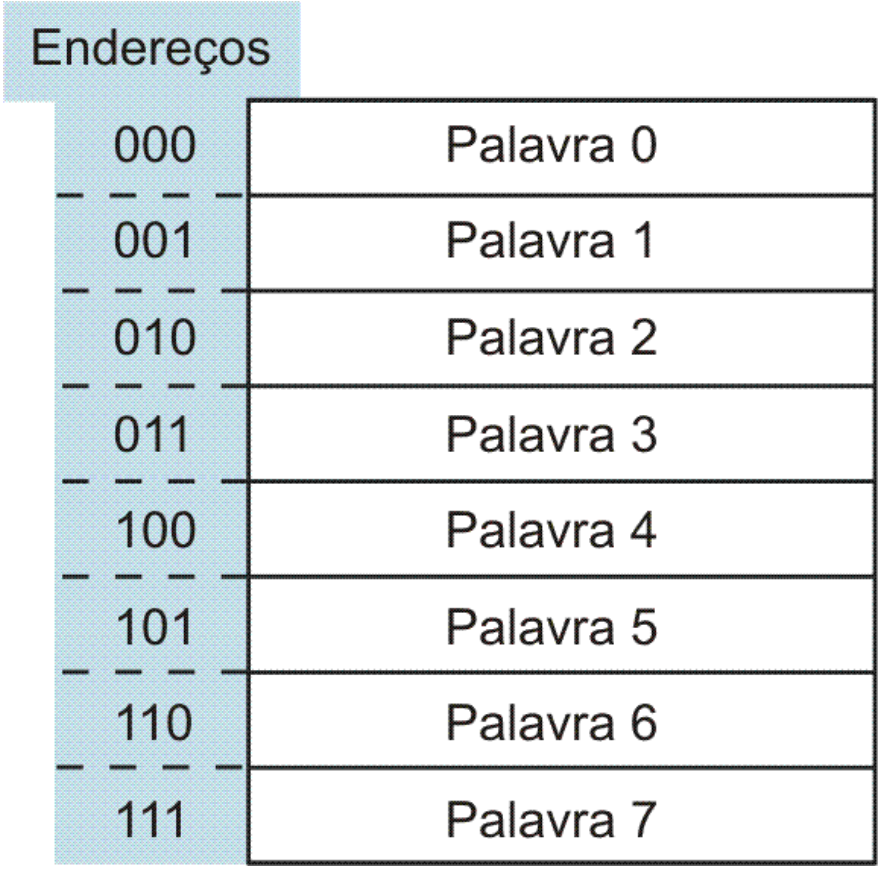


(b)

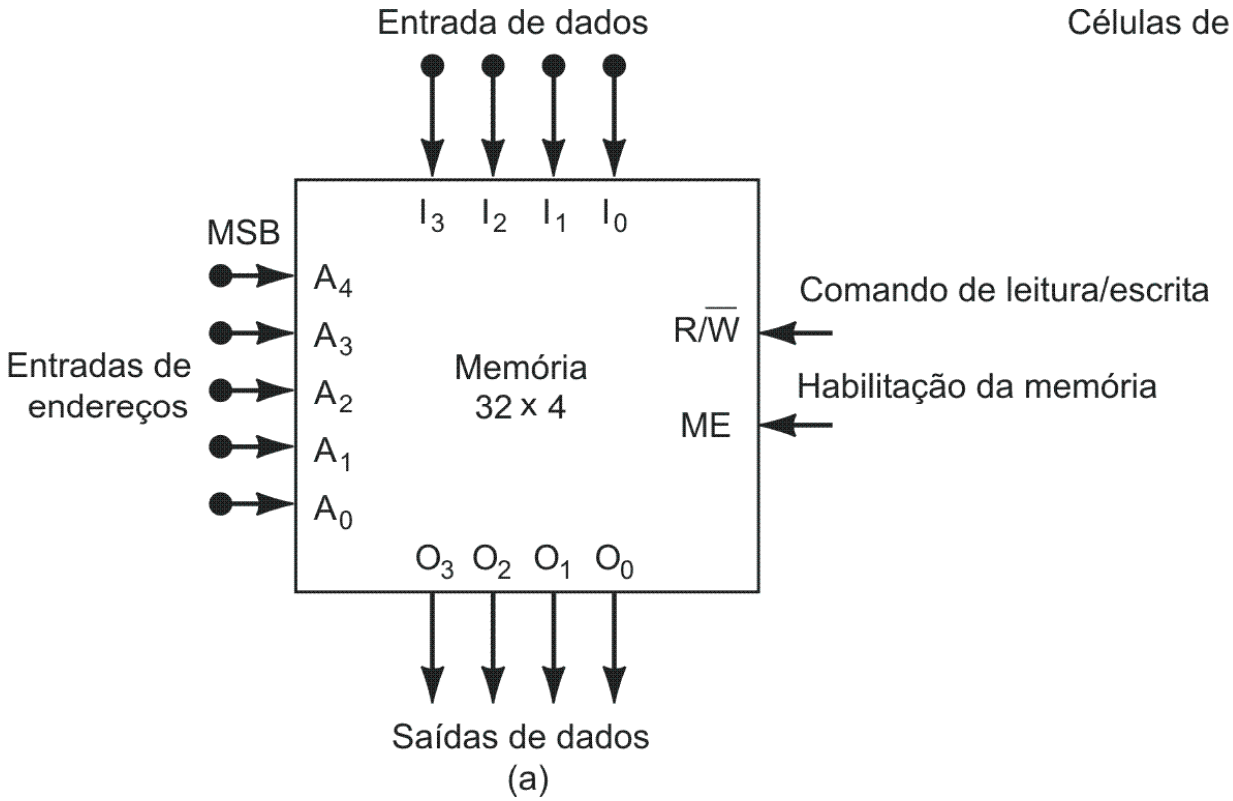
Um computador geralmente usa uma memória principal de alta velocidade e uma memória auxiliar externa mais lenta.

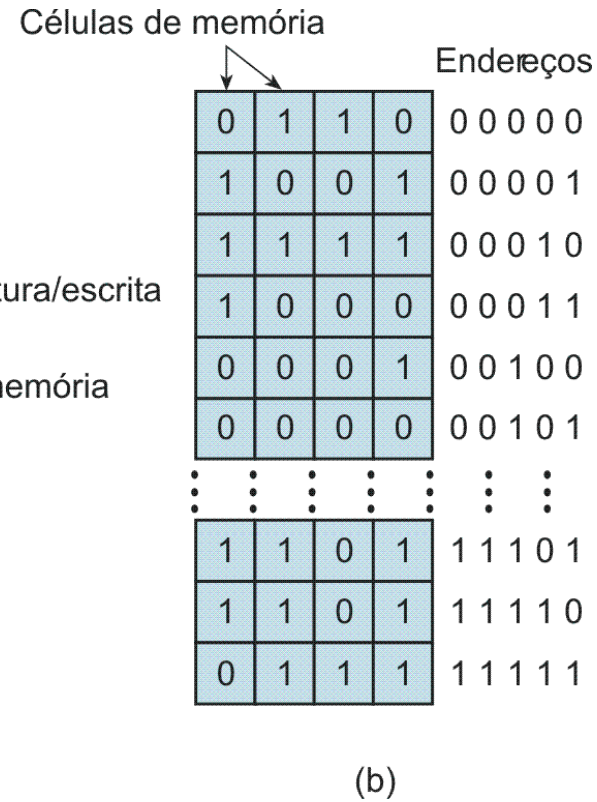


Cada posição tem um endereço binário específico

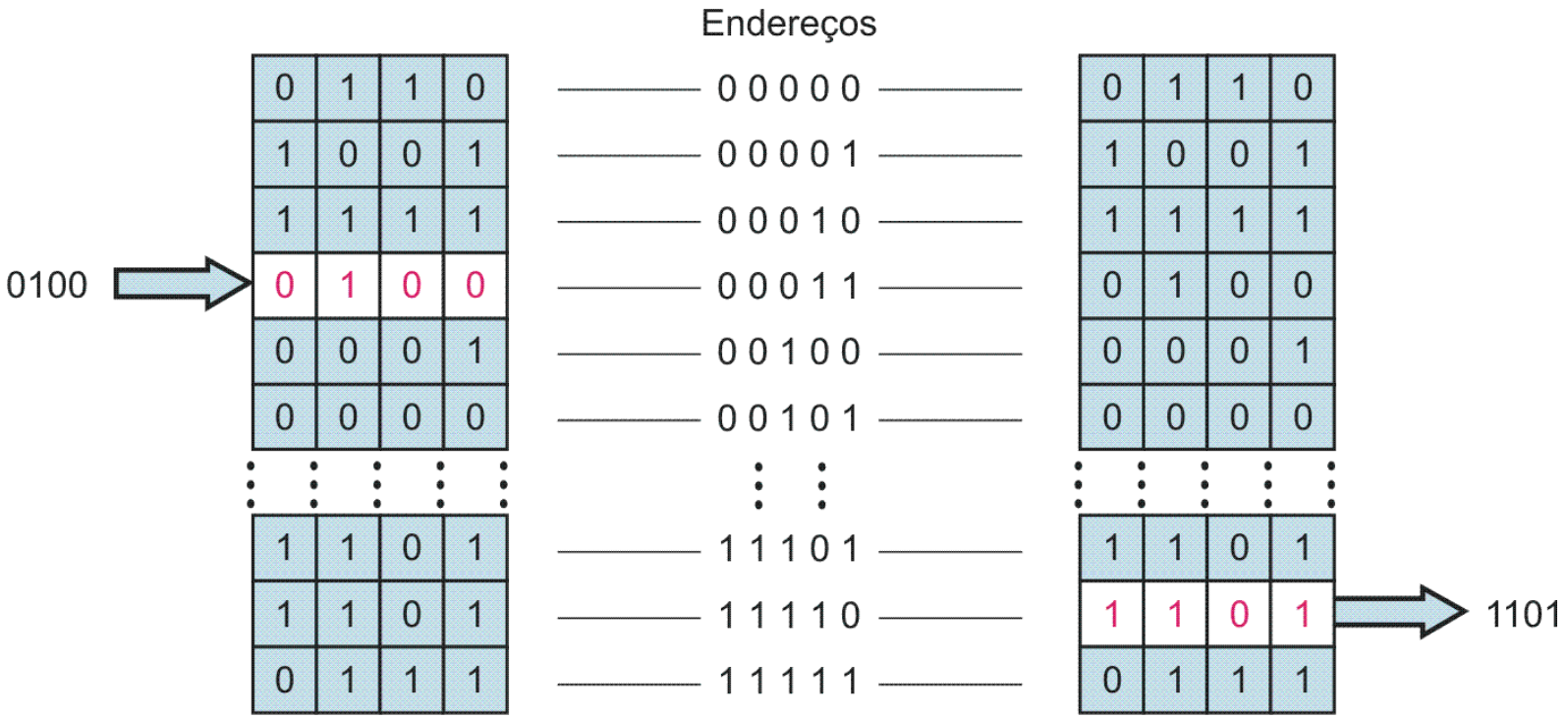


(a) Diagrama de uma memória 32 x 4
(b) Configuração virtual das células de memória em 32 palavras de 4 bits.





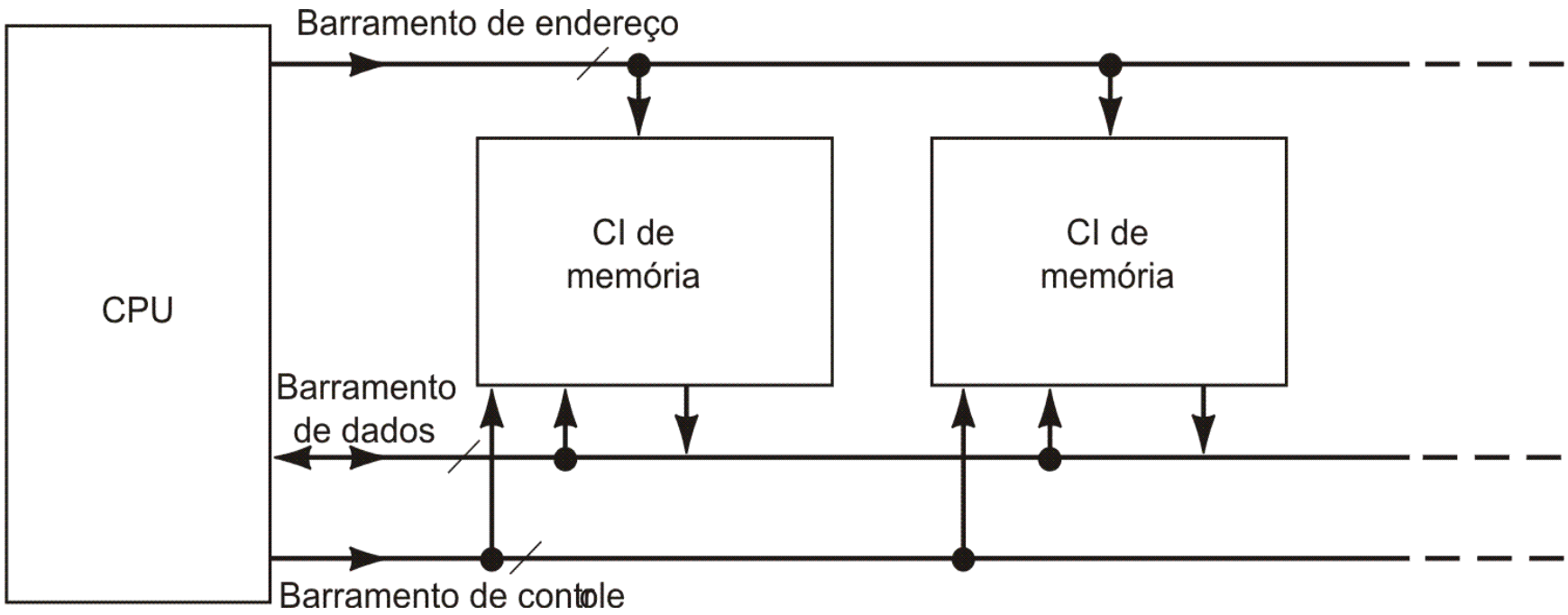
Visão simplificada das operações de leitura e de escrita em uma memória de 32 x 4



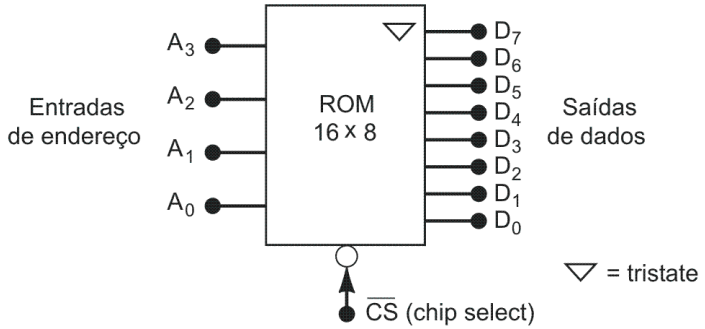
(a) ESCRITA da palavra 0100 na posição de memória 00011.

(b) LEITURA da palavra 1101 da posição de memória 11110.

Três barramentos conectando os CIs de memória principal na CPU



(a) Símbolo de uma memória ROM típica
(b) Tabela mostrando os dados binários de cada endereço
(c) A tabela em hexadecimal.



(a)

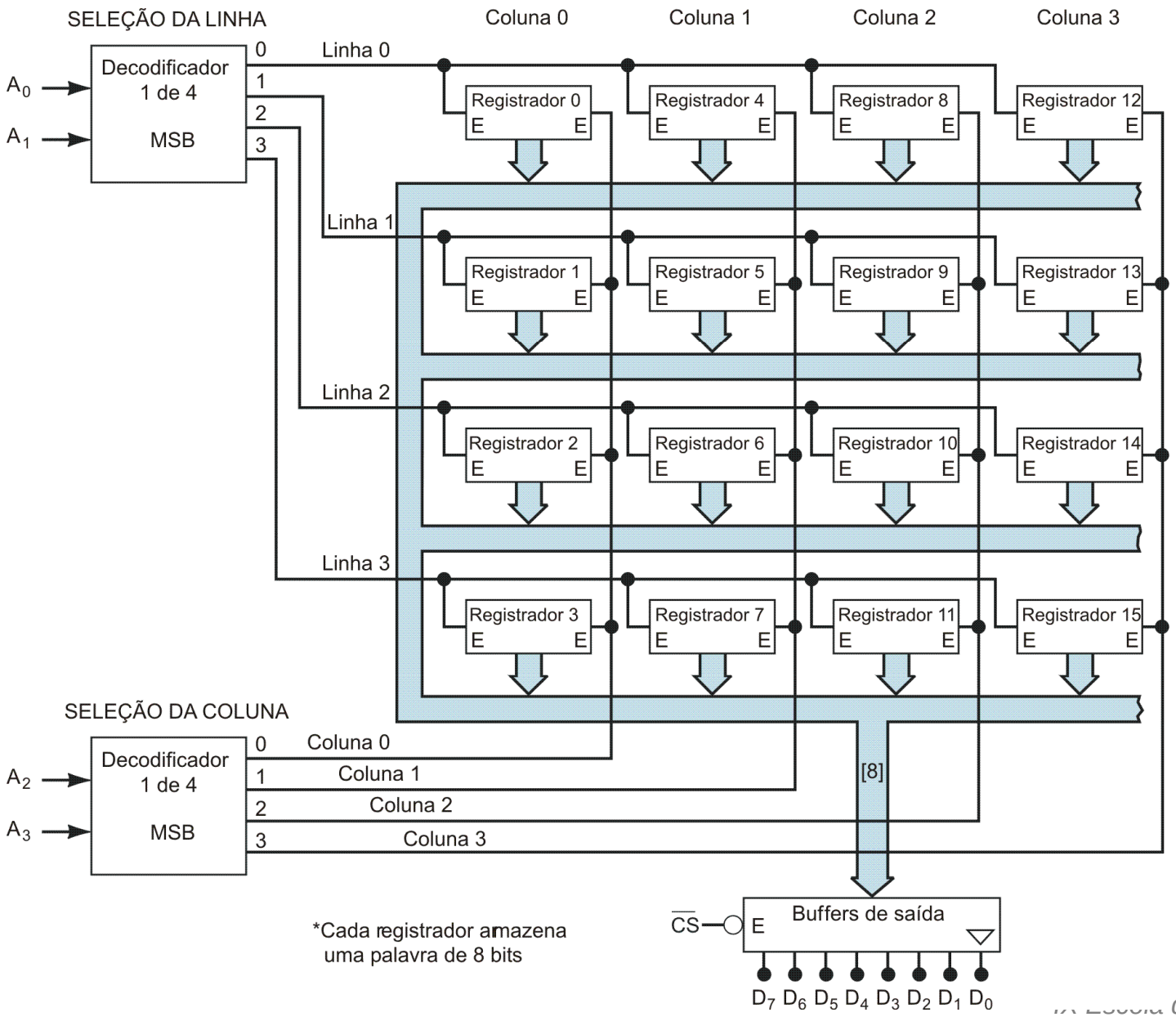
	Endereço				Dados							
Palavra	A_3	A_2	A_1	A_0	D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
0	0	0	0	0	1	1	0	1	1	1	1	0
1	0	0	0	1	0	0	1	1	1	0	1	0
2	0	0	1	0	1	0	0	0	0	1	0	1
3	0	0	1	1	1	0	1	0	1	1	1	1
4	0	1	0	0	0	0	0	1	1	0	0	1
5	0	1	0	1	0	1	1	1	1	0	1	1
6	0	1	1	0	0	0	0	0	0	0	0	0
7	0	1	1	1	1	1	1	0	1	1	0	1
8	1	0	0	0	0	0	1	1	1	1	0	0
9	1	0	0	1	1	1	1	1	1	1	1	1
10	1	0	1	0	1	0	1	1	1	0	0	0
11	1	0	1	1	1	1	0	0	0	1	1	1
12	1	1	0	0	0	0	1	0	0	1	1	1
13	1	1	0	1	0	1	1	0	1	0	1	0
14	1	1	1	0	1	1	0	1	0	0	1	0
15	1	1	1	1	0	1	0	1	1	0	1	1

(b)

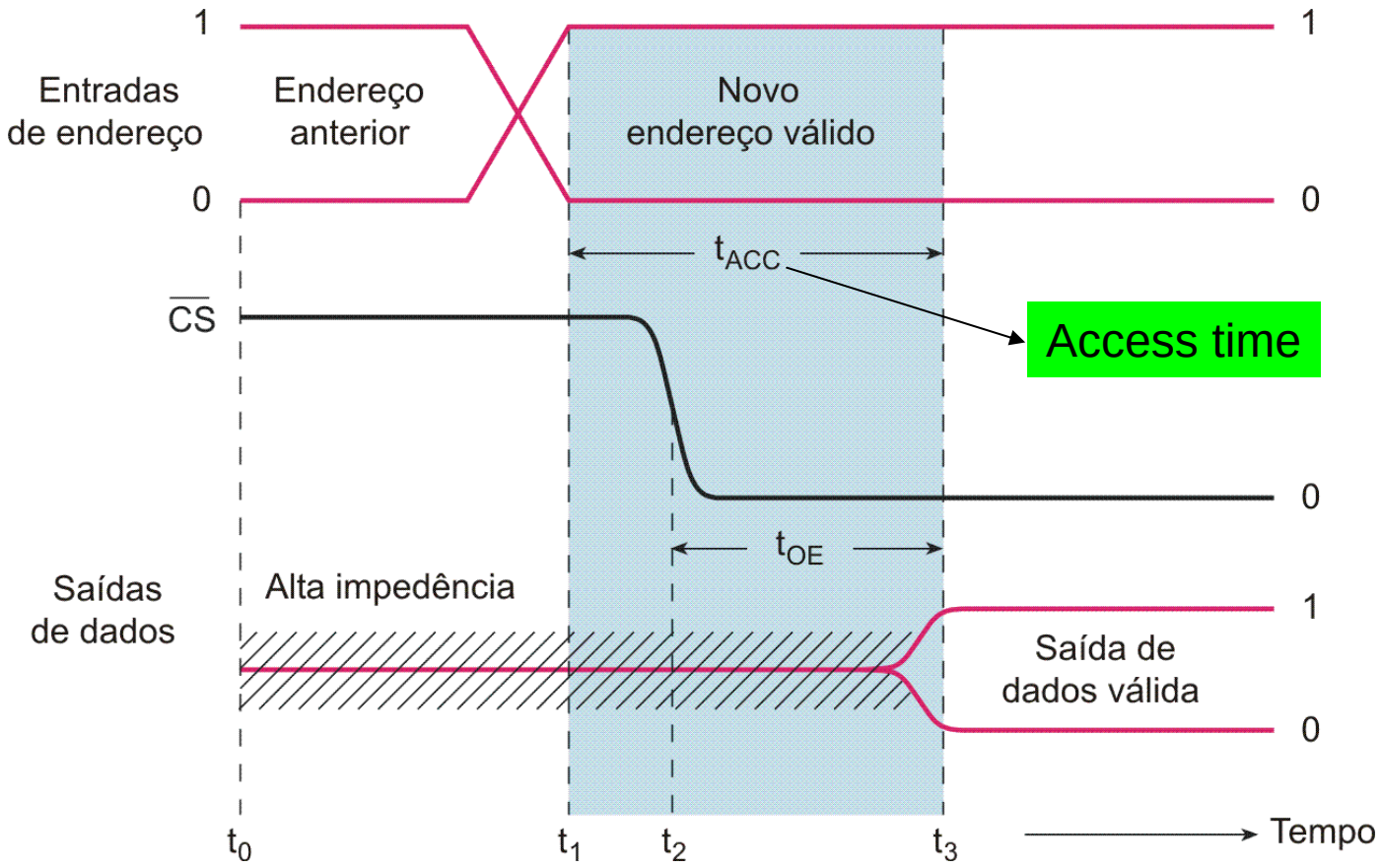
	Endereço	Dados
Palavra	A_3 A_2 A_1 A_0	D_7 D_6 D_5 D_4 D_3 D_2 D_1 D_0
0	0	DE
1	1	3A
2	2	85
3	3	AF
4	4	19
5	5	7B
6	6	00
7	7	ED
8	8	3C
9	9	FF
10	A	B8
11	B	C7
12	C	27
13	D	6A
14	E	D2
15	F	5B

(c)

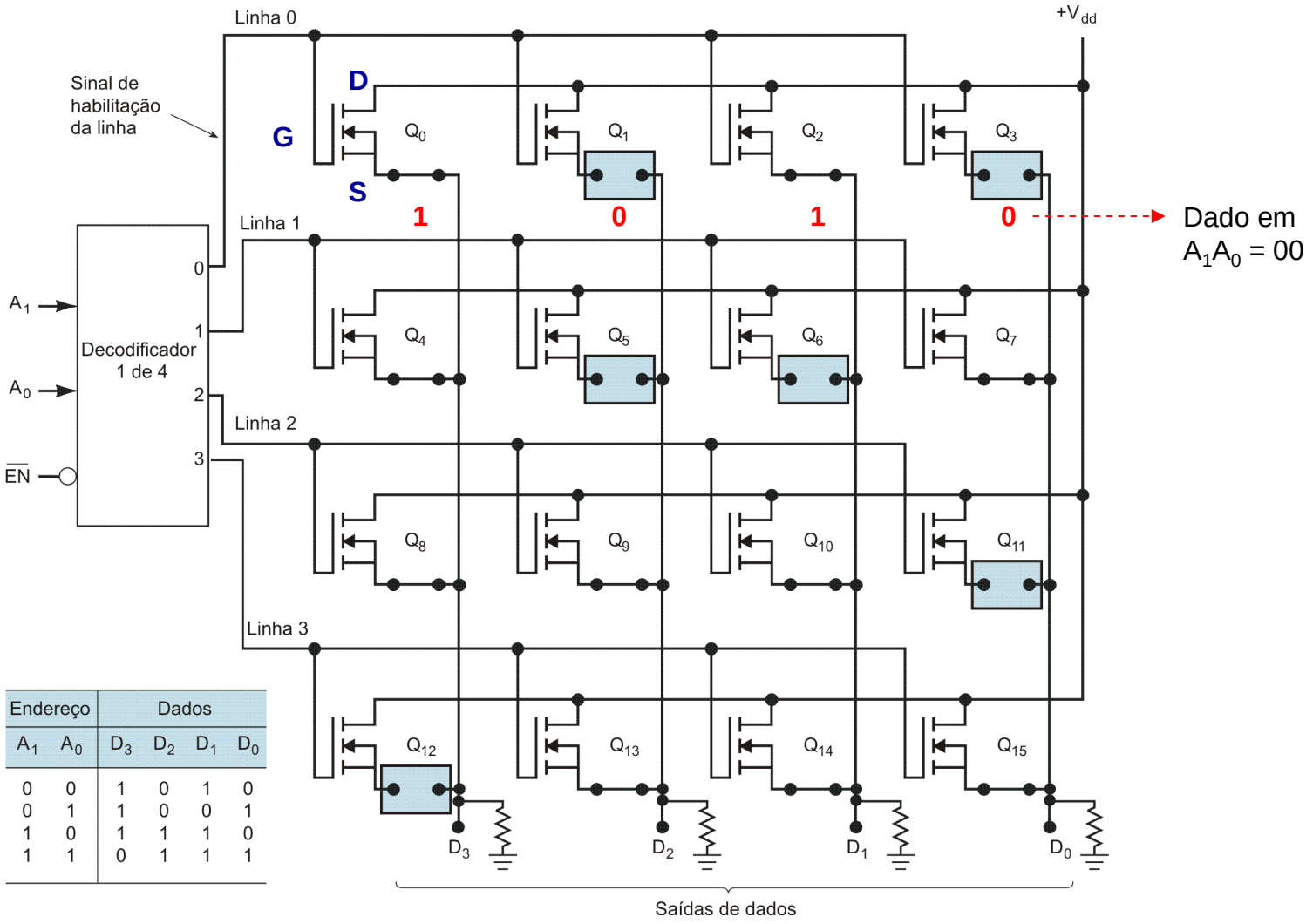
Arquitetura de uma ROM 16 × 8



Temporização típica para a operação de leitura de uma ROM

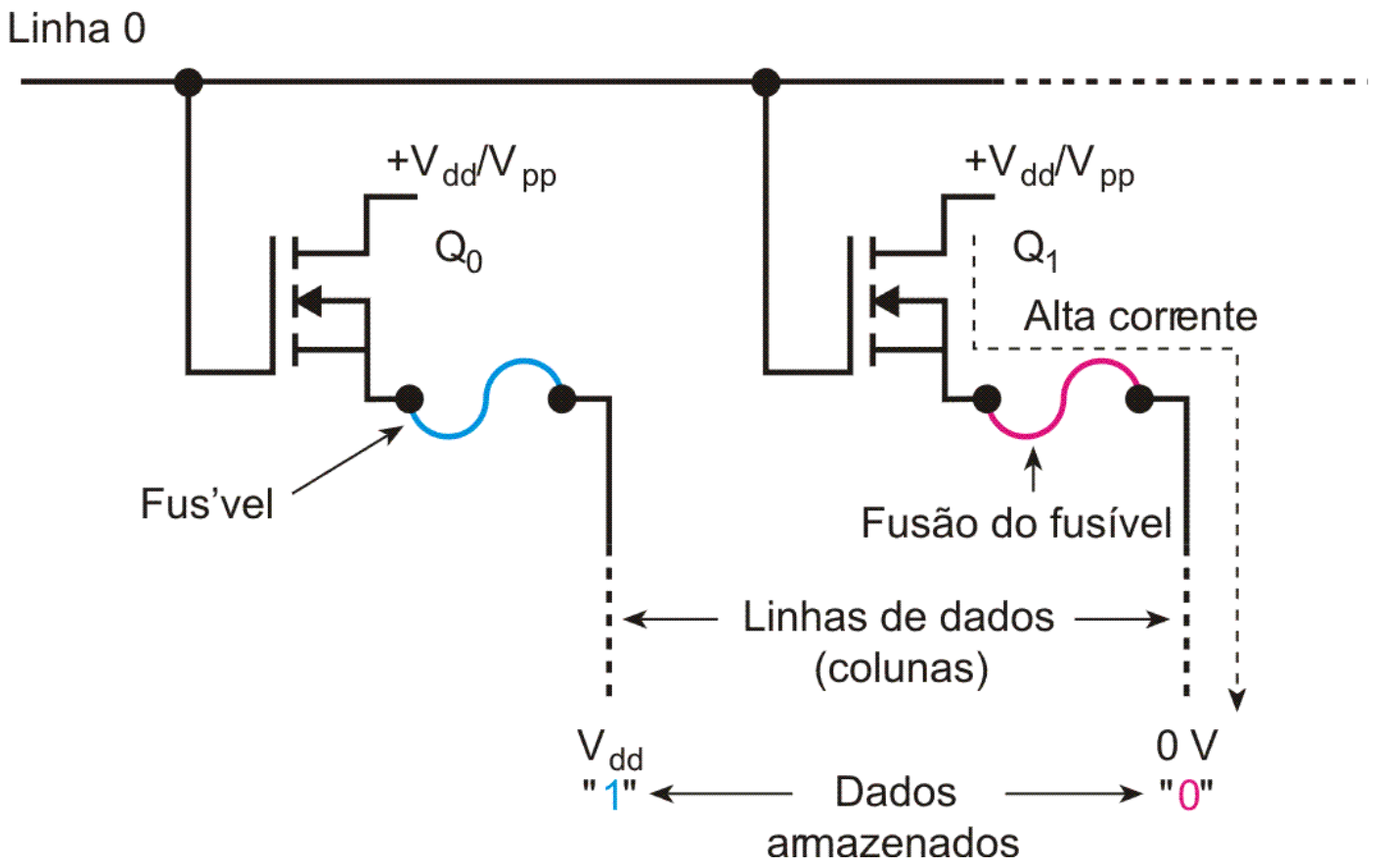


Estrutura de uma ROM, onde se usa um MOSFET para cada célula memória. Uma conexão de fonte aberta armazena ‘0’; uma conexão fechada armazena ‘1’.

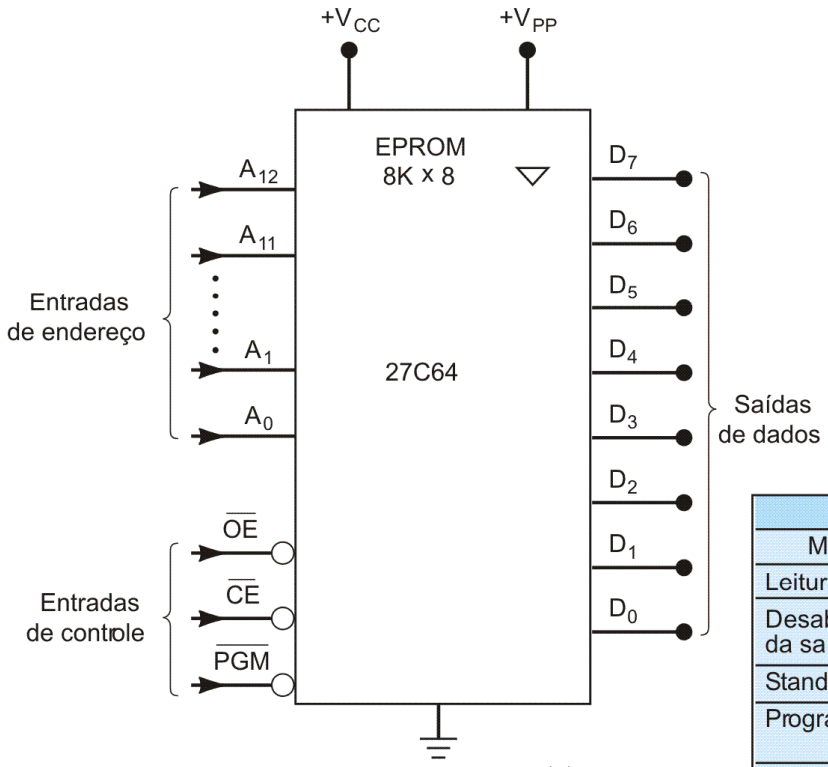


Endereço		Dados			
A_1	A_0	D_3	D_2	D_1	D_0
0	0	1	0	1	0
0	1	1	0	0	1
1	0	1	1	1	0
1	1	0	1	1	1

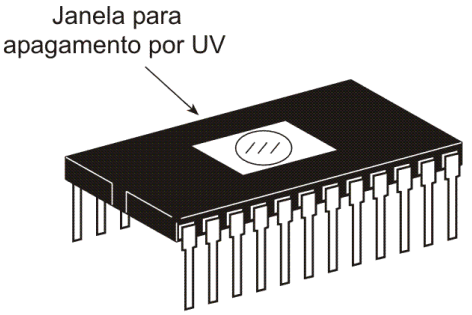
PROMS usam fusíveis que podem ser seletivamente “queimados” (abertos) pelo usuário para programar um nível lógico 0 na célula.



(a) Símbolo lógico para a EPROM 27C64
(b) Encapsulamento típico mostrando a janela para entrada de luz ultravioleta
(c) Modos de operação da 27C64.



(a)

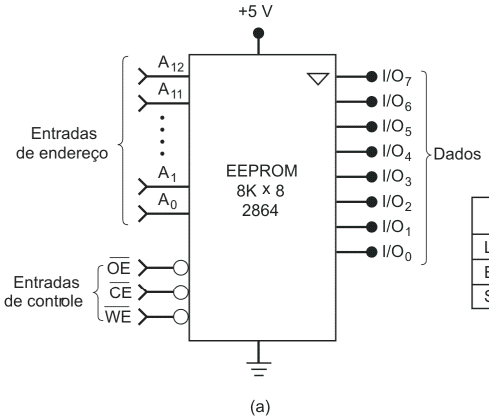


(b)

	Entradas				Saídas
Modo	$\overline{CE}$	$\overline{OE}$	$\overline{PGM}$	VPP	$D_7 - D_0$
Leitura	0	0	1	0–5V	Saída de dados
Desabilitação da saída	0	1	1	0–5V	Alta impedância
Standby	1	X	X	X	Alta impedância
Programação	0	1	0	12,75 V	Entradade dados
Verificação da Programação	0	0	1	12,75 V	Saída de dados

(c)

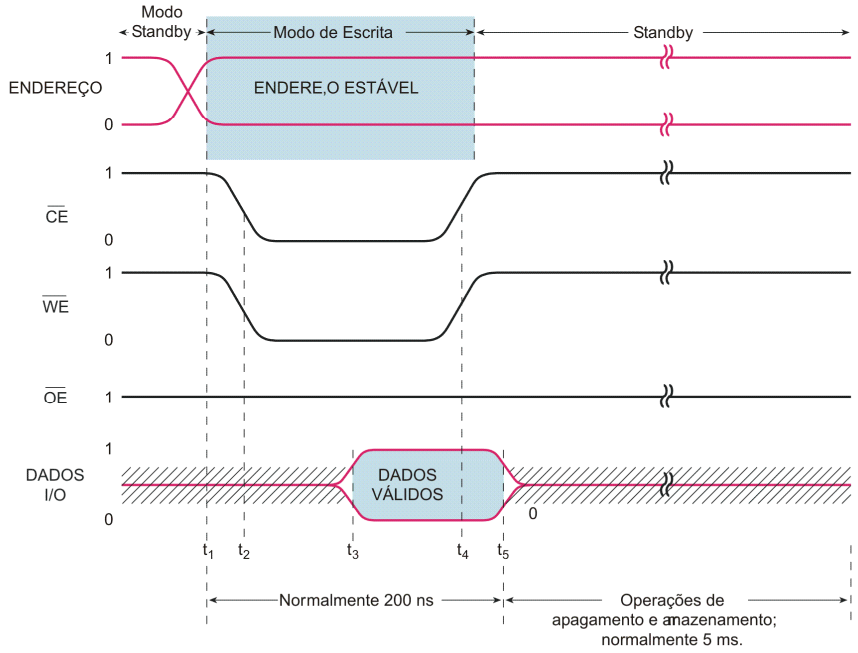
(a) Símbolo lógico para a EEPROM 2864
(b) Modos de operação
(c) Temporização para a operação de escrita



(a)

	Entradas			
Modo	$\overline{CE}$	$\overline{OE}$	$\overline{WE}$	Pinos de I/O
LEITURA	BAIXO	BAIXO	ALTO	Saída de dados
ESCRITA	BAIXO	ALTO	BAIXO	Entrada de dados
STANDBY	ALTO	X	X	Alta impedância

(b)



(c)

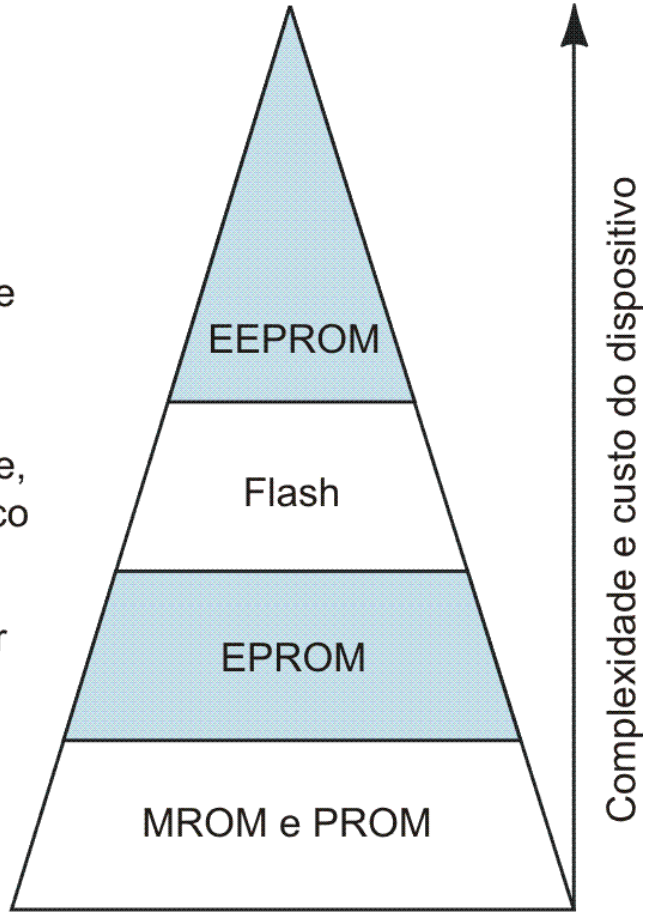
A complexidade e o custo das memórias semicondutoras não-voláteis aumentam à medida que a flexibilidade no apagamento e na programação aumenta.

Pode ser apagada eletricamente no circuito byte a byte

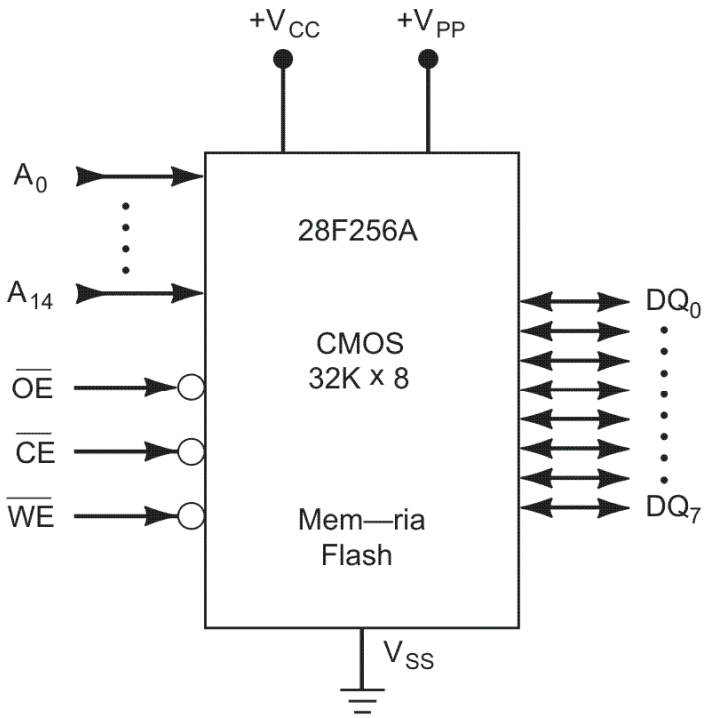
Pode ser apagada eletricamente, no circuito, por setor ou em bloco (todas as células)

Pode ser apagada em bloco por luz UV sendo apagada e reprogramada fora do circuito

Não pode ser apagada e reprogramada



(a) Símbolo lógico para o chip de memória flash 28F256A
(b) Entradas de controle (*CE*, *OE* e *WE*).



(a)

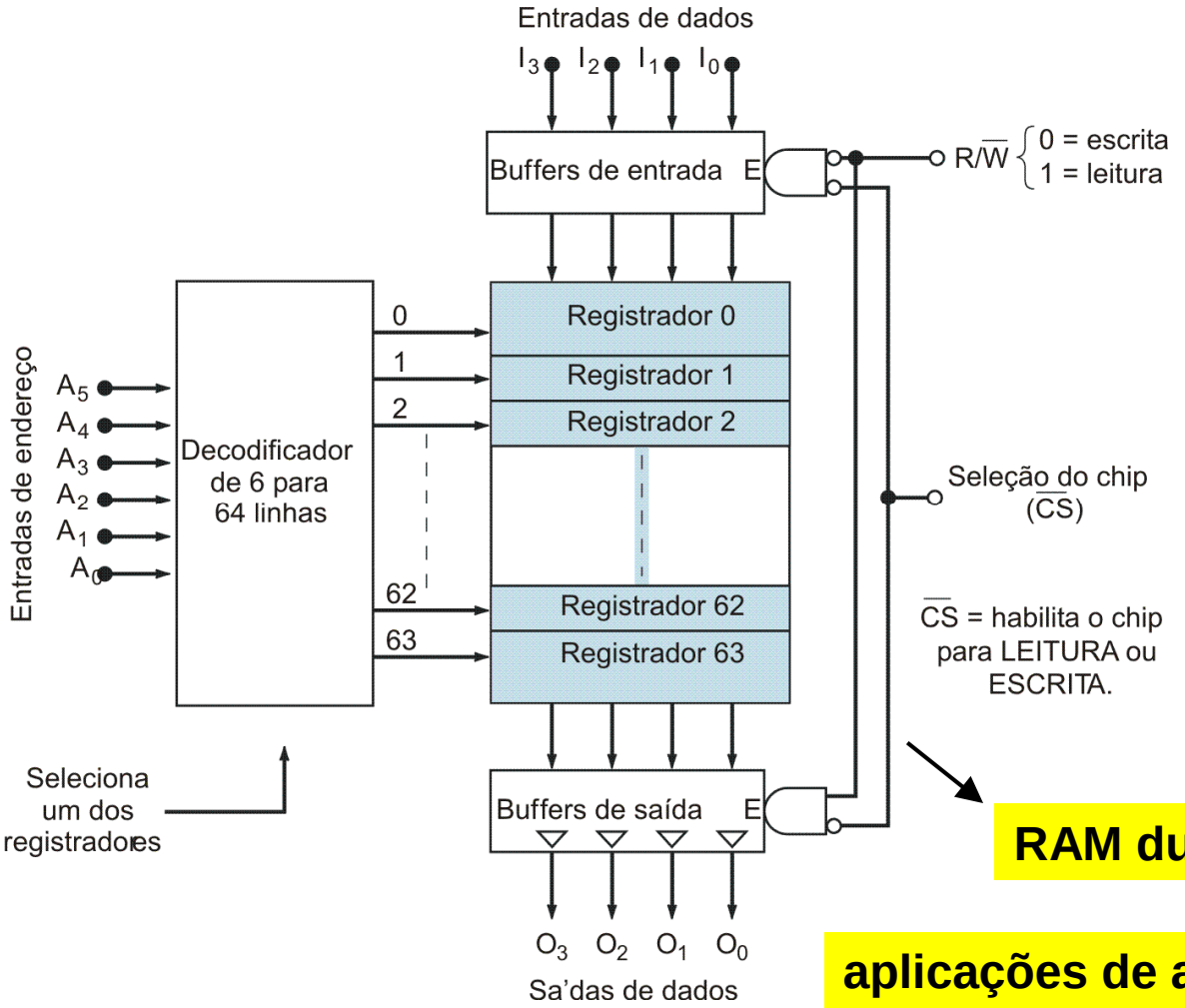
	Entradas			
Modo	$\overline{CE}$	$\overline{OE}$	$\overline{WE}$	Pinos de dados
LEITURA	BAIXO	BAIXO	ALTO	Saída de dados
STANDBY	ALTO	X	X	Alta impedância
ESCRITA*	BAIXO	ALTO	BAIXO	Entrada de dados

*Nota: Se $V_{PP} \leq 6{,}65V$, uma operação de escrita não pode ser realizada

(b)

Memória flash em um *pen-drive* USB

Organização interna de uma RAM de 64 × 4



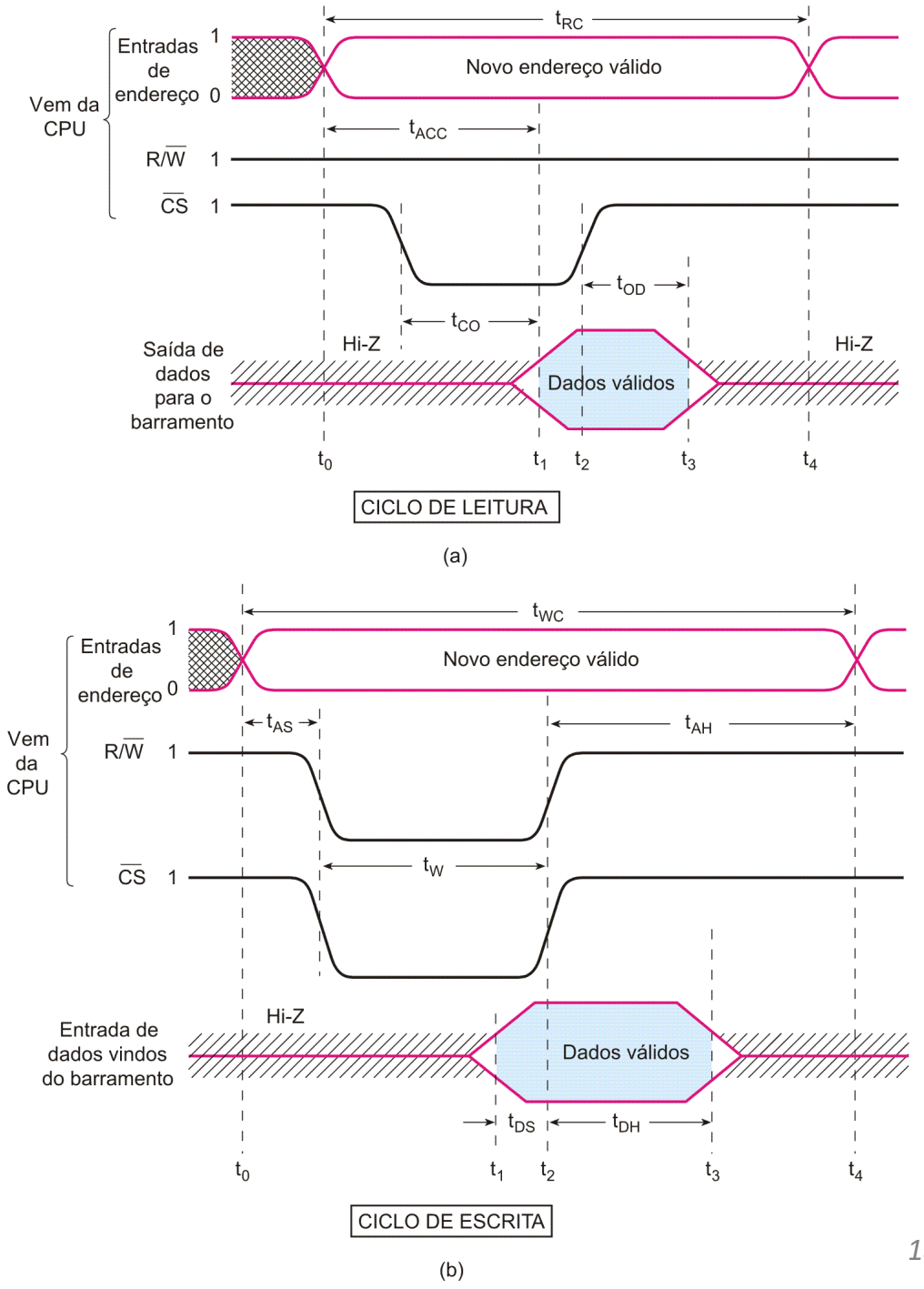
RAM dual-port

aplicações de alta velocidade

origem dos dados ≠ destino

Temporização típica de uma memória RAM

(a) ciclo de leitura (b) ciclo de escrita



RAM Estática × RAM Dinâmica

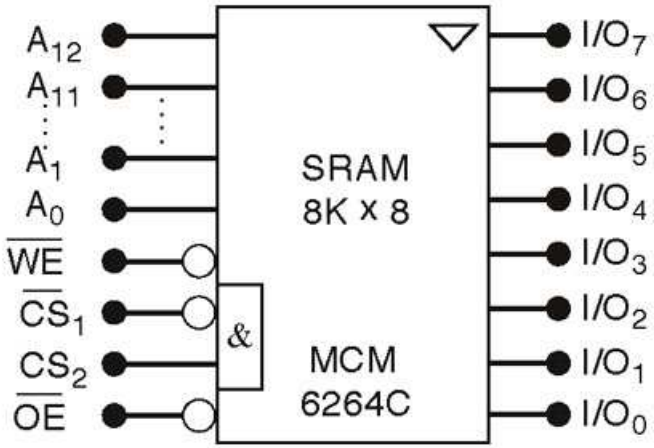
RAM Estática (*SRAM*)

- dados armazenados somente enquanto o CI estiver alimentado - **volátil**
- células de memórias formadas por **flip-flops**
- tecnologias de fabricação: bipolar, MOS ou BiCMOS
- média capacidade (< 4Mbit)
- alta velocidade (tempo de endereçamento ~ 10ns)

RAM Dinâmica (*DRAM*)

- dados armazenados somente enquanto o CI estiver alimentado - **volátil**
- células de memória utilizam **capacitores MOS** para armazenar carga
- necessitam sinal de *refresh* periódico devido à fuga de carga do capacitor
- maior capacidade
- menor consumo
- alta velocidade de acesso com tecnologias DDR, DDR2 e DDR3

Símbolo e tabela de modo de operação para a SRAM CMOS MCM6264C



Mode	Inputs				I/O pins
	$\overline{WE}$	$\overline{CS_1}$	CS_2	$\overline{OE}$	
READ	1	0	1	0	$DATA_{OUT}$
WRITE	0	0	1	X	$DATA_{IN}$
Output disable	1	X	X	1	High Z
Not selected (power down)	X	1	X	X	High Z
	X	X	0	X	

X = don't care

Exemplo de SRAM utilizada em projeto do CBPF

3.3V CMOS Static RAM
4 Meg (512K x 8-Bit)

IDT71V424S
IDT71V424L

Features

- **512K x 8 advanced high-speed CMOS Static RAM**
- **JEDEC Center Power / GND pinout for reduced noise**
- **Equal access and cycle times**
  - *Commercial and Industrial: 10/12/15ns*
- **Single 3.3V power supply**
- **One Chip Select plus one Output Enable pin**
- **Bidirectional data inputs and outputs directly TTL-compatible**
- **Low power consumption via chip deselect**
- **Available in 36-pin, 400 mil plastic SOJ package and 44-pin, 400 mil TSOP.**

Description

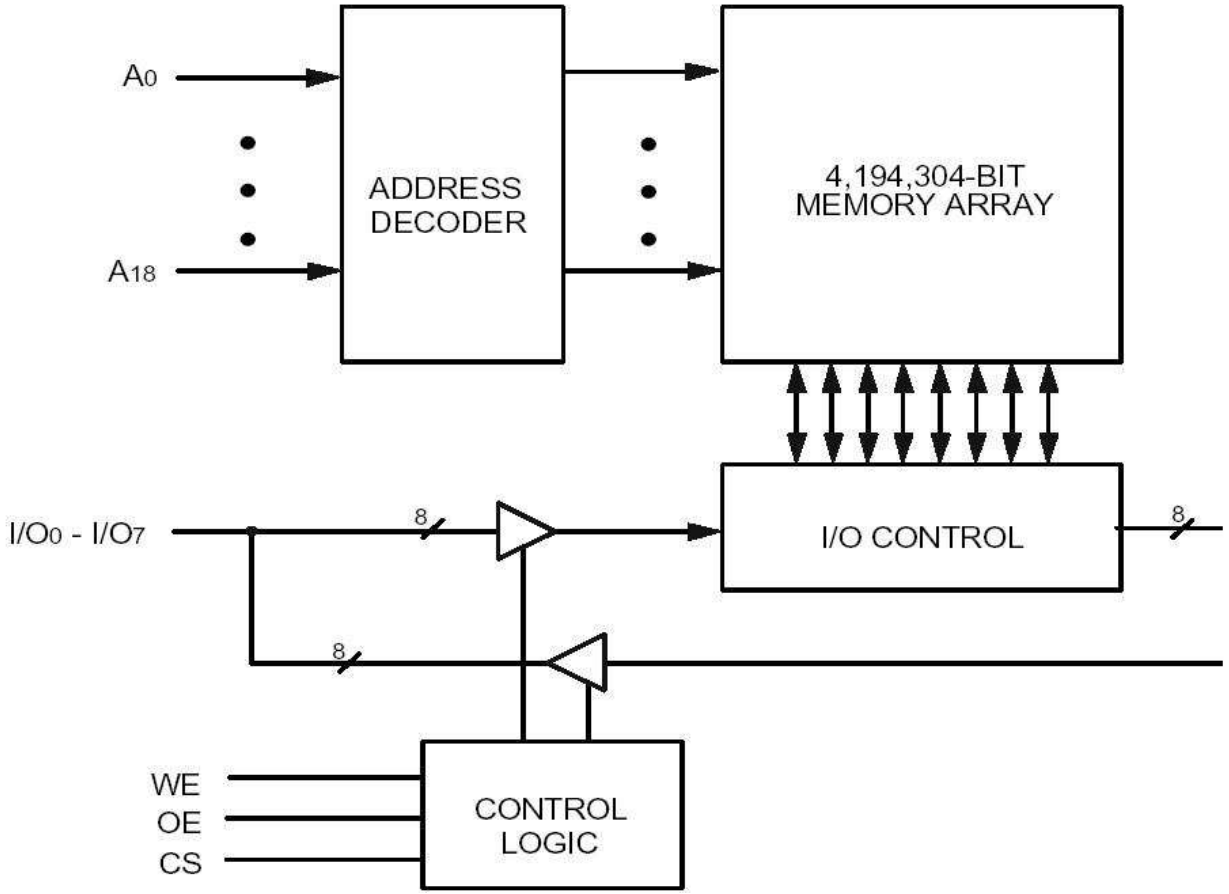
The IDT71V424 is a 4,194,304-bit high-speed Static RAM organized as 512K x 8. It is fabricated using IDT's high-perfomance, high-reliability CMOS technology. This state-of-the-art technology, combined with innovative circuit design techniques, provides a cost-effective solution for high-speed memory needs.

The IDT71V424 has an output enable pin which operates as fast as 5ns, with address access times as fast as 10ns. All bidirectional inputs and outputs of the IDT71V424 are TTL-compatible and operation is from a single 3.3V supply. Fully static asynchronous circuitry is used, requiring no clocks or refresh for operation.

The IDT71V424 is packaged in a 36-pin, 400 mil Plastic SOJ and 44-pin, 400 mil TSOP.

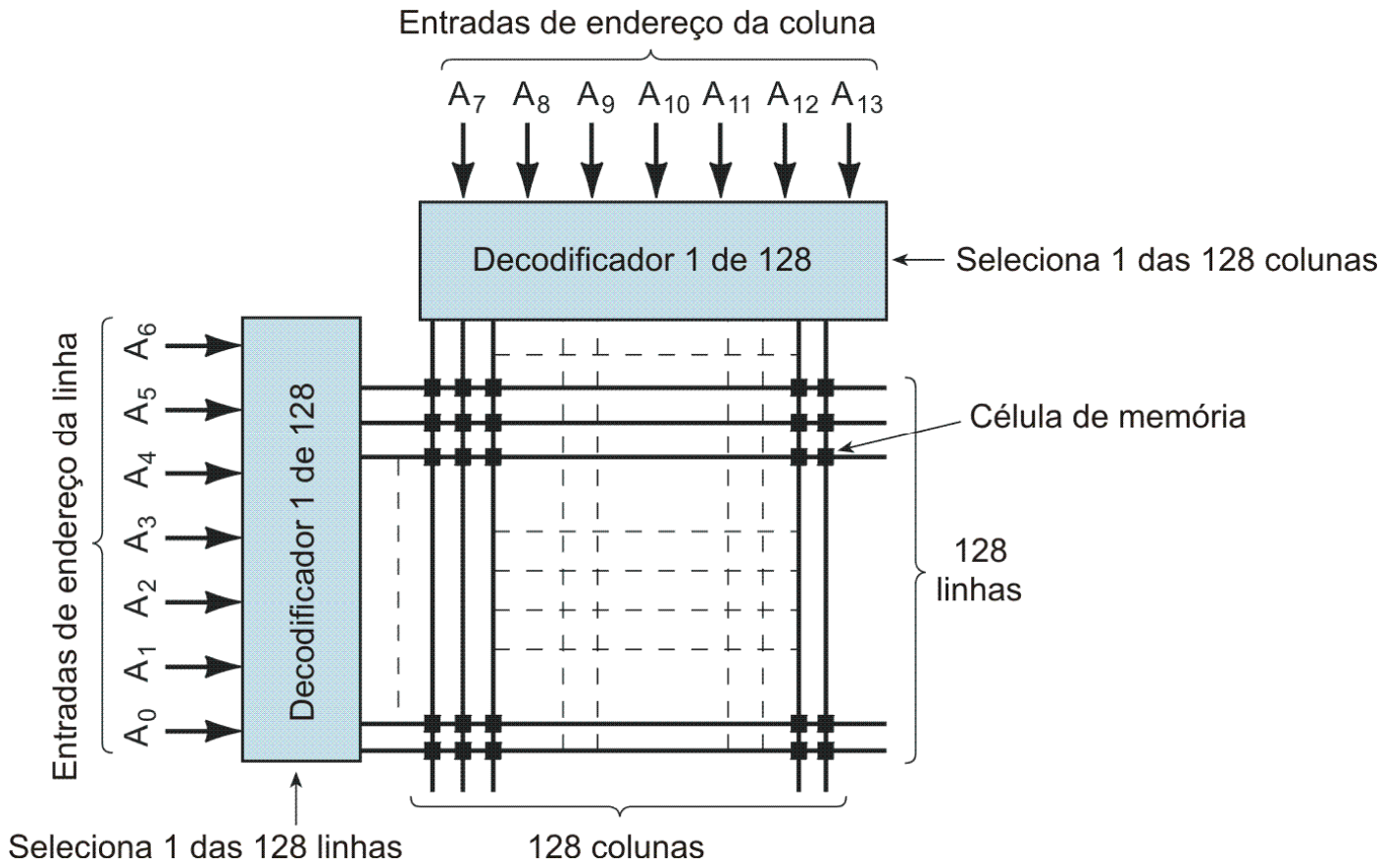
Preço Unitário = US$ 9,35
(modelo 10ns)

Exemplo de SRAM utilizada em projeto do CBPF

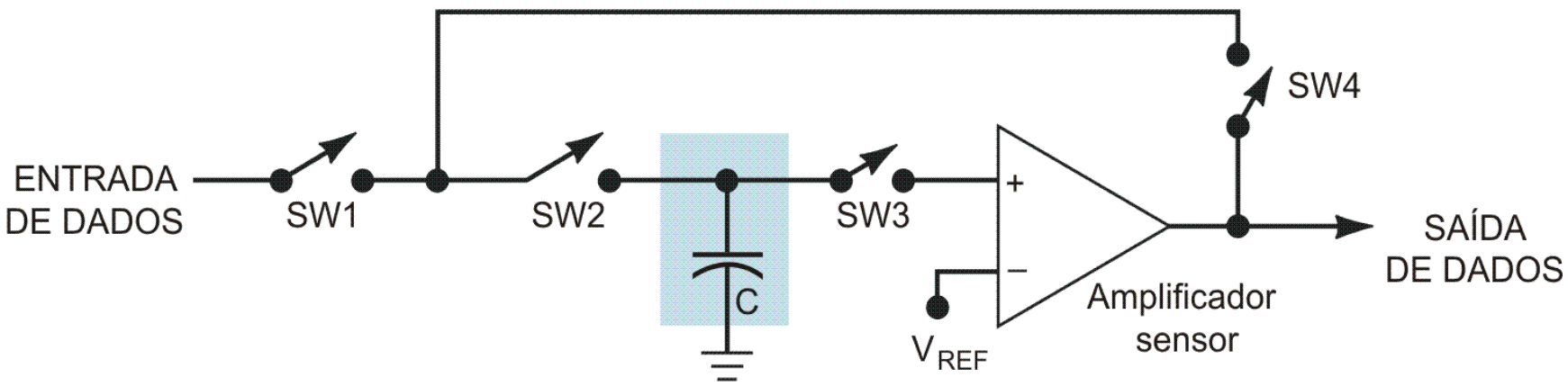


Arquitetura da memória IDT71V424

Arranjo das células em uma RAM dinâmica de 16K × 1



Célula de memória dinâmica

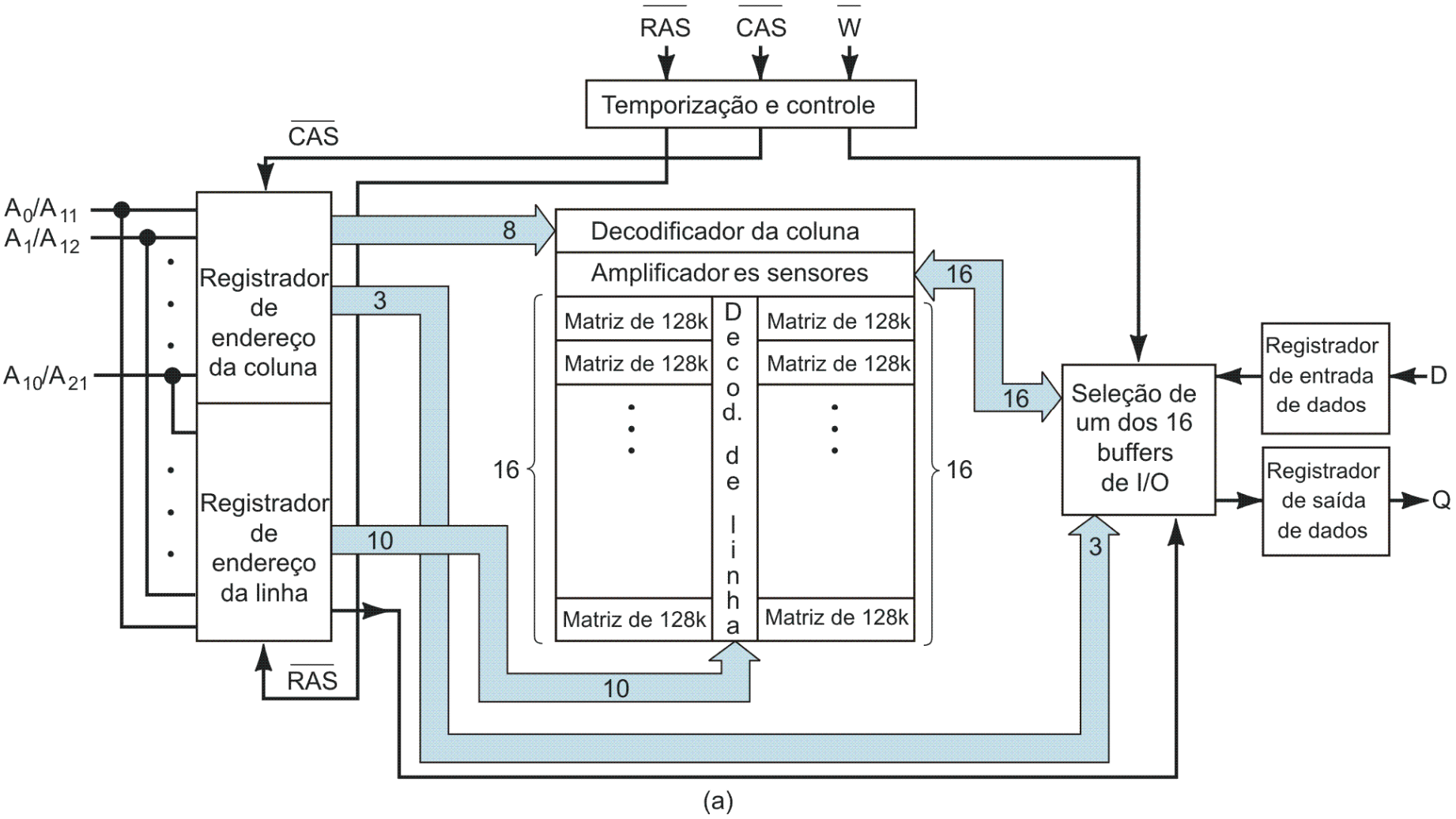


As chaves SW1 a SW4 são transístores MOSFET

Operação de escrita → **as chaves SW1 e SW2 são fechadas.**

Operação de leitura → **todas as chaves são fechadas, exceto SW1.**

Arquitetura simplificada da DRAM
TMS44100 de 4M × 1

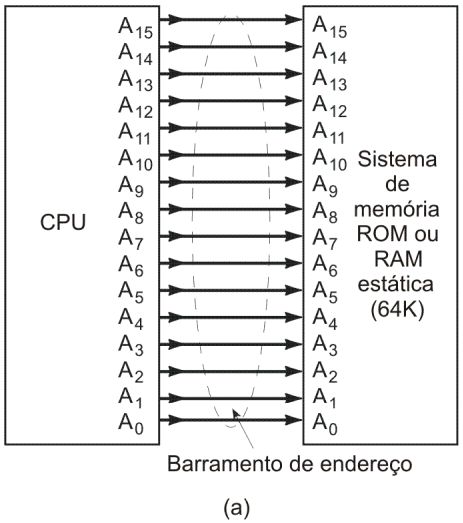


(a)

Multiplexação de endereço em memórias DRAM

ROM

O endereçamento é direto.

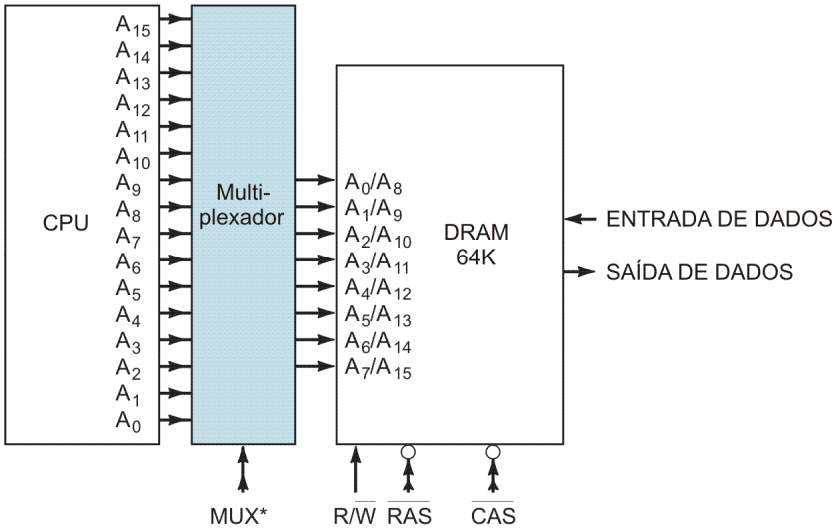


(a)

DRAM

Como as DRAMs apresentam alta capacidade, são necessários muitos bits de endereçamento, aumentando a dimensão dos CIs.

Para contornar esse problema, utiliza-se multiplexação de endereço.



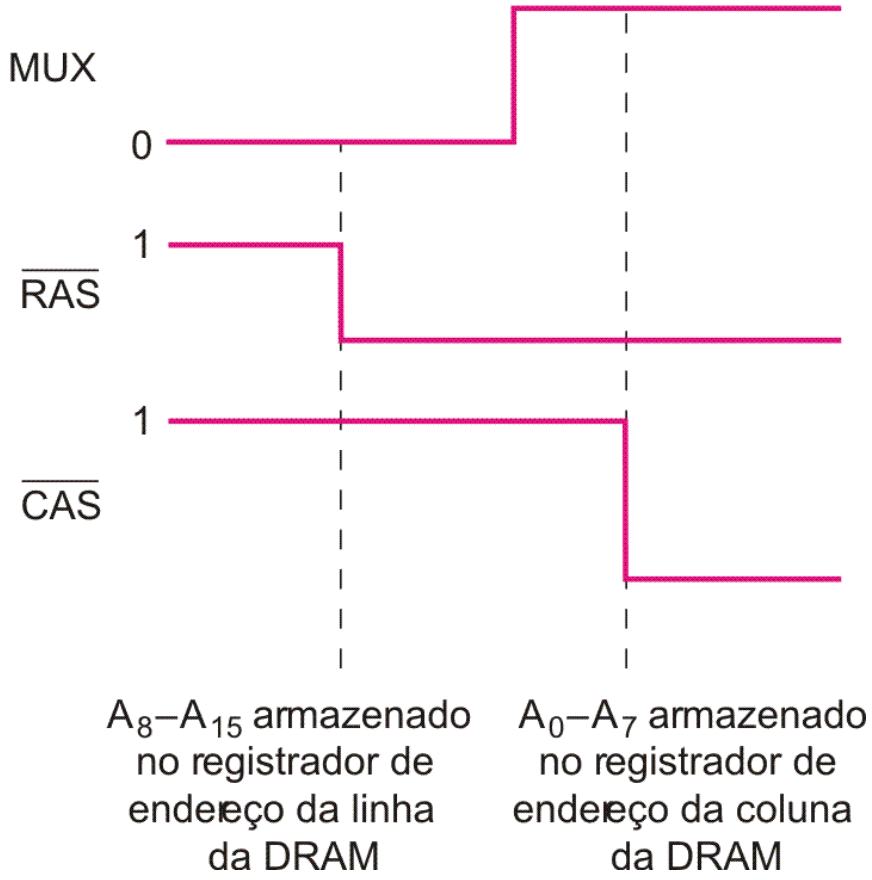
•MUX = 0 transmite o endereço. $A_8 - A_{15}$ da CPU para a DRAM. MUX = 1 transmite $A_0 - A_7$ da CPU para DRAM.

(b)

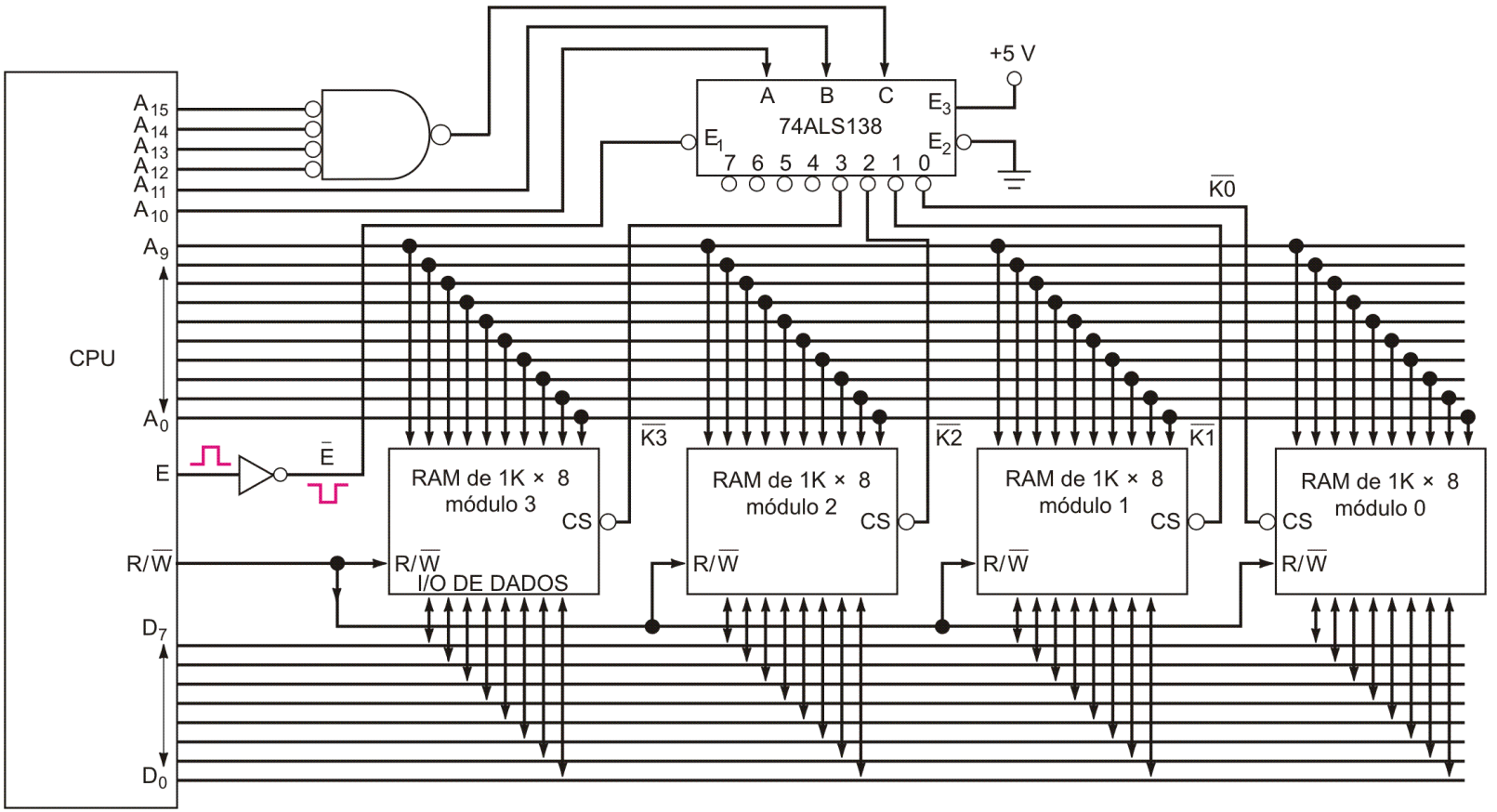
Temporização da multiplexação de endereço em memórias DRAM

Row Address Strobe

Column Address Strobe

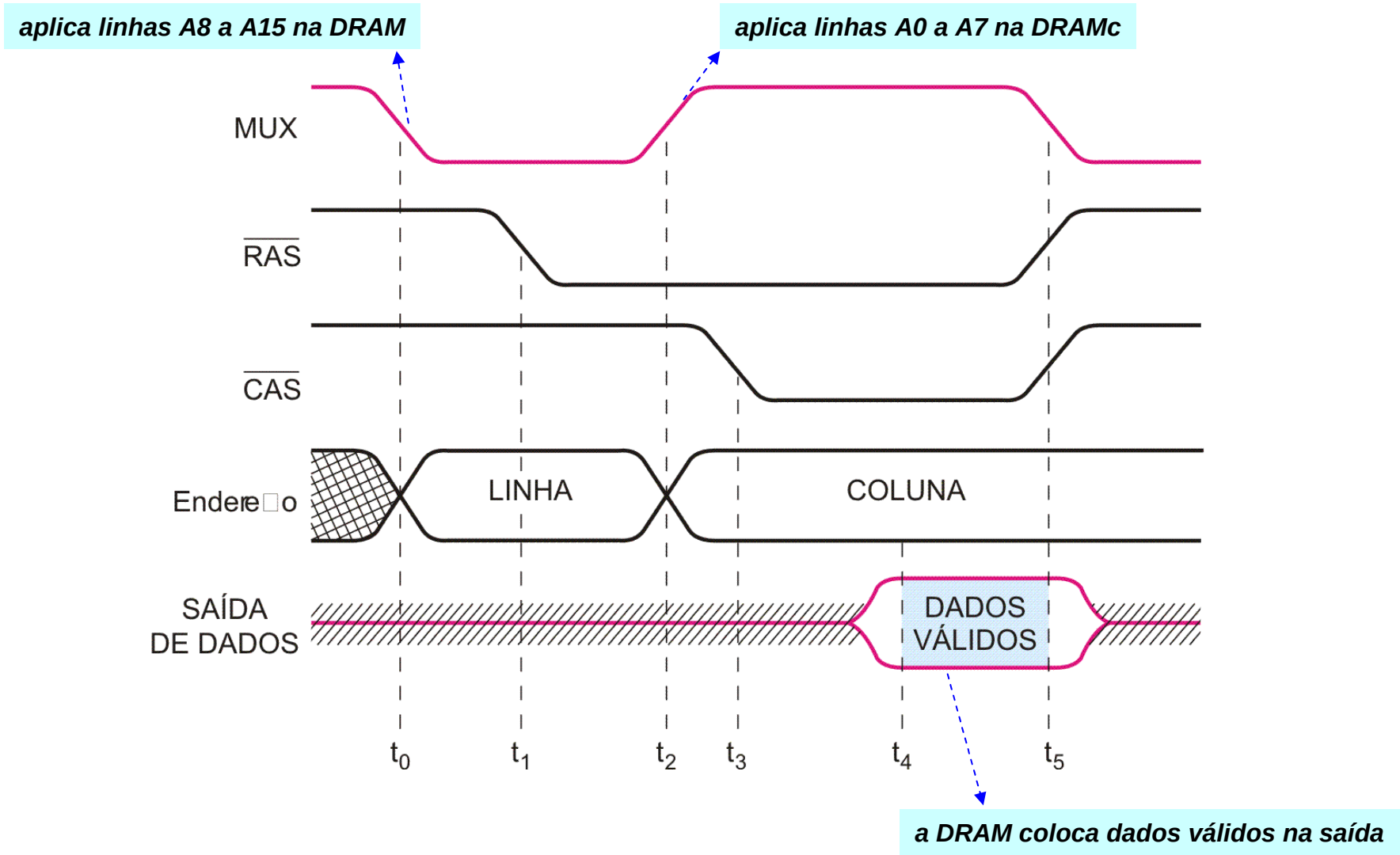


Memória RAM de 4K X 8 conectada em uma CPU

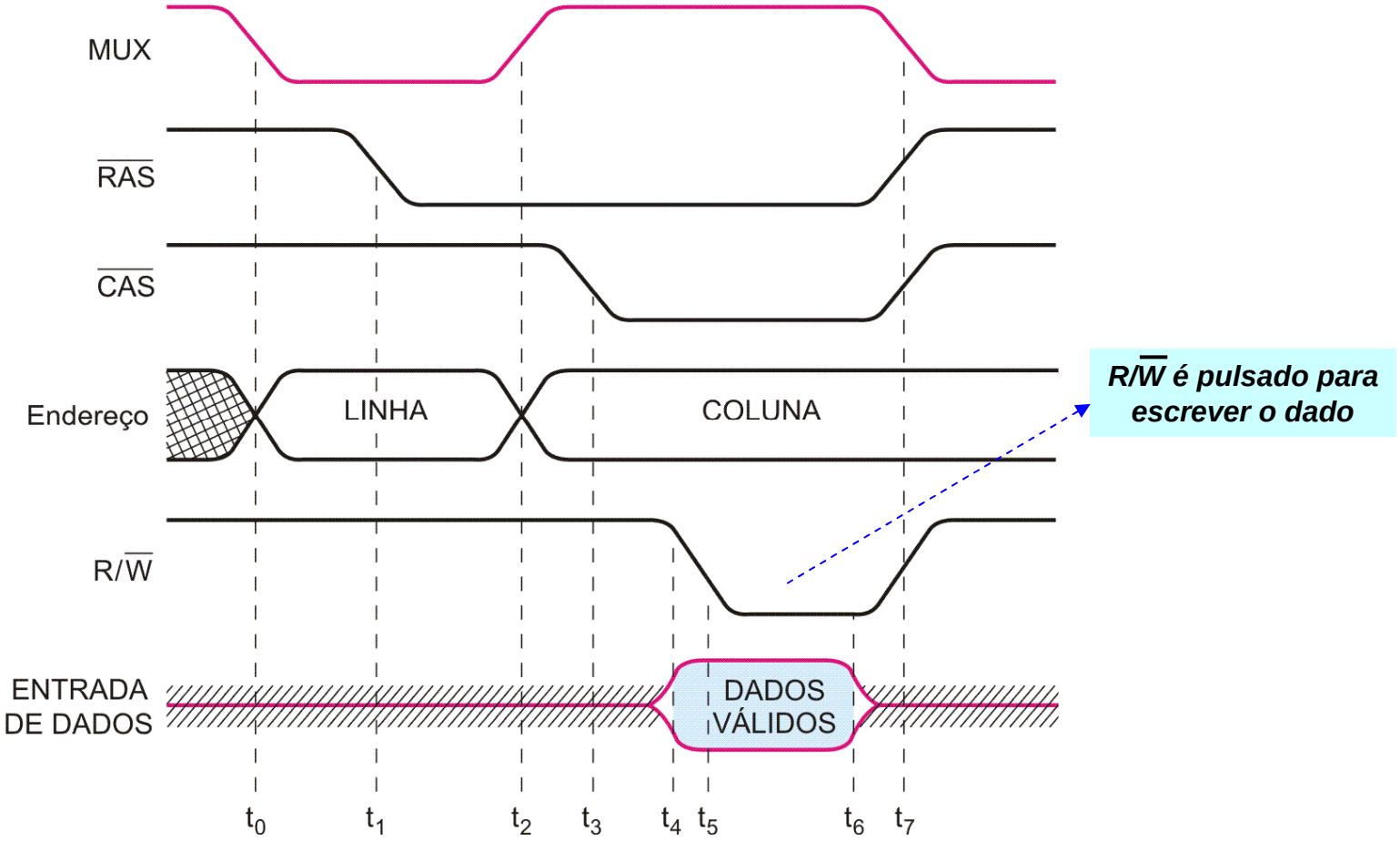


Sinais na operação de leitura em uma RAM dinâmica

(supondo a entrada R/$\overline{W}$ (não mostrada) em nível '1')



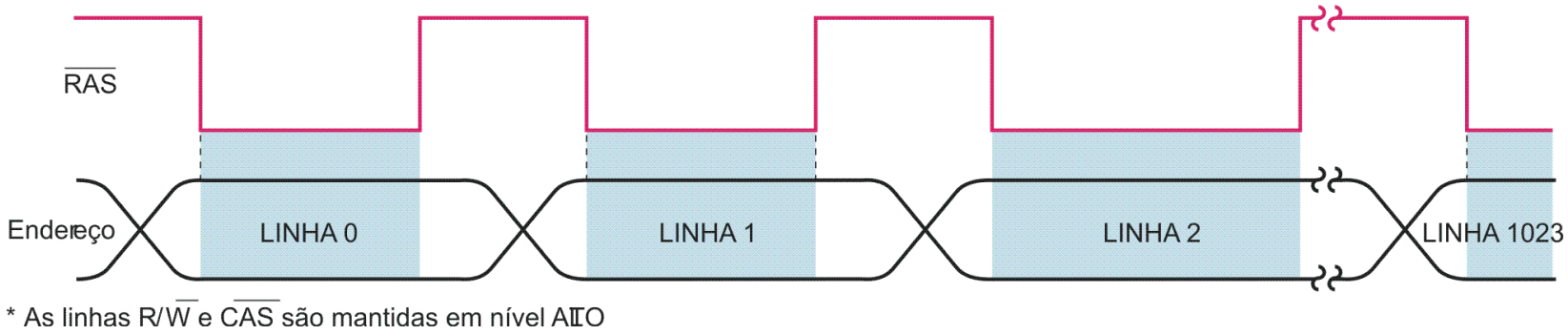
Sinais na operação de escrita em uma RAM dinâmica



Modo de refresh com o sinal RAS

• Sempre que uma operação de leitura for realizada em uma célula, todas as células daquela linha recebem um *refresh*.

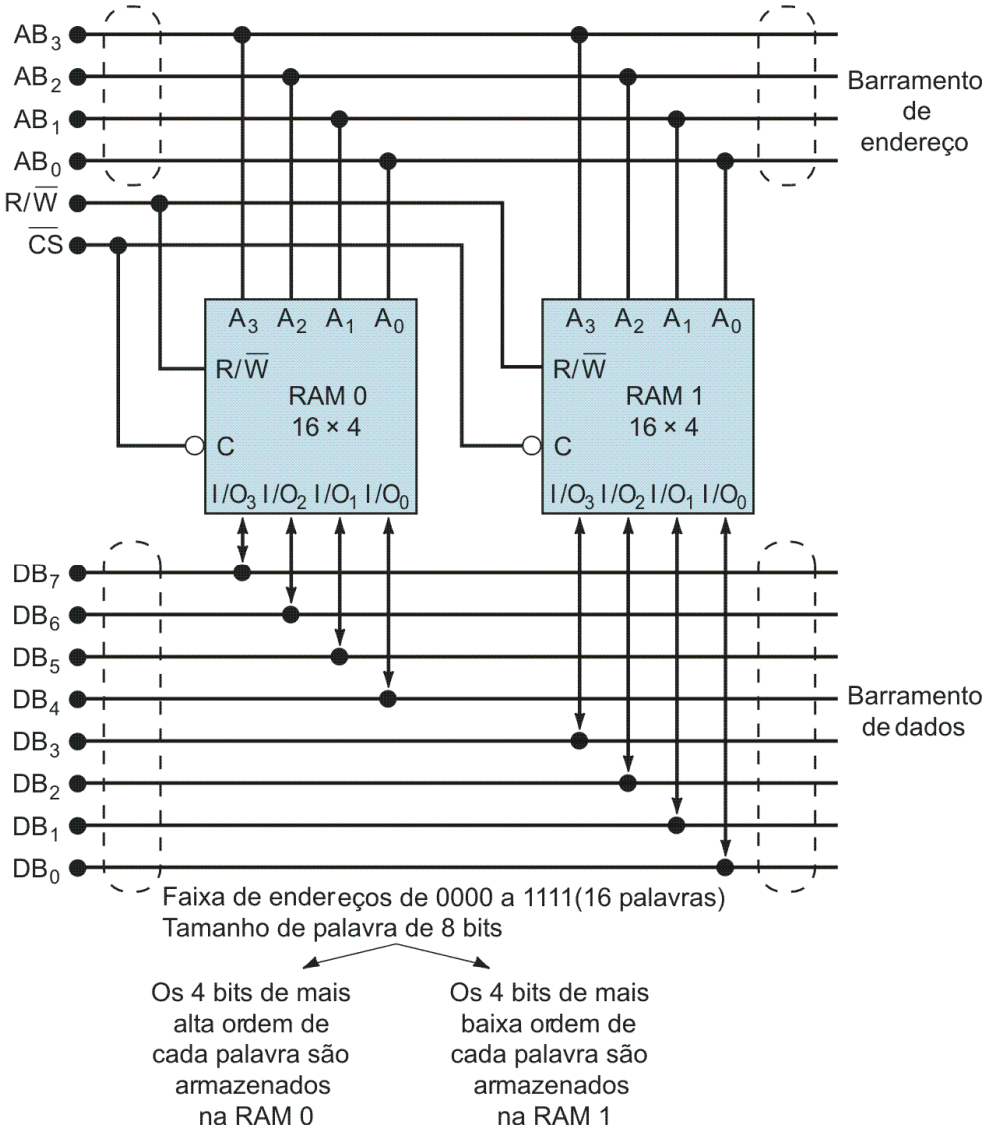
• *Refresh* com RAS é o modo mais comum, utilizando um contador de endereços.



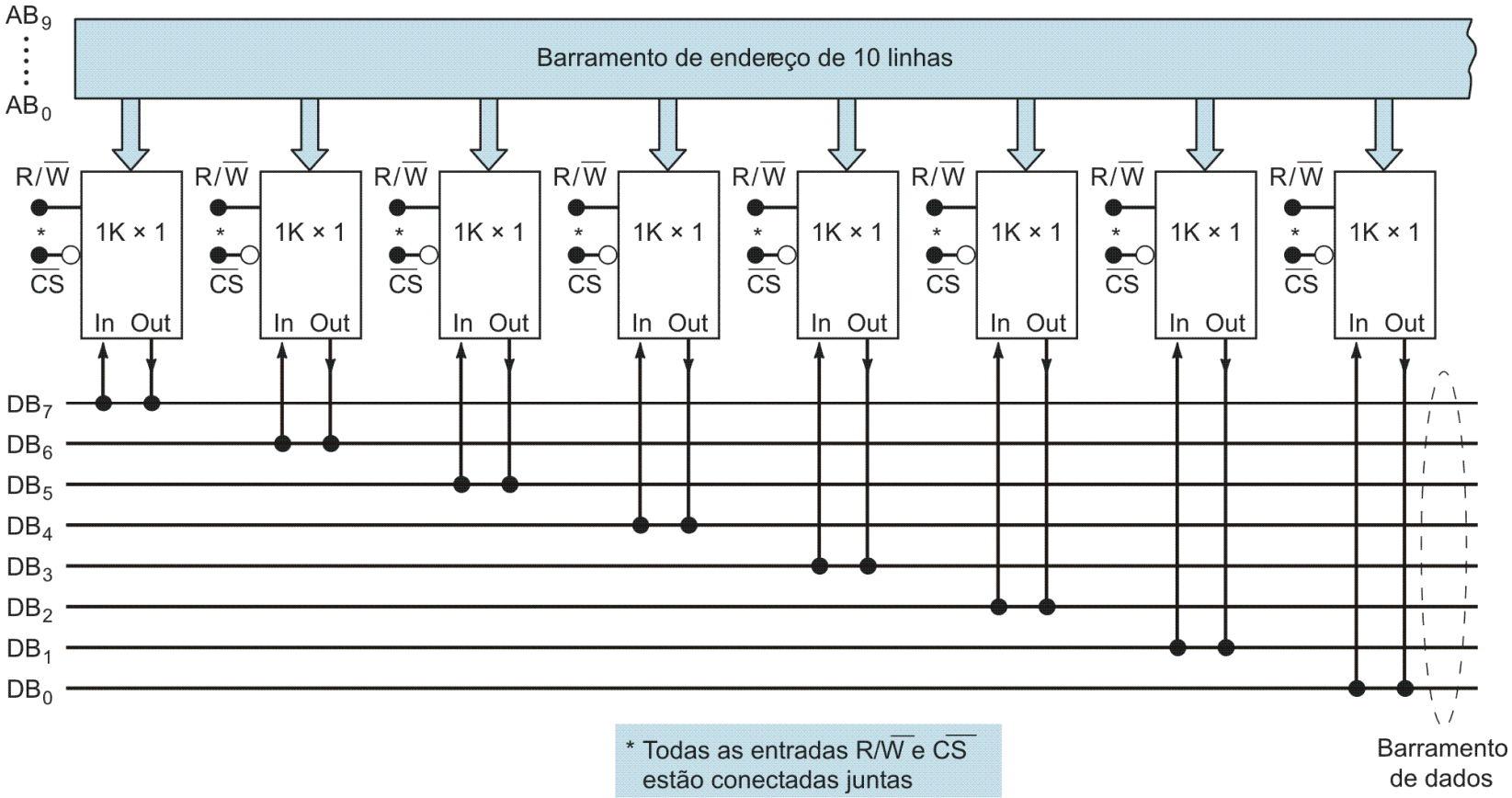
• Um Controlador de DRAM é utilizado frequentemente para controlar o processo de *refresh*, de forma que o endereçamento proveniente da CPU não sofra interferência do endereçamento do processo de *refresh*.

• A maioria das memórias DRAM atualmente já possui circuitos de *refresh* internos, o que elilmina a necessidade de fornecimento externo de endereços para *refresh*.

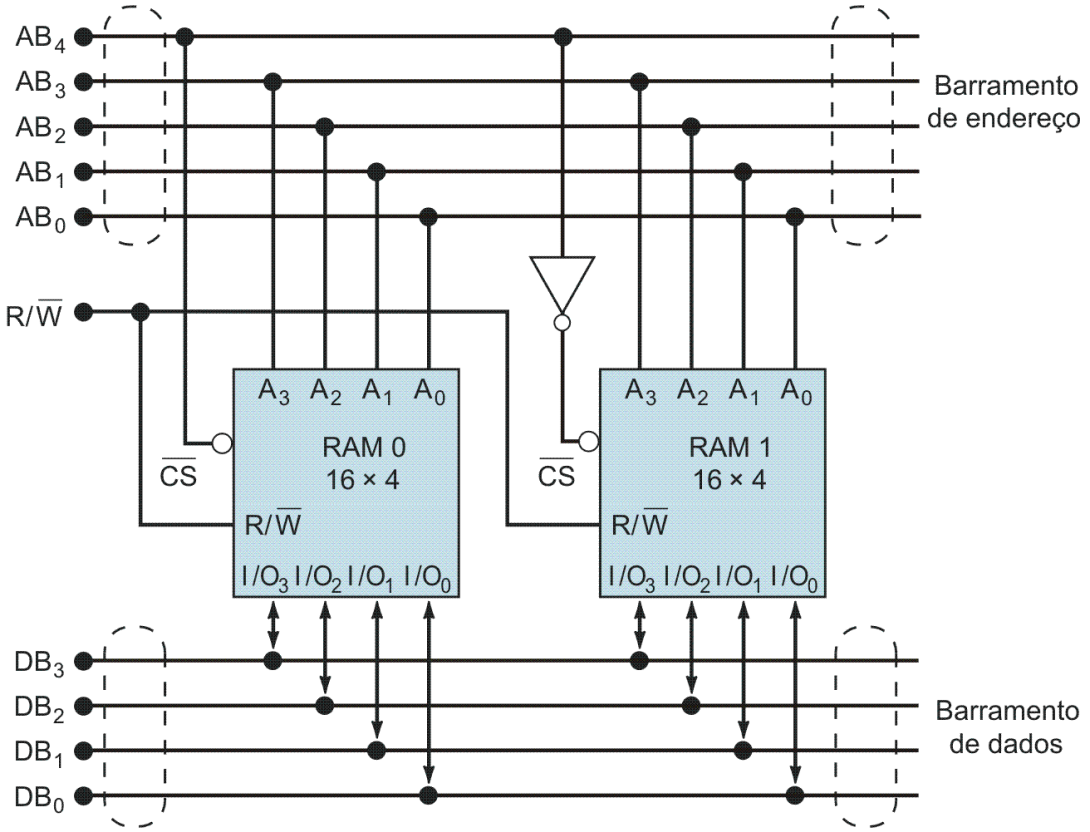
Combinando duas RAMs de 16 x 4 em um módulo de 16 x 8



Oito memórias de 1K x 1 organizadas como uma memória de 1K x 8

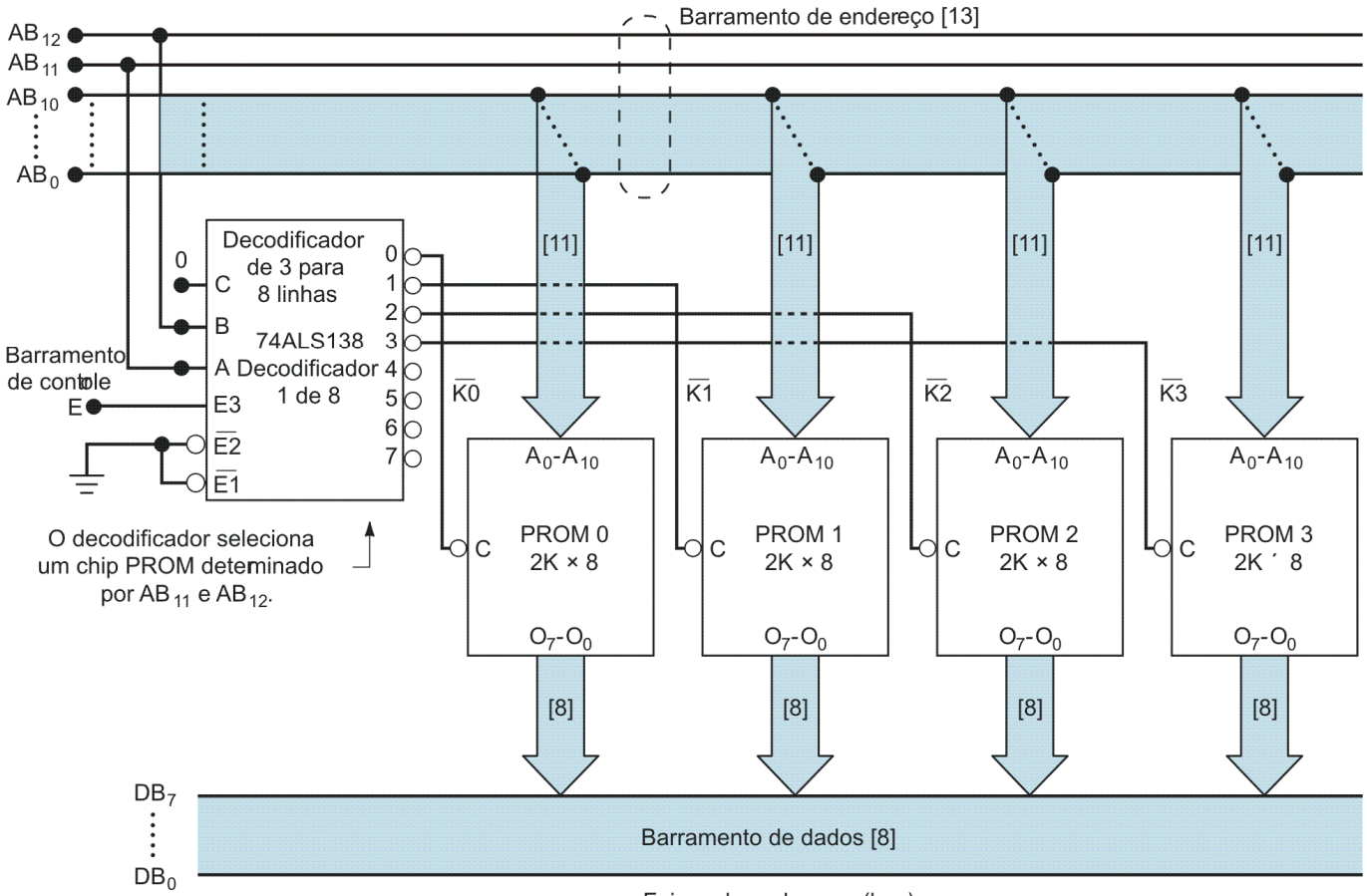


Duas memórias 16 x 4 formando uma memória de 32 x 4



Faixas de endereço:	00000 a 01111 – RAM 0 10000 a 11111 –RAM 1
Total	00000 a 11111 – (32 palavras)

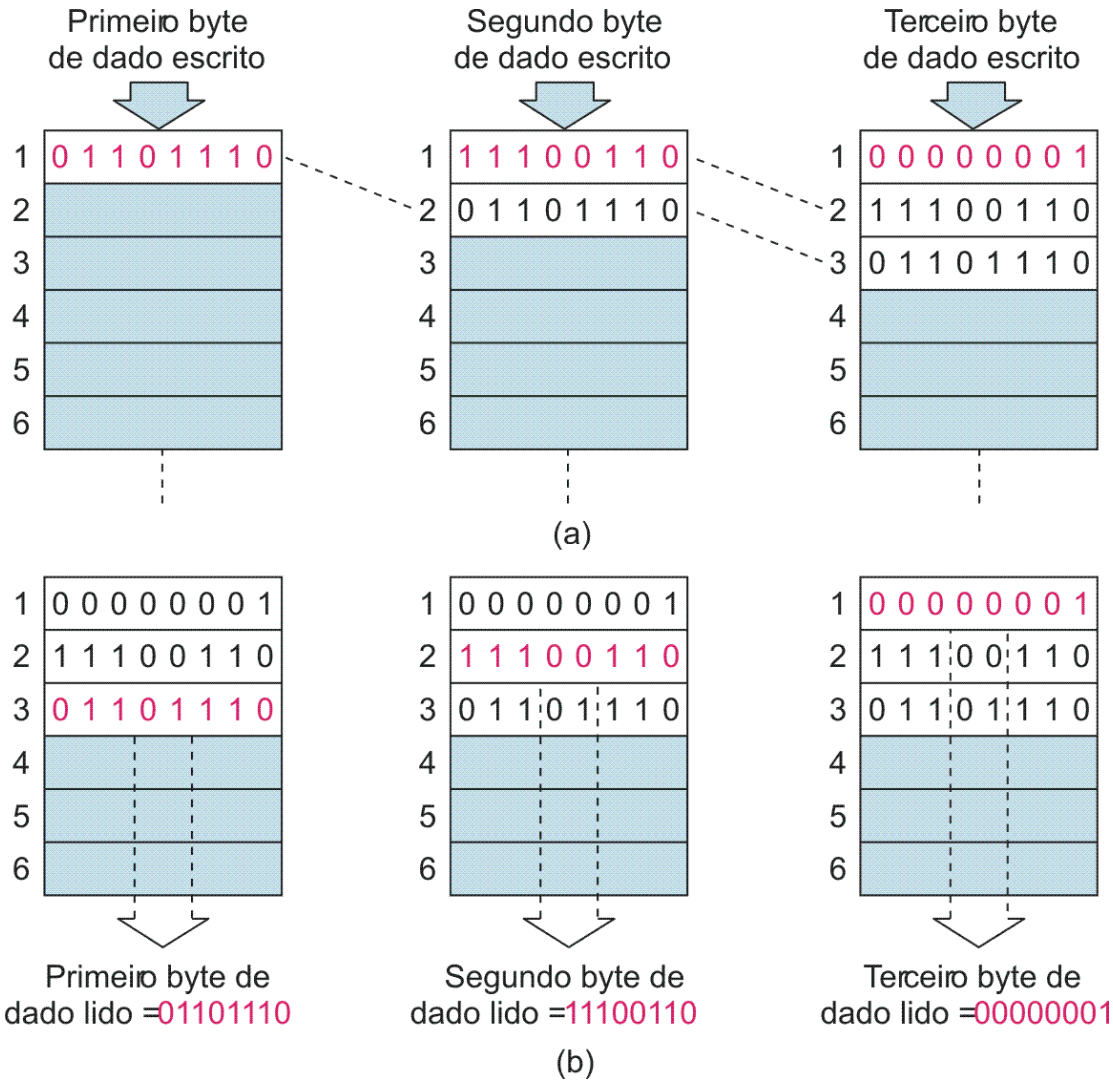
Quatro PROMs 2K x 8 organizadas para formar uma memória com capacidade de 8K x 8



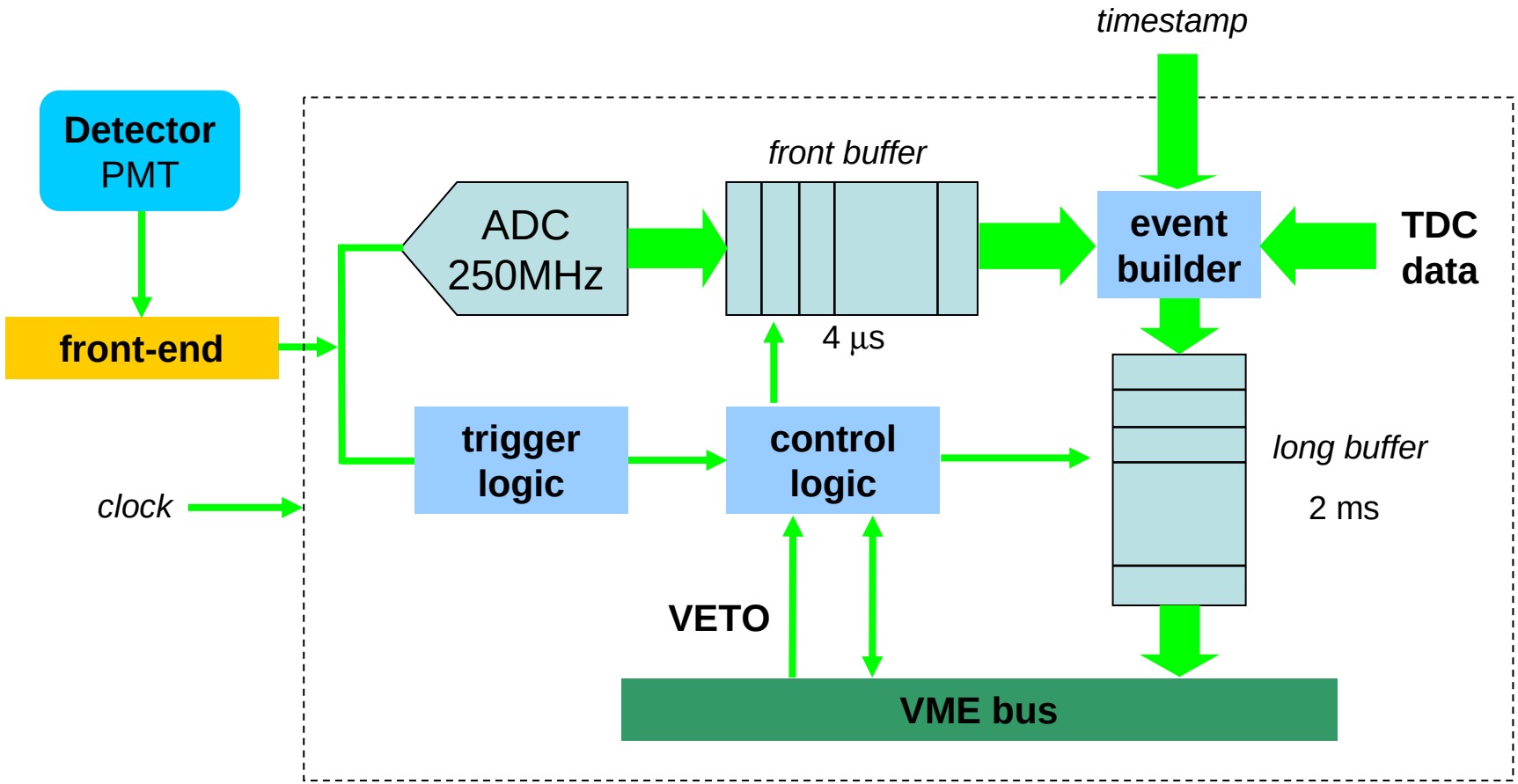
Faixas de endereços (hex)

0000 a 07FF – PROM 0
0800 a 0FFF – PROM 1
1000 a 17FF – PROM 2
1800 a 1FFF – PROM 3

Memórias FIFO: os dados são lidos (b) na mesma ordem em que foram escritos (a).



Utilização de memória FIFO em projeto no CBPF



Módulo de aquisição de dados do Projeto **Neutrinos Angra**

Utilização de memória FIFO em projeto no CBPF

3.3 VOLT HIGH-DENSITY SUPERSYNC II™
36-BIT FIFO
65,536 x 36
131,072 x 36

Preço Unitário = US$ 89,95

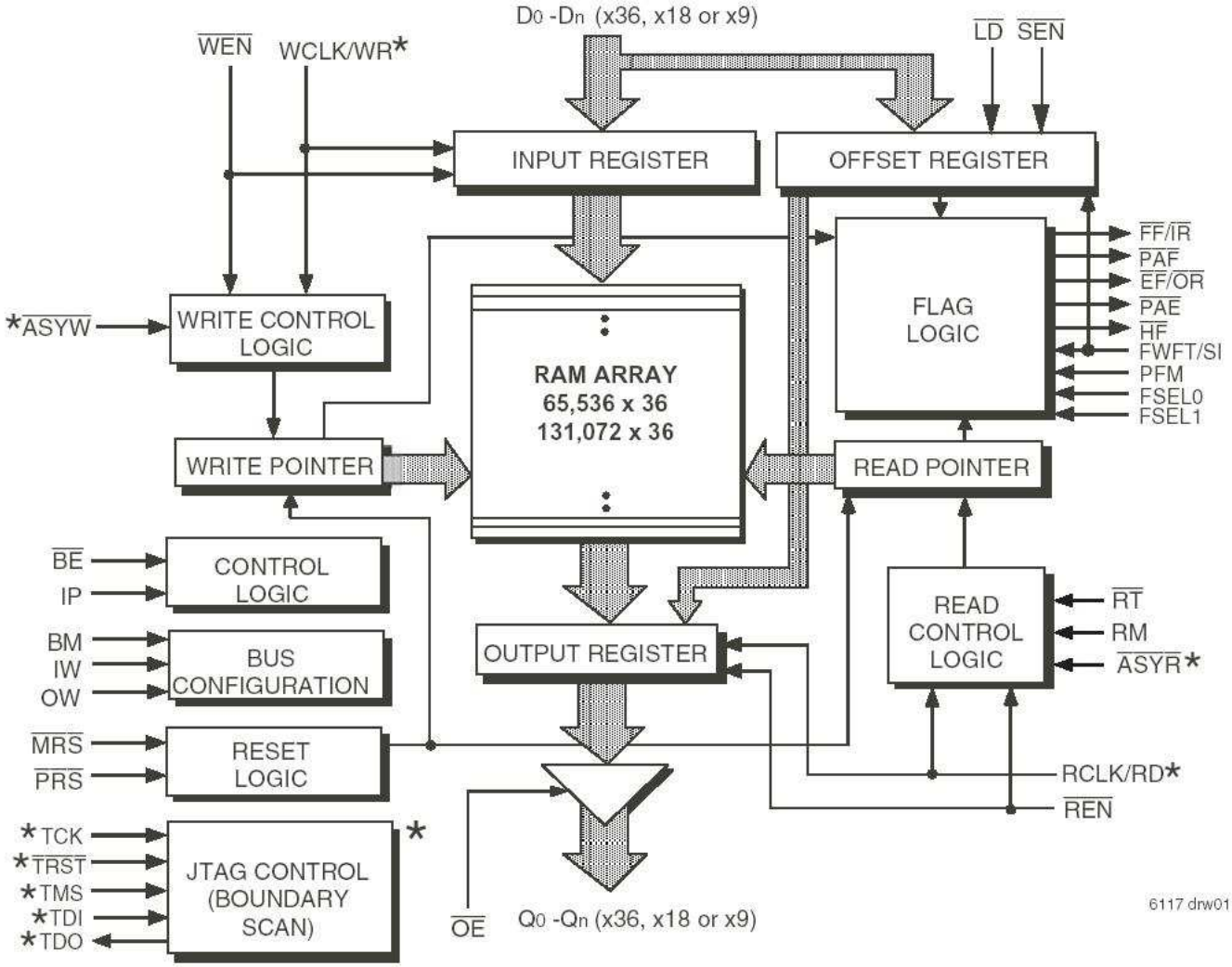
IDT72V36100
IDT72V36110

FEATURES:

- Choose among the following memory organizations:
  - IDT72V36100 — 65,536 x 36
  - IDT72V36110 — 131,072 x 36
- Higher density, 2Meg and 4Meg SuperSync II FIFOs
- Up to 166 MHz Operation of the Clocks
- User selectable Asynchronous read and/or write ports (PBGA Only)
- User selectable input and output port bus-sizing
  - *x36 in to x36 out*
  - *x36 in to x18 out*
  - *x36 in to x9 out*
  - *x18 in to x36 out*
  - *x9 in to x36 out*
- Big-Endian/Little-Endian user selectable byte representation
- 5V input tolerant
- Fixed, low first word latency
- Zero latency retransmit
- Auto power down minimizes standby power consumption
- Master Reset clears entire FIFO
- Partial Reset clears data, but retains programmable settings
- Empty, Full and Half-Full flags signal FIFO status
- Programmable Almost-Empty and Almost-Full flags, each flag can default to one of eight preselected offsets
- Selectable synchronous/asynchronous timing modes for Almost-Empty and Almost-Full flags
- Program programmable flags by either serial or parallel means
- Select IDT Standard timing (using $\overline{EF}$ and $\overline{FF}$ flags) or First Word Fall Through timing (using $\overline{OR}$ and $\overline{IR}$ flags)
- Output enable puts data outputs into high impedance state
- Easily expandable in depth and width
- JTAG port, provided for Boundary Scan function (PBGA Only)
- Independent Read and Write Clocks (permit reading and writing simultaneously)
- Available in a 128-pin Thin Quad Flat Pack (TQFP) or a 144-pin Plastic Ball Grid Array (PBGA) (with additional features)
- Pin compatible to the SuperSync II (IDT72V3640/72V3650/72V3660/72V3670/72V3680/72V3690) family
- High-performance submicron CMOS technology
- Industrial temperature range (–40°C to +85°C) is available
- Green parts available, see ordering information

Memória FIFO utilizada no Projeto **Neutrinos Angra**

Utilização de memória FIFO em projeto no CBPF



Memória FIFO do Projeto **Neutrinos Angra**

Método *checksum* para verificação de erros em uma ROM 8×8

(a) ROM com dados corretos; (b) ROM com erro nos dados