UNIOESTE
Ciência da Computação

Sistemas Digitais

Circuitos Combinacionais

Prof. Jorge Habib El Khouri
Prof. Antonio Marcos Hachisuca

2020

Referências Bibliográficas

1. *Digital Fundamentals, Thomas L. Floyd; Editora: Pearson; Edição: 11; Ano: 2015;*
2. *Sistemas Digitais Princípios e Aplicações, Ronald J. Tocci; Editora: Pearson; Edição: 11; Ano: 2011;*
3. *Computer Organization and Design, David A. Patterson; Editora: Elsevier; Edição: 1; Ano: 2017*
4. *Digital Design: Principles and Practices, John F. Wakerly; Editora: Pearson; Edição: 5; Ano: 2018;*
5. *Guide to Assembly Language Programming in Linux, Sivarama P. Dandamudi; Editora: Springer; Edição: 1; Ano: 2005.*

Regras Básicas da Álgebra Booleana

Comutativa

$$A + B = B + A$$

$$AB = BA$$

Associativa

$$(A + B) + C = A + (B + C)$$

$$(AB)C = A(BC)$$

Distributiva

$$A(B + C) = AB + AC$$

$$A + 0 = A \qquad A \oplus 0 = A$$

$$A + 1 = 1 \qquad A \oplus 1 = \bar{A}$$

$$A + A = A \qquad A \oplus A = 0$$

$$A + \bar{A} = 1 \qquad A \oplus \bar{A} = 1$$

$$\bar{\bar{A}} = A \qquad A + AB = A$$

$$A \cdot 0 = 0 \qquad A + \bar{A}B = A + B$$

$$A \cdot 1 = A \qquad (A + B)(A + C) = A + BC$$

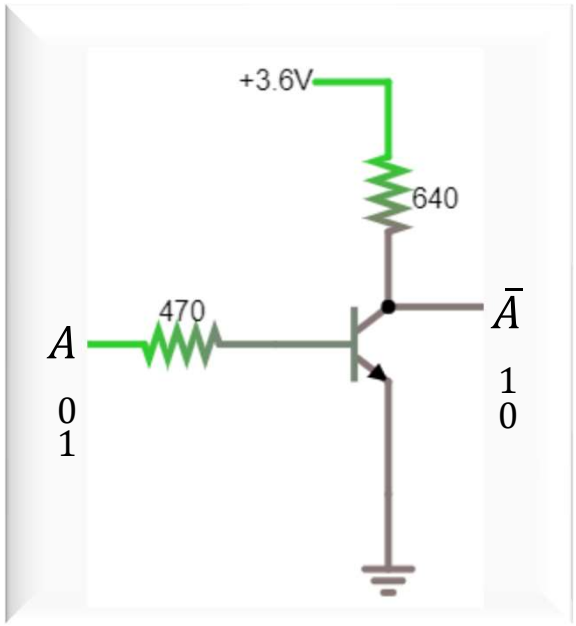
$$A \cdot A = A \qquad A \oplus B = A\bar{B} + \bar{A}B$$

$$A \cdot \bar{A} = 0 \qquad \overline{A \oplus B} = AB + \bar{A}\bar{B}$$

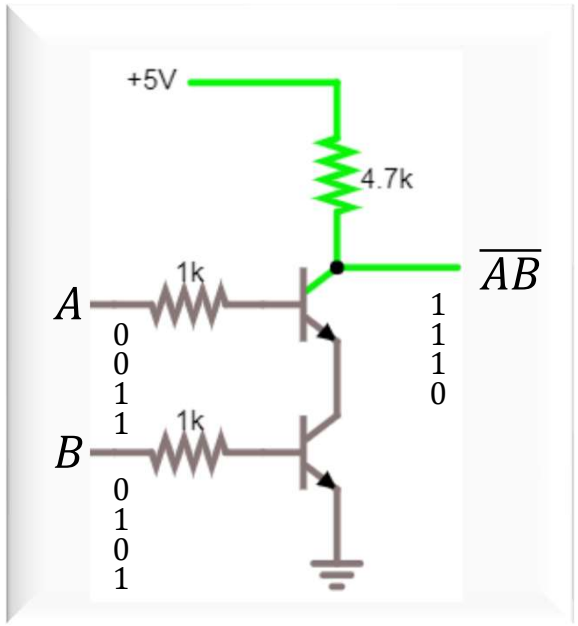
$$\overline{X \cdot Y \cdot Z} = \bar{X} + \bar{Y} + \bar{Z} \qquad \overline{X + Y + Z} = \bar{X} \cdot \bar{Y} \cdot \bar{Z}$$

Elementos Lógicos Universais

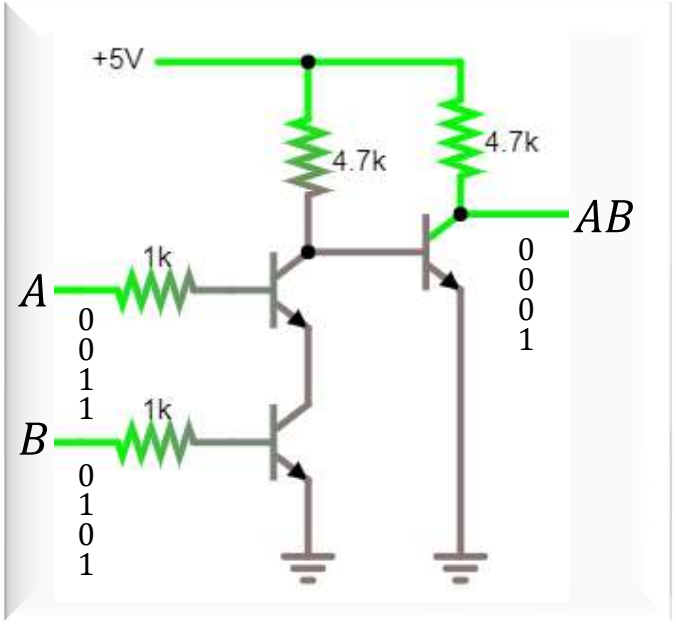
- A porta NAND é considerada universal porque pode ser utilizada para produzir todas as demais portas lógicas;
- A implementação de algumas Portas pode ser feita como segue:



$NOT\ (1T)$

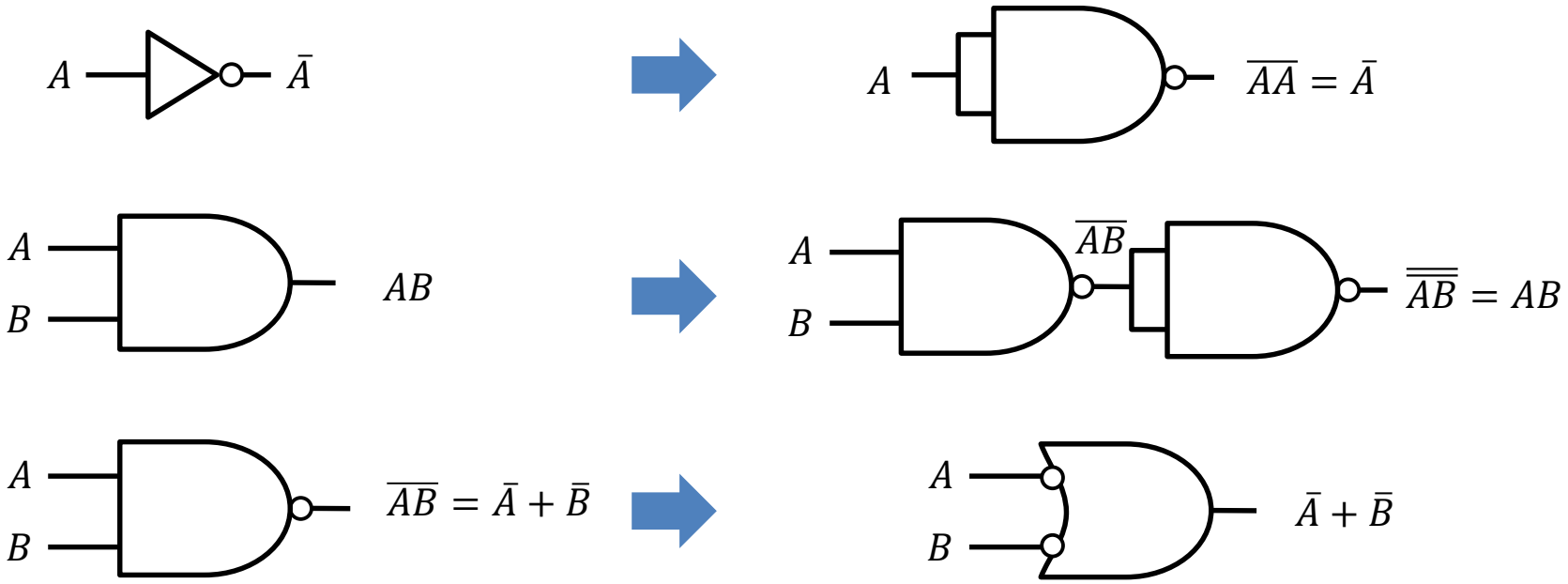


$NAND\ (2T)$



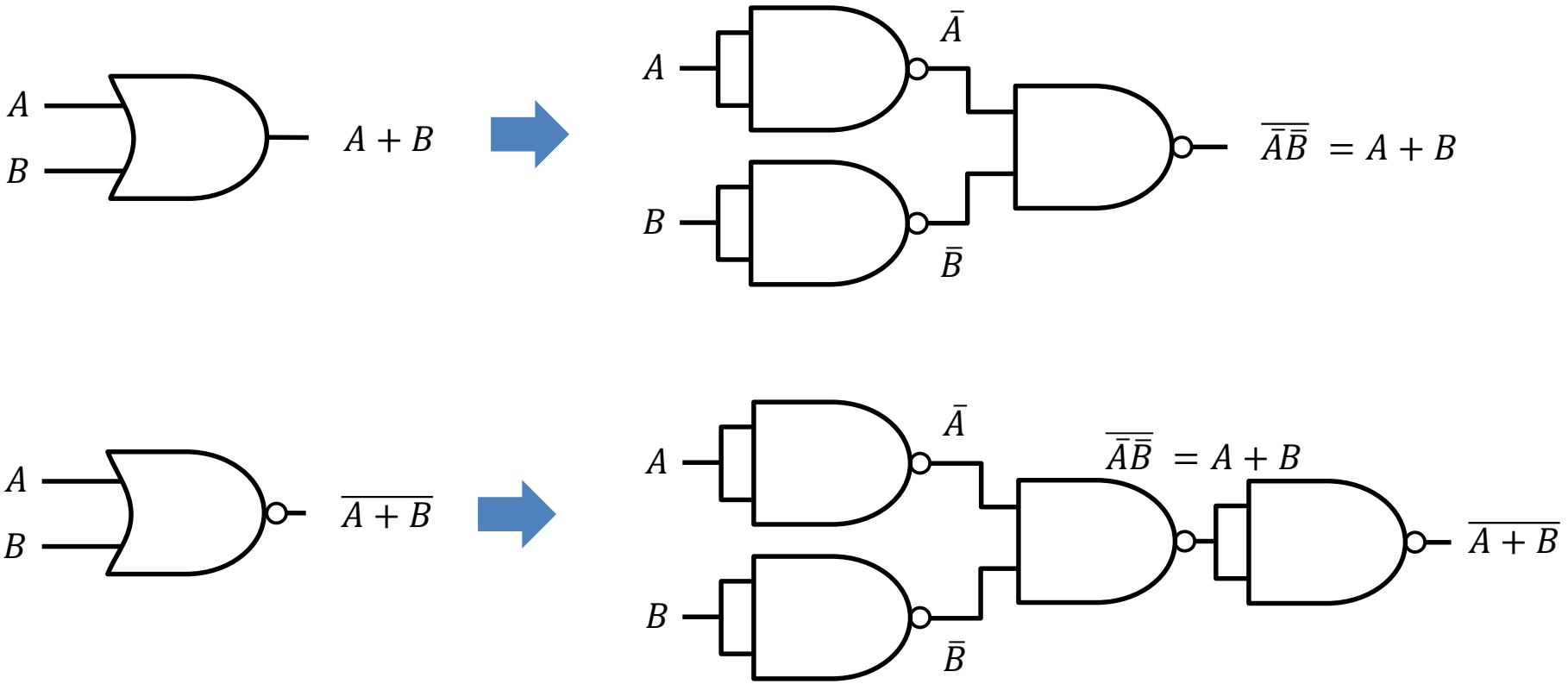
$AND\ (3T)$

Equivalência de Portas
Porta NAND



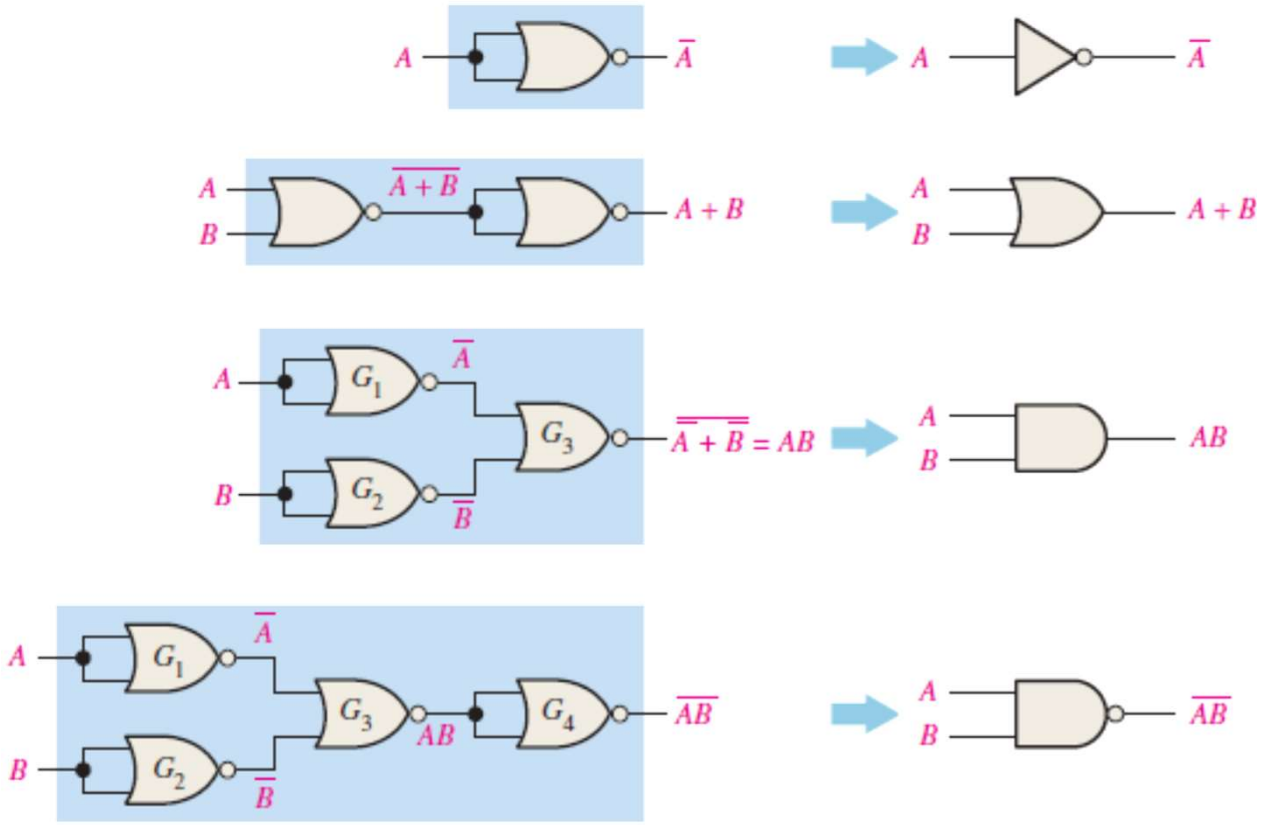
Digital Fundamentals, Thomas L. Floyd; Editora: Pearson; Edição: 11; Ano: 2015

Equivalência de Portas
Porta NAND



Digital Fundamentals, Thomas L. Floyd; Editora: Pearson; Edição: 11; Ano: 2015

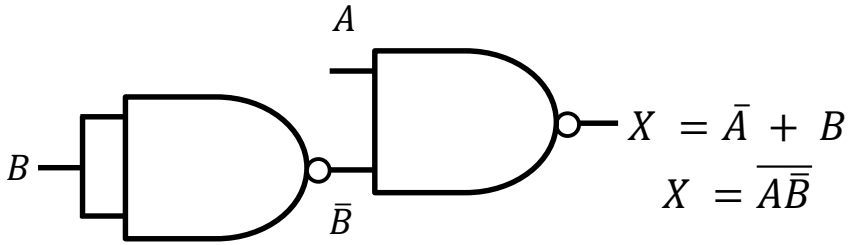
Equivalência de Portas
Porta NOR



Digital Fundamentals, Thomas L. Floyd; Editora: Pearson; Edição: 11; Ano: 2015

Equivalência de Portas
Exemplo

$$X = \bar{A} + B$$



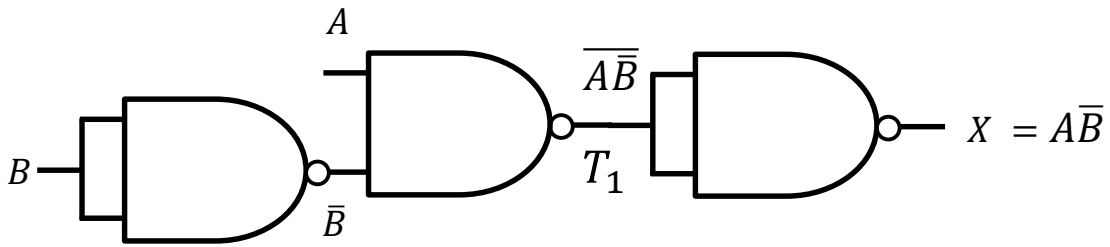
$$X = A * (B * B)$$

$* \; representando \; o \; NAND$

Digital Fundamentals, Thomas L. Floyd; Editora: Pearson; Edição: 11; Ano: 2015

Equivalência de Portas
Exemplo

$X = A\bar{B}$



$T_1 = A * (B * B)$

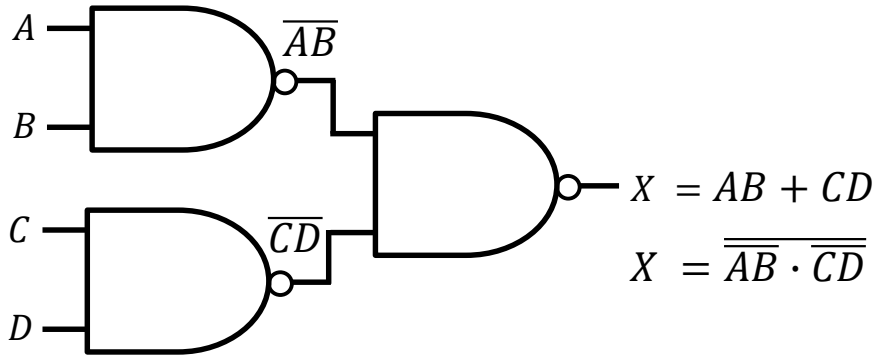
$X = T_1 * T_1$

$* representando\ o\ NAND$

Digital Fundamentals, Thomas L. Floyd; Editora: Pearson; Edição: 11; Ano: 2015

Equivalência de Portas
Exemplo

$$X = AB + CD$$



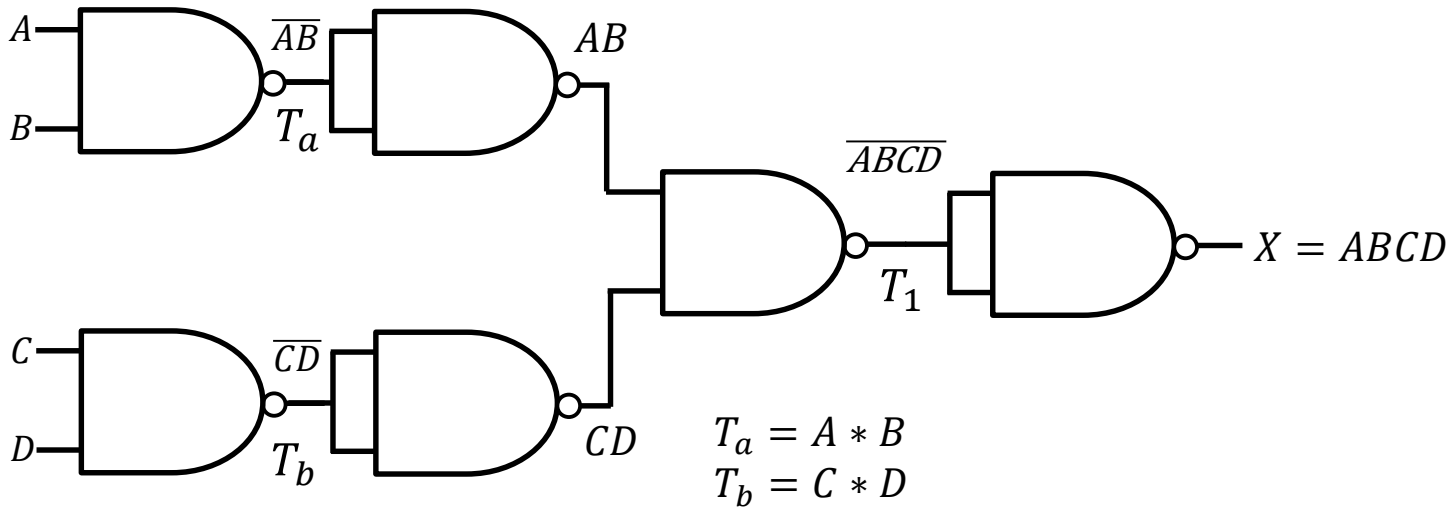
$$X = (A * B) * (C * D)$$

$* \; representando \; o \; NAND$

Digital Fundamentals, Thomas L. Floyd; Editora: Pearson; Edição: 11; Ano: 2015

Equivalência de Portas
Exemplo

$X = ABCD$

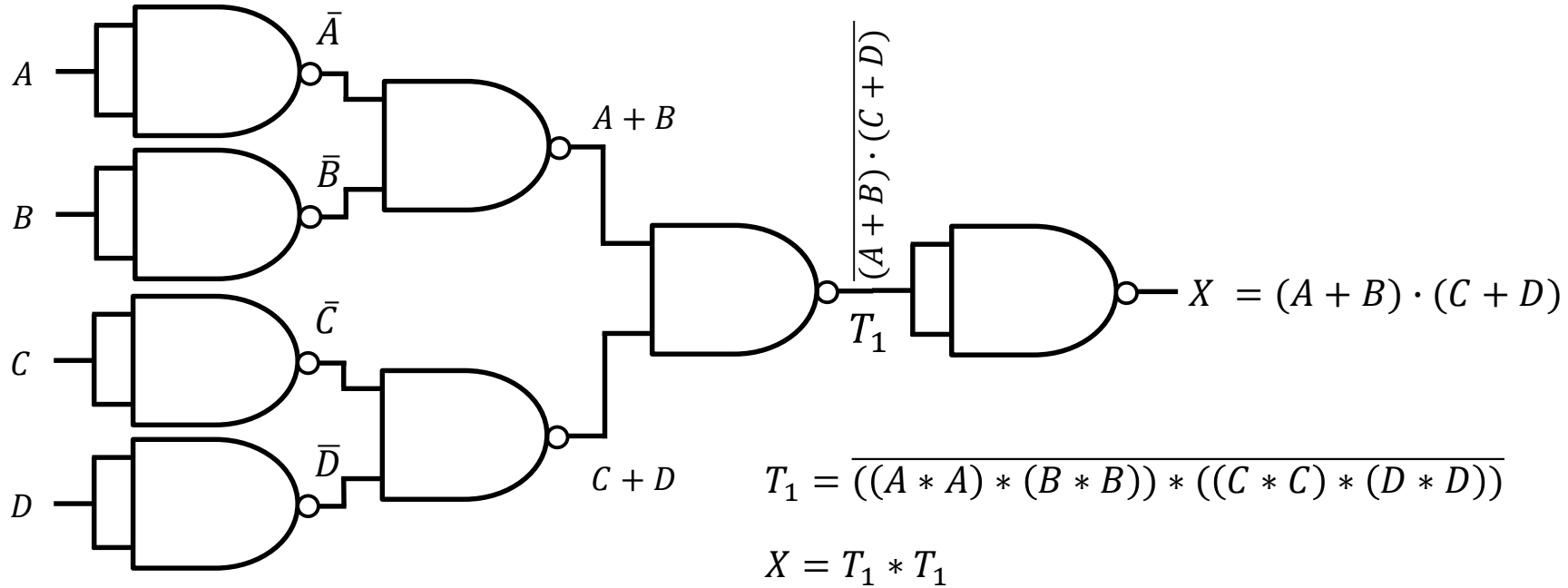


$T_a = A * B$
$T_b = C * D$
$T_1 = (T_a * T_a) * (T_b * T_b)$
$X = T_1 * T_1$
$* \; representando \; o \; NAND$

Digital Fundamentals, Thomas L. Floyd; Editora: Pearson; Edição: 11; Ano: 2015

Equivalência de Portas
Exemplo

$$X = (A + B) \cdot (C + D)$$



$$T_1 = \overline{((A * A) * (B * B)) * ((C * C) * (D * D))}$$

$$X = T_1 * T_1$$

$* \; representando \; o \; NAND$

Digital Fundamentals, Thomas L. Floyd; Editora: Pearson; Edição: 11; Ano: 2015

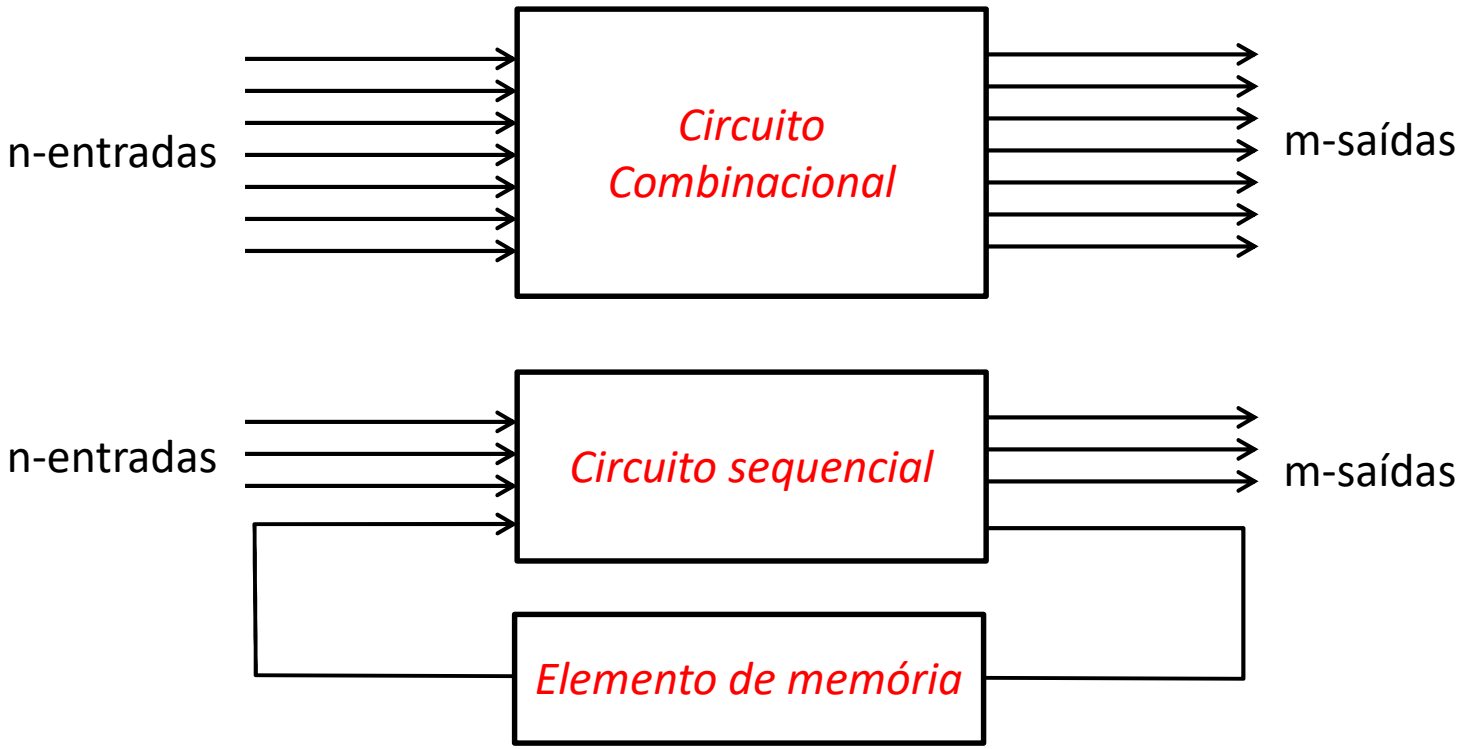
CIRCUITOS COMBINACIONAIS

- Um circuito combinacional consiste em portas lógicas cujas saídas, em qualquer momento, são determinadas pela combinação dos valores das entradas
- Para n variáveis de entrada, existem 2^n combinações de entrada binária possíveis
- Para cada combinação binária das variáveis de entrada, existe uma saída possível

Circuitos Combinacionais vs circuitos sequenciais

- Os circuitos combinacionais não possuem memória interna
  - O valor de saída depende apenas dos valores atuais de entrada
- Os circuitos sequenciais contem lógica combinacional, e elementos de memória (usados para armazenar estados de circuito)
  - As saídas dependem dos valores de entrada atuais e dos valores de entrada anteriores (mantidos nos elementos de memória)

Circuitos Combinacionais vs circuitos sequenciais



Circuitos Combinacionais vs circuitos sequenciais

CIRCUITOS COMBINACIONAIS
1. Codificador/decodificador 2. Comparadores 3. Geradores de paridade 4. Multiplexador/Demux 5. PLAs 6. Memórias ROM 7. Somador / Subtrator 8. ULA 9. Multiplicadores / Divisores

CIRCUITOS SEQUENCIAIS
1. Latches 2. Flip-Flop 3. Registradores 4. Contadores 5. Máquina de Estados 6. Geradores de clock 7. Memória RAM 8. Sequenciadores

SOMADOR

Circuito Somador

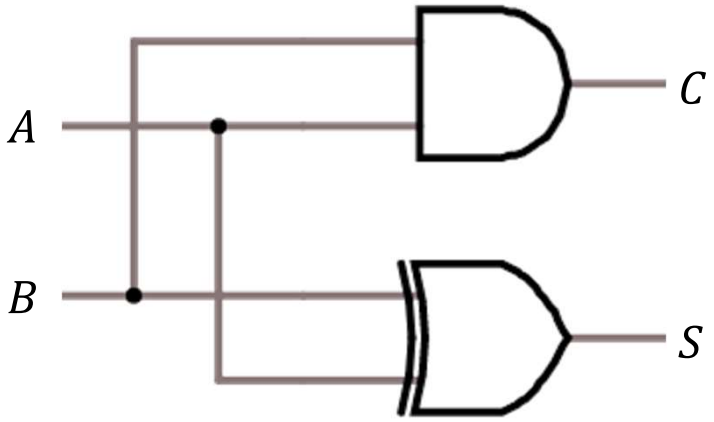
Half-Adder

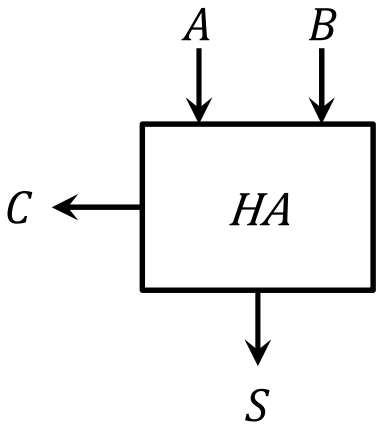
$$\begin{array}{r} C \\ A \\ + \quad B \\ \hline S \end{array}$$

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

$$S = A \oplus B$$

$$C = AB$$





Circuito Somador

Full-Adder

C_{in}	A	B	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

$$S = A \oplus B \oplus C_{in}$$

$$C_{out} = ??$$

AB \ C	$\overline{C_{in}}$	C_{in}
$\bar{A}\bar{B}$ (00)	0	1
$\bar{A}B$ (01)	2	1 3
AB (11)	1 6	1 7
$A\bar{B}$ (10)	4	1 5

$$C_{out}\ C_{in}$$
$$+\ A$$
$$\ \ B$$
$$\overline{\ \ \ \ \ \ S}$$

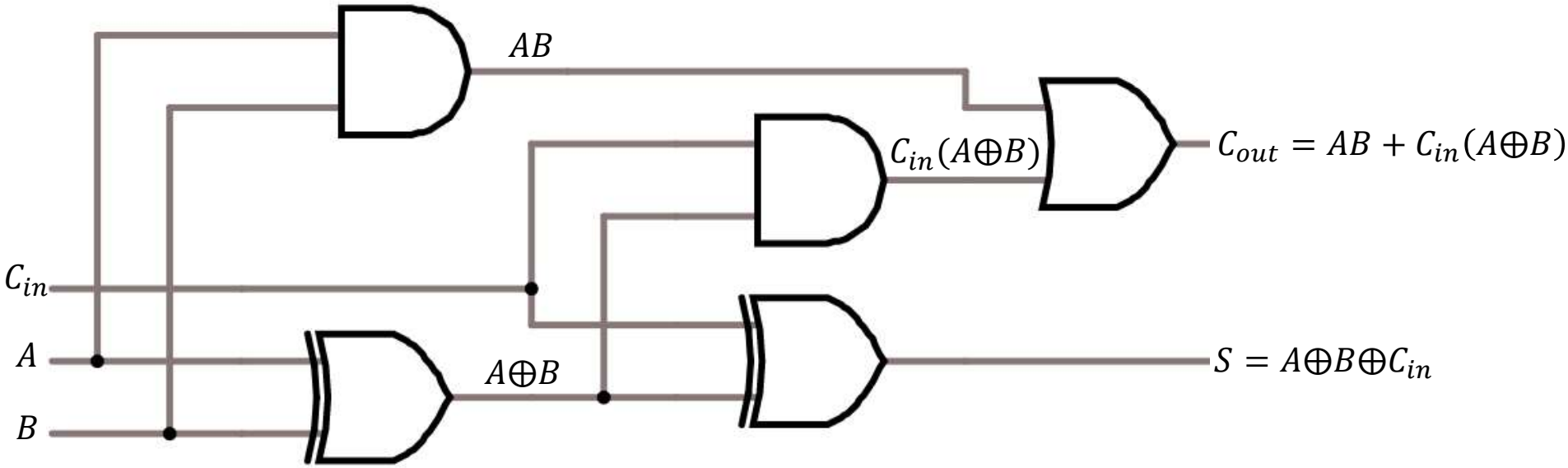
$$C_{out} = AB\bar{C}_{in} + \bar{A}BC_{in} + A\bar{B}C_{in} + ABC_{in}$$
$$C_{out} = AB(\bar{C}_{in} + C_{in}) + C_{in}(\bar{A}B + A\bar{B})$$
$$C_{out} = AB + C_{in}(A \oplus B)$$

$$AB + AC_{in} + BC_{in}$$

$$C_{out} = AB + C_{in}(A + B)$$

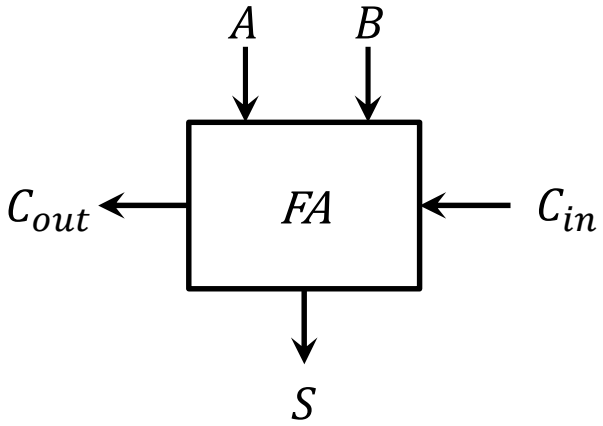
Circuito Somador

Full-Adder



Circuito Somador

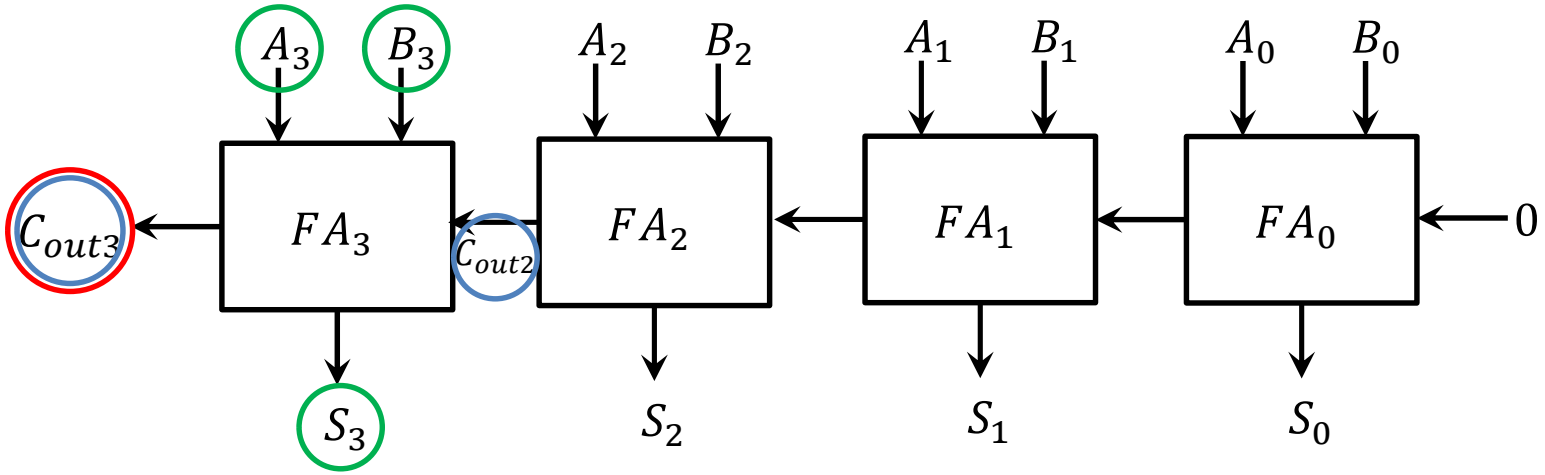
Full-Adder



Circuito Somador

Full-Adder

Soma de dois números de 4 bits: $S_{0\ldots3} = A_{0\ldots3} + B_{0\ldots3}$

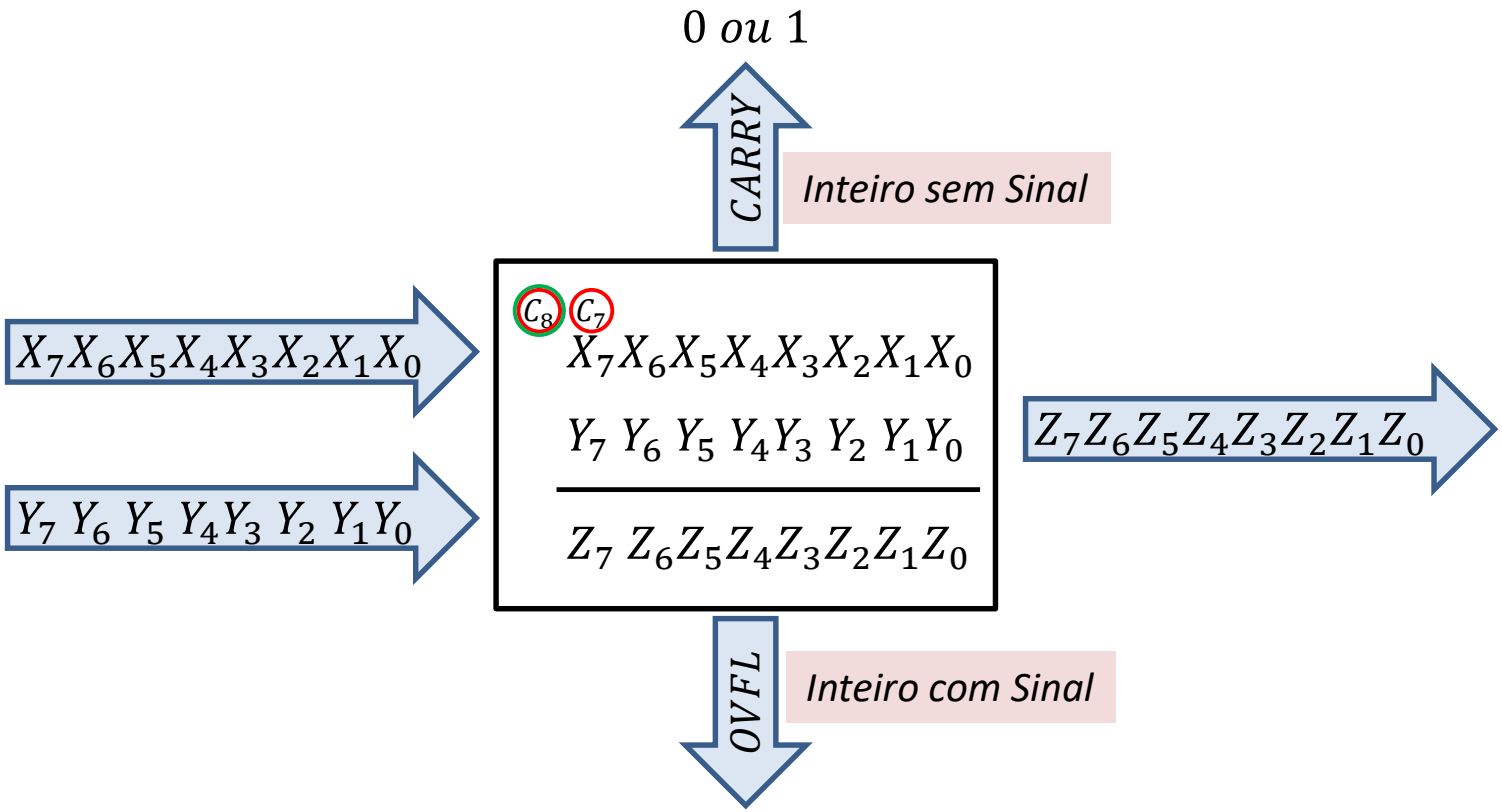


$CARRY$ = flag para soma de inteiros sem sinal: se 1 então Erro

$OVFL$ = flag para soma de inteiros com sinal: se 1 então Erro

$OVFL$ = flag para soma de inteiros com sinal: se 1 então Erro

Soma de Inteiros



Soma de Inteiros com Sinal

OVERFLOW

SINAIS			
A	B	R	$OVFL$
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	1
1	1	1	0

$$OVFL = AB\bar{R} + \bar{A}\bar{B}R$$

$$OVFL = A_3B_3\bar{S}_3 + \bar{A}_3\bar{B}_3S_3$$

		$\bar{R}$	R
	AB \ C	0	1
$\bar{A}\bar{B}$	00	0	1 (1)
$\bar{A}B$	01	2	3
AB	11	1 (6)	7
$A\bar{B}$	10	4	5

$$AB\bar{R} + \bar{A}\bar{B}R$$

Soma de Inteiros com Sinal

OVERFLOW

SINAIS				
A	*B*	*R*	*OVFL*	$C_3 \oplus C_2$
0	0	0	0	0
0	0	1	1	1
0	1	0	0	0
0	1	1	0	0
1	0	0	0	0
1	0	1	0	0
1	1	0	1	1
1	1	1	0	0

$C_3C_2C_1C_0$

```
  0001        0111        1101        0001
   0101        0111        0101        0101
+  0001     +  0001     +  1101     +  1001
   ----        ----        ----        ----
   0110        1000        0010        1110

  1111        0011        1001        1101
   1111        1011        1101        1101
+  0001     +  0001     +  1001     +  1101
   ----        ----        ----        ----
   0000        1100        0110        1010
```

$$OVFL = C_3 \oplus C_2$$

Circuito Subtrator

Full-Subtractor

B_{in}	A	B	S	B_{out}
0	0	0	0	0
0	0	1	1	1
0	1	0	1	0
0	1	1	0	0
1	0	0	1	1
1	0	1	0	1
1	1	0	0	0
1	1	1	1	1

$$S = A \oplus B \oplus B_{in}$$

$$B_{out} = ??$$

AB \ C	$\overline{B_{in}}$	B_{in}
$\bar{A}\bar{B}$ 00	0	1 (1)
$\bar{A}B$ 01	1 (2)	1 (3)
AB 11	6	1 (7)
$A\bar{B}$ 10	4	5

-1 0 _ 0 1 1	-1 -1 _ 0 0 1	-1 -1 _ 0 1 0

$$B_{out} = \bar{A}B\bar{B}_{in} + \bar{A}\bar{B}B_{in} + \bar{A}BB_{in} + ABB_{in}$$

$$B_{out} = \bar{A}B(B_{in} + \bar{B}_{in}) + B_{in}(\bar{A}\bar{B} + AB)$$

$$B_{out} = \bar{A}B + B_{in}(\overline{A \oplus B})$$

$$\bar{A}B + BB_{in} + \bar{A}B_{in}$$

$$B_{out} = \bar{A}B + B_{in}(\bar{A} + B)$$

Circuito Subtrator

Full-Subtractor

B_{in}	A	B	S	B_{out}
0	0	0	0	0
0	0	1	1	1
0	1	0	1	0
0	1	1	0	0
1	0	0	1	1
1	0	1	0	1
1	1	0	0	0
1	1	1	1	1

$$S = A \oplus B \oplus B_{in}$$

$$B_{out} = ??$$

	$\overline{B_{in}}$	B_{in}
AB \ C	0	1
$\bar{A}\bar{B}$	00 (0)	1 (1)
$\bar{A}B$	01 (2)	1 (3)
AB	11 (6)	1 (7)
$A\bar{B}$	10 (4)	(5)

$$\bar{A}B + ABB_{in} + \bar{A}\bar{B}B_{in}$$

-1 0	-1 -1	-1 -1
0	0	0
1	0	1
1	1	0

$$B_{out} = \bar{A}B + B_{in}(AB + \bar{A}\bar{B}) \qquad B_{out} = \bar{A}B + B_{in}(\overline{A \oplus B})$$

REPRESENTAÇÃO DE DADOS

INTEIROS SEM E COM SINAL

Representação de Dados

Inteiros

- Como obter um Binário que represente um número inteiro?
- Inteiro sem sinal e Inteiro com sinal;
- Para os casos exemplos será utilizado $N = 4bits$;
- Inteiro sem sinal contém apenas a magnitude do número:
  - o binário correspondente pode ser obtido pela conversão do número para a base 2;

- $X = 8$ $BIN = 1000$
- $X = 13$ $BIN = 1101$
- $X = 19$ $BIN = 10011$
- $BIN = 1001$ $X = 9$
- $BIN = 0000$ $X = 0$

Assim, a faixa de representação é dada por:

$$0 \leq X_4 \leq 15$$

$$0 \leq X_{16} \leq 65535$$

$$0 \leq X_N \leq 2^N - 1$$

Representação de Dados

Inteiros

- Para número inteiro com sinal há a necessidade de codificar o sinal e a magnitude:
- O método Complemento de 2 ou $C-2$ consiste em:
- Se o sinal de X for positivo, converter o número para a base 2 e verificar se o MSB resultou em 0:

$X = +4$	`0100`	→ Sinal Ok.
$X = +9$	`1001`	→ Sinal inconsistente
$X = +0$	`0000`	→ Sinal Ok.
$X = +7$	`0111`	→ Sinal Ok.

Representação de Dados

Inteiros

- Se o sinal de X for negativo, seguir o processo indicado abaixo e verificar se o MSB resultou em 1:

$X = -5$

$(|X|)_2$: 0101
$0 \Leftrightarrow 1$: 1010
+ 1

BIN: **1011** ✓

$X = -0$

$(|X|)_2$: 0000
$0 \Leftrightarrow 1$: 1111
+ 1

BIN: **0000** ✓

$X = -10$

$(|X|)_2$: 1010
$0 \Leftrightarrow 1$: 0101
+ 1

BIN: **0110** ×

$X = -1$

$(|X|)_2$: 0001
$0 \Leftrightarrow 1$: 1110
+ 1

BIN: **1111** ✓

$X = -8$

$(|X|)_2$: 1000
$0 \Leftrightarrow 1$: 0111
+ 1

BIN: **1000** ✓

$X = -9$

$(|X|)_2$: 1001
$0 \Leftrightarrow 1$: 0110
+ 1

BIN: **0111** ×

Assim, a faixa de representação para $C-2$ é dada por:

$$-8 \leq X_4 \leq +7$$

$$-32768 \leq X_{16} \leq +32767$$

$$-2^{N-1} \leq X_N \leq +(2^{N-1}-1)$$

O método $C-2$ proporciona a mesma representação para o $+0$ e -0.

Representação de Dados

Inteiros

- Decodificar os seguintes binários que representam inteiros com sinal em $C-2$:

$BIN = 0010$

$X = +2$

$BIN = 1101$

$X = -3$

```
BIN:          1101
 -               1
              ----
              1100
0 ⇔ 1:        0011 (|X|)2
```

```
BIN:          1101
0 ⇔ 1:        0010
            +    1
              ----
(|X|)2:       0011
```

$BIN = 1001$

$X = -7$

```
BIN:          1001
0 ⇔ 1:        0110
            +    1
              ----
(|X|)2:       0111
```

$BIN = 0111$

$X = +7$

$BIN = 1100$

$X = -4$

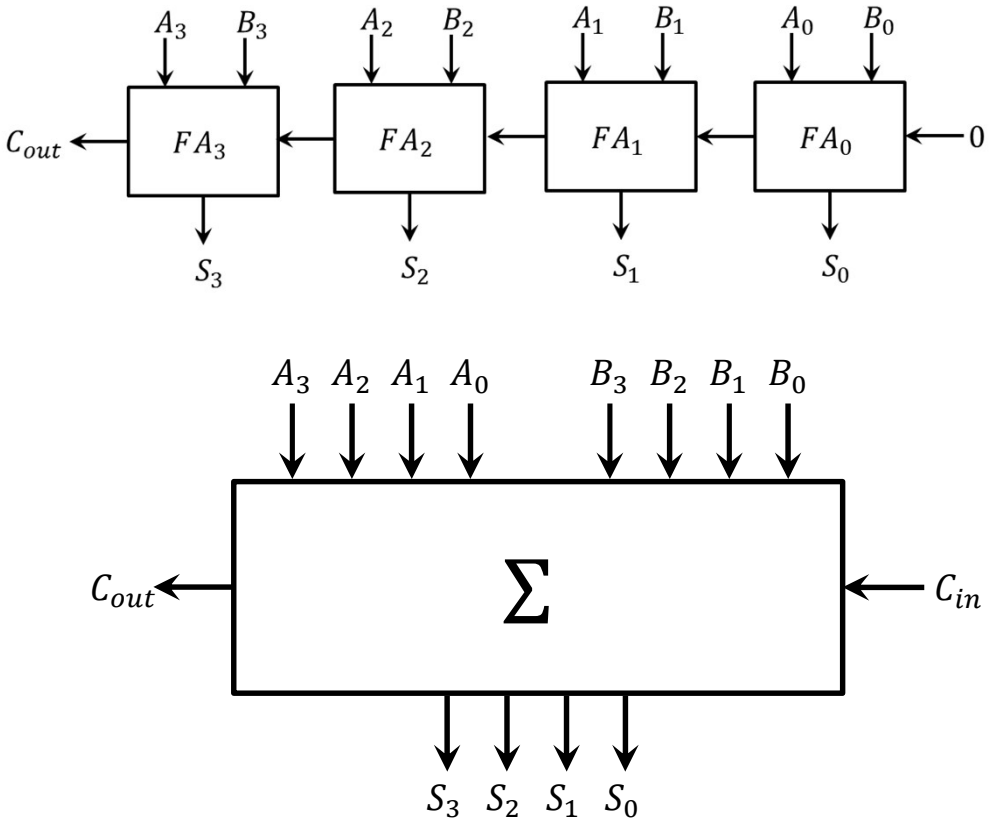
```
BIN:          1100
0 ⇔ 1:        0011
            +    1
              ----
(|X|)2:       0100
```

ENCADEANDO SOMADORES

INTEGRANDO SOMADOR/SUBTRATOR

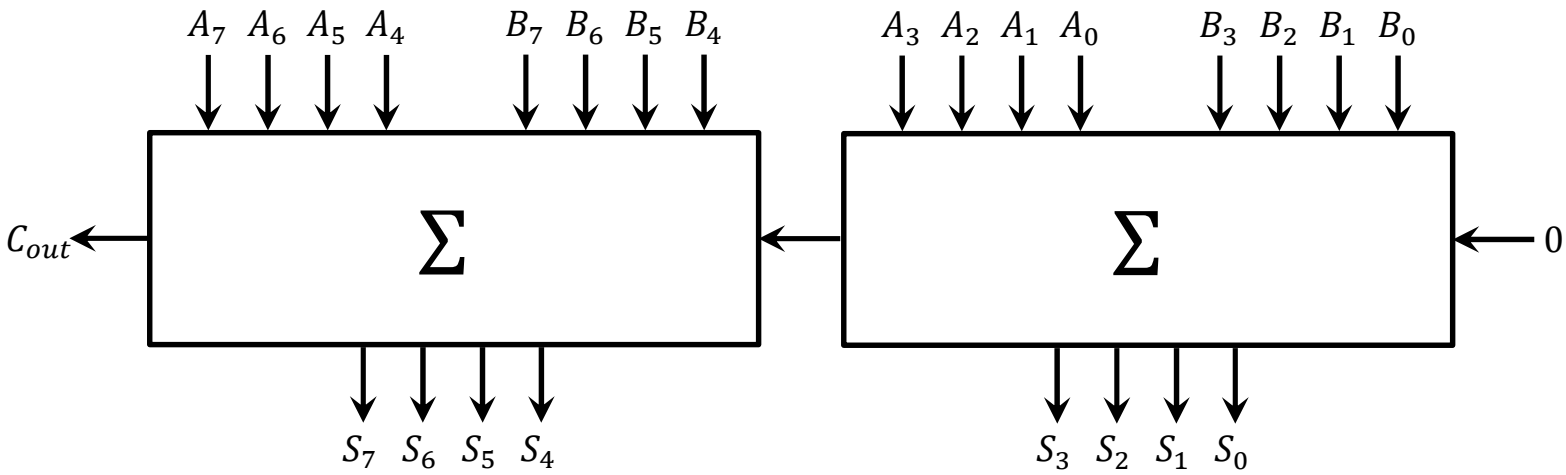
Circuito Somador

Somador de 4 Bits



Circuito Somador

Somador de 8 Bits



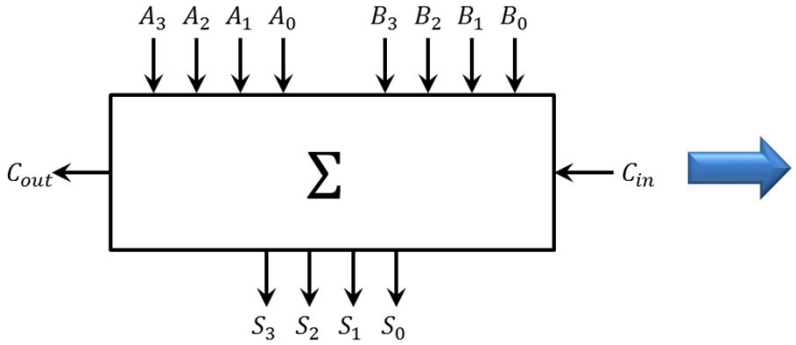
Integrando Somador/Subtrator

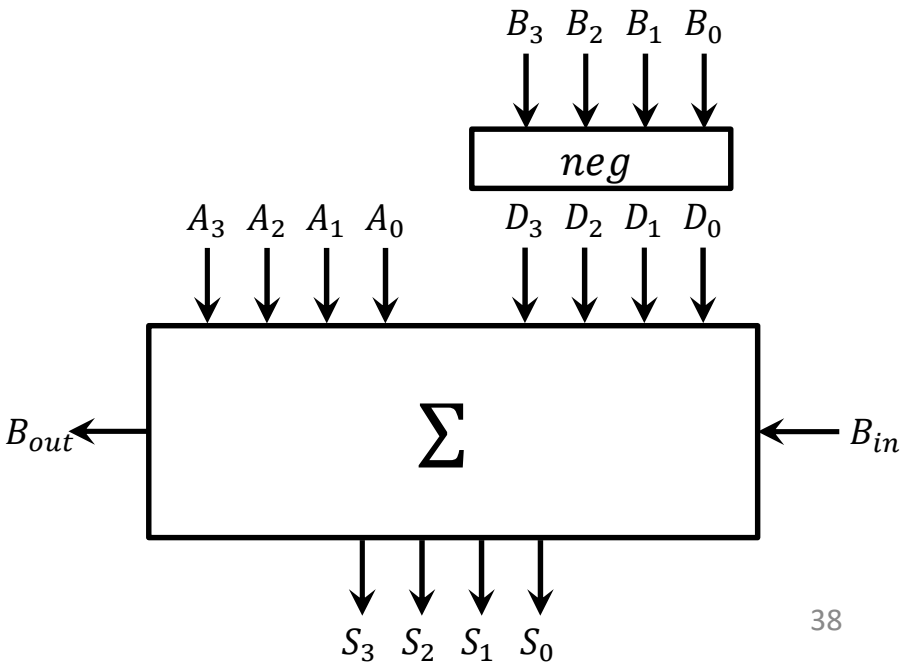
Somador de 4 Bits

É possível implementar o $SUBTRATOR$ com o $SOMADOR$?

$$S[3 \ldots 0] = A[3 \ldots 0] - B[3..0]$$

$$S[3 \ldots 0] = A[3 \ldots 0] + (-B[3..0])$$





Circuito Subtrator

Subtrator de 4 Bits

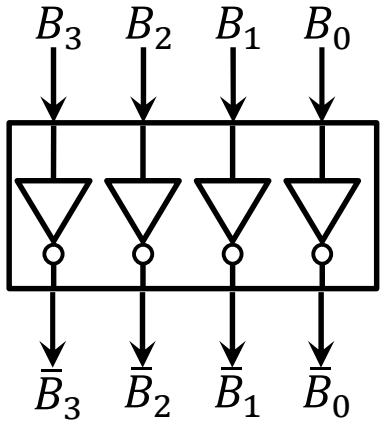
É possível implementar o $SUBTRATOR$ com o $SOMADOR$?

$D[3 \ldots 0] = -B[3..0]$

A transformação deverá converter um inteiro sem sinal em seu equivalente negativo em $C - 2$.

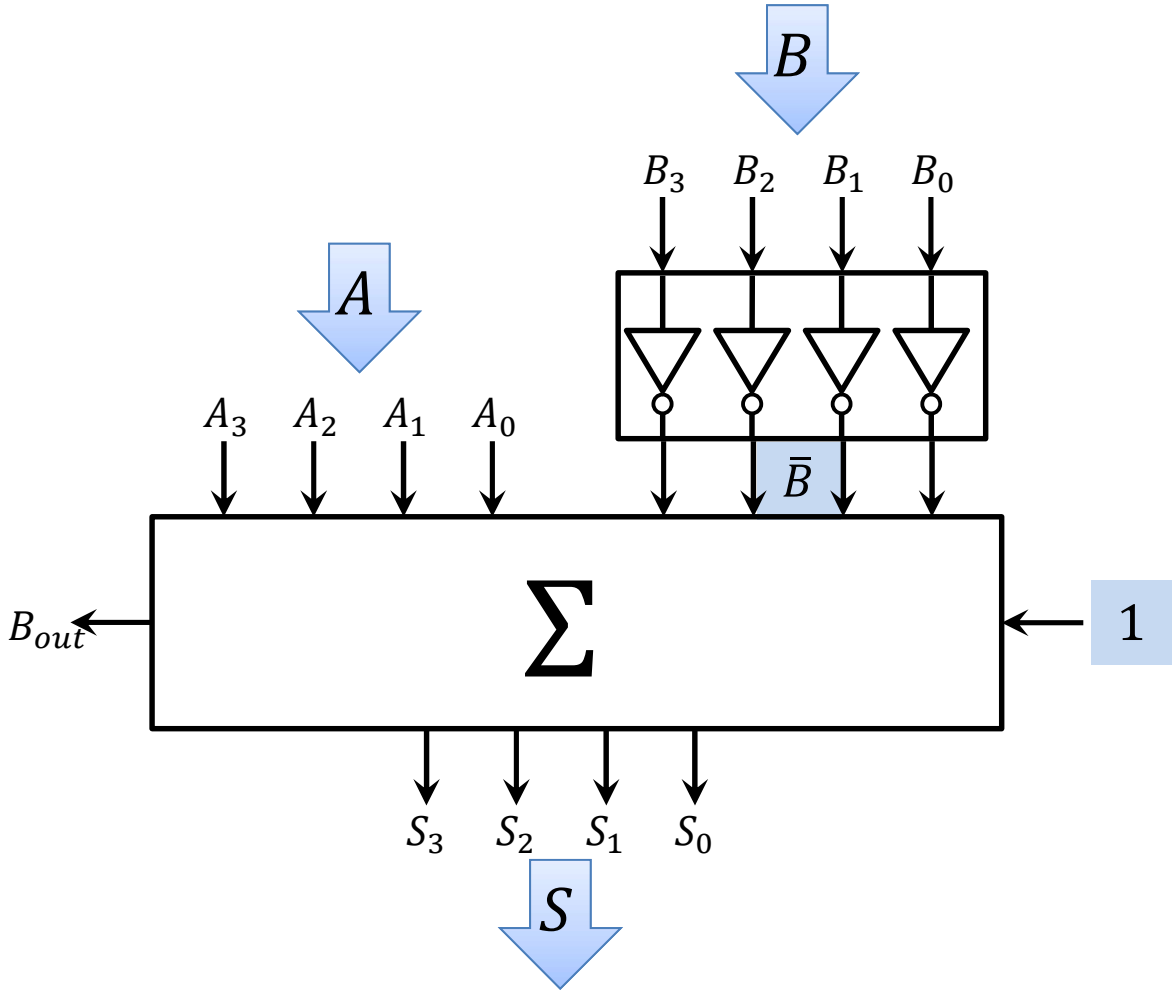
Exemplo: 5 em -5.

$$\begin{array}{r} B = 0101 \\ \bar{B} = 1010 \\ + \qquad 1 \\ \hline -B = 1011 \end{array}$$



Circuito Subtrator

Subtrator de 4 Bits

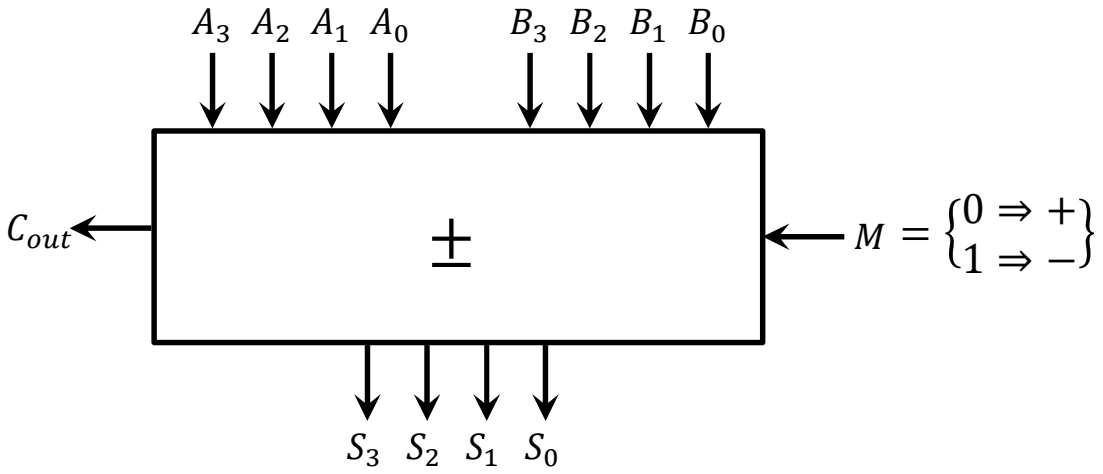


$\bar{B} + 1 = -B$

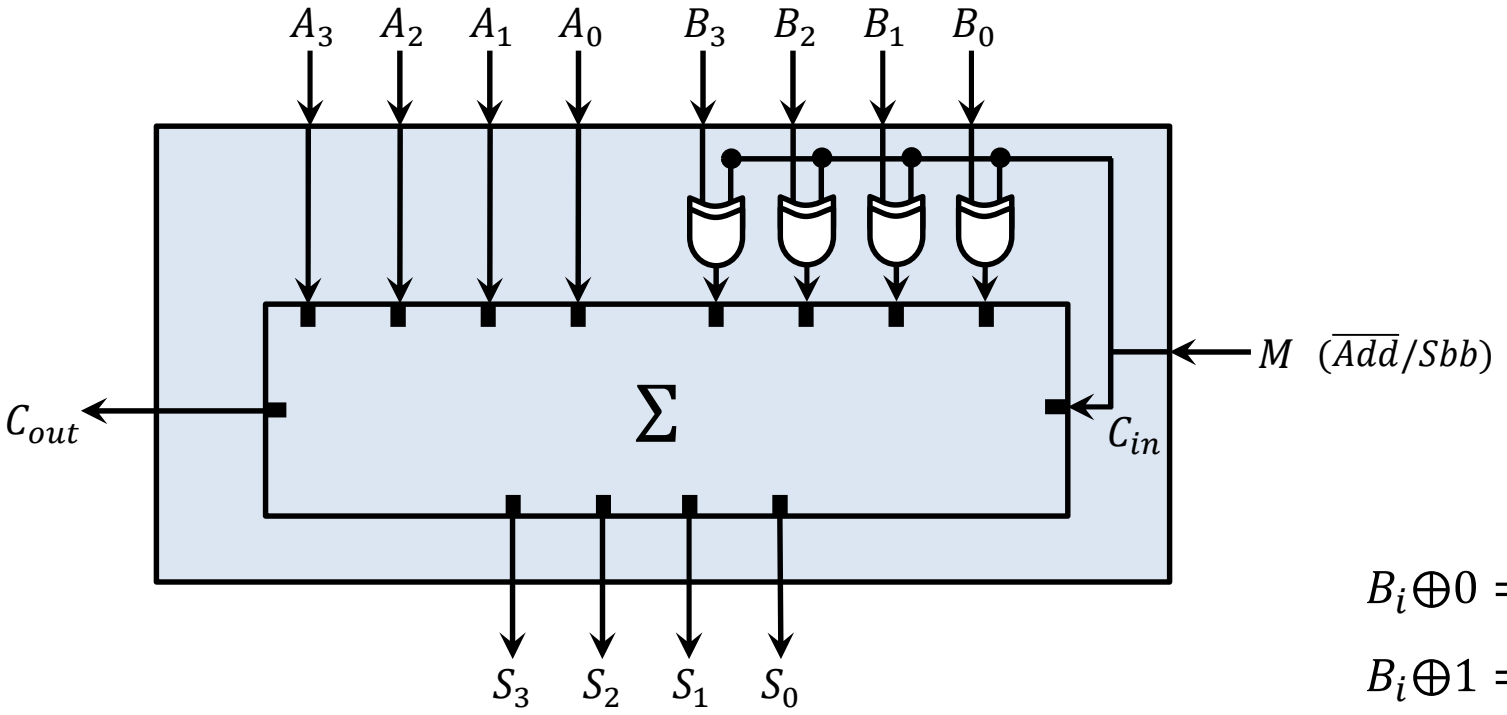
Em $C - 2$

Circuito Somador/Subtrator

Podemos pensar em um circuito que integra a $SOMA$ e $SUBTRAÇÃO$, acrescentando um $flag\ M$ que seleciona a operação desejada.



Circuito Somador/Subtrator



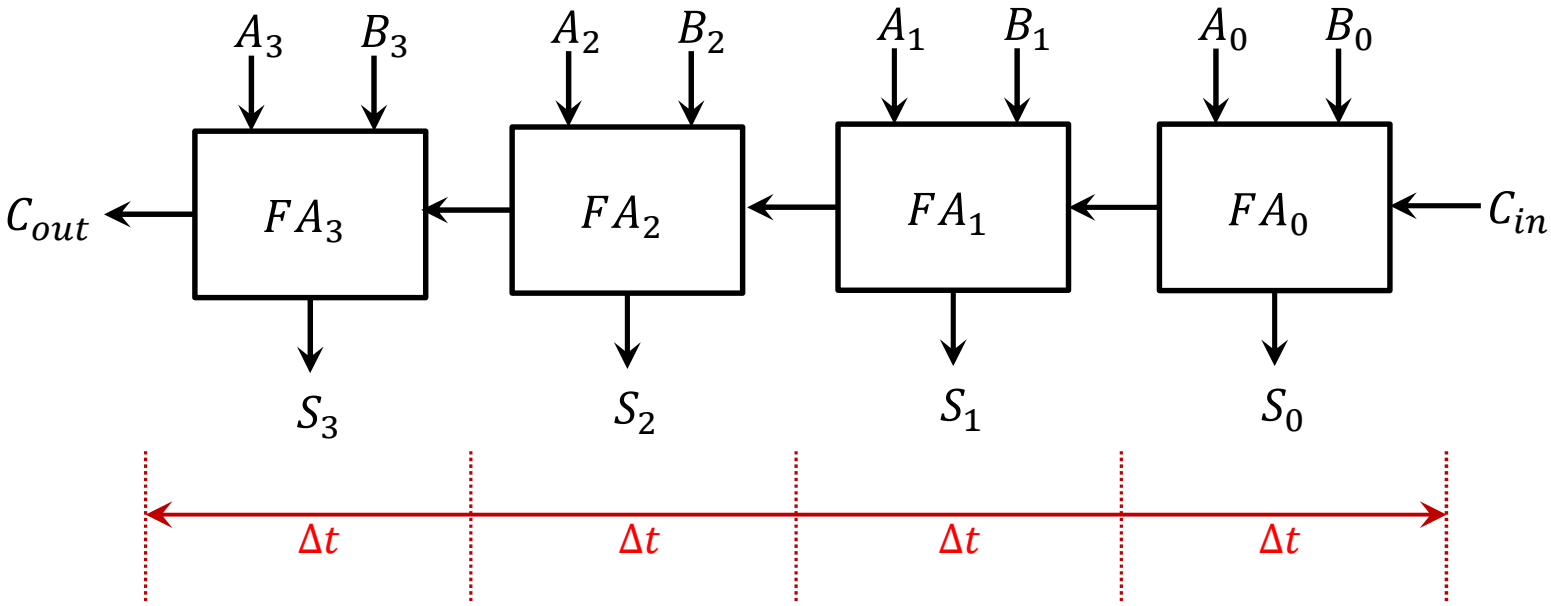
$$B_i \oplus 0 = B_i$$

$$B_i \oplus 1 = \bar{B}_i$$

OTIMIZAÇÃO DO SOMADOR

Circuito Somador/Subtrator
Otimização

- O circuito $SOMADOR$ forma um importante componente da Unidade de Aritmética e Lógica;
- Assim, é importante avaliar o possibilidade de otimização da lógica utilizada.

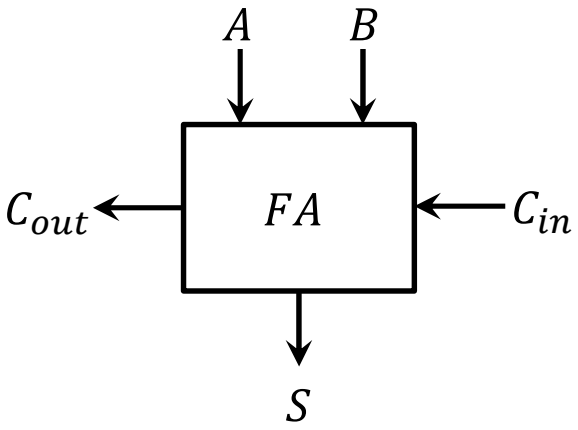


Circuito Somador/Subtrator
Otimização

- Nesta arquitetura, denominada *Ripple Carry Adder* (Somador com Propagação de $Carry$), cada estágio consome um tempo Δt, em função da dependência do $Carry$ produzido pelo estágio anterior;
- A estratégia é buscar uma forma de obtenção do $Carry$ em paralelo com cálculo da soma;
- A nova arquitetura, denominada *Look-Ahead Carry Adder* (Somador com Antecipação do $Carry$), implementa uma lógica que calcula o $Carry$ final em paralelo com as somas de cada estágio;

Circuito Somador/Subtrator
Otimização

- Observando um estágio do $SOMADOR$, temos:

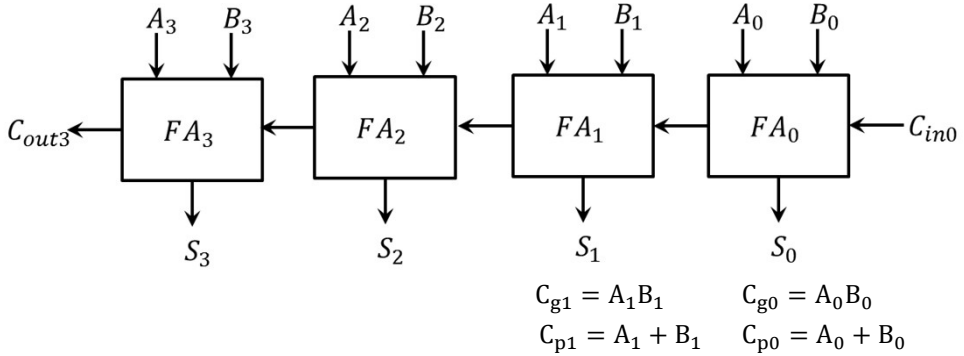


- O C_{out} será 1 em uma destas situações:
  - ❑ Quando A e B forem 1; ou
  - ❑ Quando A ou B for 1, sendo C_{in} também 1;
- Assim, duas variáveis de apoio são criadas:
  - ❑ *Carry generation:* $C_g = AB$
  - ❑ *Carry propagagion:* $C_p = A + B$

$$C_{out} = C_g + C_p C_{in}$$

Circuito Somador/Subtrator
Otimização

Para cada estágio, começando em FA_0, serão calculados os respectivos C_{out}, até alcançarmos o C_{out} final.



- Estágio FA_0

$$C_{outo} = C_{g0} + C_{p0}C_{in0}$$

- Estágio FA_1

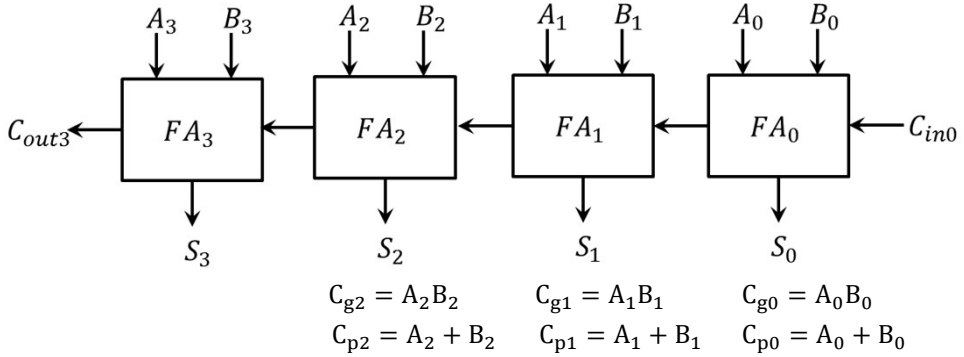
$$C_{in1} = C_{out0}$$

$$C_{out1} = C_{g1} + C_{p1}C_{in1} = C_{g1} + C_{p1}C_{out0} = C_{g1} + C_{p1}(C_{g0} + C_{p0}C_{in0})$$

$$C_{out1} = C_{g1} + C_{p1}C_{g0} + C_{p1}C_{p0}C_{in0}$$

Circuito Somador/Subtrator
Otimização

Para cada estágio, começando em FA_0, serão calculados os respectivos C_{out}, até alcançarmos o C_{out} final.



- Estágio FA_2

$C_{in2} = C_{out1}$

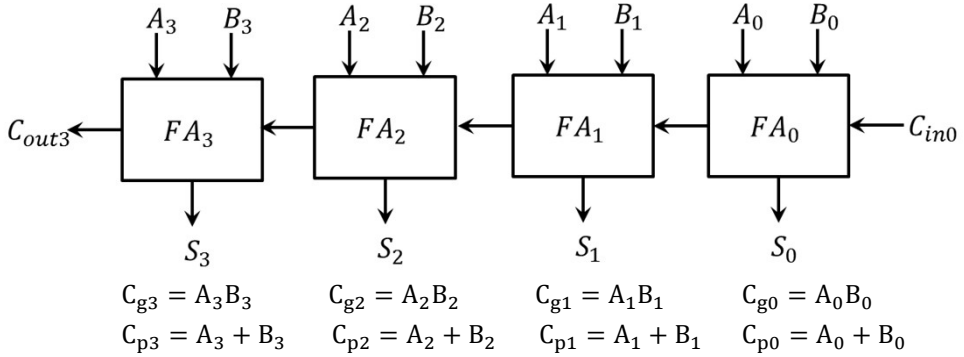
$C_{out1} = C_{g1} + C_{p1}C_{g0} + C_{p1}C_{p0}C_{in0}$

$C_{out2} = C_{g2} + C_{p2}C_{in2} = C_{g2} + C_{p2}C_{out1} = C_{g2} + C_{p2}(C_{g1} + C_{p1}C_{g0} + C_{p1}C_{p0}C_{in0})$

$C_{out2} = C_{g2} + C_{p2}C_{g1} + C_{p2}C_{p1}C_{g0} + C_{p2}C_{p1}C_{p0}C_{in0}$

Circuito Somador/Subtrator
Otimização

Para cada estágio, começando em FA_0, serão calculados os respectivos C_{out}, até alcançarmos o C_{out} final.



- Estágio FA_3

$$C_{in3} = C_{out2}$$

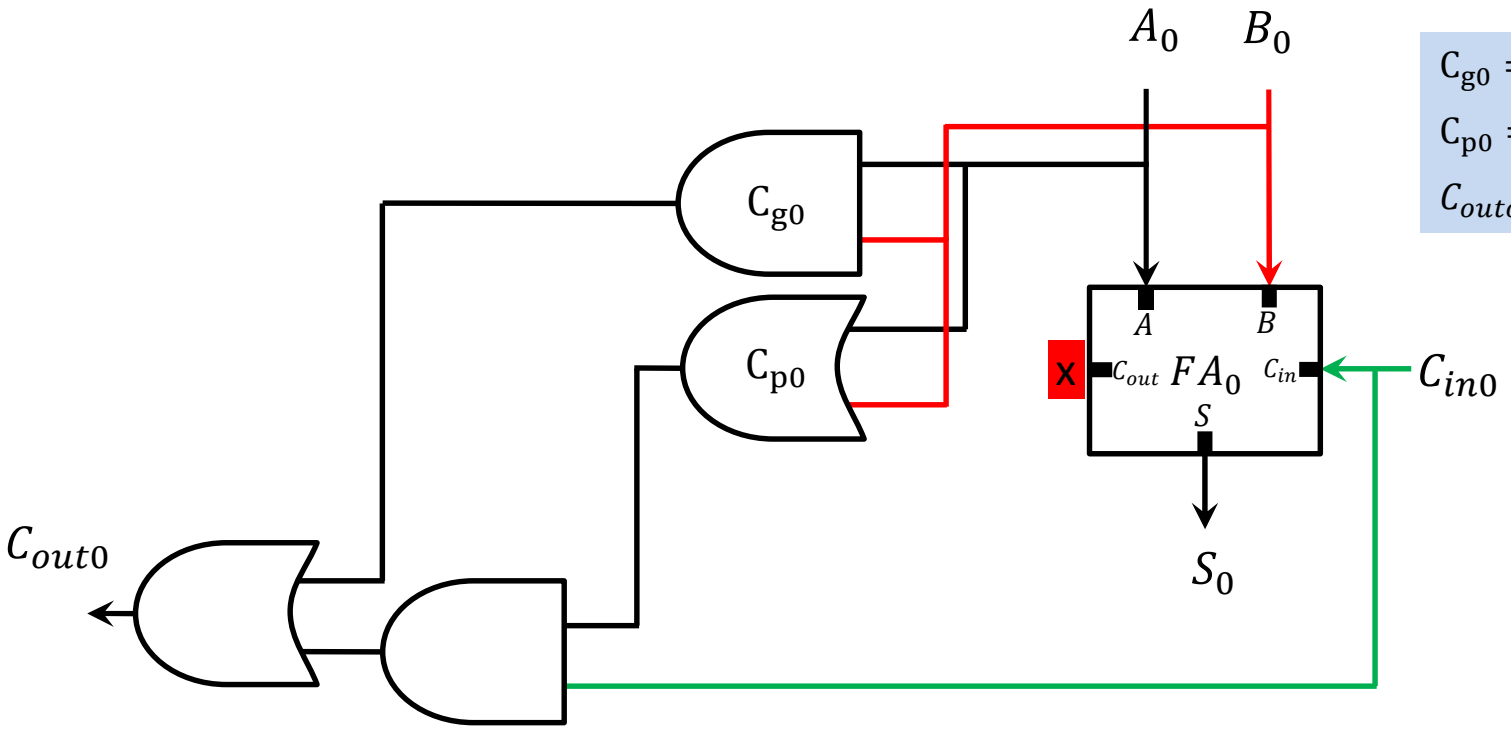
$$C_{out2} = C_{g2} + C_{p2}C_{g1} + C_{p2}C_{p1}C_{g0} + C_{p2}C_{p1}C_{p0}C_{in0}$$

$$C_{out3} = C_{g3} + C_{p3}C_{in3} = C_{g3} + C_{p3}C_{out2}$$

$$C_{out3} = C_{g3} + C_{p3}(C_{g2} + C_{p2}C_{g1} + C_{p2}C_{p1}C_{g0} + C_{p2}C_{p1}C_{p0}C_{in0})$$

$$C_{out3} = C_{g3} + C_{p3}C_{g2} + C_{p3}C_{p2}C_{g1} + C_{p3}C_{p2}C_{p1}C_{g0} + C_{p3}C_{p2}C_{p1}C_{p0}C_{in0}$$

Circuito Somador/Subtrator
Look-Ahead Carry



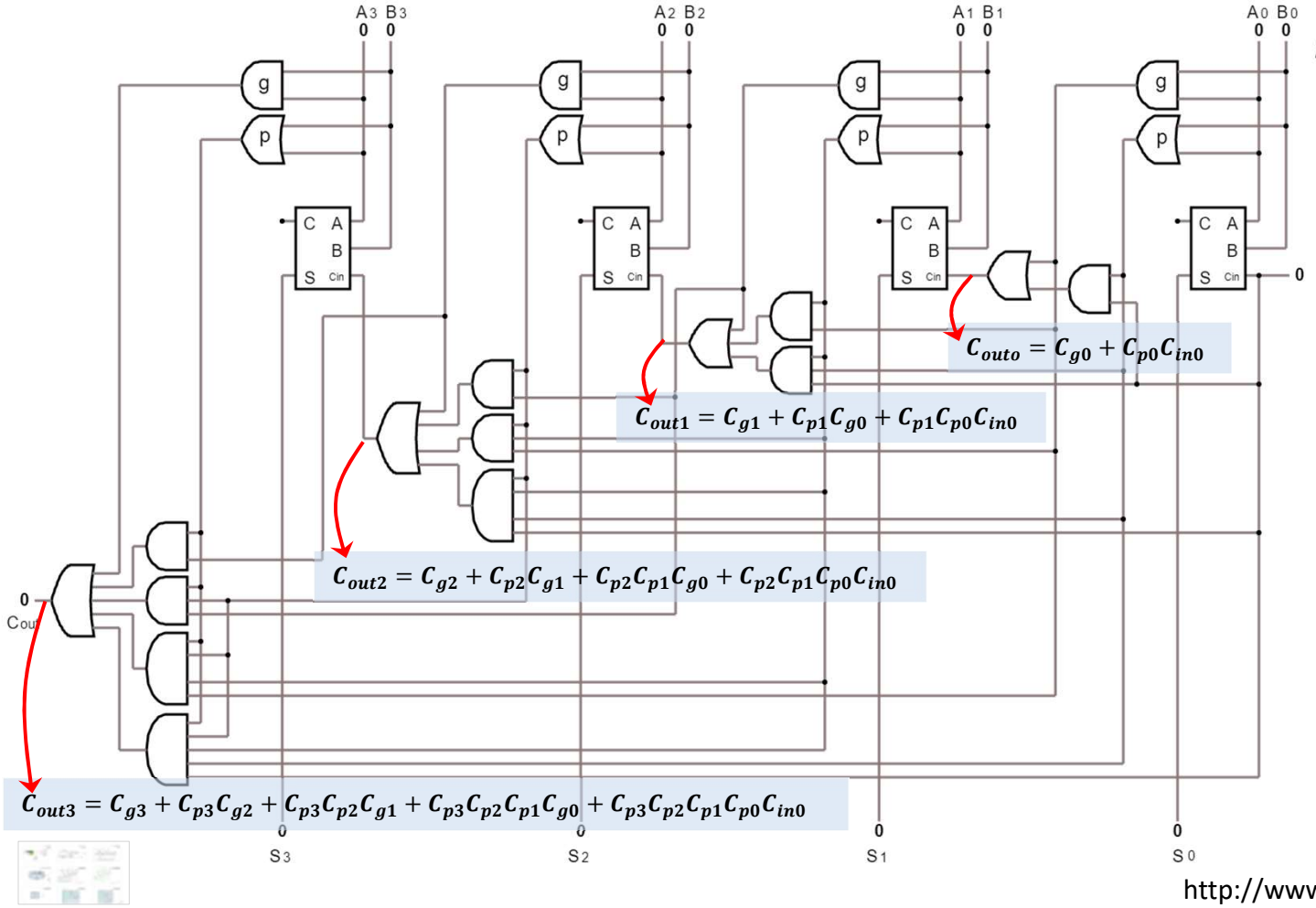
$C_{g0} = A_0 B_0$

$C_{p0} = A_0 + B_0$

$C_{outo} = C_{g0} + C_{p0} C_{in0}$

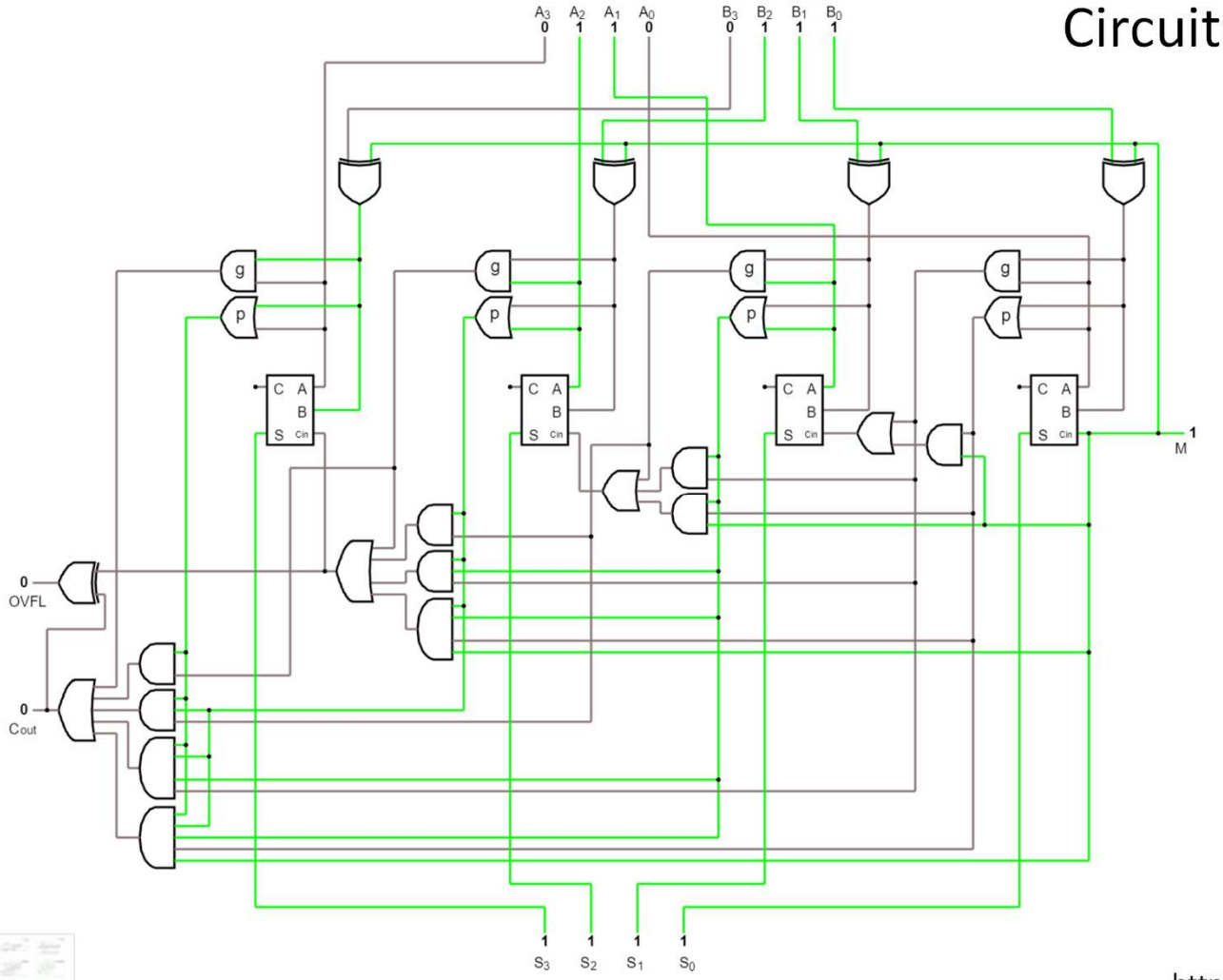
http://www.falstad.com/circuit/

Circuito Somador
Look-Ahead Carry



http://www.falstad.com/circuit/

Circuito Somador/Subtrator
Look-Ahead Carry

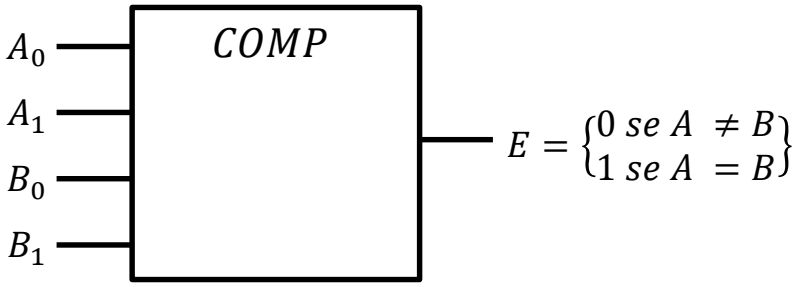


http://www.falstad.com/circuit/

COMPARADOR

Circuito Comparador

- A função de um circuito comparador é comparar duas quantidades binárias para determinar a relação entre elas.
- A funcionalidade mais simples é determinar se dois números são iguais.
- Segue um comparador de igualdade de 2 bits:

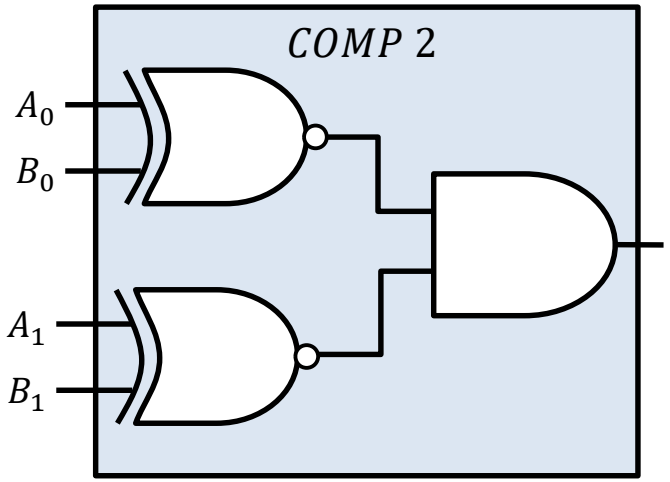


XOR

A	B	X	$\overline{X}$
0	0	0	1
0	1	1	0
1	0	1	0
1	1	0	1

Circuito Comparador

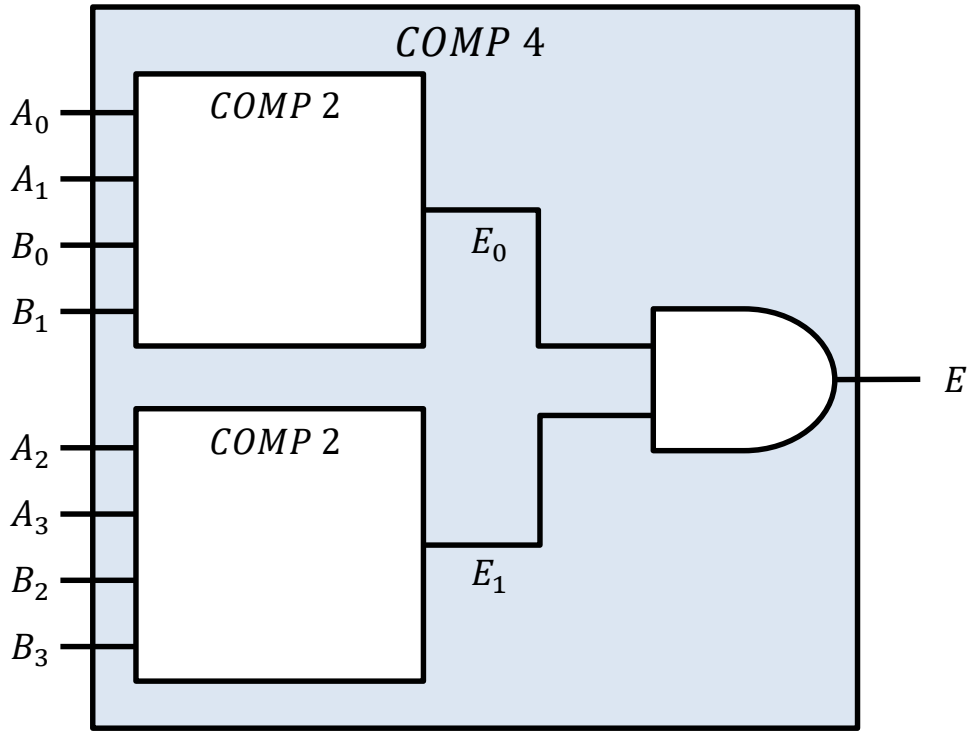
Igualdade - 2 bits



$$E = \begin{cases} 0 \; se \; A \neq B \\ 1 \; se \; A = B \end{cases}$$

Circuito Comparador

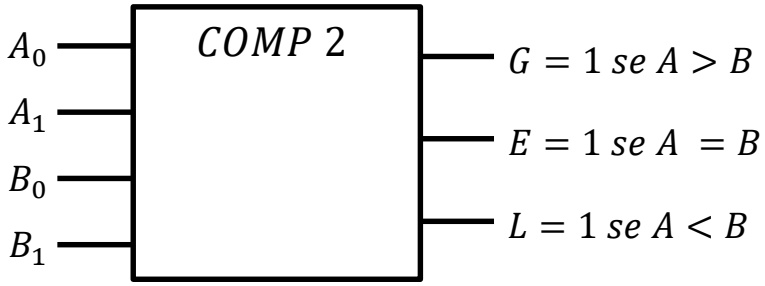
Igualdade - 4 bits



Circuito Comparador

Completo – 2 bits

- Um comparador completo testa dois inteiros A e B, informando se $A = B$, $A < B$ ou $A > B$.
- Segue um comparador completo de 2 bits:

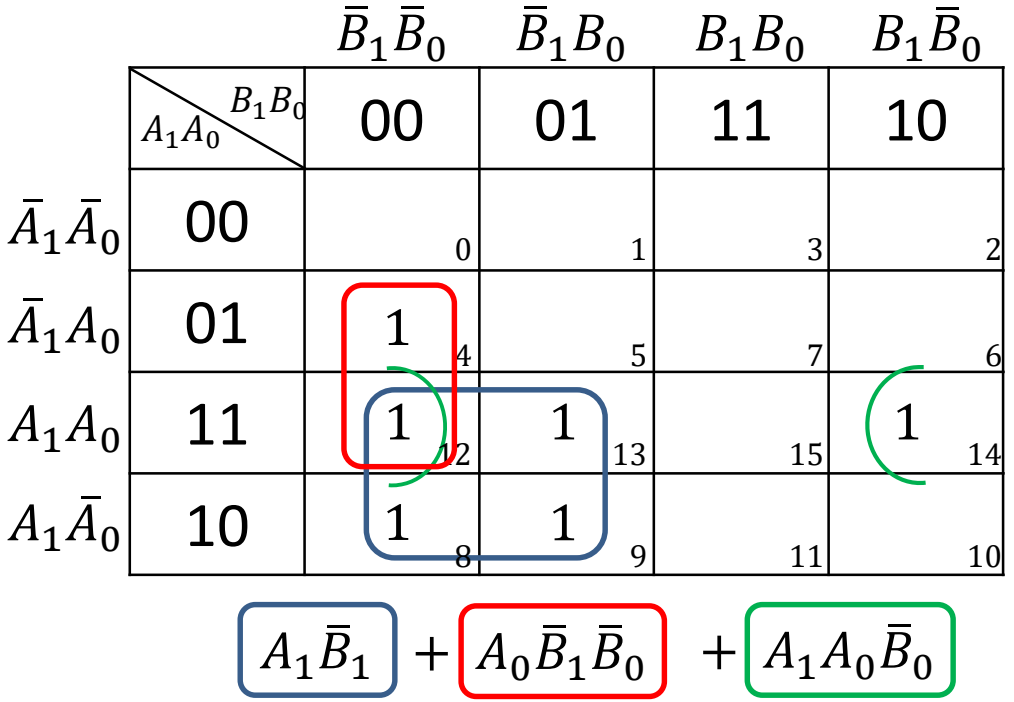


- Se $A_1 > B_1$ então $A > B$, senão se $A_1 < B_1$ então $A < B$, senão, a comparação é repetida para o próximo par de bits.

Circuito Comparador

Completo – 2 bits

A_1	A_0	B_1	B_0	G	E	L
0	0	0	0		1	
0	0	0	1			1
0	0	1	0			1
0	0	1	1			1
0	1	0	0	1		
0	1	0	1		1	
0	1	1	0			1
0	1	1	1			1
1	0	0	0	1		
1	0	0	1	1		
1	0	1	0		1	
1	0	1	1			1
1	1	0	0	1		
1	1	0	1	1		
1	1	1	0	1		
1	1	1	1		1	

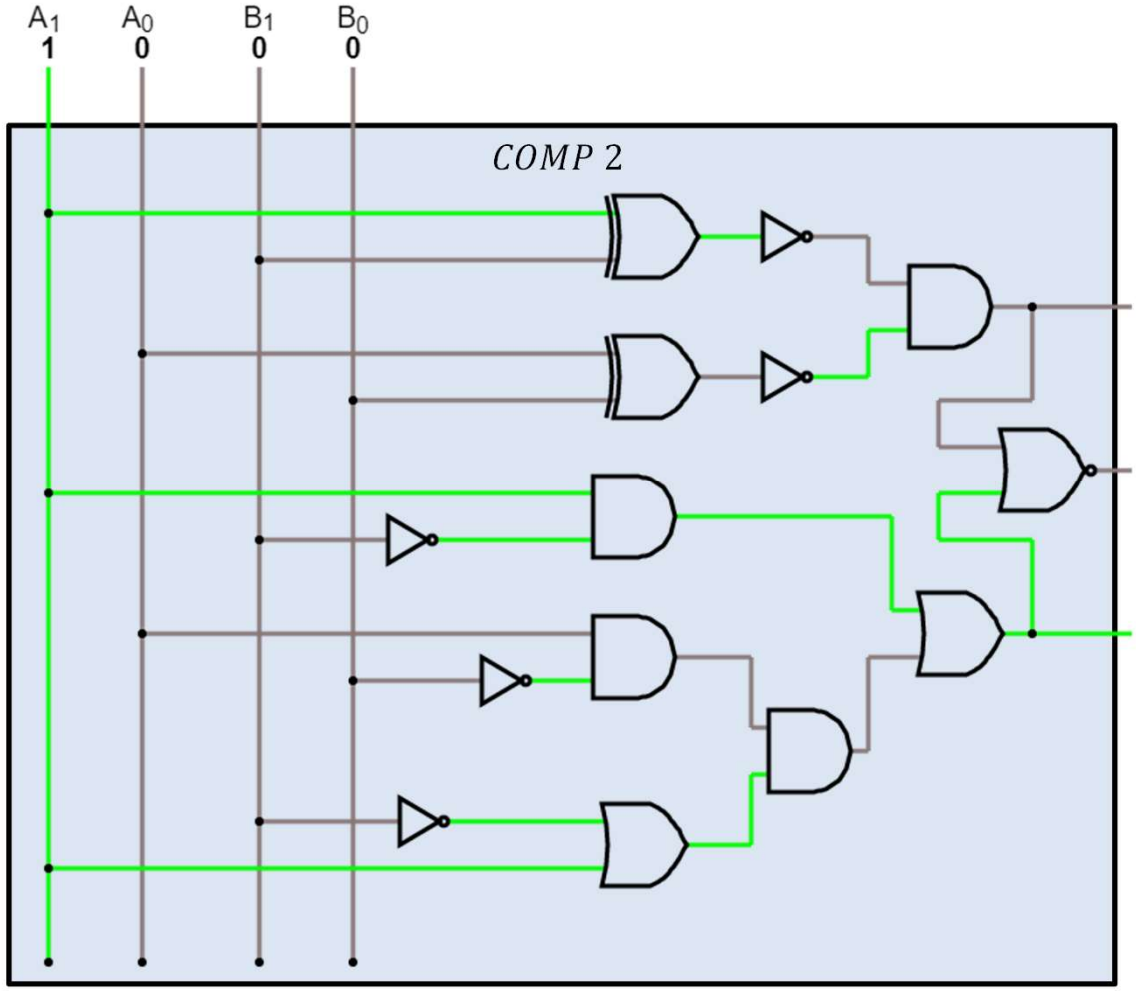


$$A_1\bar{B}_1 + A_0\bar{B}_1\bar{B}_0 + A_1A_0\bar{B}_0$$

$$G = A_1\bar{B}_1 + A_0\,\bar{B}_0(\bar{B}_1 + A_1)$$

Circuito Comparador

Completo – 2 bits



$$E = \overline{A_0 \oplus B_0} \cdot \overline{A_1 \oplus B_1}$$

$$G = A_1 \bar{B}_1 + A_0 \bar{B}_0 (\bar{B}_1 + A_1)$$

Circuito Comparador

Completo – 2 bits

A_1	A_0	B_1	B_0	G	E	L
0	0	0	0		1	
0	0	0	1			1
0	0	1	0			1
0	0	1	1			1
0	1	0	0	1		
0	1	0	1		1	
0	1	1	0			1
0	1	1	1			1
1	0	0	0	1		
1	0	0	1	1		
1	0	1	0		1	
1	0	1	1			1
1	1	0	0	1		
1	1	0	1	1		
1	1	1	0	1		
1	1	1	1		1	

		$\bar{B}_1\bar{B}_0$	$\bar{B}_1B_0$	B_1B_0	$B_1\bar{B}_0$
	A_1A_0 \ B_1B_0	00	01	11	10
$\bar{A}_1\bar{A}_0$	00	0	1	3	2
$\bar{A}_1A_0$	01	1 (4)	5	7	6
A_1A_0	11	1 (12)	1 (13)	15	1 (14)
$A_1\bar{A}_0$	10	1 (8)	1 (9)	11	10

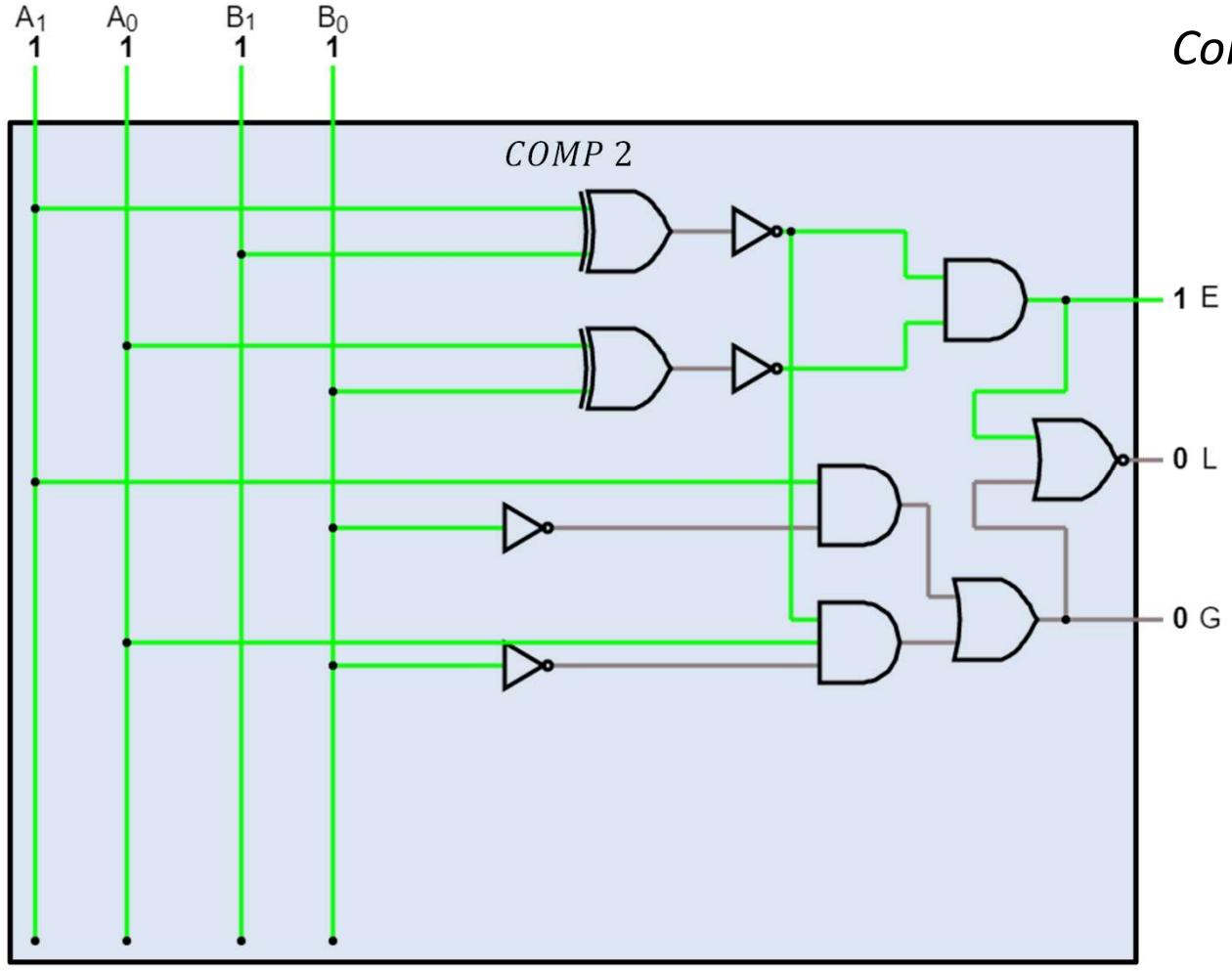
$$A_1\bar{B}_1 + \bar{A}_1A_0\bar{B}_1\bar{B}_0 + A_1A_0B_1\bar{B}_0$$

$$G = A_1\bar{B}_1 + A_0\,\bar{B}_0(\bar{A}_1\bar{B}_1 + A_1B_1)$$

$$G = A_1\bar{B}_1 + A_0\,\bar{B}_0(\overline{A_1 \oplus B_1})$$

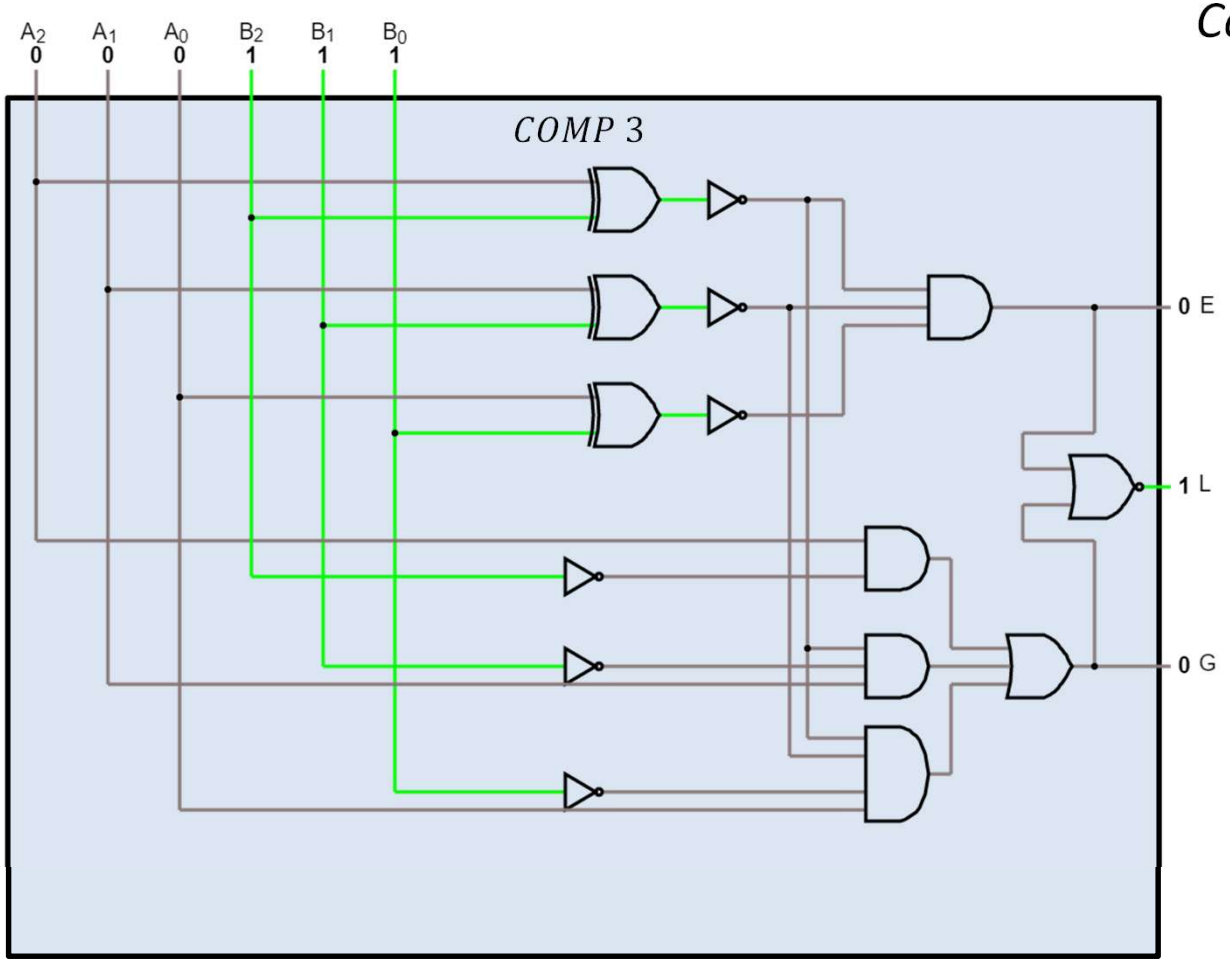
Circuito Comparador

Completo – 2 bits



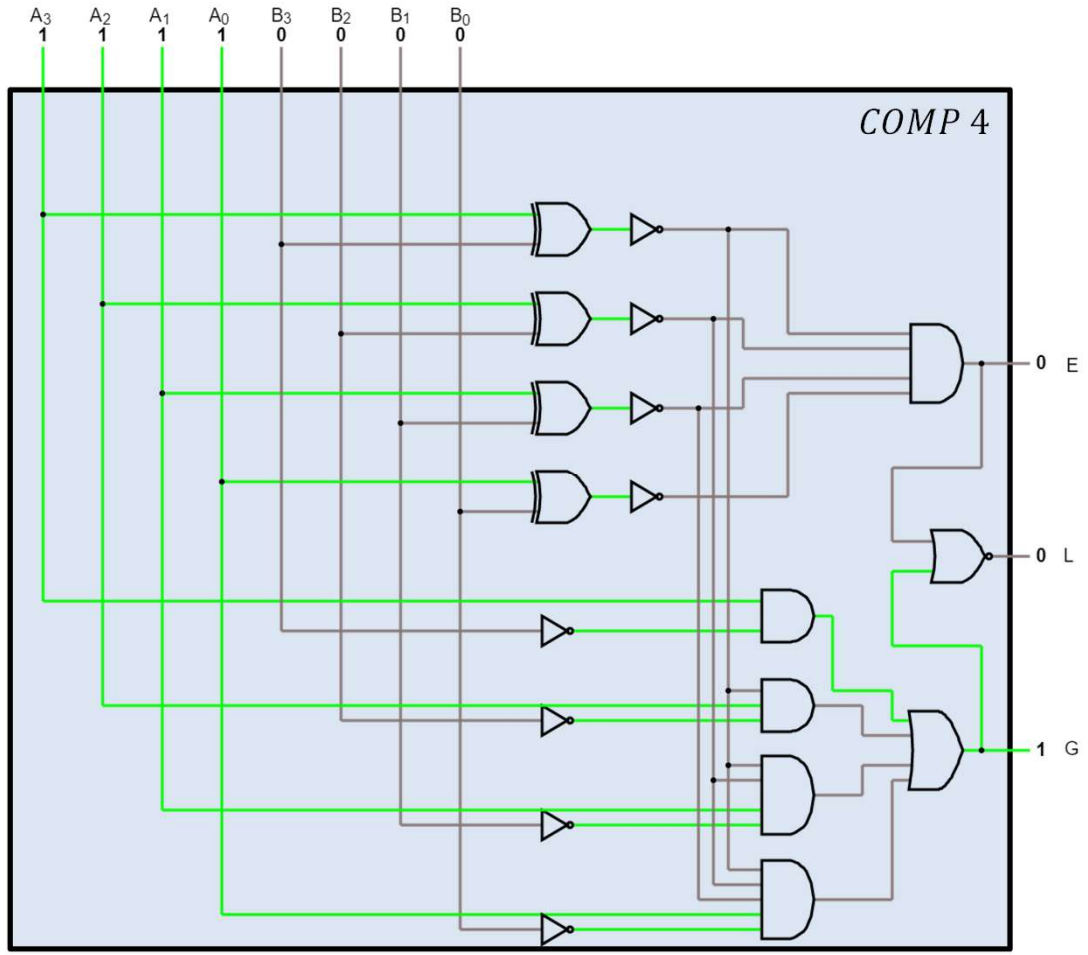
Circuito Comparador

Completo – 3 bits



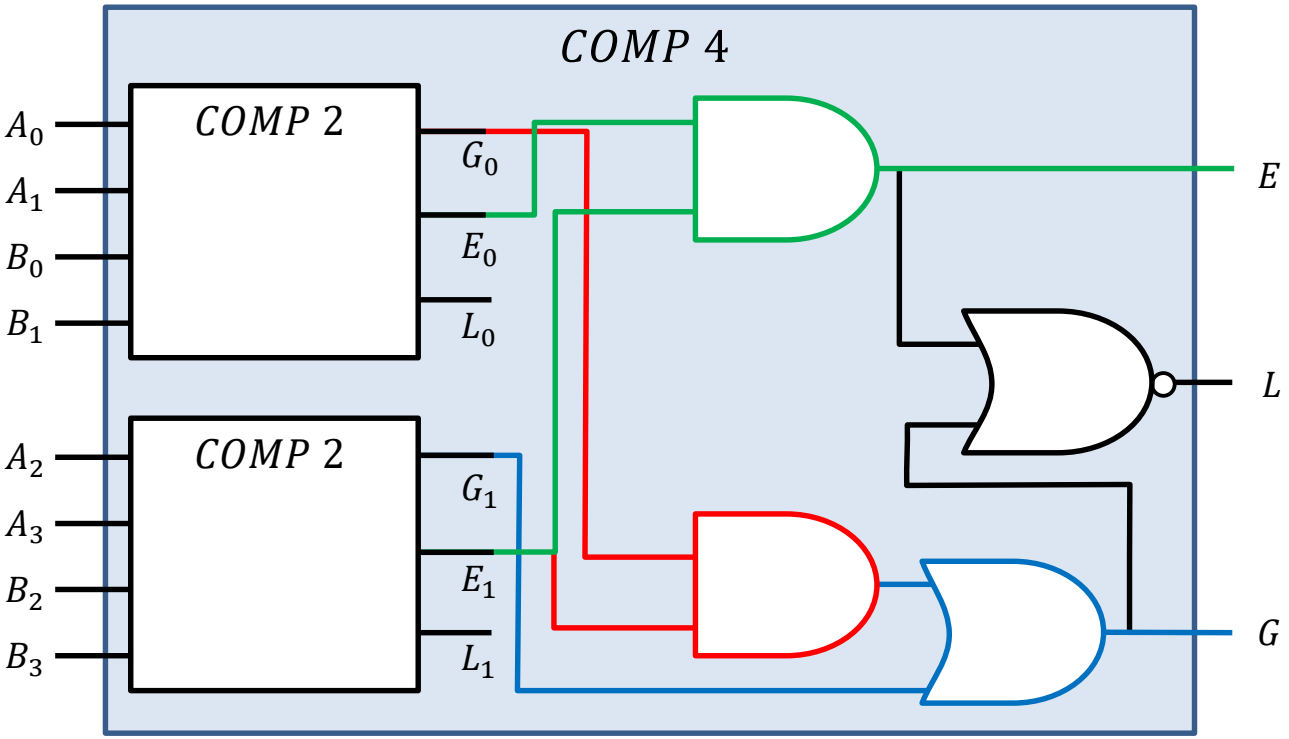
Circuito Comparador

Completo – 4 bits



Circuito Comparador

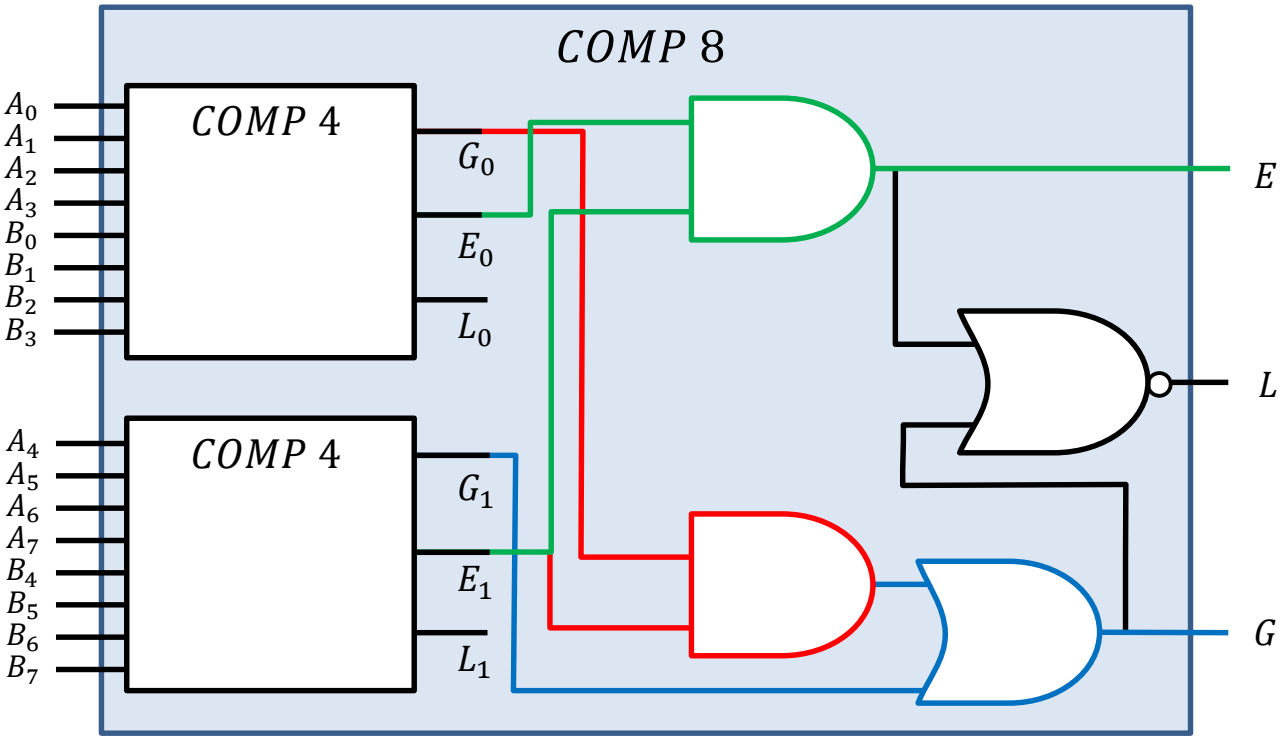
Completo – 4 bits



$$G = G_1 + E_1 G_0$$

Circuito Comparador

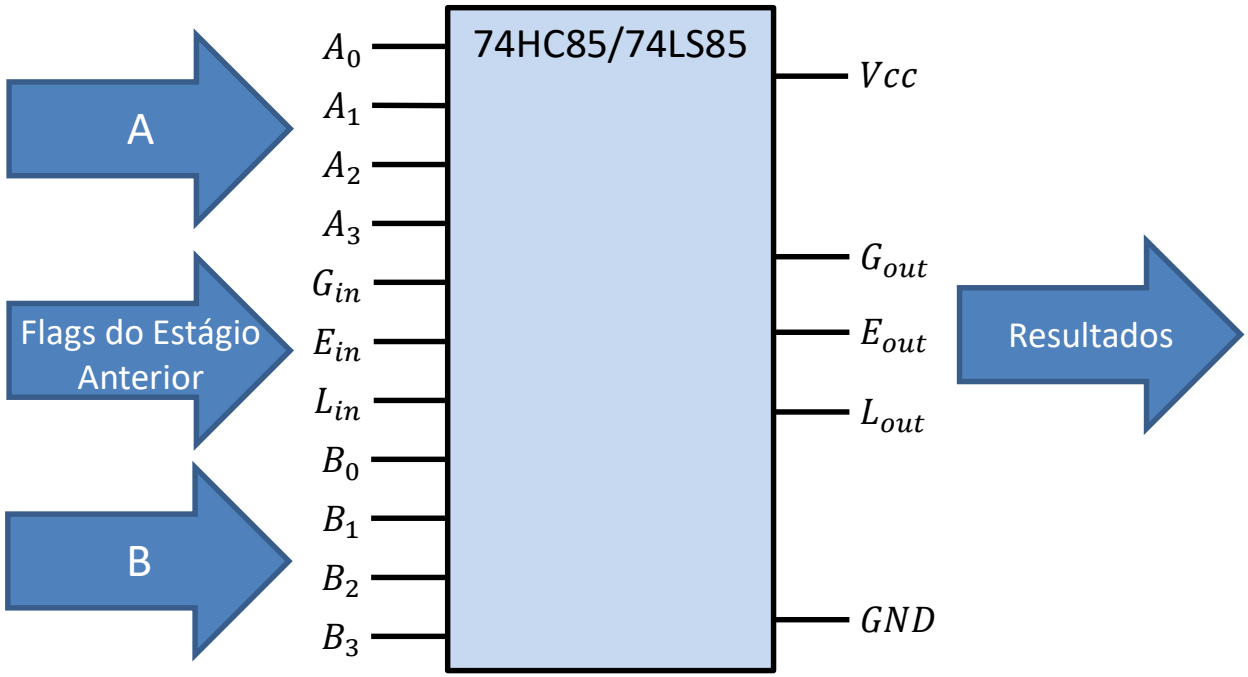
Completo – 8 bits



$$G = G_1 + E_1 G_0$$

Circuito Comparador

Completo – 4 bits com Cascata

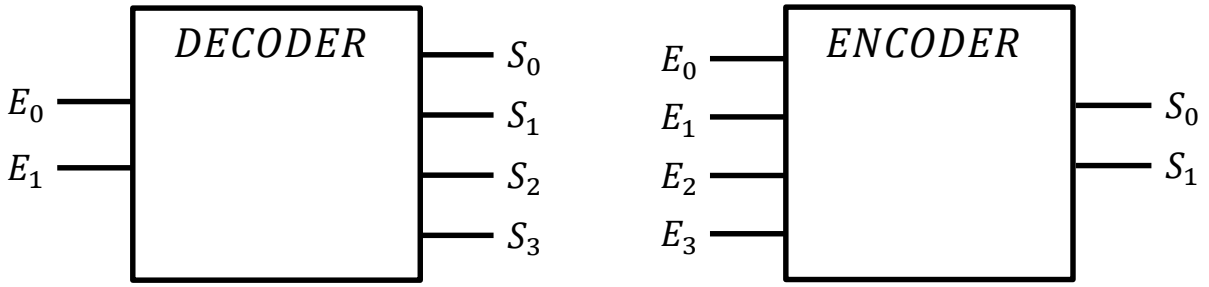


DECODER

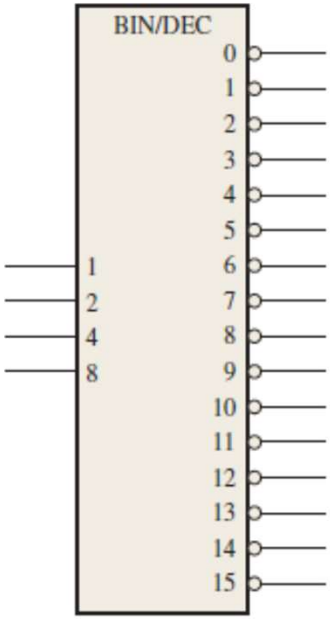
Decoder

- Codificador é um circuito que mapeia um conjunto de entradas em um conjunto de saídas de acordo com uma determinada lógica de codificação
- É um circuito que transforma uma informação de um formato para outro;

  ✓ Decodificador $n \Rightarrow 2^n$
  ✓ Codificador $2^n \Rightarrow n$



Decoder

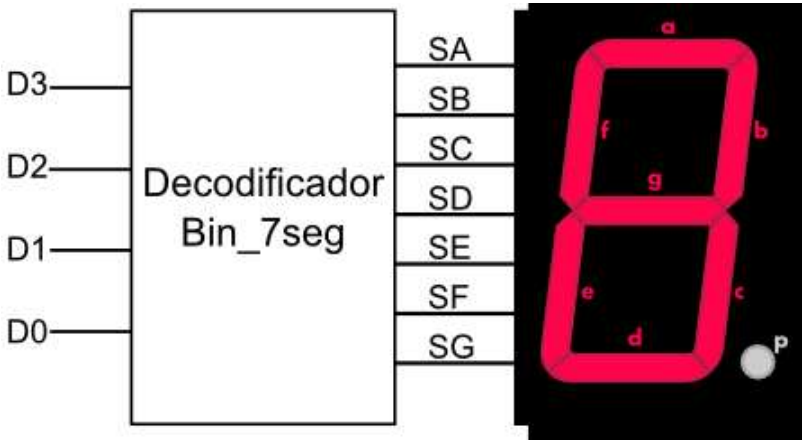


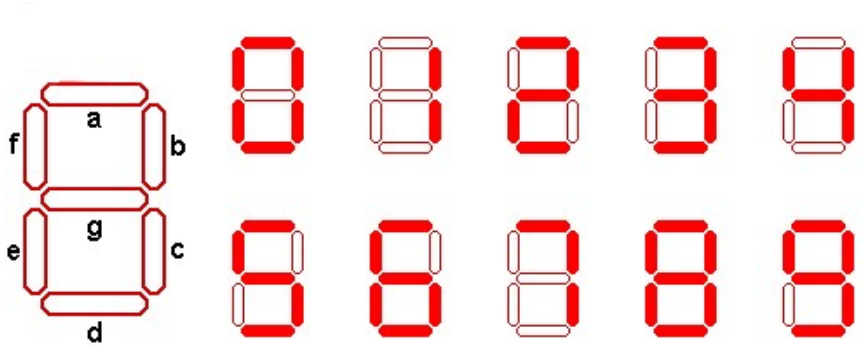
Decimal Digit	Binary Inputs A_3	A_2	A_1	A_0	Decoding Function	Outputs 0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	0	0	0	0	$\overline{A}_3\overline{A}_2\overline{A}_1\overline{A}_0$	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	0	0	0	1	$\overline{A}_3\overline{A}_2\overline{A}_1A_0$	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1
2	0	0	1	0	$\overline{A}_3\overline{A}_2A_1\overline{A}_0$	1	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1
3	0	0	1	1	$\overline{A}_3\overline{A}_2A_1A_0$	1	1	1	0	1	1	1	1	1	1	1	1	1	1	1	1
4	0	1	0	0	$\overline{A}_3A_2\overline{A}_1\overline{A}_0$	1	1	1	1	0	1	1	1	1	1	1	1	1	1	1	1
5	0	1	0	1	$\overline{A}_3A_2\overline{A}_1A_0$	1	1	1	1	1	0	1	1	1	1	1	1	1	1	1	1
6	0	1	1	0	$\overline{A}_3A_2A_1\overline{A}_0$	1	1	1	1	1	1	0	1	1	1	1	1	1	1	1	1
7	0	1	1	1	$\overline{A}_3A_2A_1A_0$	1	1	1	1	1	1	1	0	1	1	1	1	1	1	1	1
8	1	0	0	0	$A_3\overline{A}_2\overline{A}_1\overline{A}_0$	1	1	1	1	1	1	1	1	0	1	1	1	1	1	1	1
9	1	0	0	1	$A_3\overline{A}_2\overline{A}_1A_0$	1	1	1	1	1	1	1	1	1	0	1	1	1	1	1	1
10	1	0	1	0	$A_3\overline{A}_2A_1\overline{A}_0$	1	1	1	1	1	1	1	1	1	1	0	1	1	1	1	1
11	1	0	1	1	$A_3\overline{A}_2A_1A_0$	1	1	1	1	1	1	1	1	1	1	1	0	1	1	1	1
12	1	1	0	0	$A_3A_2\overline{A}_1\overline{A}_0$	1	1	1	1	1	1	1	1	1	1	1	1	0	1	1	1
13	1	1	0	1	$A_3A_2\overline{A}_1A_0$	1	1	1	1	1	1	1	1	1	1	1	1	1	0	1	1
14	1	1	1	0	$A_3A_2A_1\overline{A}_0$	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	1
15	1	1	1	1	$A_3A_2A_1A_0$	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0

Decoder

Display de 7 Segmentos

Exemplo: Display de 7 segmentos: Função para o segmento a.





$$f_a = \sum(0,2,3,5,6,7,8,9)$$

Decoder

Display de 7 Segmentos

Exemplo: Como agrupar e qual a função? $f_a = \sum(0,2,3,5,6,7,8,9)$

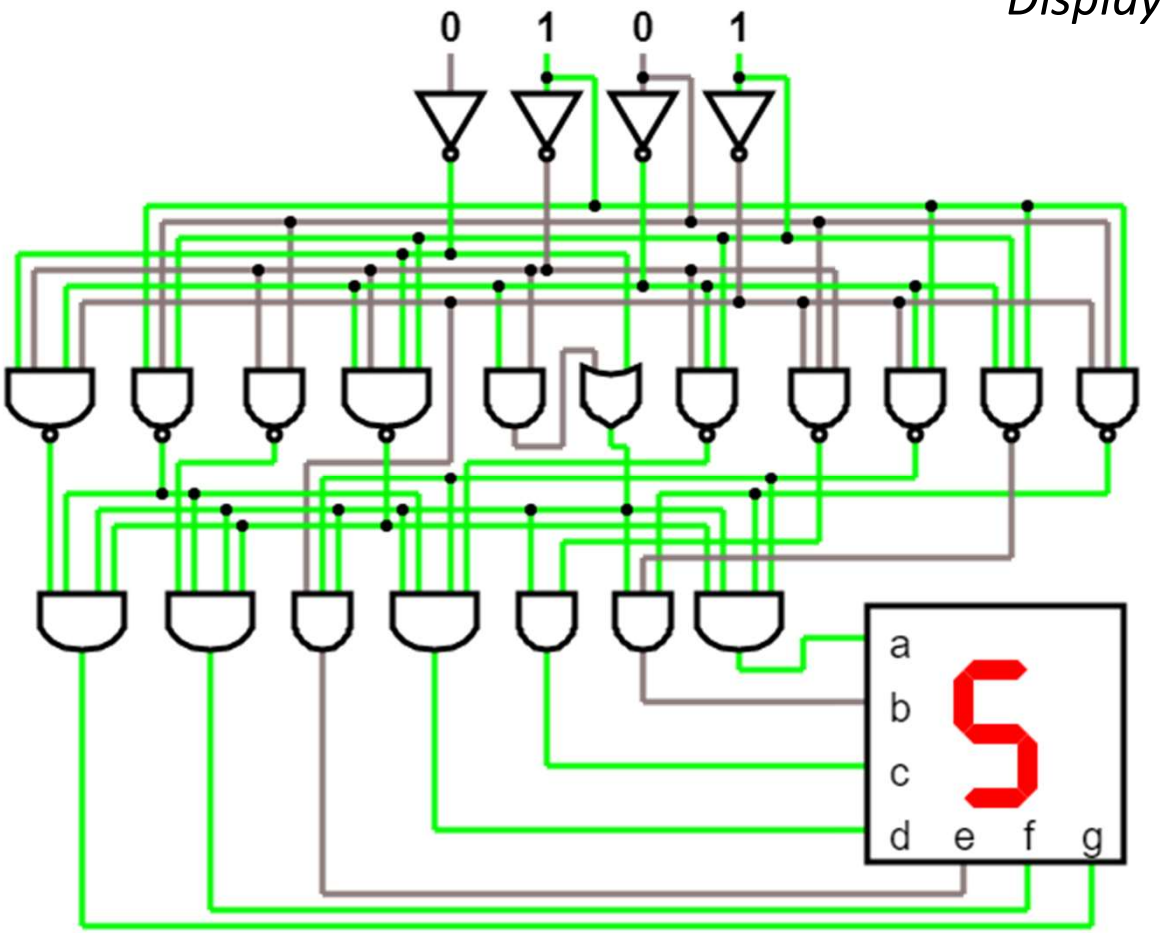
AB \ CD	$\bar{C}\bar{D}$ 00	$\bar{C}D$ 01	CD 11	$C\bar{D}$ 10
$\bar{A}\bar{B}$ 00	1 (0)	(1)	1 (3)	1 (2)
$\bar{A}B$ 01	(4)	1 (5)	1 (7)	1 (6)
AB 11	X (12)	X (13)	X (15)	X (14)
$A\bar{B}$ 10	1 (8)	1 (9)	X (11)	X (10)

$A + C + BD + \bar{B}\bar{D}$

$\boldsymbol{X} = Don't\ Care$

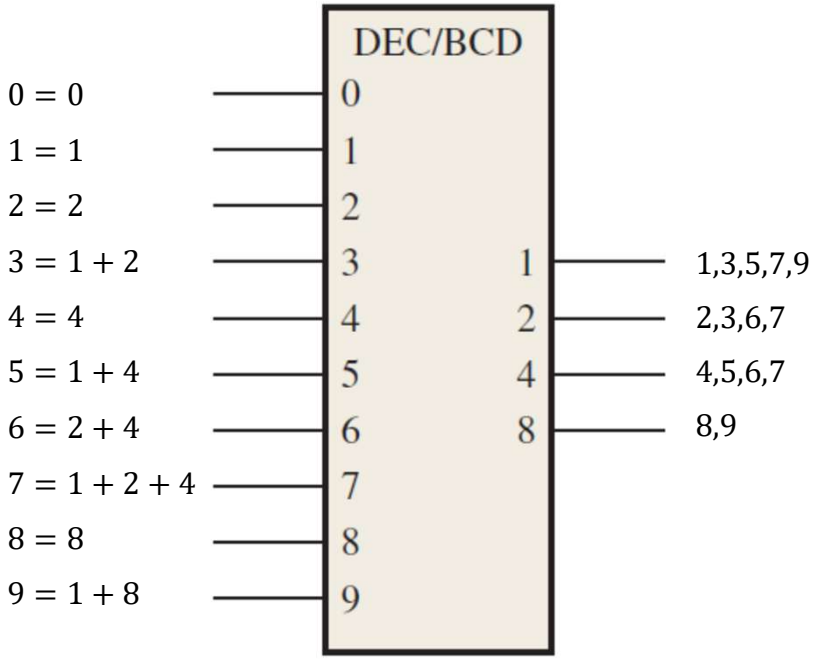
Decoder

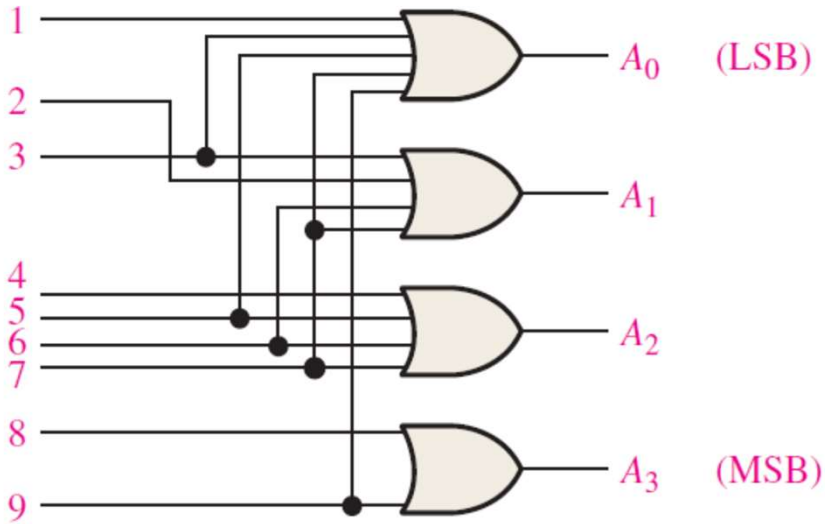
Display de 7 Segmentos



ENCODER

Encoder

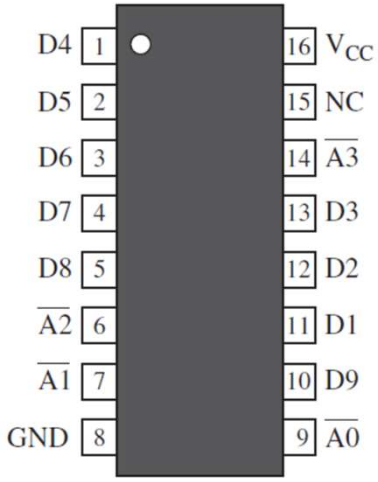


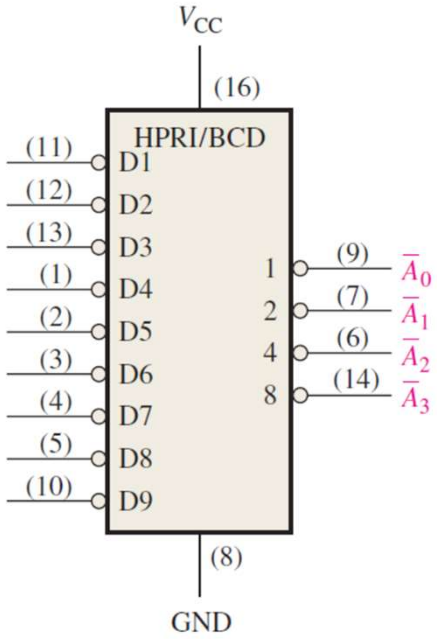


Considerando que apenas uma entrada será ativada.

Encoder

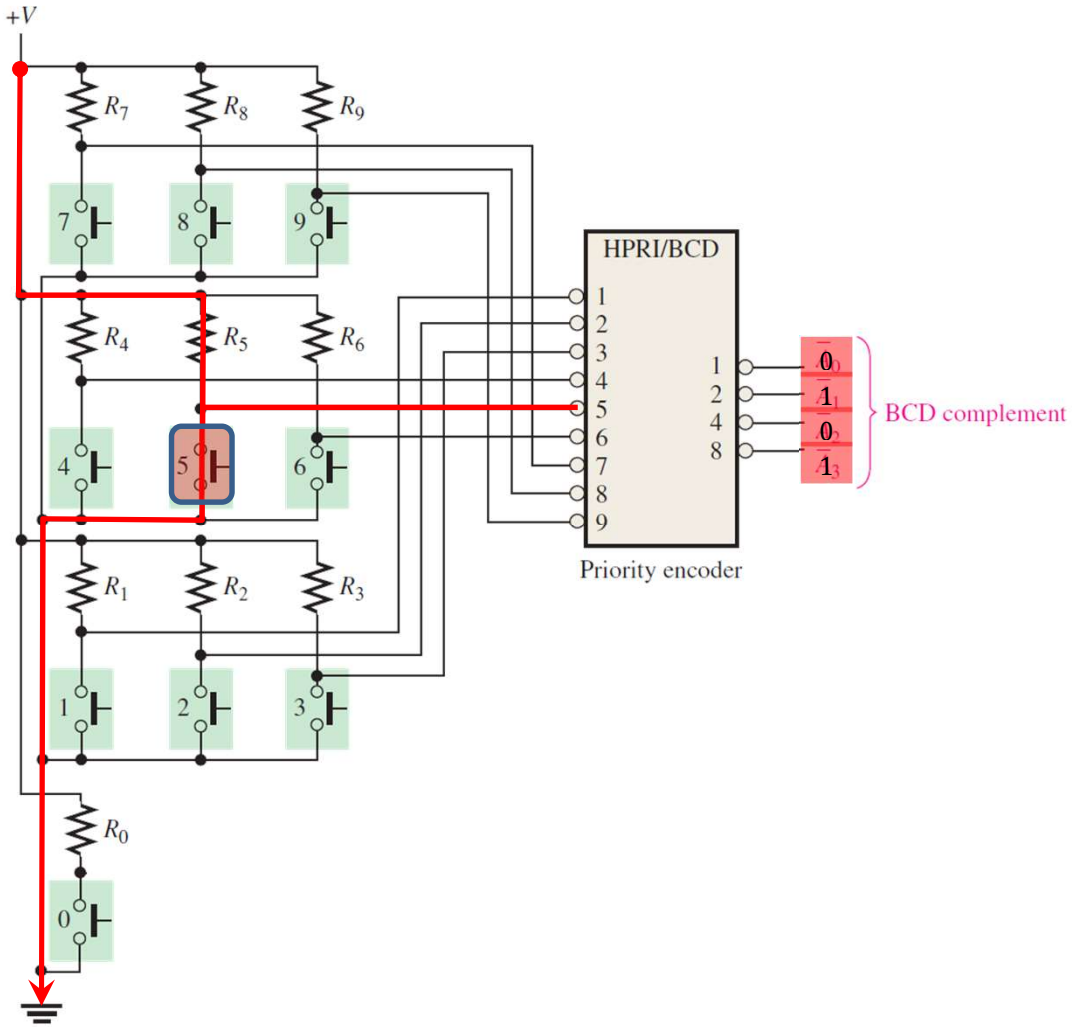
- O $74HC147$ é um codificador com prioridade e entradas ativas em LOW (0).
- As saídas são codificações em BCD, ativas em LOW.

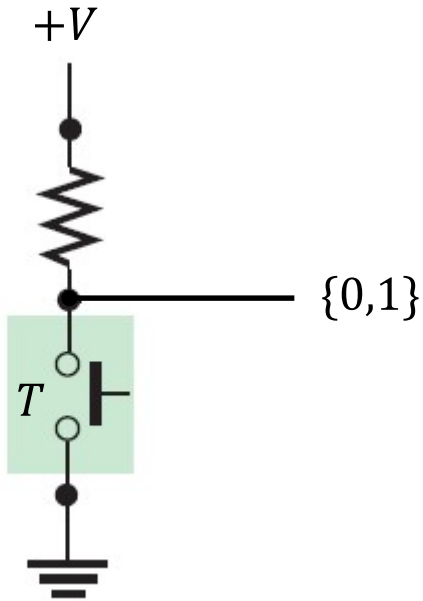




Havendo mais de uma entrada ativa em LOW, a codificação na saída será do maior valor.

Encoder

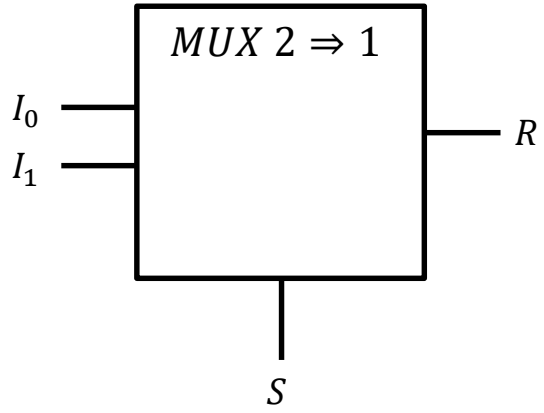

+V
R7
R8
R9
7
8
9
R4
R5
R6
4
5
6
R1
R2
R3
1
2
3
R0
0
HPRI/BCD
1
2
3
4
5
6
7
8
9
1
2
4
8
BCD complement
Priority encoder

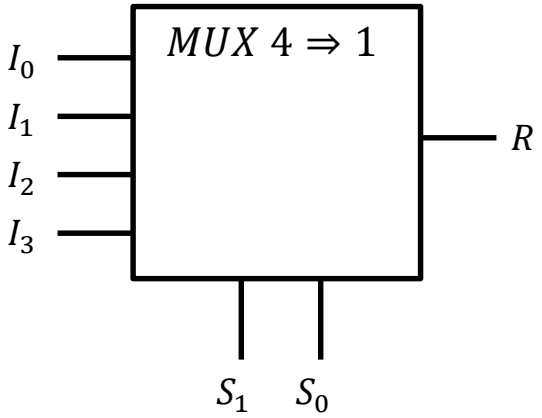

+V
T
{0,1}

MULTIPLEXADOR

Multiplexador

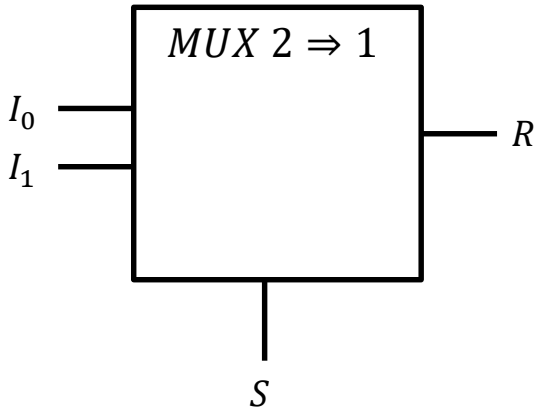
- Multiplexador é um circuito que seleciona, a partir de uma entrada específica, um dos sinais de entrada a ser transferido para a saída;
- Também referenciado como Multiplex, Multiplexer ou MUX.





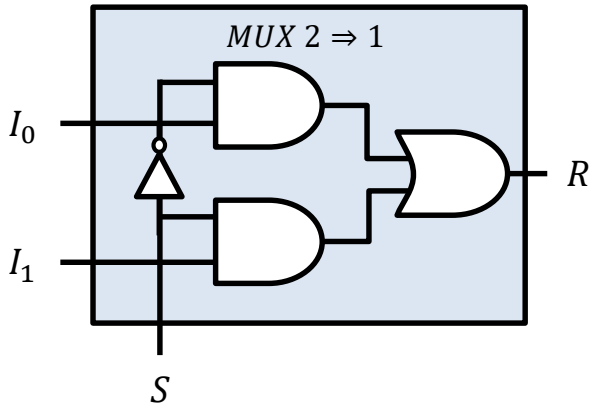
Multiplexador

$MUX\ 2 \Rightarrow 1$



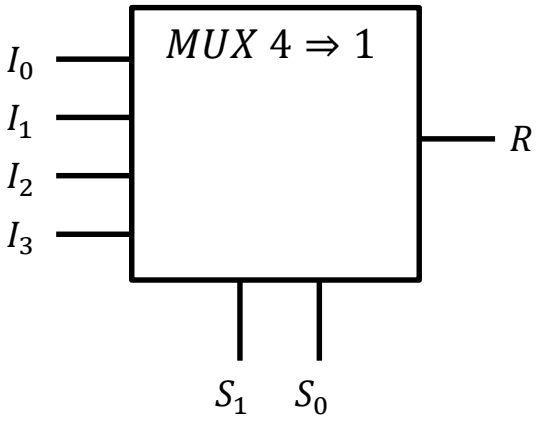
S	R
0	I_0
1	I_1

$$R = \bar{S}I_0 + SI_1$$

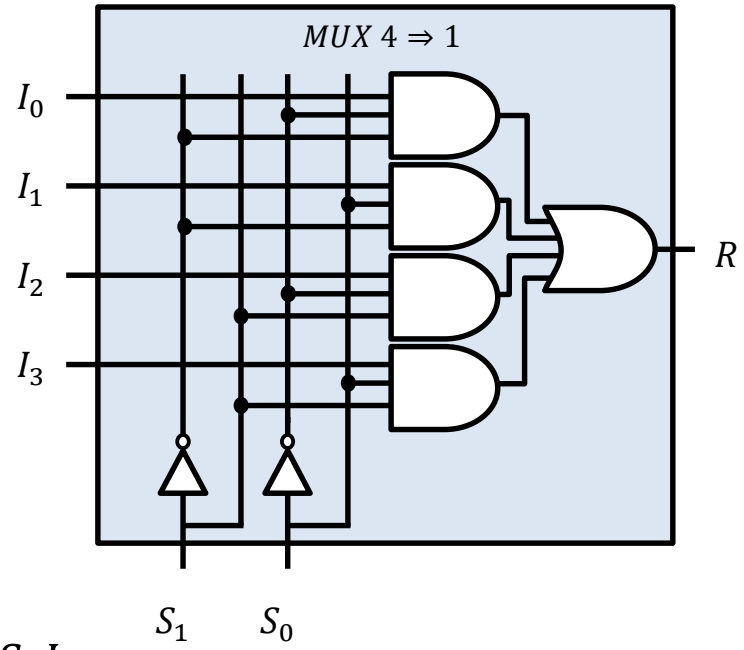


Multiplexador

$MUX\ 4 \Rightarrow 1$



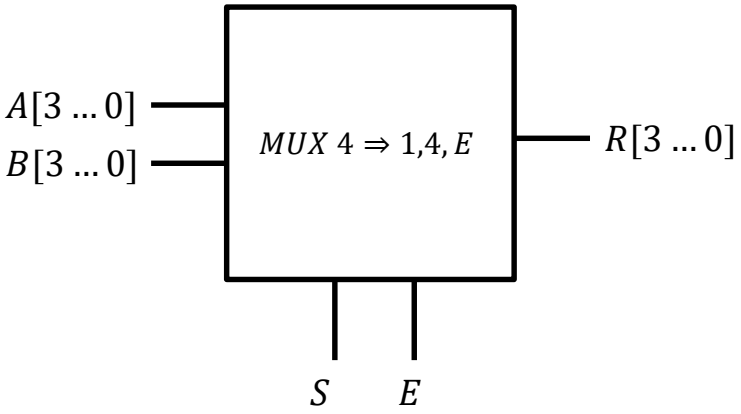
S_1	S_0	R
0	0	I_0
0	1	I_1
1	0	I_2
1	1	I_3



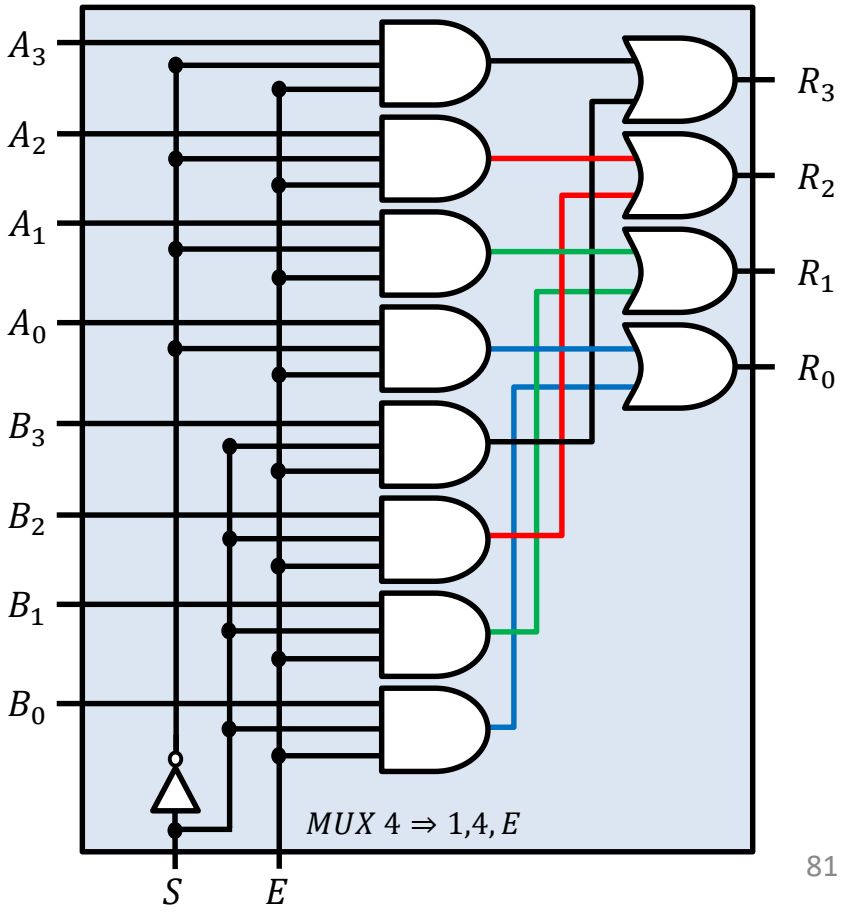
$$R = \bar{S}_1\bar{S}_0I_0 + \bar{S}_1S_0I_1 + S_1\bar{S}_0I_2 + S_1S_0I_3$$

Multiplexador

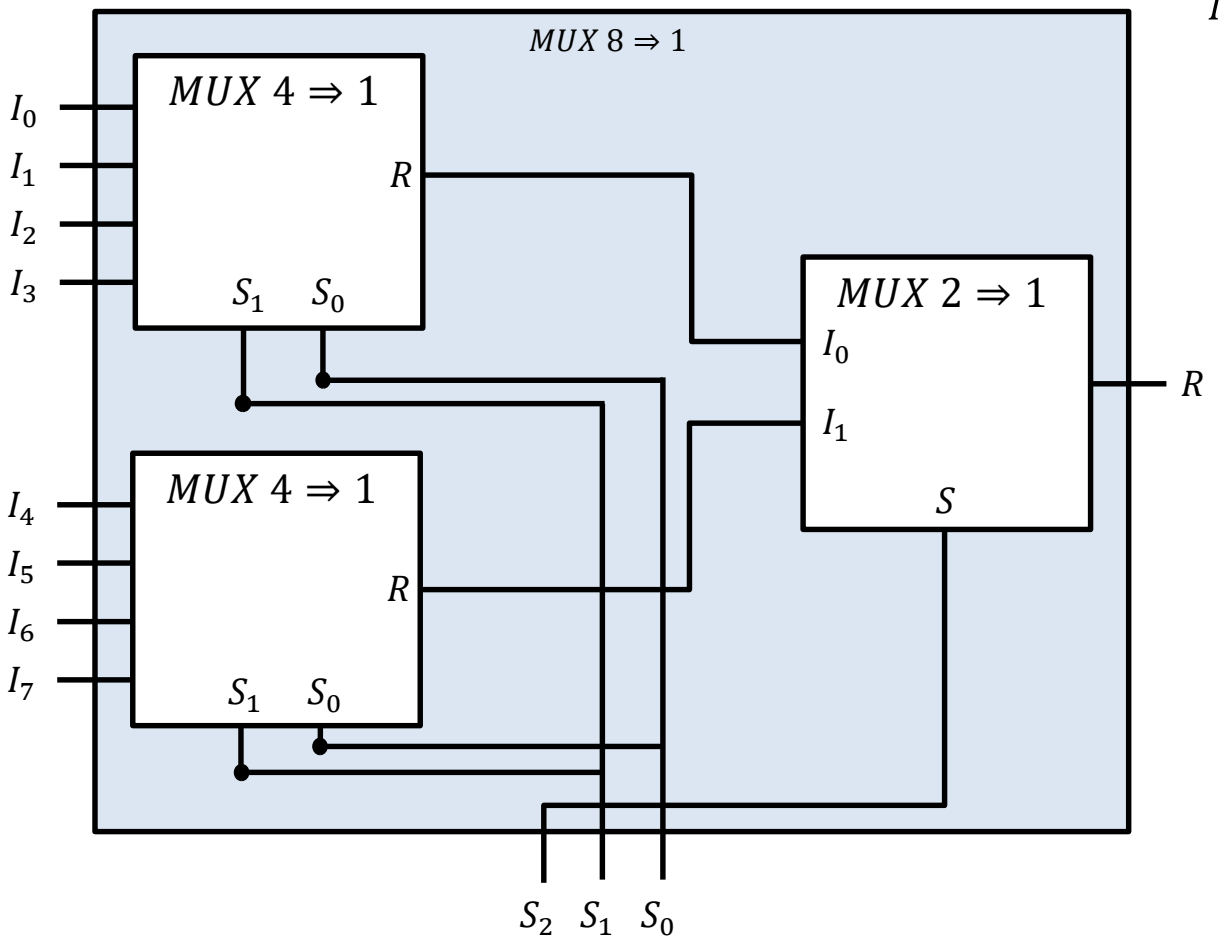
$MUX\ 2 \Rightarrow 1\ -4\ bits\ e\ Enable$



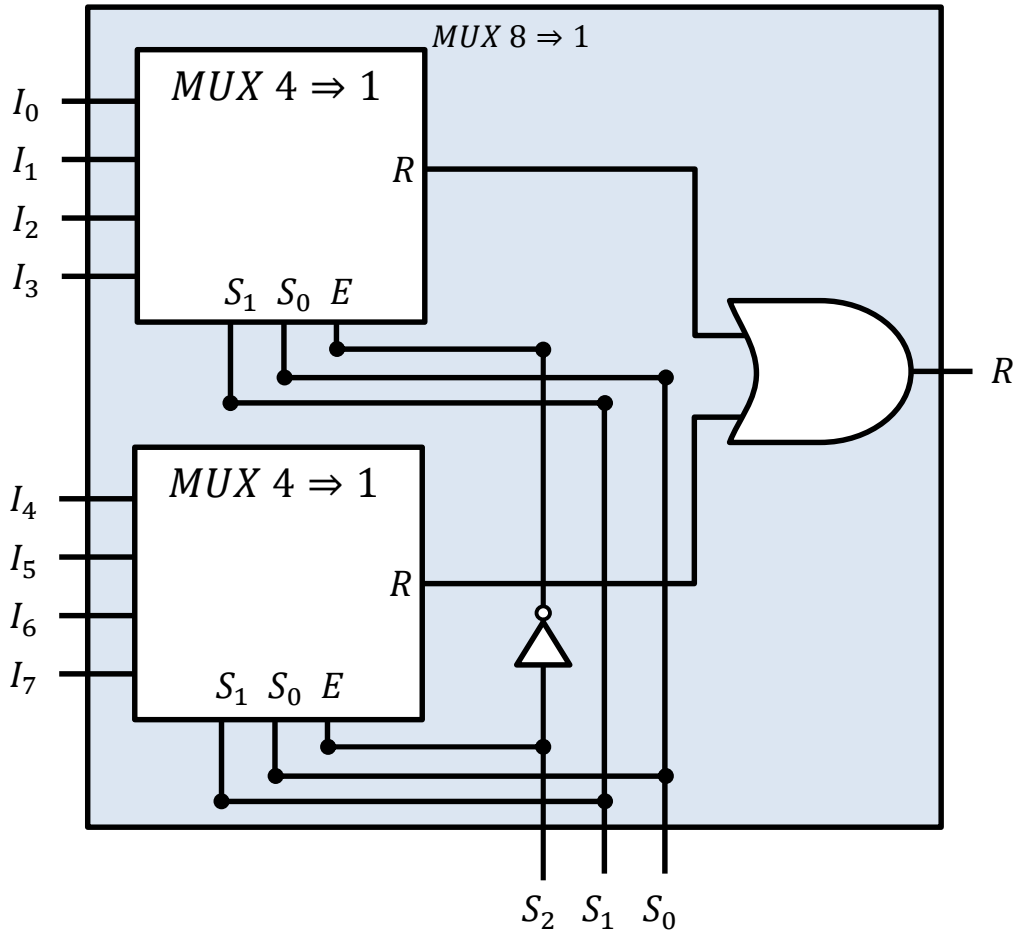
$$R = E\bar{S}A + ESB$$



Multiplexador
$MUX\ 8 \Rightarrow 1$



Multiplexador
$MUX\ 8 \Rightarrow 1$

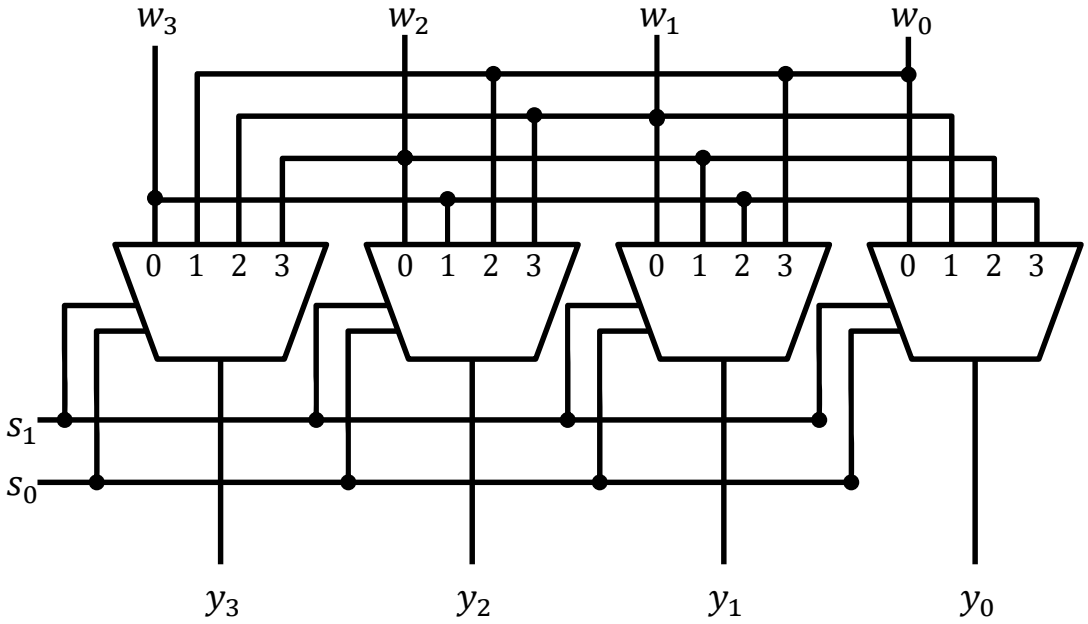


EXERCÍCIOS

Multiplexador

Exercício

1. O circuito abaixo é composto por 4 multiplexadores. As entradas são: uma palavra de dados $w[3 \ldots 0]$ e dois bits de controle ($s_1 - s_0$). A saída é o vetor $y[3 \ldots 0]$. Preencha a tabela verdade correspondente a este circuito, e interprete o que este circuito realiza. No preenchimento da tabela verdade utilizar os valores $w_3, w_2, w_1 \; ou \; w_0$.

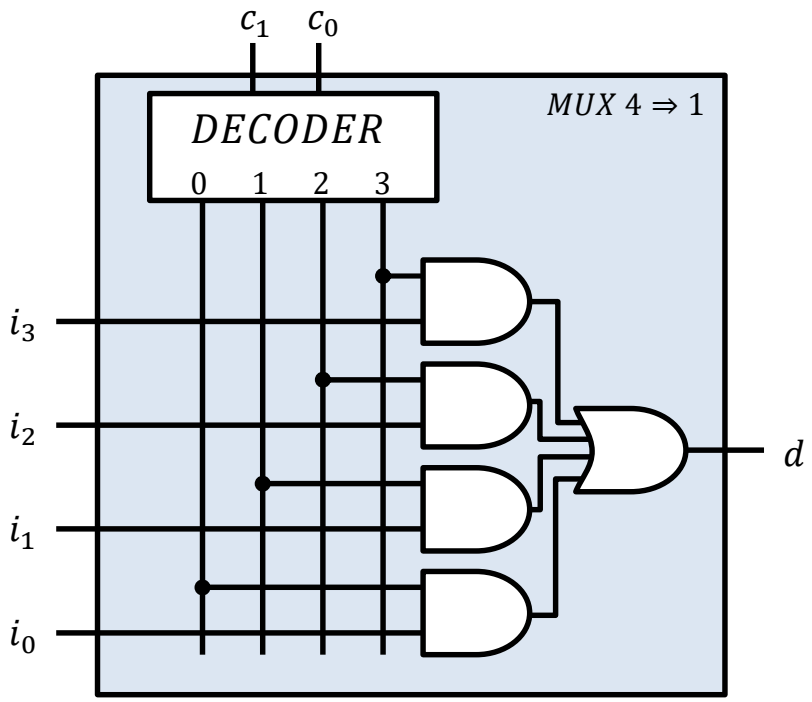


s_1	s_0	y_3	y_2	y_1	y_0
0	0	w_3	w_2	w_1	w_0
0	1	w_0	w_3	w_2	w_1
1	0	w_1	w_0	w_3	w_2
1	1	w_2	w_1	w_0	w_3

Multiplexador

Exercício

2. Determine a saída d em função de c_1 e c_0.



c_1	c_0	d
0	0	i_0
0	1	i_1
1	0	i_2
1	1	i_3

Multiplexador

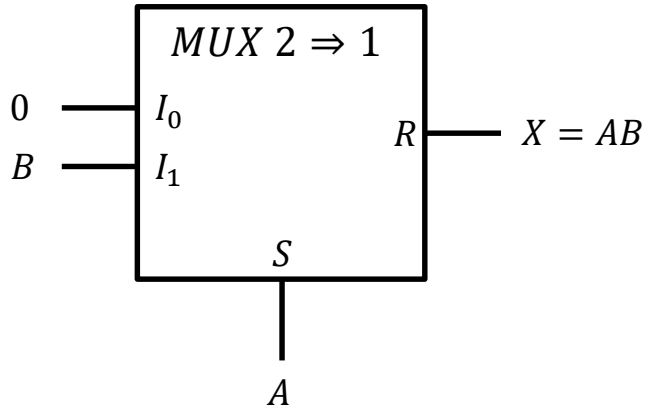
Exercício

3. Implementar as seguintes funções lógicas:

a) $Y = AB$

A	B	X
0	0	0
0	1	0
1	0	0
1	1	1

A	R
0	0
1	B



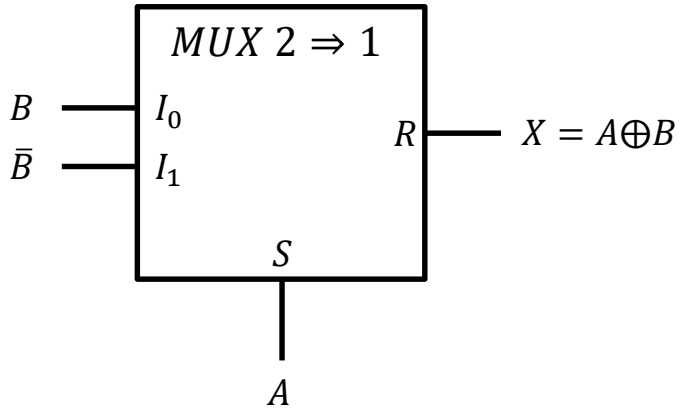
Multiplexador

Exercício

b) $Y = A \oplus B$

A	B	X
0	0	0
0	1	1
1	0	1
1	1	0

A	R
0	B
1	$\overline{B}$



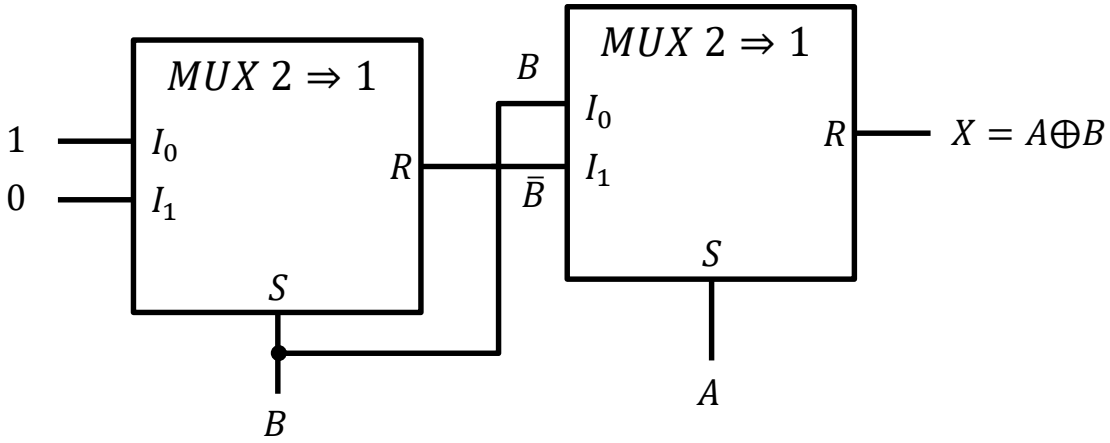
Multiplexador

Exercício

b) $Y = A \oplus B$

A	B	X
0	0	0
0	1	1
1	0	1
1	1	0

A	R
0	B
1	$\overline{B}$

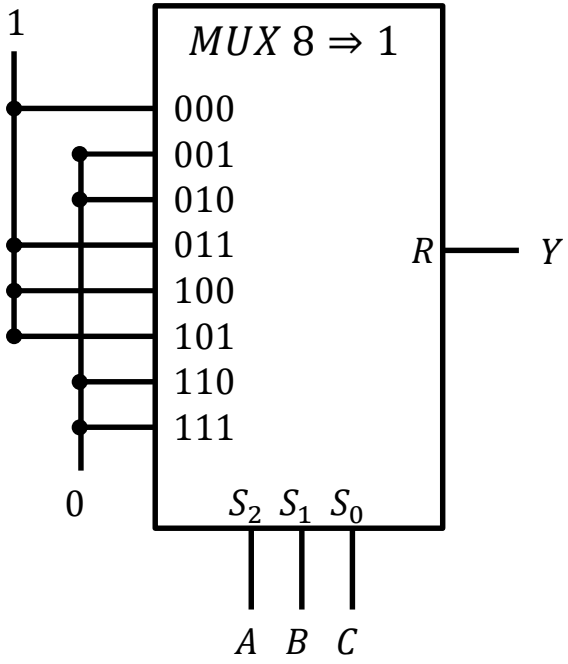


Multiplexador

Exercício

c) $Y = A\bar{B} + \bar{B}\bar{C} + \bar{A}BC$

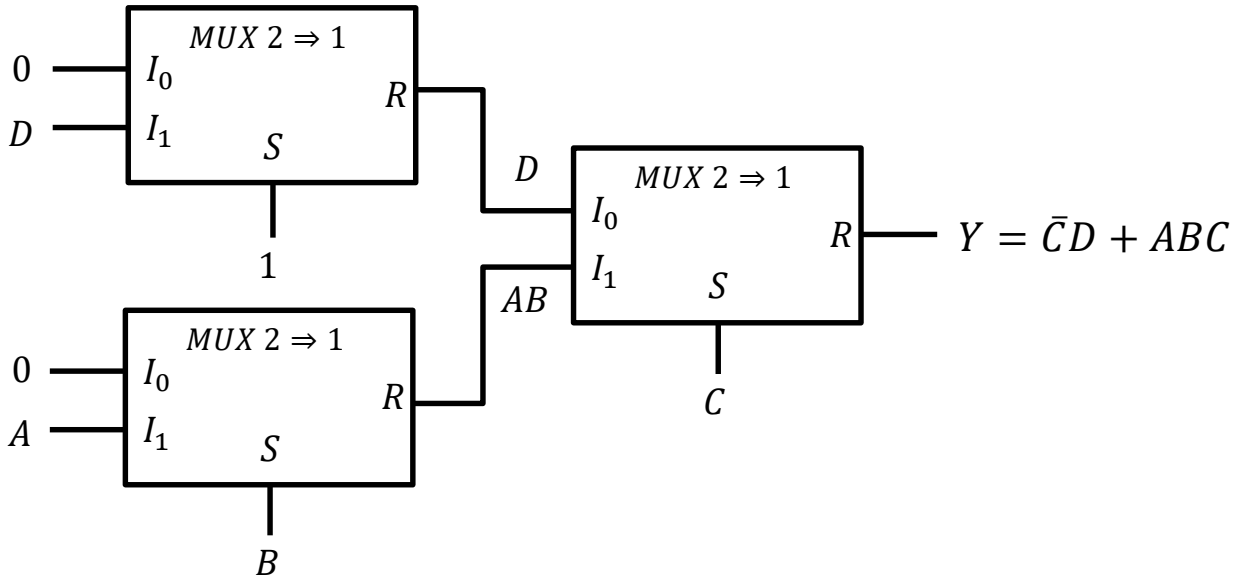
	A	B	C	Y
$\bar{B}\bar{C}$	0	0	0	1
	0	0	1	0
	0	1	0	0
$\bar{A}BC$	0	1	1	1
$\bar{B}\bar{C}$ $A\bar{B}$	1	0	0	1
$A\bar{B}$	1	0	1	1
	1	1	0	0
	1	1	1	0



Multiplexador

Exercício

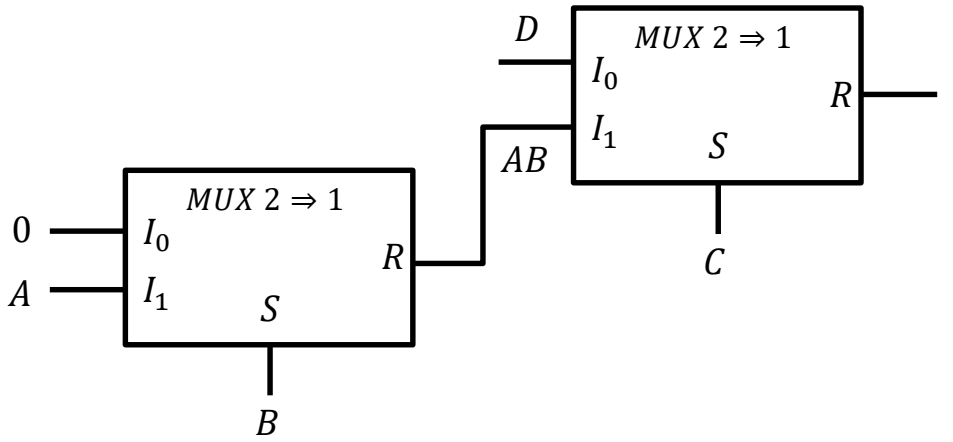
d) $Y = \bar{C}D + ABC$



Multiplexador

Exercício

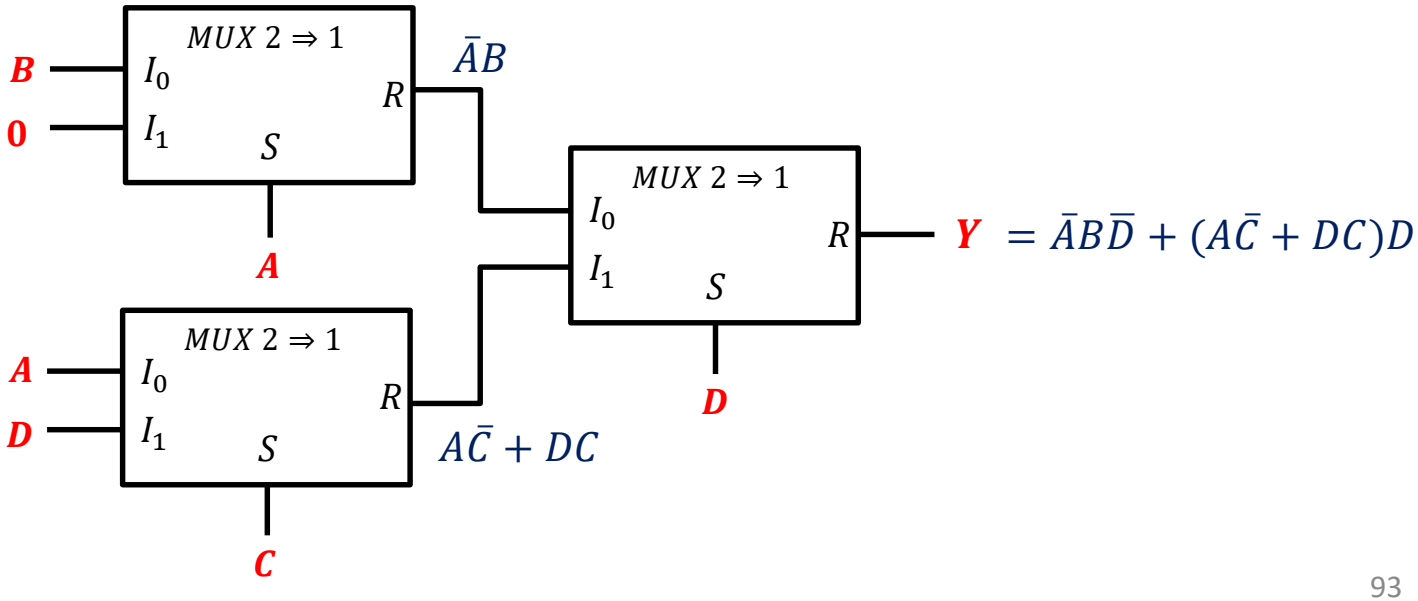
d) $Y = \bar{C}D + ABC$



Multiplexador

Exercício

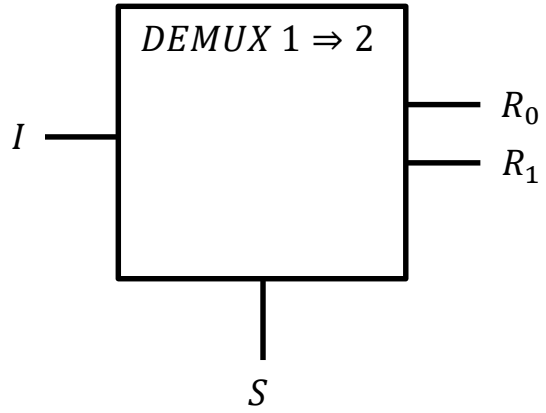
e) Qual a função Y gerada abaixo?

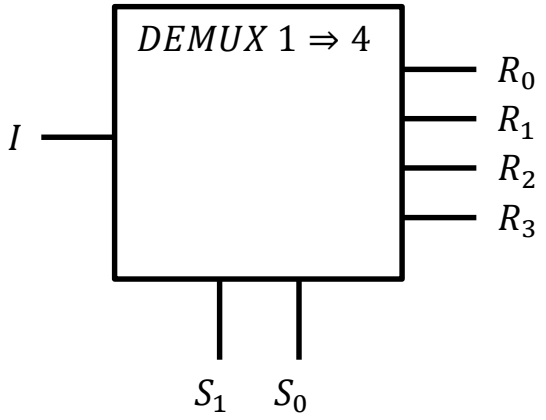


DEMULTIPLEXADOR

Demultiplexador

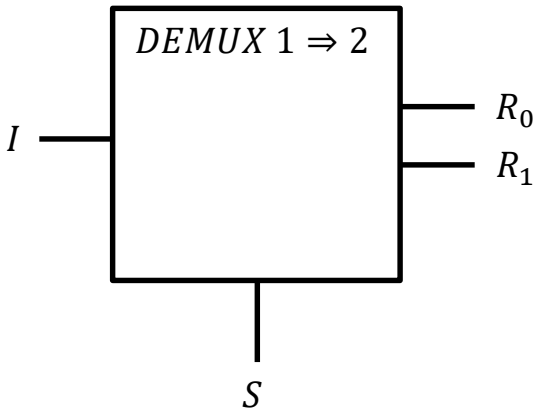
- Demultiplexador é um circuito que permite selecionar uma das entradas e transferi-la para a saída. É um distribuidor de dados;
- Decoders podem também serem utilizados como demultiplexadores.
- Também referenciado como Demultiplex, Demultiplexer ou DEMUX.





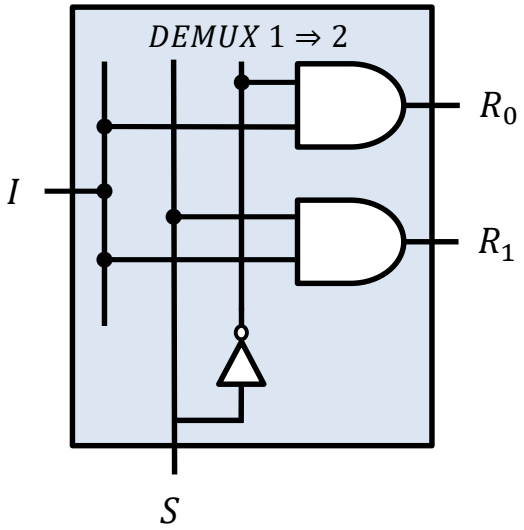
Demultiplexador

$DEMUX\ 1 \Rightarrow 2$



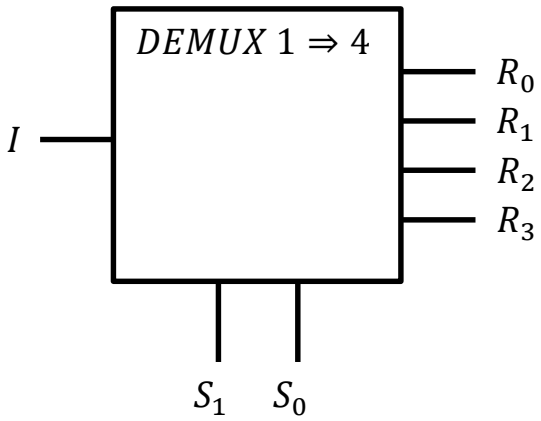
S	R_1	R_0
0	0	I
1	I	0

$$R_0 = \bar{S}I$$
$$R_1 = SI$$



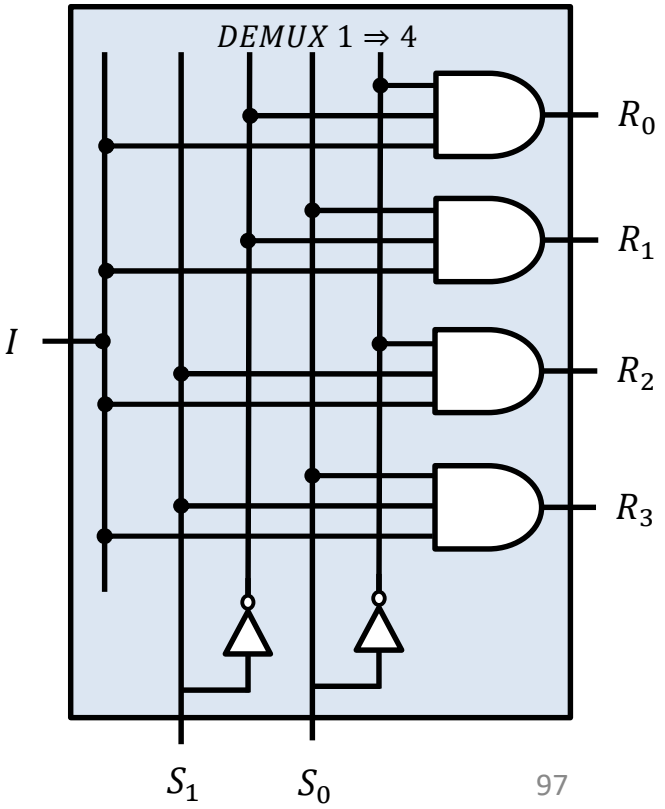
Demultiplexador

$DEMUX\ 1 \Rightarrow 4$



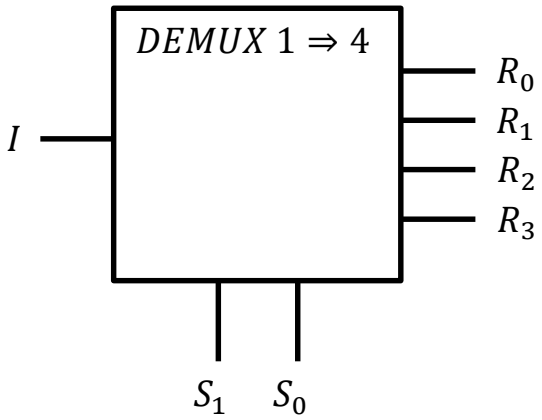
S_1	S_0	R_3	R_3	R_1	R_0
0	0	0	0	0	***I***
0	1	0	0	***I***	0
1	0	0	***I***	0	0
1	1	***I***	0	0	0

$$R_0 = \bar{S}_1 \bar{S}_0 I$$
$$R_1 = \bar{S}_1 S_0 I$$
$$R_2 = S_1 \bar{S}_0 I$$
$$R_3 = S_1 S_0 I$$

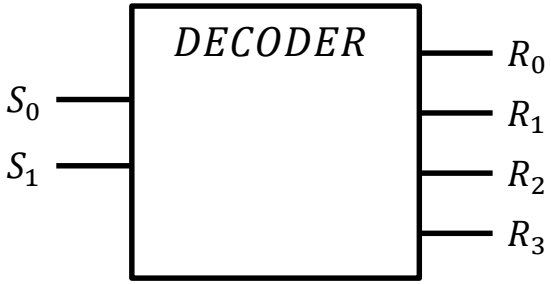


Demultiplexador

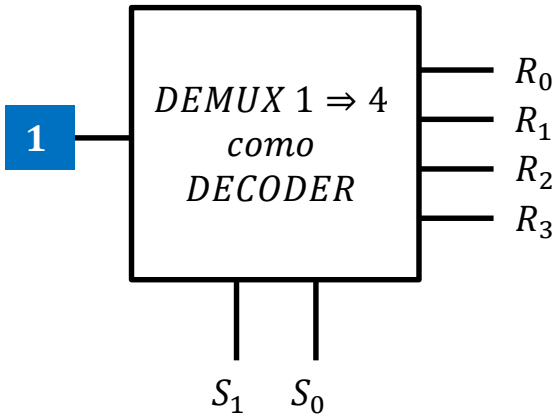
$DEMUX\ 1 \Rightarrow 4\ como\ DECODER$



$$R_0 = \bar{S}_1 \bar{S}_0 I$$
$$R_1 = \bar{S}_1 S_0 I$$
$$R_2 = S_1 \bar{S}_0 I$$
$$R_3 = S_1 S_0 I$$



$$R_0 = \bar{S}_1 \bar{S}_0$$
$$R_1 = \bar{S}_1 S_0$$
$$R_2 = S_1 \bar{S}_0$$
$$R_3 = S_1 S_0$$

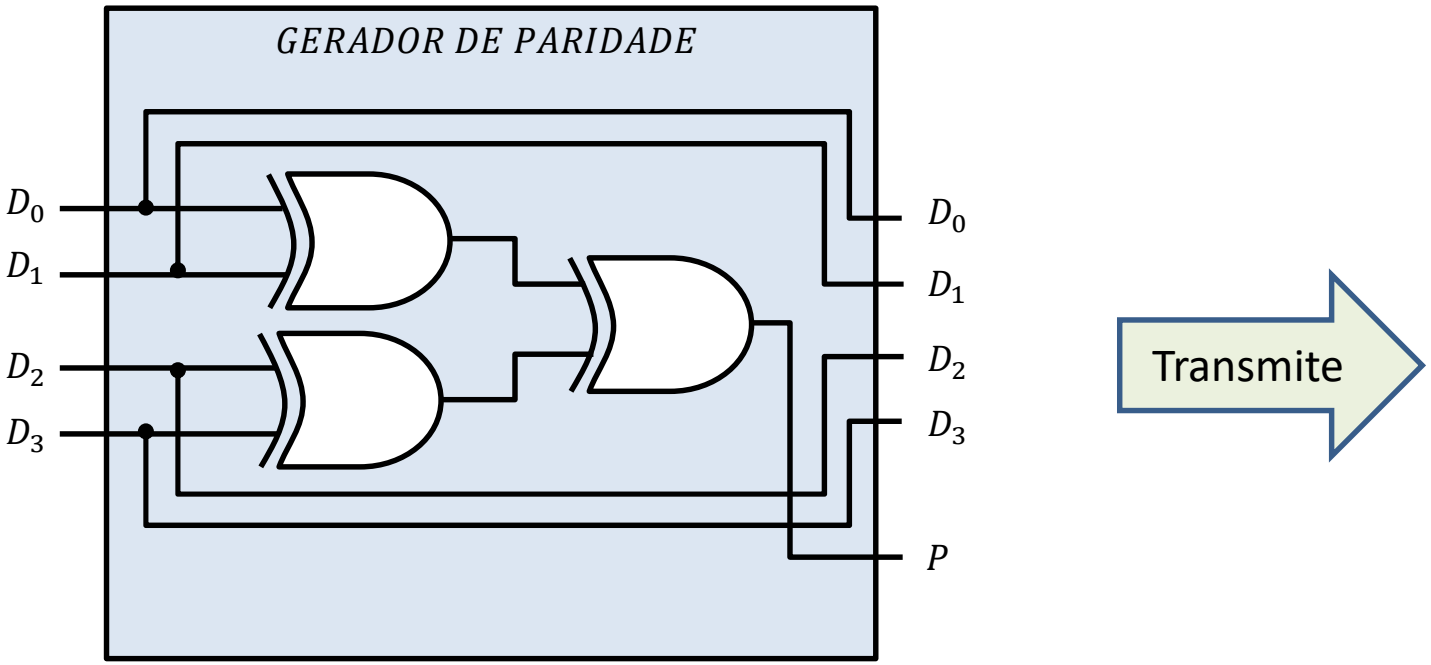


GERADORES DE PARIDADE

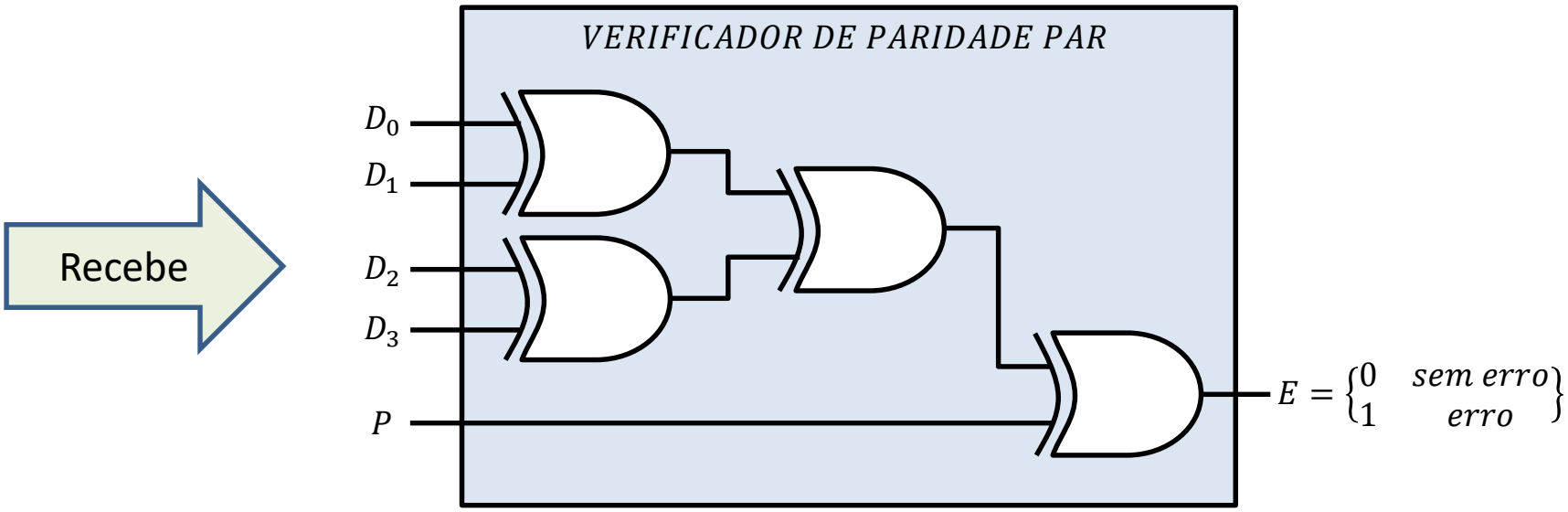
Gerador de Paridade

- Normalmente utilizado na comunicação de dados;
- Gerador de paridade - cria o $parity\ bit$, como bit extra, adicionado a uma determinada palavra;
- Verificador de paridade – calcula e compara a paridade de uma palavra a fim de garantir a correção do dado recebido.
- Paridade par: o valor do bit de paridade é determinado para que o número total de $1s$ na palavra seja par;
- Paridade ímpar: o valor do bit de paridade é determinado para que o número total de $1s$ na palavra seja ímpar;

Gerador de Paridade Par

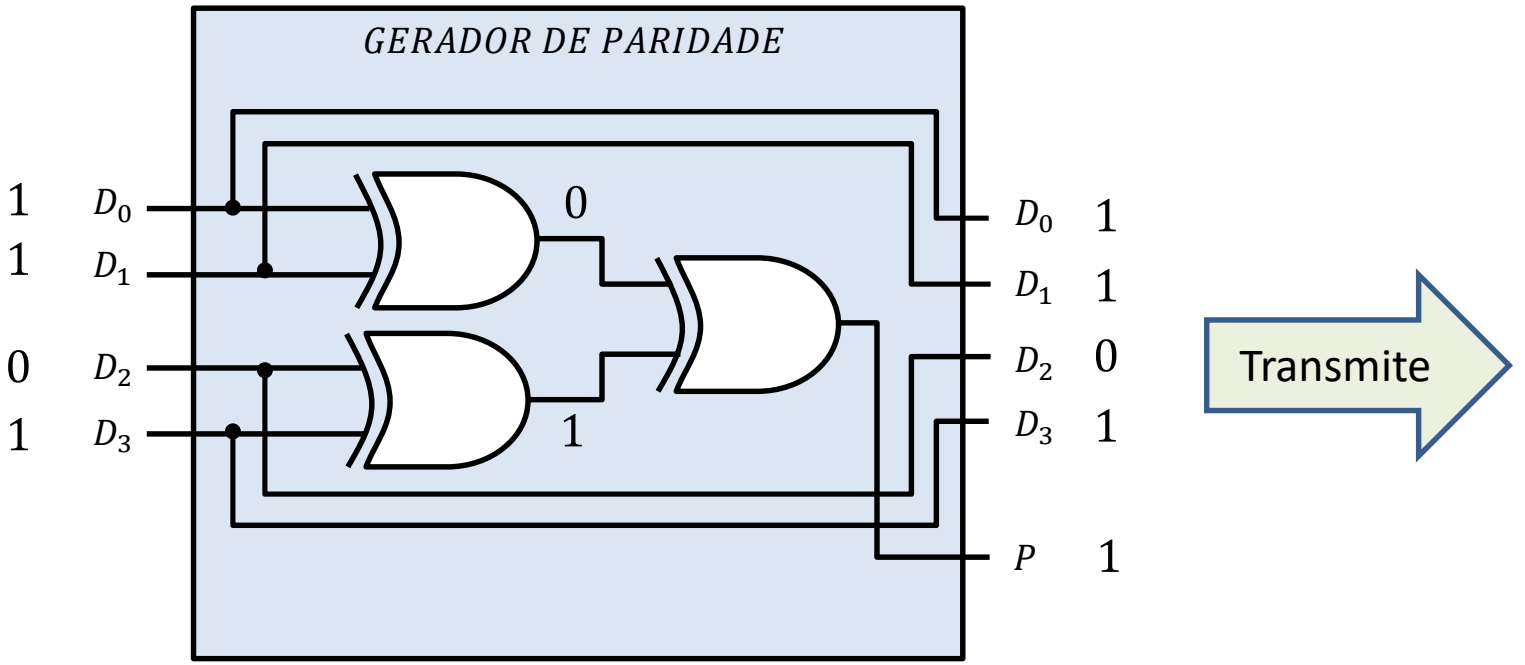


Verificador de Paridade Par



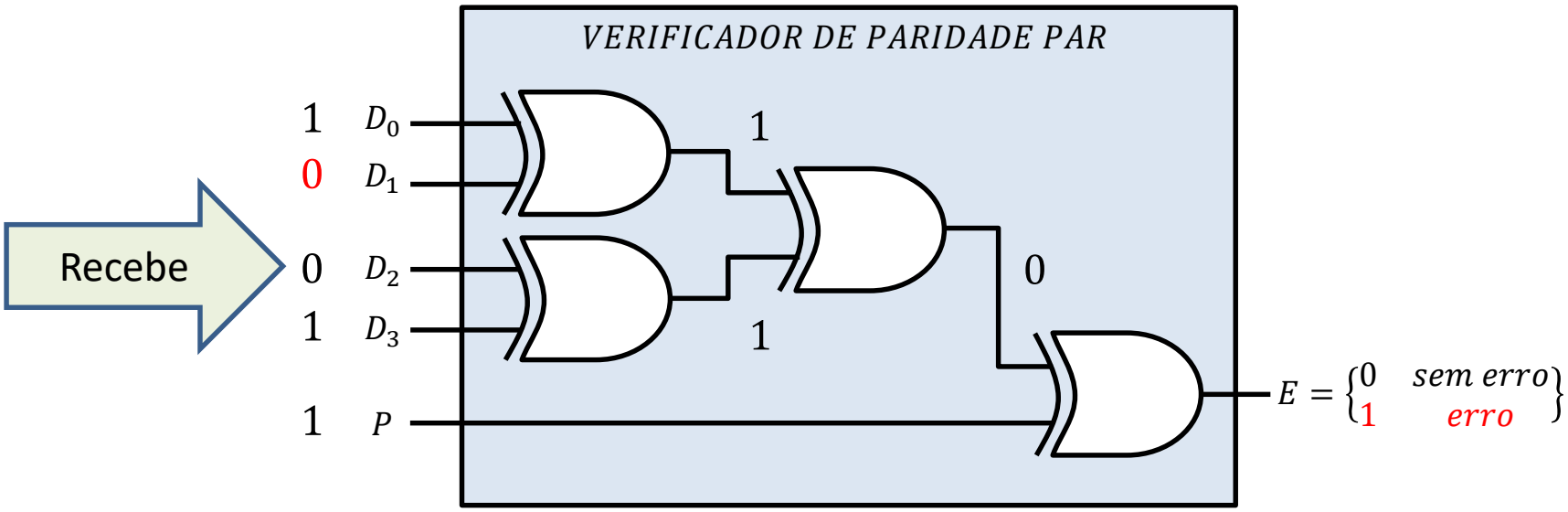
Gerador de Paridade Par

Exemplo



Verificador de Paridade Par

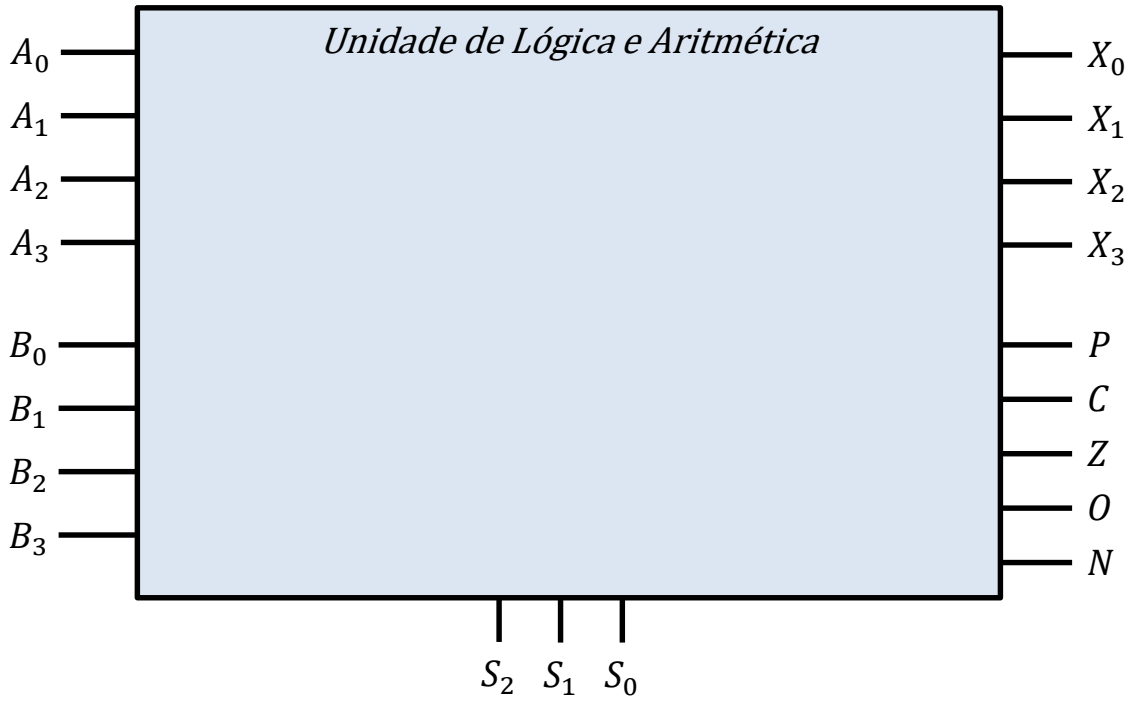
Exemplo



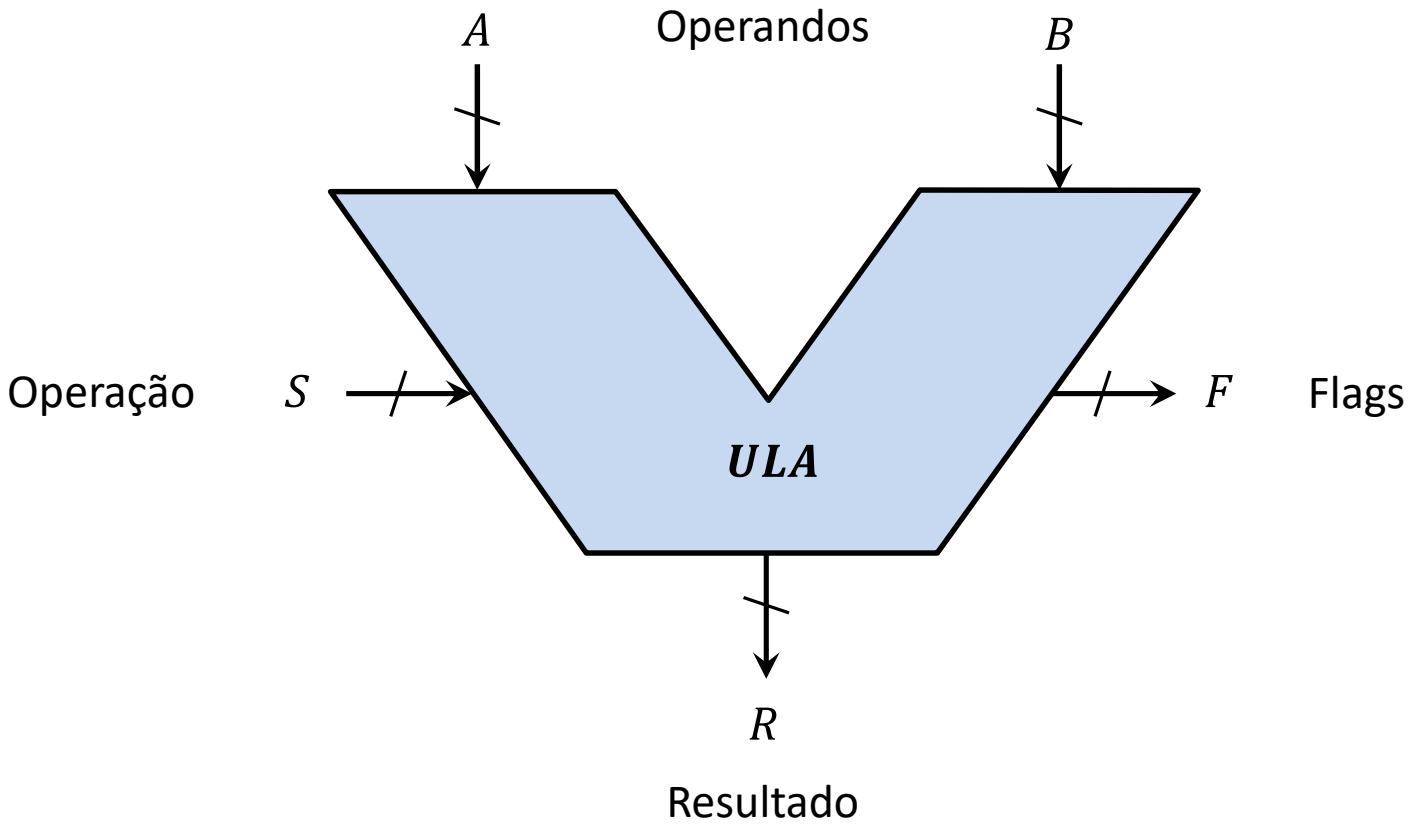
TRABALHO

Trabalho

- Elaborar um circuito com as seguintes interfaces:



Trabalho



Trabalho

- Significado dos bits de $S_{2\ldots0}$:

S_2	S_1	S_0	*Operação*
0	0	0	$X = A + B$
0	0	1	$X = A - B$
0	1	0	$X = A\ and\ B$
0	1	1	$X = A\ or\ B$
1	0	0	$X = A\ xor\ B$
1	0	1	$X = -A$ ou $X = -B$
1	1	0	$X = A + 1$
1	1	1	$X = A - 1$

- Significado dos Flags de qualidade: P, C, Z, O e S:

P	C	Z	O	N
$*$	$*$	$*$	$*$	$*$
$*$	$*$	$*$	$*$	$*$
$*$	0	$*$	0	$*$
$*$	0	$*$	0	$*$
$*$	0	$*$	0	$*$
$*$	0	$*$	0	$*$
$*$	0	$*$	0	$*$
$*$	0	$*$	0	$*$

Flags	*Significado*
P	Paridade par de X
C	$Carry$ da última operação
Z	Todos os bits de X são 0
O	Operação gerou $Overflow$
N	Bit de Sinal de X

Trabalho

- Significado dos bits de $S_{2\ldots0}$:

S_2	S_1	S_0	*Operação*
0	0	0	$X = A + B$
0	0	1	$X = A - B$
0	1	0	$X = -A\ ou\ X = -B$
0	1	1	$X = A + 1$
1	0	0	$X = not\ A\ ou\ not\ B$
1	0	1	$X = A\ and\ B$
1	1	0	$X = A\ or\ B$
1	1	1	$X = A\ xor\ B$

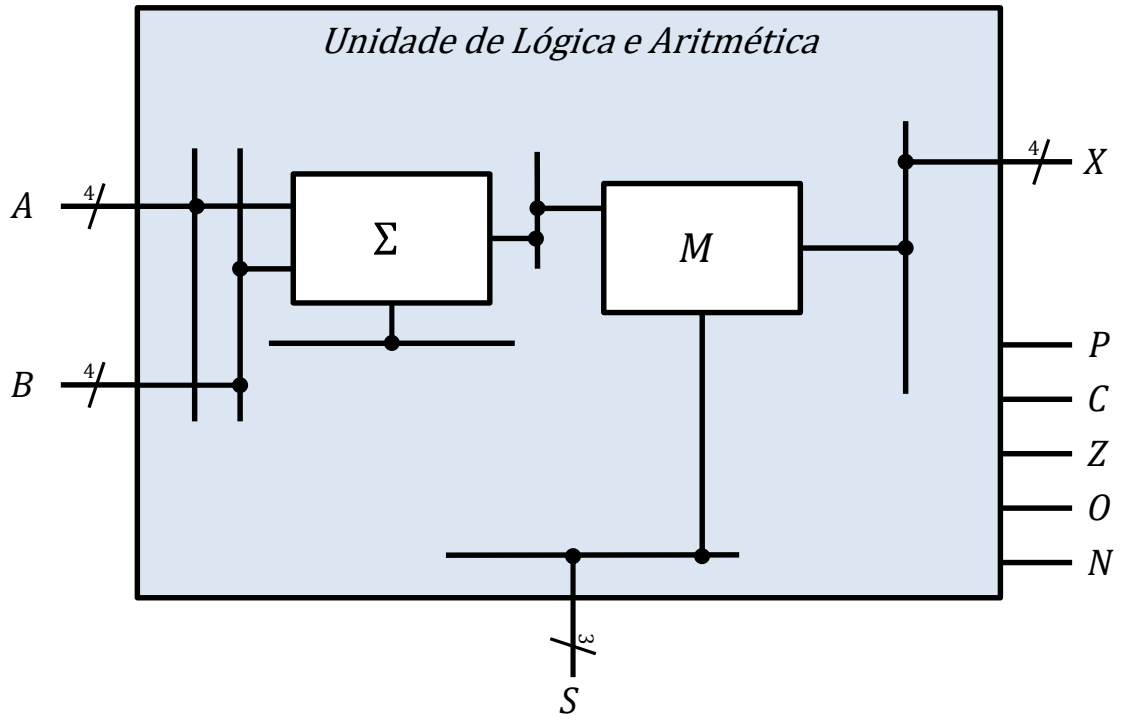
- Significado dos Flags de qualidade: P, C, Z, O e S:

P	C	Z	O	N
$*$	$*$	$*$	$*$	$*$
$*$	$*$	$*$	$*$	$*$
$*$	0	$*$	0	$*$
$*$	0	$*$	0	$*$
$*$	0	$*$	0	$*$
$*$	0	$*$	0	$*$
$*$	0	$*$	0	$*$
$*$	0	$*$	0	$*$

Flags	*Significado*
P	Paridade par de X
C	$Carry$ da última operação
Z	Todos os bits de X são 0
O	Operação gerou $Overflow$
N	Bit de Sinal de X

Trabalho

- Para o dia $15/Jun$ apresentar um diagrama simplificado da proposta:



Trabalho

- ~~Para o dia $22/Jun$ apresentar uma solução de 1 bit;~~
- ~~Para o dia $29/Jun$ apresentar uma solução de 2 bits;~~
- ~~Para o dia $06/Jul$ apresentar uma solução de 4 bits;~~

- Para o dia $29/Jun$ apresentar uma solução de 2 bits;
- Para o dia $06/Jul$ apresentar uma solução de 4 bits;

Unidade de Lógica e Aritmética

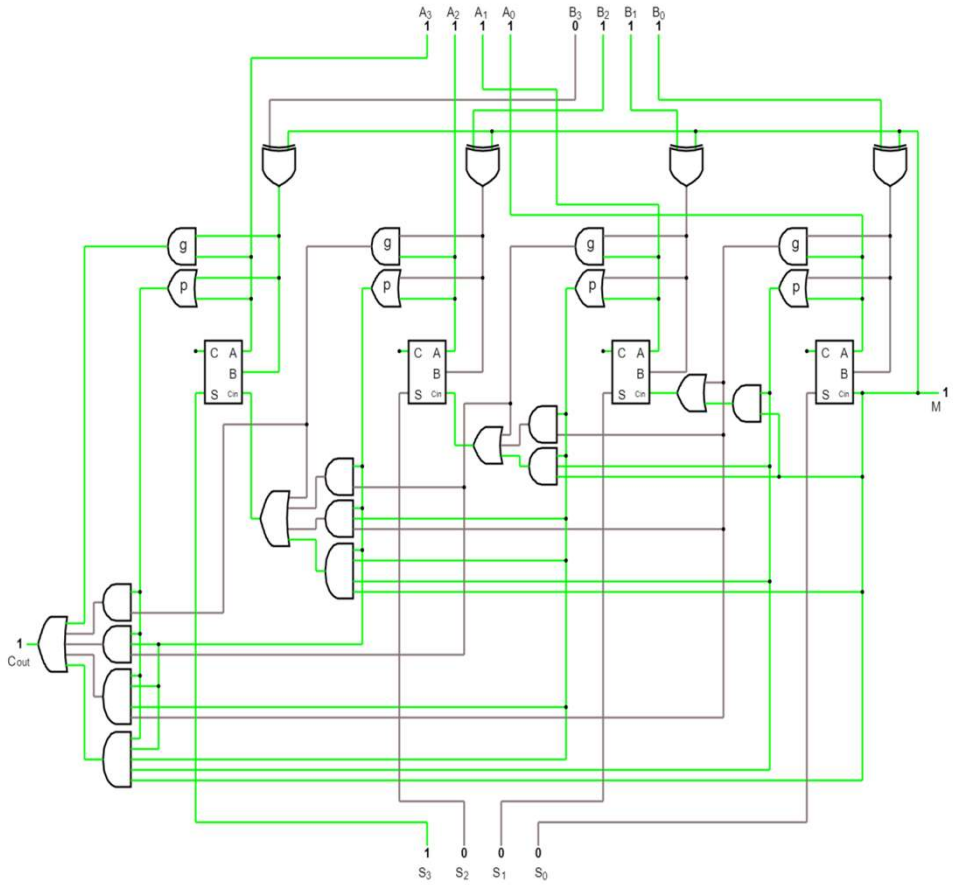
add como add/sub/neg/inc/dec

add	$X = A + B$
sub	$X = A + (-B)$
dec	$X = A + (-1)$
inc	$X = A + 1$
neg	$X = 0 + (-B)$

$$X = \begin{bmatrix} A \\ 0 \end{bmatrix} + \begin{bmatrix} B \\ -B \\ 1 \\ -1 \end{bmatrix}$$

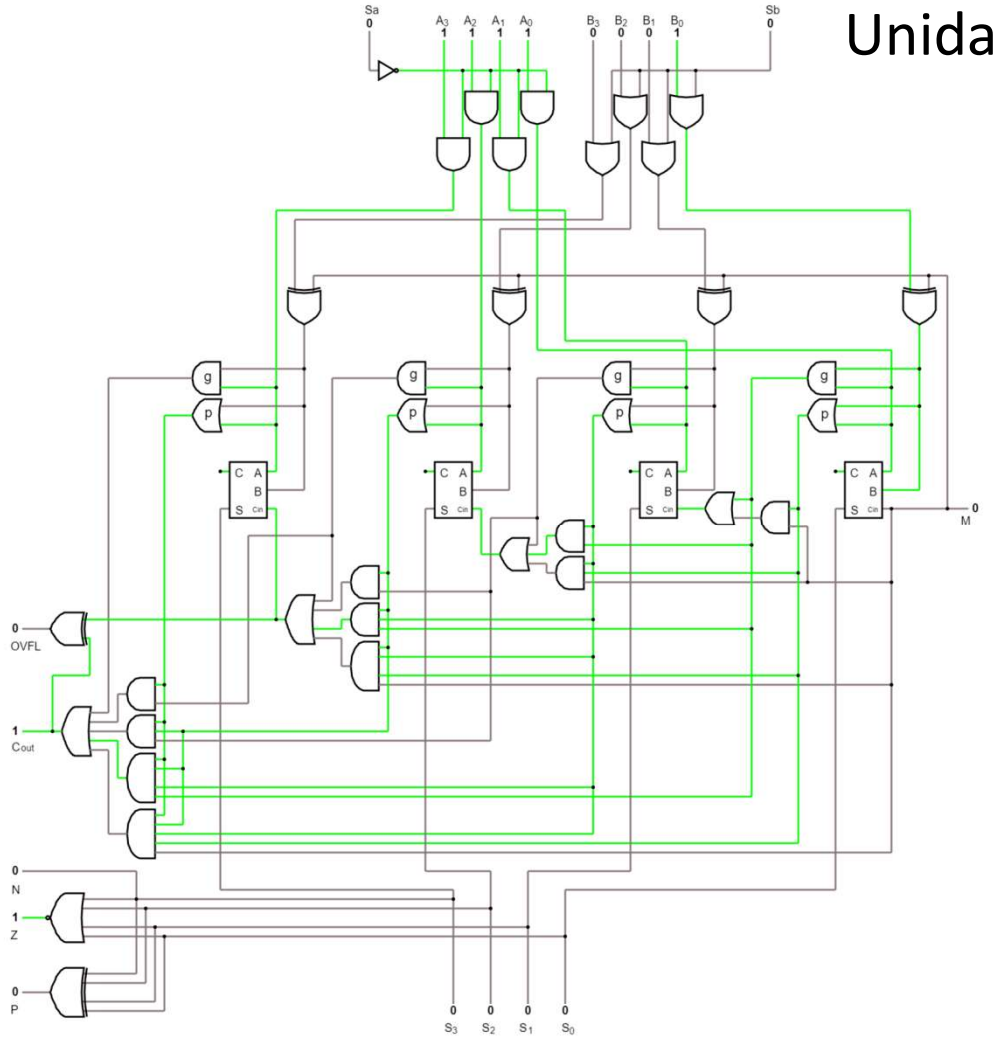
Unidade de Lógica e Aritmética

add como add/sub/neg/inc/dec



Unidade de Lógica e Aritmética

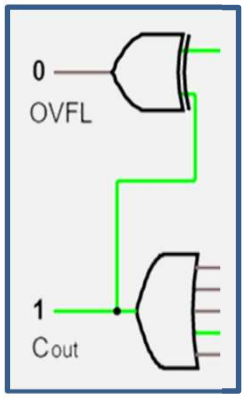
add como add/sub/neg/inc/dec

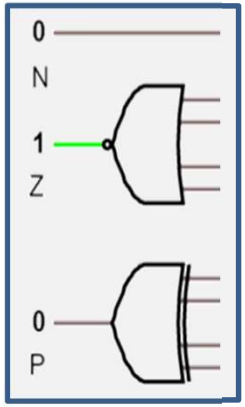


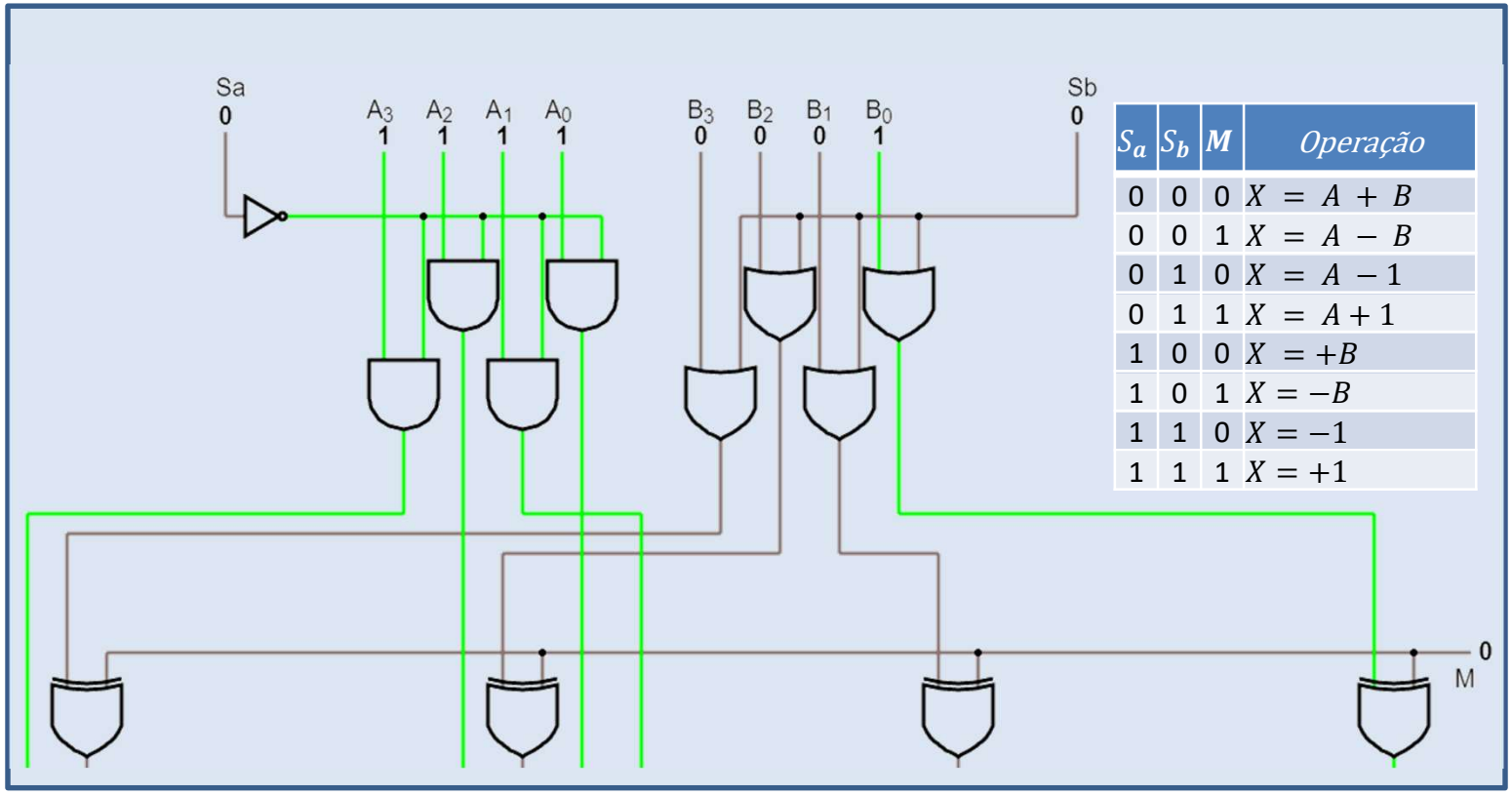
S_a	S_b	M	*Operação*
0	0	0	$X = A + B$
0	0	1	$X = A - B$
0	1	0	$X = A - 1$
0	1	1	$X = A + 1$
1	0	0	$X = +B$
1	0	1	$X = -B$
1	1	0	$X = -1$
1	1	1	$X = 1$

Unidade de Lógica e Aritmética

add como add/sub/neg/inc/dec



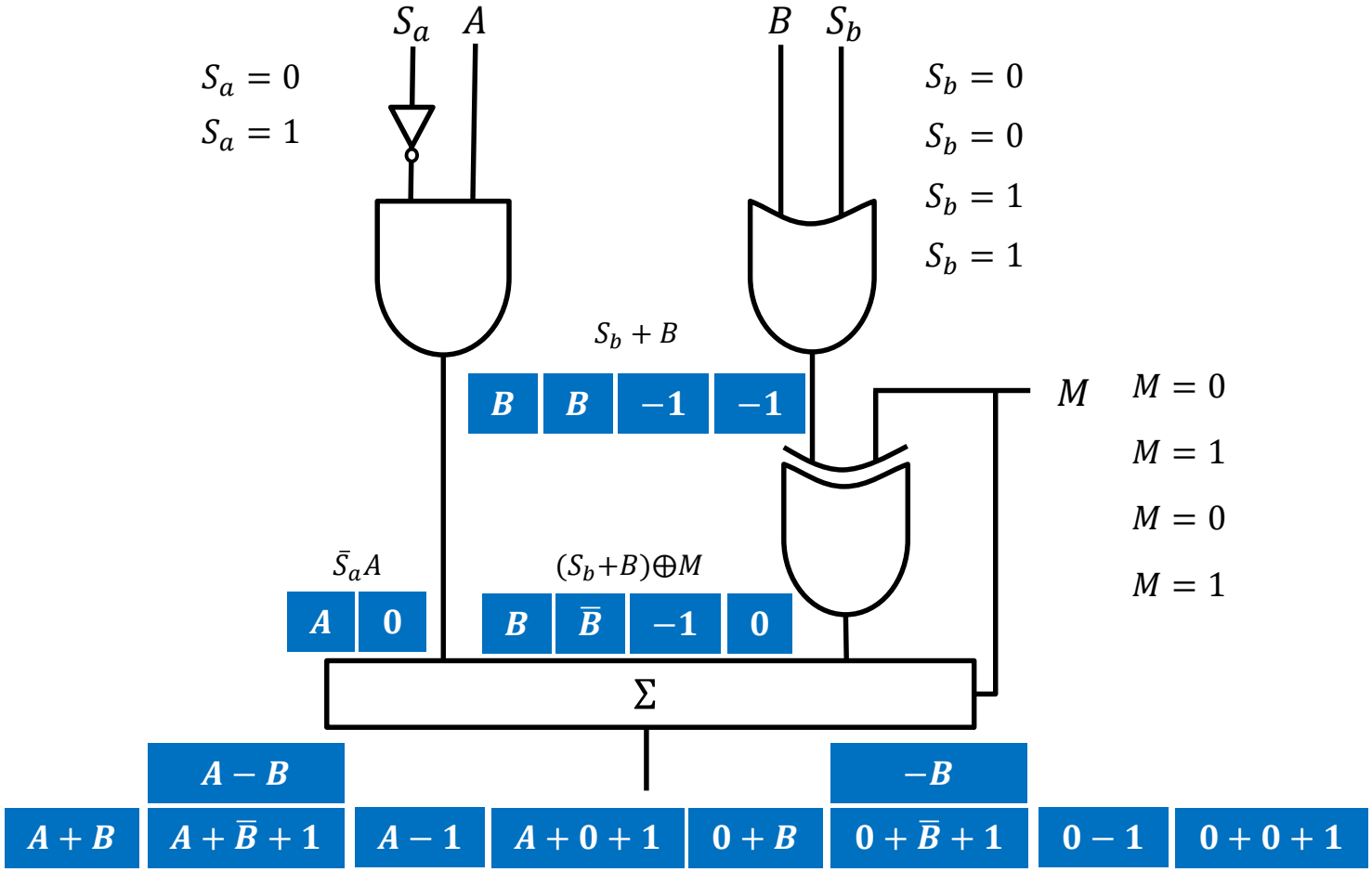




S_a	S_b	M	*Operação*
0	0	0	$X = A + B$
0	0	1	$X = A - B$
0	1	0	$X = A - 1$
0	1	1	$X = A + 1$
1	0	0	$X = +B$
1	0	1	$X = -B$
1	1	0	$X = -1$
1	1	1	$X = +1$

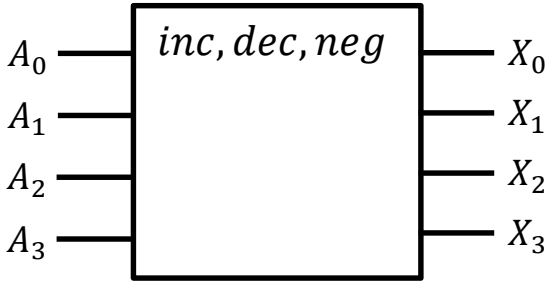
Unidade de Lógica e Aritmética

add como add/sub/neg/inc/dec



S_a	S_b	M	*Operação*
0	0	0	$X = A + B$
0	0	1	$X = A - B$
0	1	0	$X = A - 1$
0	1	1	$X = A + 1$
1	0	0	$X = +B$
1	0	1	$X = -B$
1	1	0	$X = -1$
1	1	1	$X = +1$

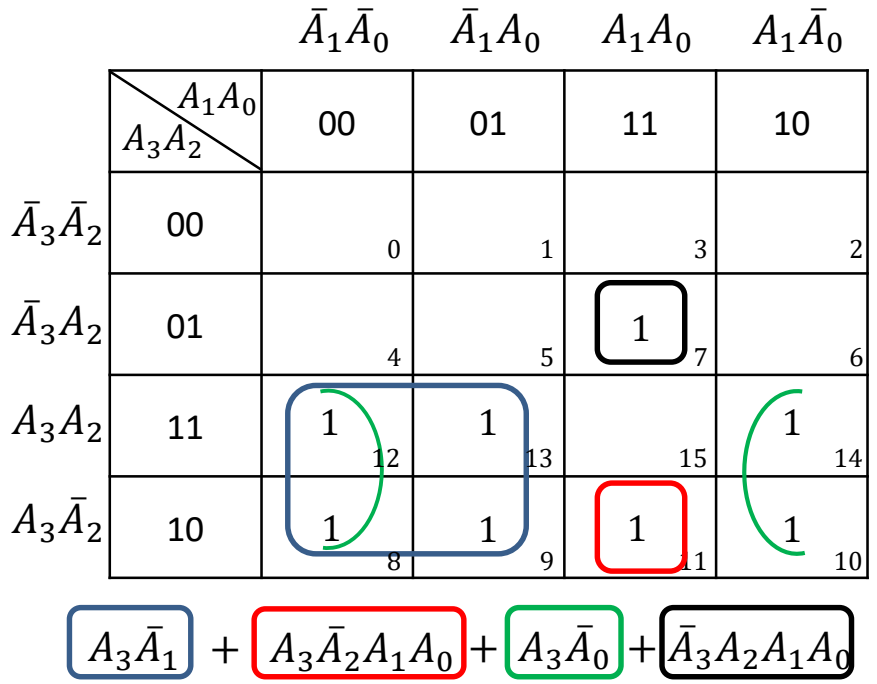
inc, dec e neg exclusivos



Unidade de Lógica e Aritmética

inc

A_3	A_2	A_1	A_0	X_3	X_2	X_1	X_0
0	0	0	0	0	0	0	1
0	0	0	1	0	0	1	0
0	0	1	0	0	0	1	1
0	0	1	1	0	1	0	0
0	1	0	0	0	1	0	1
0	1	0	1	0	1	1	0
0	1	1	0	0	1	1	1
0	1	1	1	1	0	0	0
1	0	0	0	1	0	0	1
1	0	0	1	1	0	1	0
1	0	1	0	1	0	1	1
1	0	1	1	1	1	0	0
1	1	0	0	1	1	0	1
1	1	0	1	1	1	1	0
1	1	1	0	1	1	1	1
1	1	1	1	0	0	0	0



$$A_3\bar{A}_1 + A_3\bar{A}_2A_1A_0 + A_3\bar{A}_0 + \bar{A}_3A_2A_1A_0$$

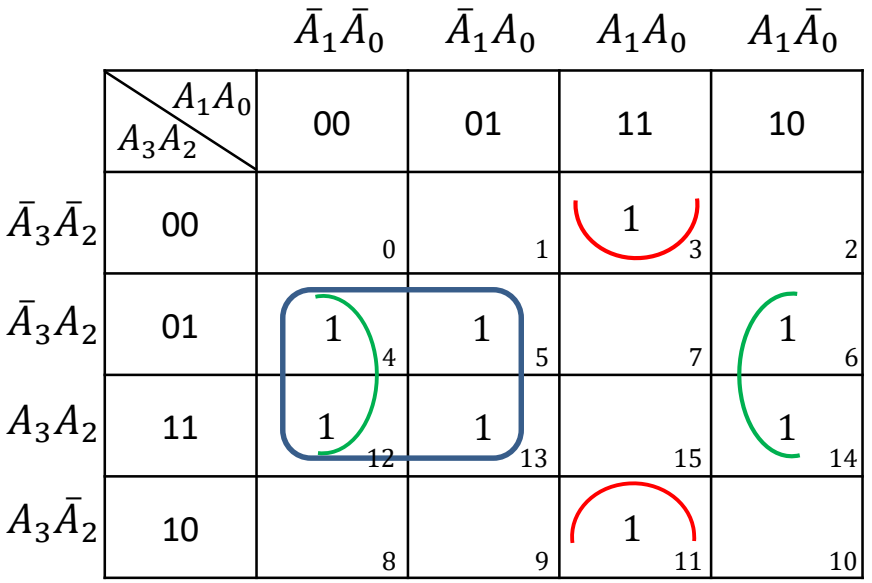
$$X_3 = A_3\bar{A}_1 + A_3\bar{A}_2A_1A_0 + A_3\,\bar{A}_0 + \bar{A}_3A_2A_1A_0$$

$$X_3 = A_3(\bar{A}_0 + \bar{A}_1) + A_1A_0(A_3 \oplus A_2)$$

Unidade de Lógica e Aritmética

inc

A_3	A_2	A_1	A_0	X_3	X_2	X_1	X_0
0	0	0	0	0	0	0	1
0	0	0	1	0	0	1	0
0	0	1	0	0	0	1	1
0	0	1	1	0	1	0	0
0	1	0	0	0	1	0	1
0	1	0	1	0	1	1	0
0	1	1	0	0	1	1	1
0	1	1	1	1	0	0	0
1	0	0	0	1	0	0	1
1	0	0	1	1	0	1	0
1	0	1	0	1	0	1	1
1	0	1	1	1	1	0	0
1	1	0	0	1	1	0	1
1	1	0	1	1	1	1	0
1	1	1	0	1	1	1	1
1	1	1	1	0	0	0	0



$$A_2\bar{A}_1 + A_0A_1\bar{A}_2 + \bar{A}_0A_2$$

$$X_2 = A_2\bar{A}_1 + A_0A_1\bar{A}_2 + \bar{A}_0A_2$$

$$X_2 = A_2(\bar{A}_0 + \bar{A}_1) + A_0A_1\bar{A}_2$$

Unidade de Lógica e Aritmética

inc

A_3	A_2	A_1	A_0	X_3	X_2	X_1	X_0
0	0	0	0	0	0	0	1
0	0	0	1	0	0	1	0
0	0	1	0	0	0	1	1
0	0	1	1	0	1	0	0
0	1	0	0	0	1	0	1
0	1	0	1	0	1	1	0
0	1	1	0	0	1	1	1
0	1	1	1	1	0	0	0
1	0	0	0	1	0	0	1
1	0	0	1	1	0	1	0
1	0	1	0	1	0	1	1
1	0	1	1	1	1	0	0
1	1	0	0	1	1	0	1
1	1	0	1	1	1	1	0
1	1	1	0	1	1	1	1
1	1	1	1	0	0	0	0

		$\bar{A}_1\bar{A}_0$	$\bar{A}_1A_0$	A_1A_0	$A_1\bar{A}_0$
	A_3A_2 \ A_1A_0	00	01	11	10
$\bar{A}_3\bar{A}_2$	00	0	1 1	3	1 2
$\bar{A}_3A_2$	01	4	1 5	7	1 6
A_3A_2	11	12	1 13	15	1 14
$A_3\bar{A}_2$	10	8	1 9	11	1 10

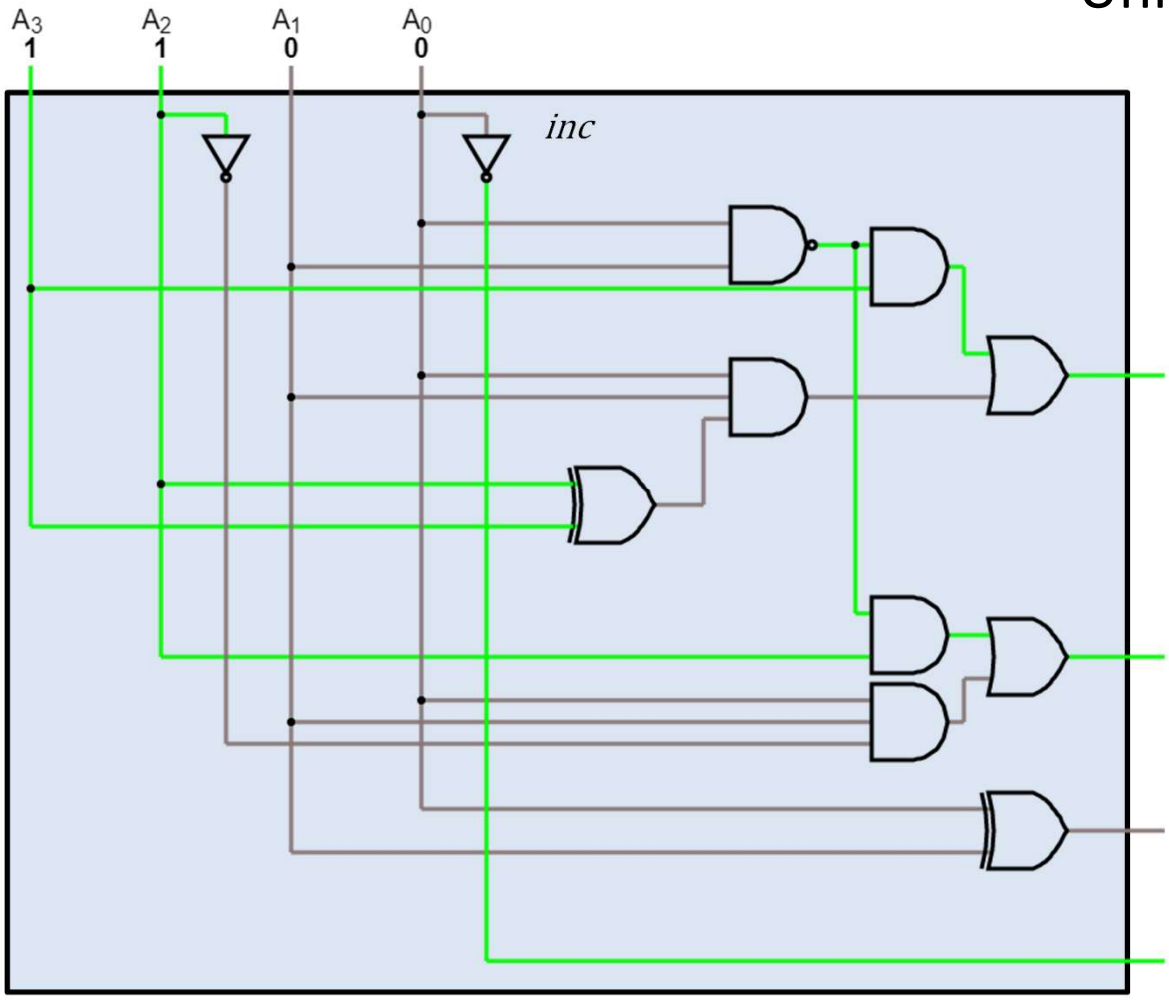
$\bar{A}_1A_0 + A_1\bar{A}_0$

$X_1 = A_0 \oplus A_1$

$X_0 = \bar{A}_0$

Unidade de Lógica e Aritmética

inc



$X = A + 1$

1 $X_3 = A_3(\bar{A}_0 + \bar{A}_1) + A_0A_1(A_2 \oplus A_3)$

$X_3 = A_3(\overline{A_0A_1}) + A_0A_1(A_2 \oplus A_3)$

1 $X_2 = A_2(\bar{A}_0 + \bar{A}_1) + A_0A_1\bar{A}_2$

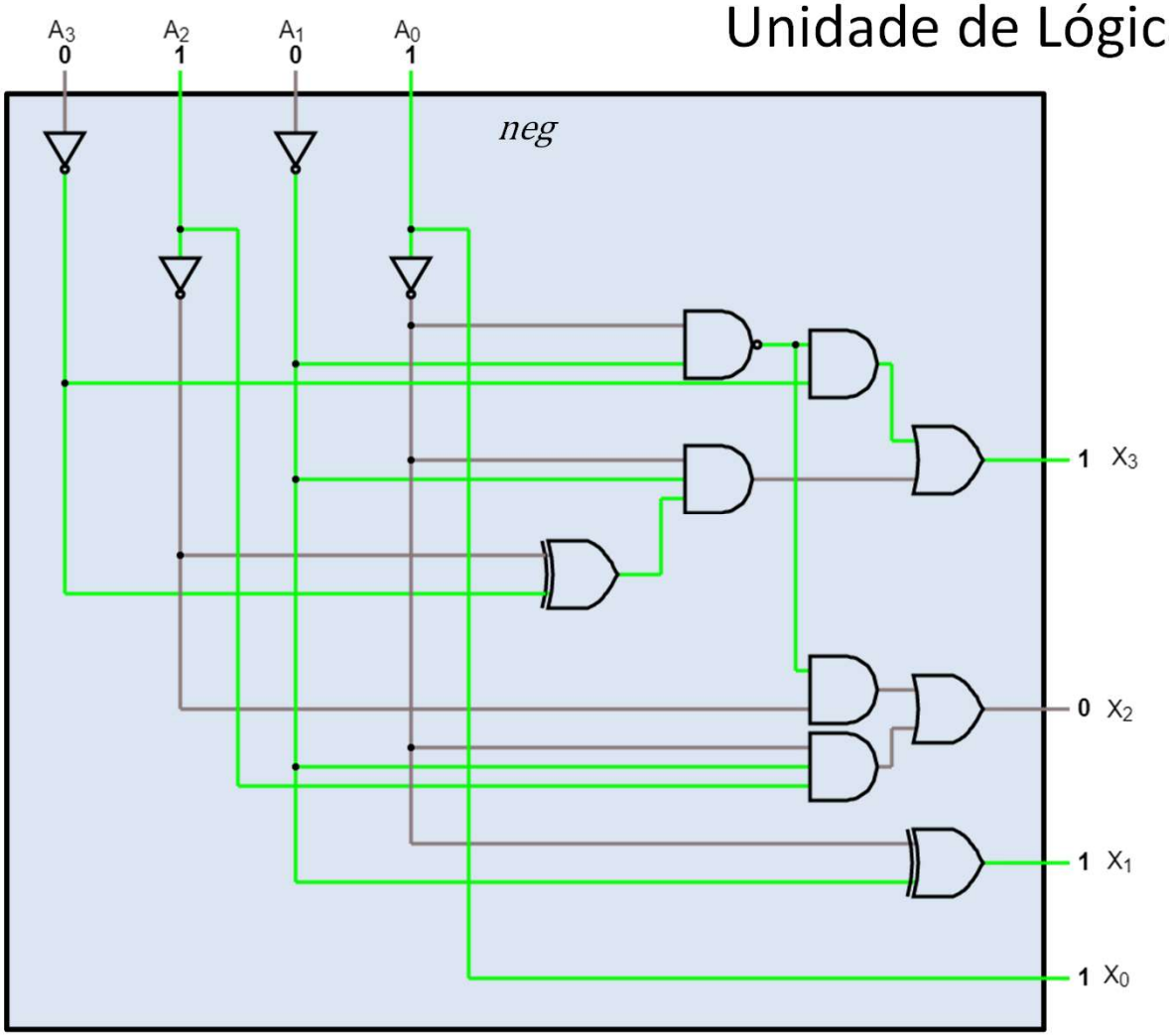
$X_2 = A_2(\overline{A_0A_1}) + A_0A_1\bar{A}_2$

0 $X_1 = A_0 \oplus A_1$

1 $X_0 = \bar{A}_0$

Unidade de Lógica e Aritmética

neg



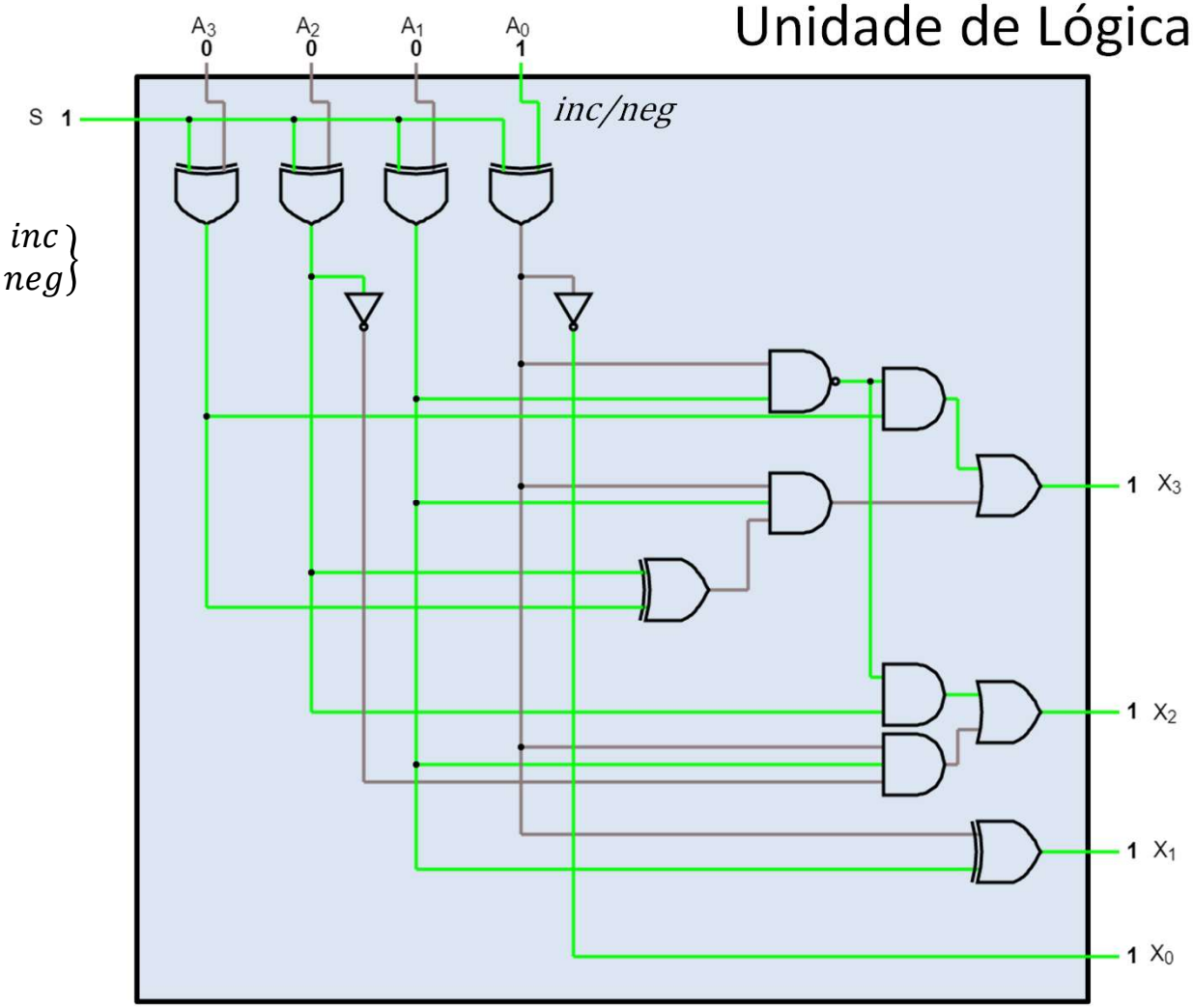
$$X = -A$$

$$X = \bar{A} + 1$$

Unidade de Lógica e Aritmética

inc/neg

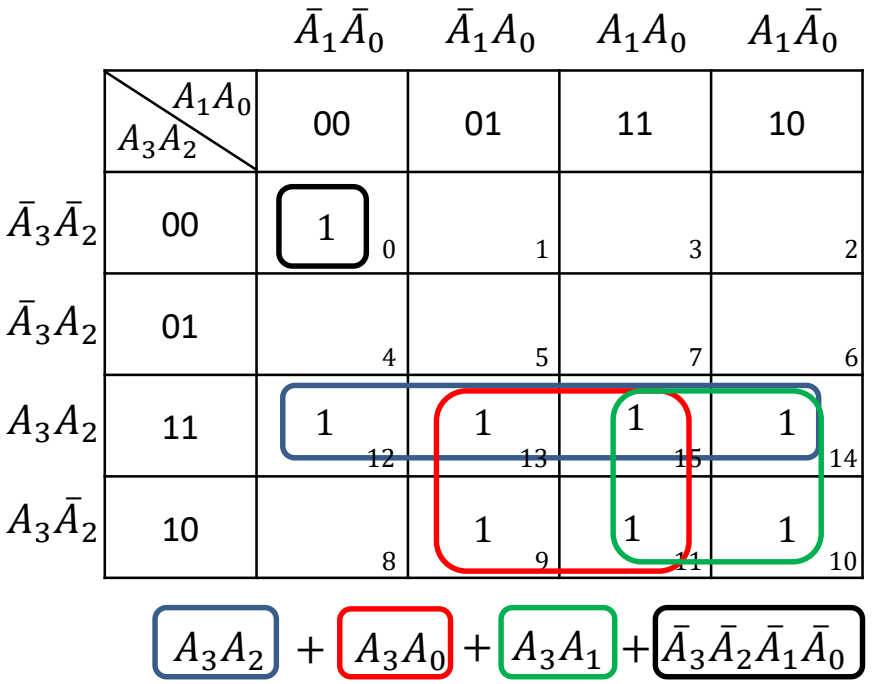
$$S = \begin{cases} 0 \Rightarrow & inc \\ 1 \Rightarrow & neg \end{cases}$$



Unidade de Lógica e Aritmética

dec

A_3	A_2	A_1	A_0	X_3	X_2	X_1	X_0
0	0	0	0	1	1	1	1
0	0	0	1	0	0	0	0
0	0	1	0	0	0	0	1
0	0	1	1	0	0	1	0
0	1	0	0	0	0	1	1
0	1	0	1	0	1	0	0
0	1	1	0	0	1	0	1
0	1	1	1	0	1	1	0
1	0	0	0	0	1	1	1
1	0	0	1	1	0	0	0
1	0	1	0	1	0	0	1
1	0	1	1	1	0	1	0
1	1	0	0	1	0	1	1
1	1	0	1	1	1	0	0
1	1	1	0	1	1	0	1
1	1	1	1	1	1	1	0



$$X_3 = A_3A_2 + A_3A_0 + A_3A_1 + \bar{A}_3\bar{A}_2\bar{A}_1\bar{A}_0$$

$$X_3 = A_3(A_0 + A_1 + A_2) + \bar{A}_0\bar{A}_1\bar{A}_2\bar{A}_3$$

Unidade de Lógica e Aritmética

dec

A_3	A_2	A_1	A_0	X_3	X_2	X_1	X_0
0	0	0	0	1	1	1	1
0	0	0	1	0	0	0	0
0	0	1	0	0	0	0	1
0	0	1	1	0	0	1	0
0	1	0	0	0	0	1	1
0	1	0	1	0	1	0	0
0	1	1	0	0	1	0	1
0	1	1	1	0	1	1	0
1	0	0	0	0	1	1	1
1	0	0	1	1	0	0	0
1	0	1	0	1	0	0	1
1	0	1	1	1	0	1	0
1	1	0	0	1	0	1	1
1	1	0	1	1	1	0	0
1	1	1	0	1	1	0	1
1	1	1	1	1	1	1	0

		$\bar{A}_1\bar{A}_0$	$\bar{A}_1A_0$	A_1A_0	$A_1\bar{A}_0$
	A_3A_2 \ A_1A_0	00	01	11	10
$\bar{A}_3\bar{A}_2$	00	1 0	1	3	2
$\bar{A}_3A_2$	01	4	5	7	6
A_3A_2	11	1 12	1 13	1 15	1 14
$A_3\bar{A}_2$	10	8	1 9	1 11	1 10

$$A_3A_2\bar{A}_1\bar{A}_0 + A_3A_0 + A_3A_1 + \bar{A}_3\bar{A}_2\bar{A}_1\bar{A}_0$$

$$X_3 = A_3A_2\bar{A}_1\bar{A}_0 + A_3A_0 + A_3A_1 + \bar{A}_3\bar{A}_2\bar{A}_1\bar{A}_0$$

$$X_3 = A_3(A_0 + A_1) + \bar{A}_0\bar{A}_1\overline{(A_2 \oplus A_3)}$$

Unidade de Lógica e Aritmética

dec

A_3	A_2	A_1	A_0	X_3	X_2	X_1	X_0
0	0	0	0	1	1	1	1
0	0	0	1	0	0	0	0
0	0	1	0	0	0	0	1
0	0	1	1	0	0	1	0
0	1	0	0	0	0	1	1
0	1	0	1	0	1	0	0
0	1	1	0	0	1	0	1
0	1	1	1	0	1	1	0
1	0	0	0	0	1	1	1
1	0	0	1	1	0	0	0
1	0	1	0	1	0	0	1
1	0	1	1	1	0	1	0
1	1	0	0	1	0	1	1
1	1	0	1	1	1	0	0
1	1	1	0	1	1	0	1
1	1	1	1	1	1	1	0

		$\bar{A}_1\bar{A}_0$	$\bar{A}_1A_0$	A_1A_0	$A_1\bar{A}_0$
	A_3A_2 \ A_1A_0	00	01	11	10
$\bar{A}_3\bar{A}_2$	00	1 0	1	3	2
$\bar{A}_3A_2$	01	4	1 5	1 7	1 6
A_3A_2	11	12	1 13	1 15	1 14
$A_3\bar{A}_2$	10	1 8	9	11	10

$$A_2A_0 + \bar{A}_1\bar{A}_0\bar{A}_2 + A_1A_2$$

$$X_2 = A_2A_0 + A_1A_2 + \bar{A}_1\bar{A}_0\bar{A}_2$$

$$X_2 = A_2(A_0 + A_1) + \bar{A}_0\bar{A}_1\bar{A}_2$$

Unidade de Lógica e Aritmética

dec

A_3	A_2	A_1	A_0	X_3	X_2	X_1	X_0
0	0	0	0	1	1	1	1
0	0	0	1	0	0	0	0
0	0	1	0	0	0	0	1
0	0	1	1	0	0	1	0
0	1	0	0	0	0	1	1
0	1	0	1	0	1	0	0
0	1	1	0	0	1	0	1
0	1	1	1	0	1	1	0
1	0	0	0	0	1	1	1
1	0	0	1	1	0	0	0
1	0	1	0	1	0	0	1
1	0	1	1	1	0	1	0
1	1	0	0	1	0	1	1
1	1	0	1	1	1	0	0
1	1	1	0	1	1	0	1
1	1	1	1	1	1	1	0

		$\bar{A}_1\bar{A}_0$	$\bar{A}_1A_0$	A_1A_0	$A_1\bar{A}_0$
	A_3A_2 \ A_1A_0	00	01	11	10
$\bar{A}_3\bar{A}_2$	00	1 (0)	1	1 (3)	2
$\bar{A}_3A_2$	01	1 (4)	5	1 (7)	6
A_3A_2	11	1 (12)	13	1 (15)	14
$A_3\bar{A}_2$	10	1 (8)	9	1 (11)	10

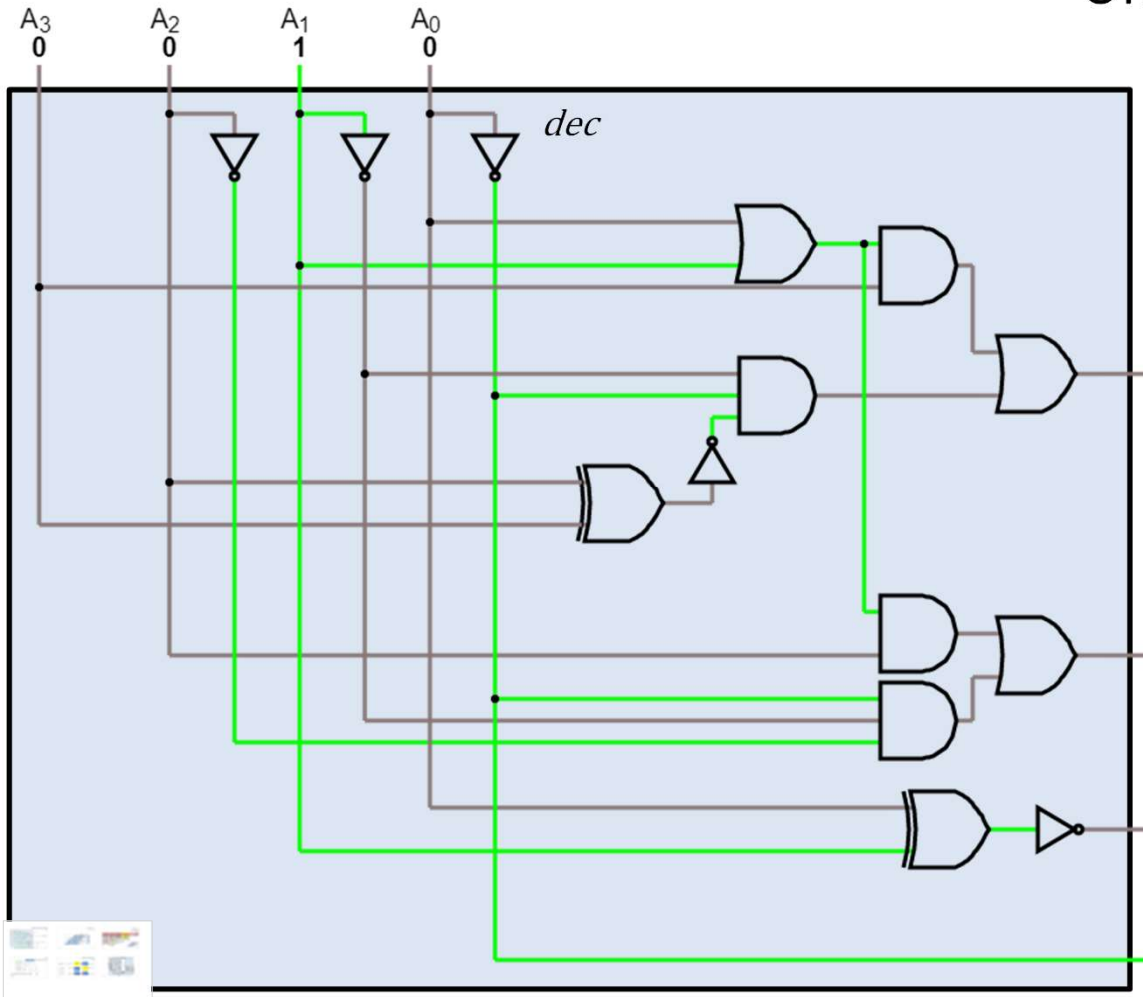
$$\bar{A}_1\bar{A}_0 + A_1A_0$$

$$X_1 = \overline{A_0 \oplus A_1}$$

$$X_0 = \bar{A}_0$$

Unidade de Lógica e Aritmética

dec



$X = A - 1$

0 $X_3 = A_3(A_0 + A_1) + \bar{A}_0\bar{A}_1\overline{(A_2 \oplus A_3)}$

0 $X_2 = A_2(A_0 + A_1) + \bar{A}_0\bar{A}_1\bar{A}_2$

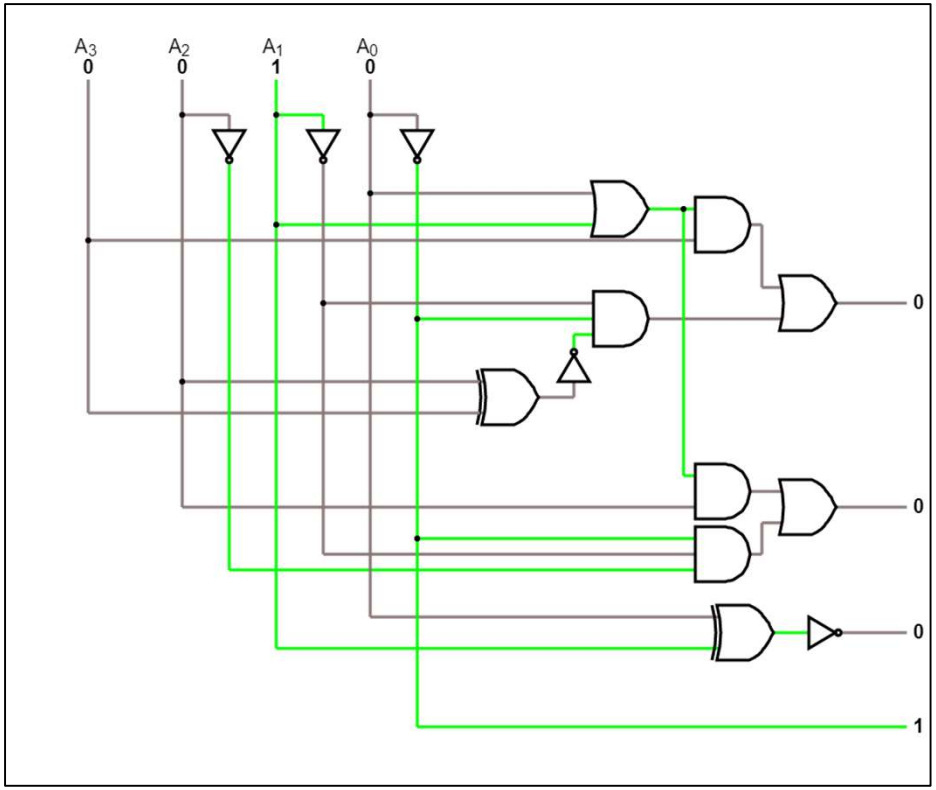
0 $X_1 = \overline{A_0 \oplus A_1}$

1 $X_0 = \bar{A}_0$

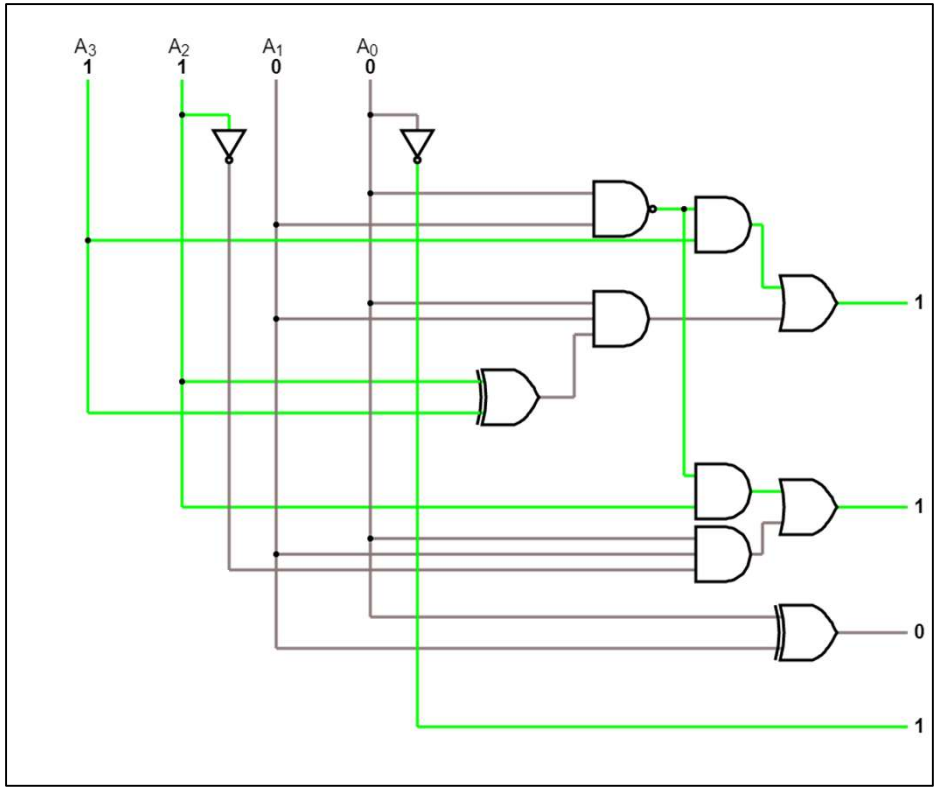
Unidade de Lógica e Aritmética

inc/dec

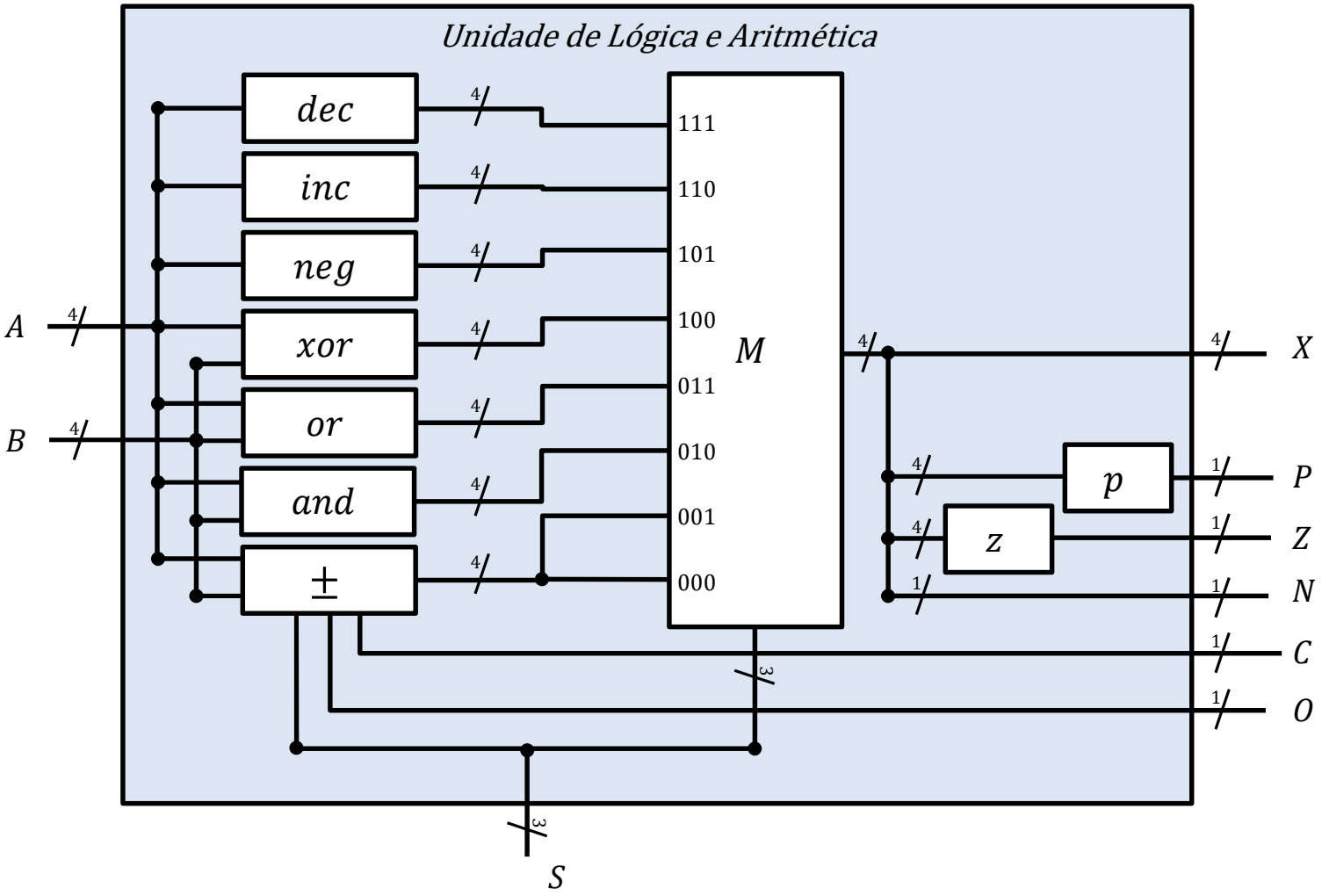
$X = A - 1$



$X = A + 1$



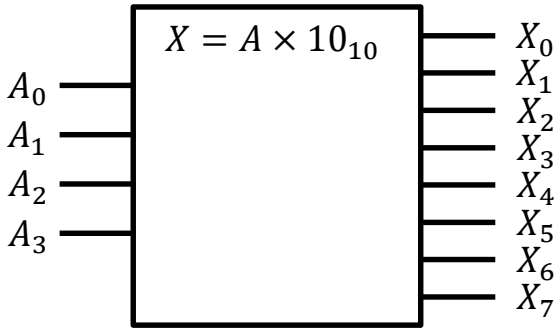
ARQUITETURA DE UMA *ULA*


Unidade de Lógica e Aritmética
dec
inc
neg
xor
or
and
±
A
B
M
111
110
101
100
011
010
001
000
p
z
X
P
Z
N
C
O
S

EXERCÍCIO

Exercício

- Elaborar um circuito que realize a multiplicação por 10:



Exercício

$X = A \times 10$

$X = A \times 10$

$X = A \times (2 + 8)$

$X = A \times 2 + A \times 8$

$X = A \times 2^1 + A \times 2^3$

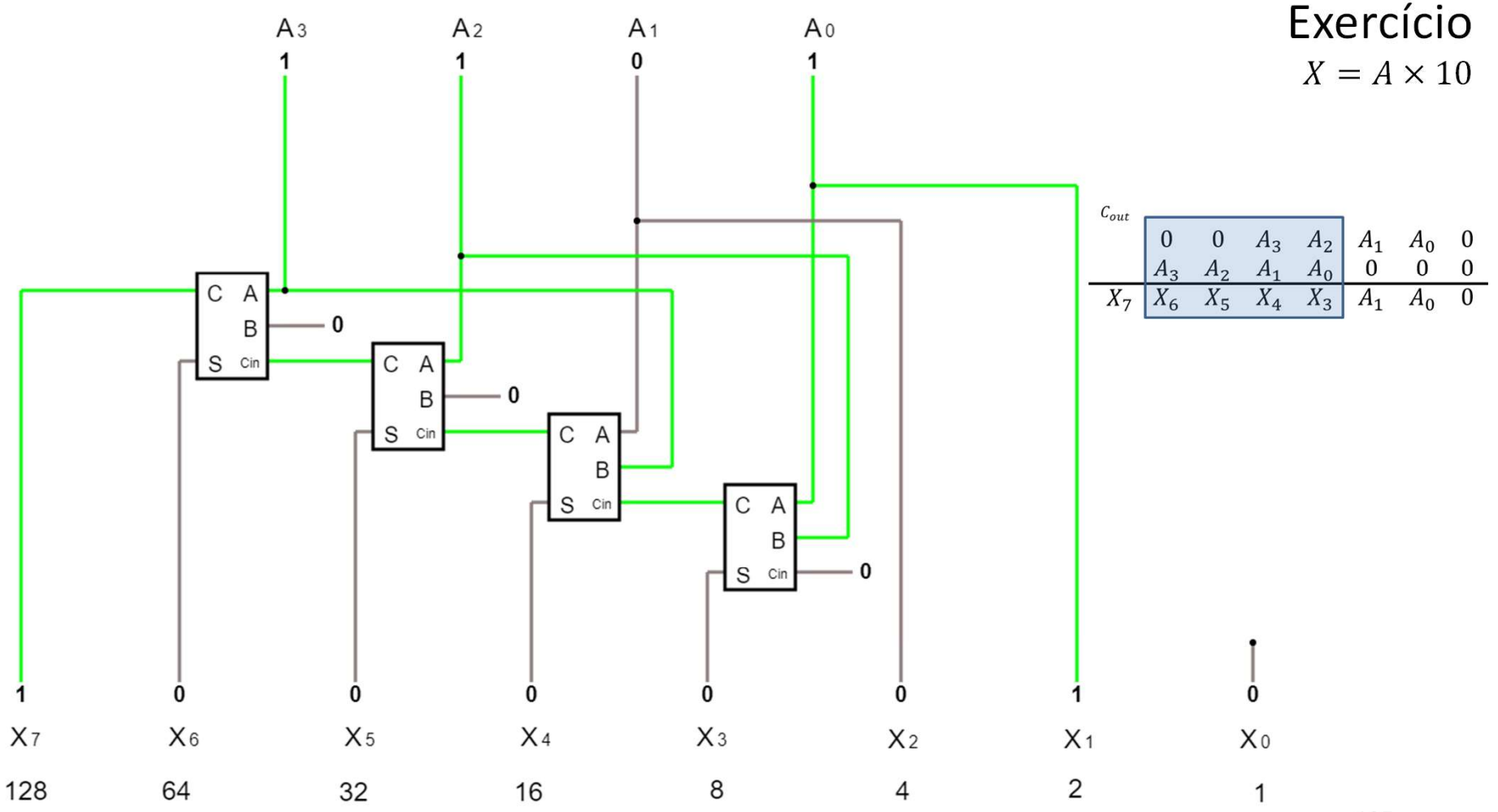
$X = A_3A_2A_1A_0 \times 2^1 + A_3A_2A_1A_0 \times 2^3$

$X = A_3A_2A_1A_0 0 + A_3A_2A_1A_0 000$

C_{out}							
	0	0	A_3	A_2	A_1	A_0	0
	A_3	A_2	A_1	A_0	0	0	0
X_7	X_6	X_5	X_4	X_3	A_1	A_0	0

Exercício

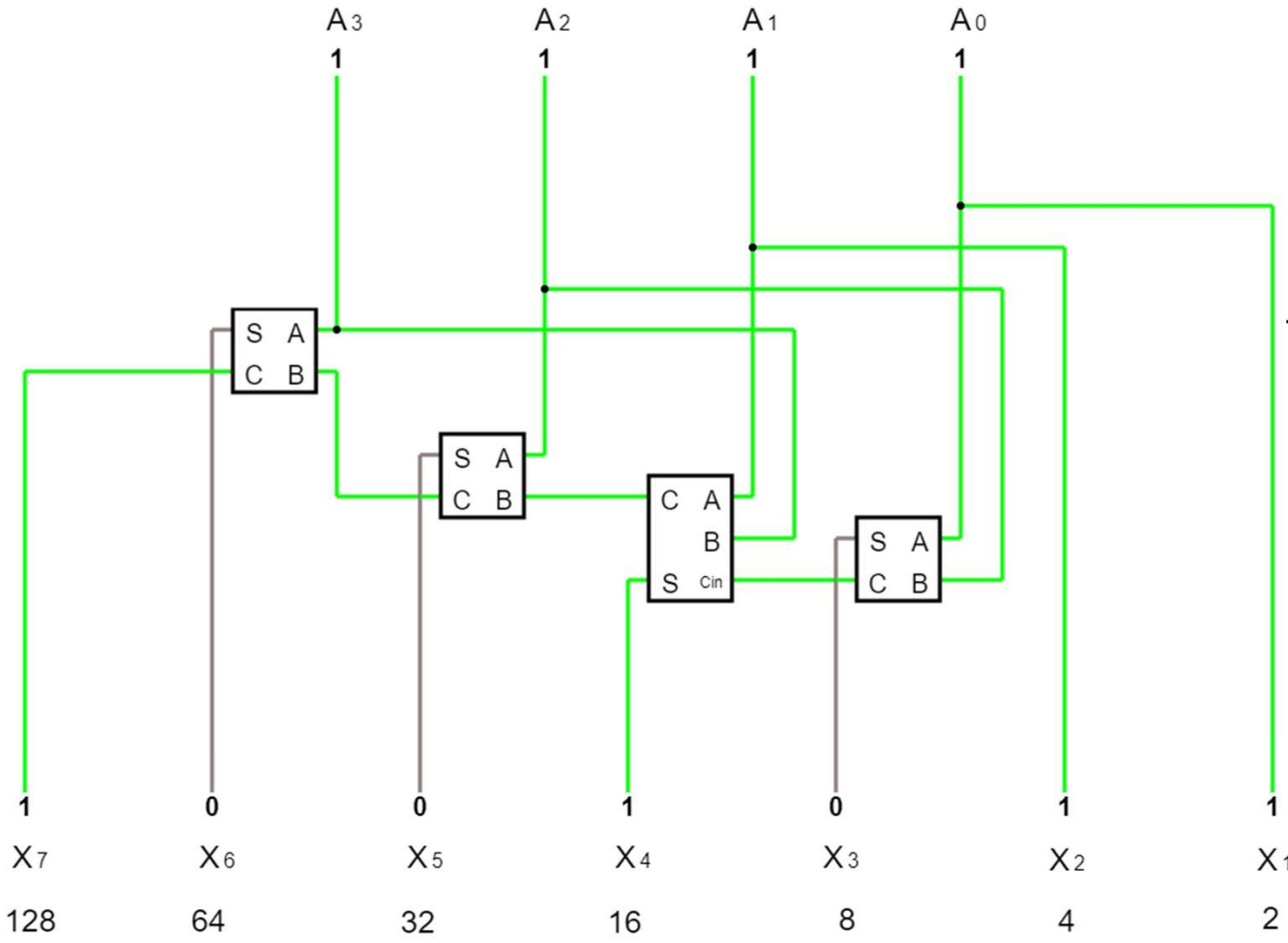
$X = A \times 10$



C_{out}							
	0	0	A_3	A_2	A_1	A_0	0
	A_3	A_2	A_1	A_0	0	0	0
X_7	X_6	X_5	X_4	X_3	A_1	A_0	0

Exercício

$X = A \times 10$

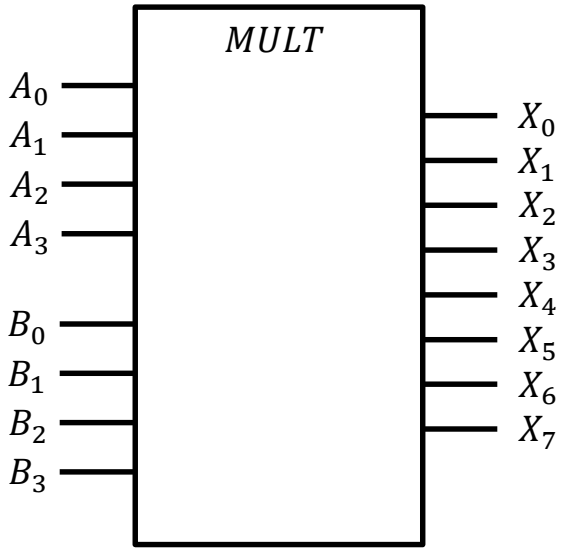


C_{out}							
	0	0	A_3	A_2	A_1	A_0	0
	A_3	A_2	A_1	A_0	0	0	0
X_7	X_6	X_5	X_4	X_3	A_1	A_0	0

MULTIPLICADOR

Multiplicação

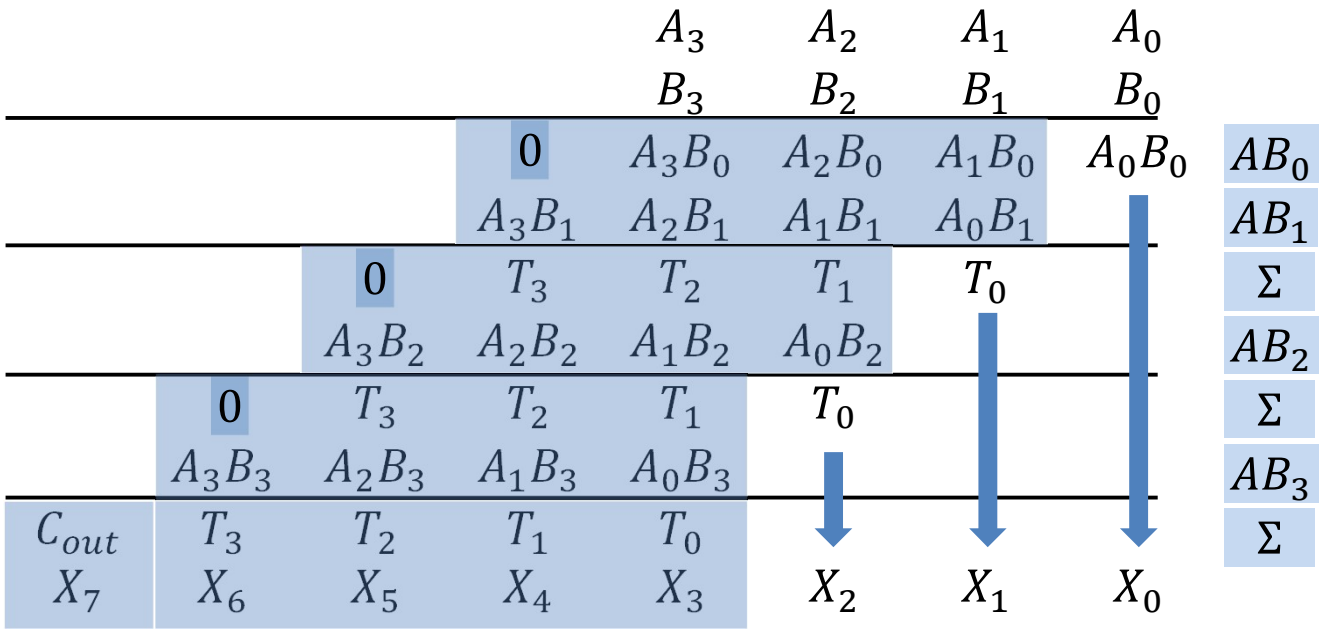
- A multiplicação de dois inteiros sem sinal de n bits produz um resultado de $2 \times n$ bits:



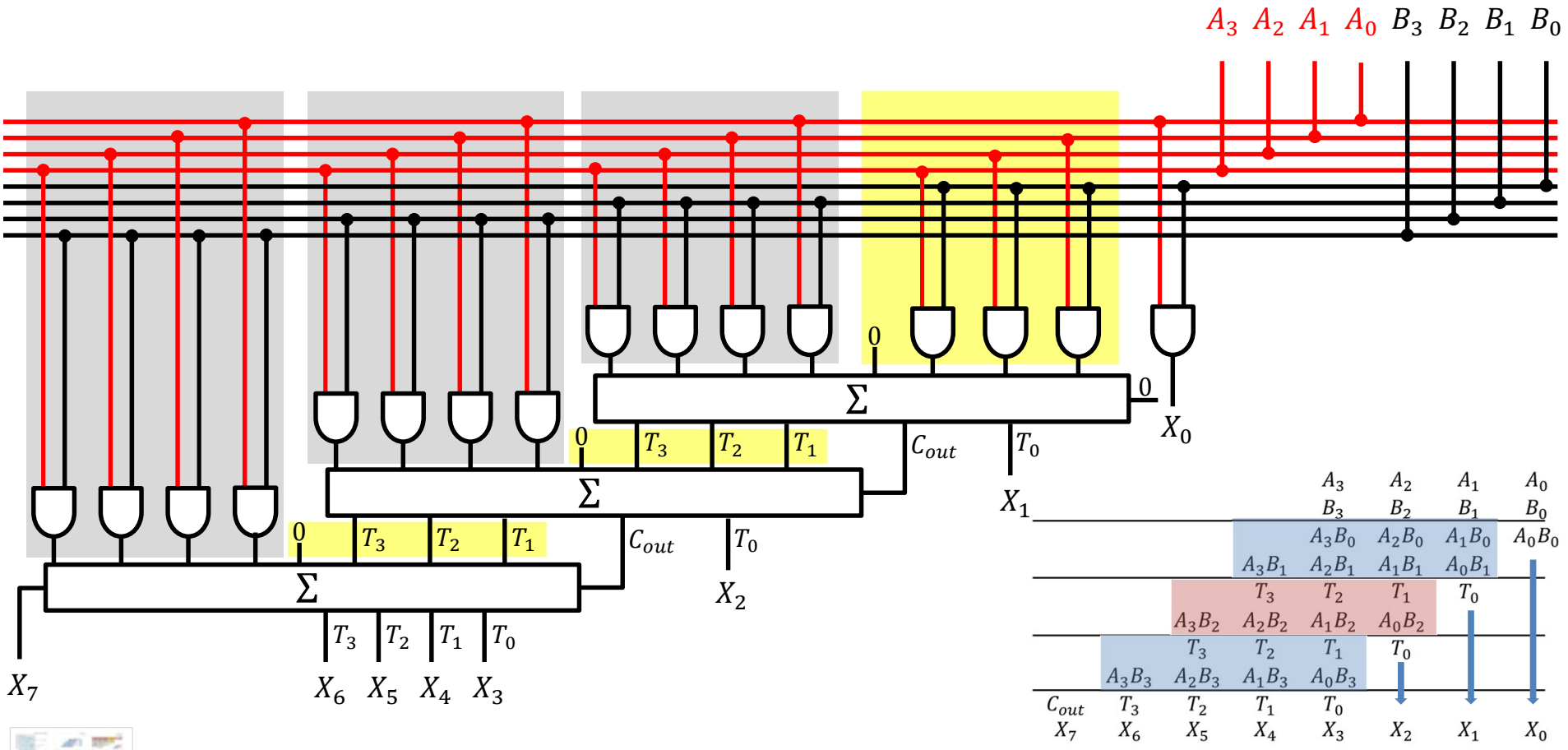
Multiplicação

				A_3	A_2	A_1	A_0	
				B_3	B_2	B_1	B_0	
				A_3B_0	A_2B_0	A_1B_0	A_0B_0	AB_0
			A_3B_1	A_2B_1	A_1B_1	A_0B_1		AB_1
		A_3B_2	A_2B_2	A_1B_2	A_0B_2			AB_2
	A_3B_3	A_2B_3	A_1B_3	A_0B_3				AB_3
X_7	X_6	X_5	X_4	X_3	X_2	X_1	X_0	

Multiplicação



Multiplicação



PLD

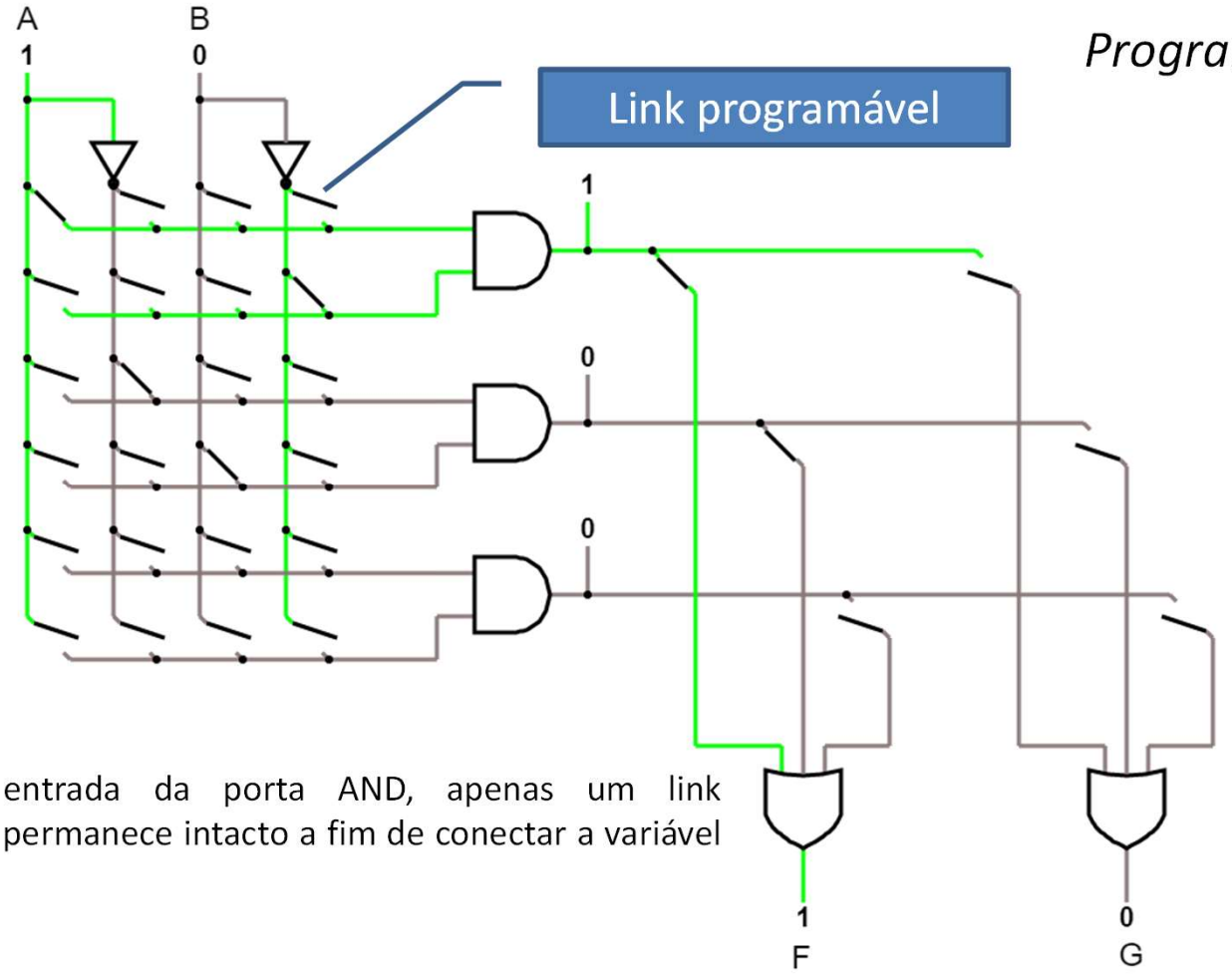
Programmable Logic Device

PLA
Programmable Logic Array

- *Programmable Logic Array* ou Matriz Lógica Programável são circuitos com arranjo lógico pré-formado;
- Disponibiliza um mecanismo de conexão das lógicas;
- É uma alternativa rápida para a criação de circuitos;
- As funções são implementadas no formato Soma de Produtos;
- As funções podem combinar variáveis e seus complementos;

PLA

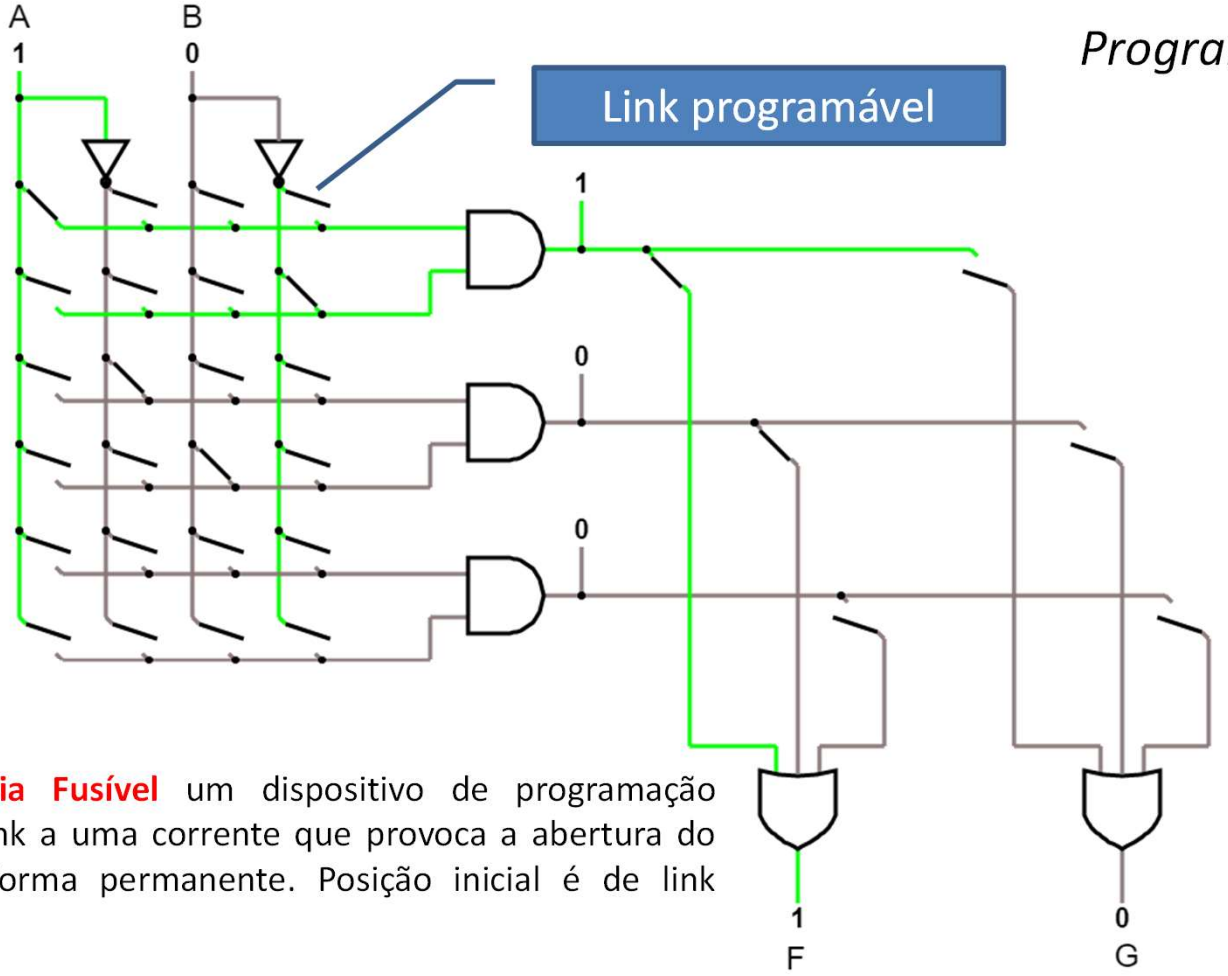
Programmable Logic Array

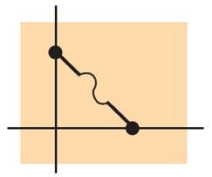


Para cada entrada da porta AND, apenas um link programável permanece intacto a fim de conectar a variável desejada.

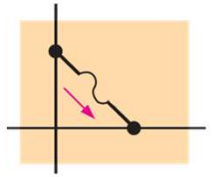
PLA

Programmable Logic Array

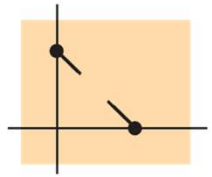




(a) Fuse intact before programming



(b) Programming current

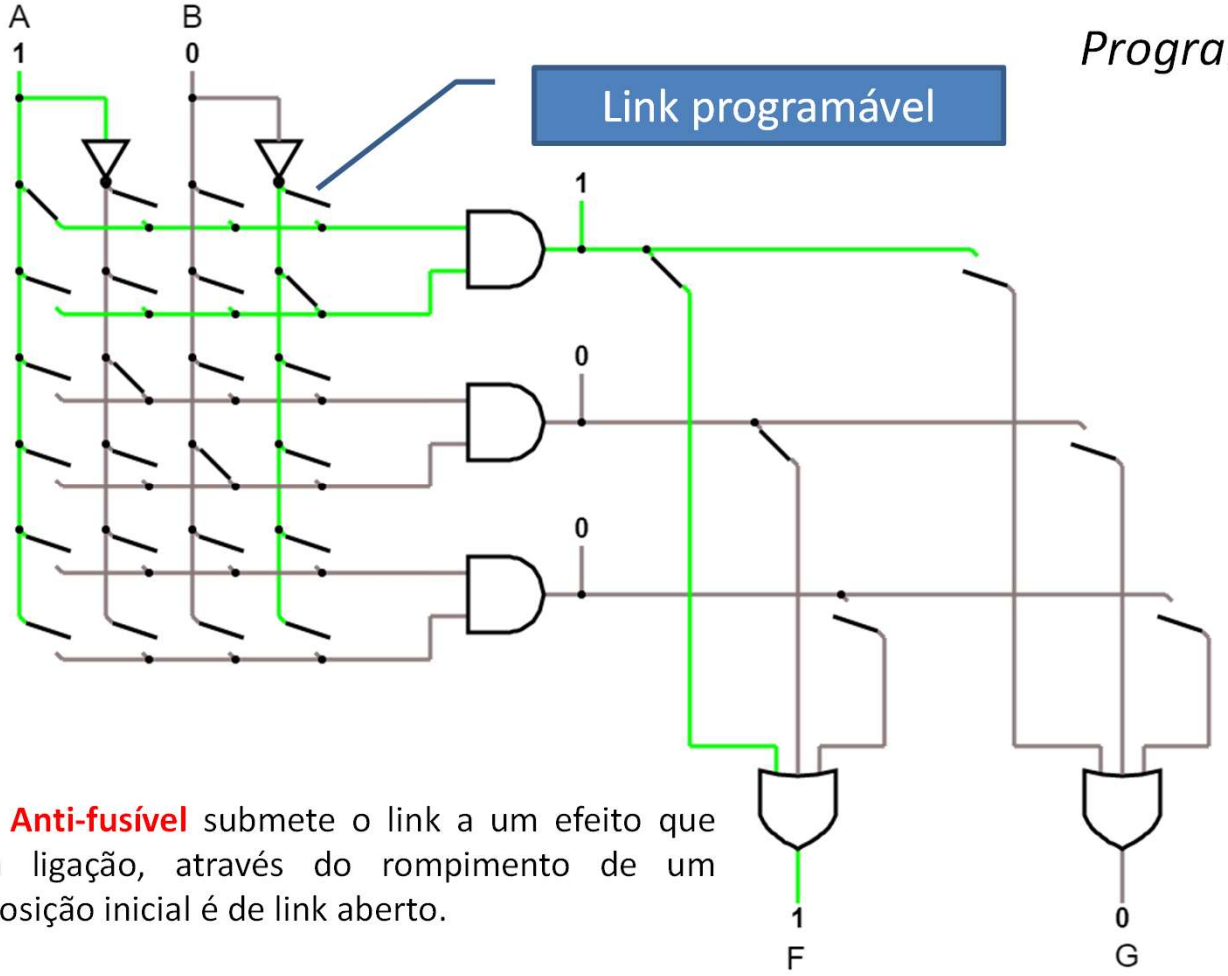


(c) Fuse open after programming

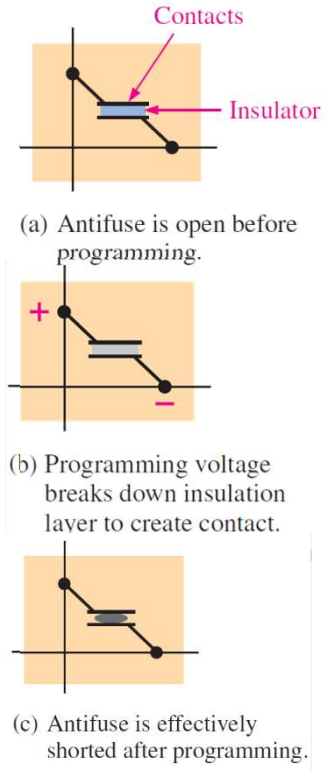
Na **Tecnologia Fusível** um dispositivo de programação submete o link a uma corrente que provoca a abertura do circuito de forma permanente. Posição inicial é de link fechado.

PLA

Programmable Logic Array



A **Tecnologia Anti-fusível** submete o link a um efeito que estabelece a ligação, através do rompimento de um isolamento. Posição inicial é de link aberto.



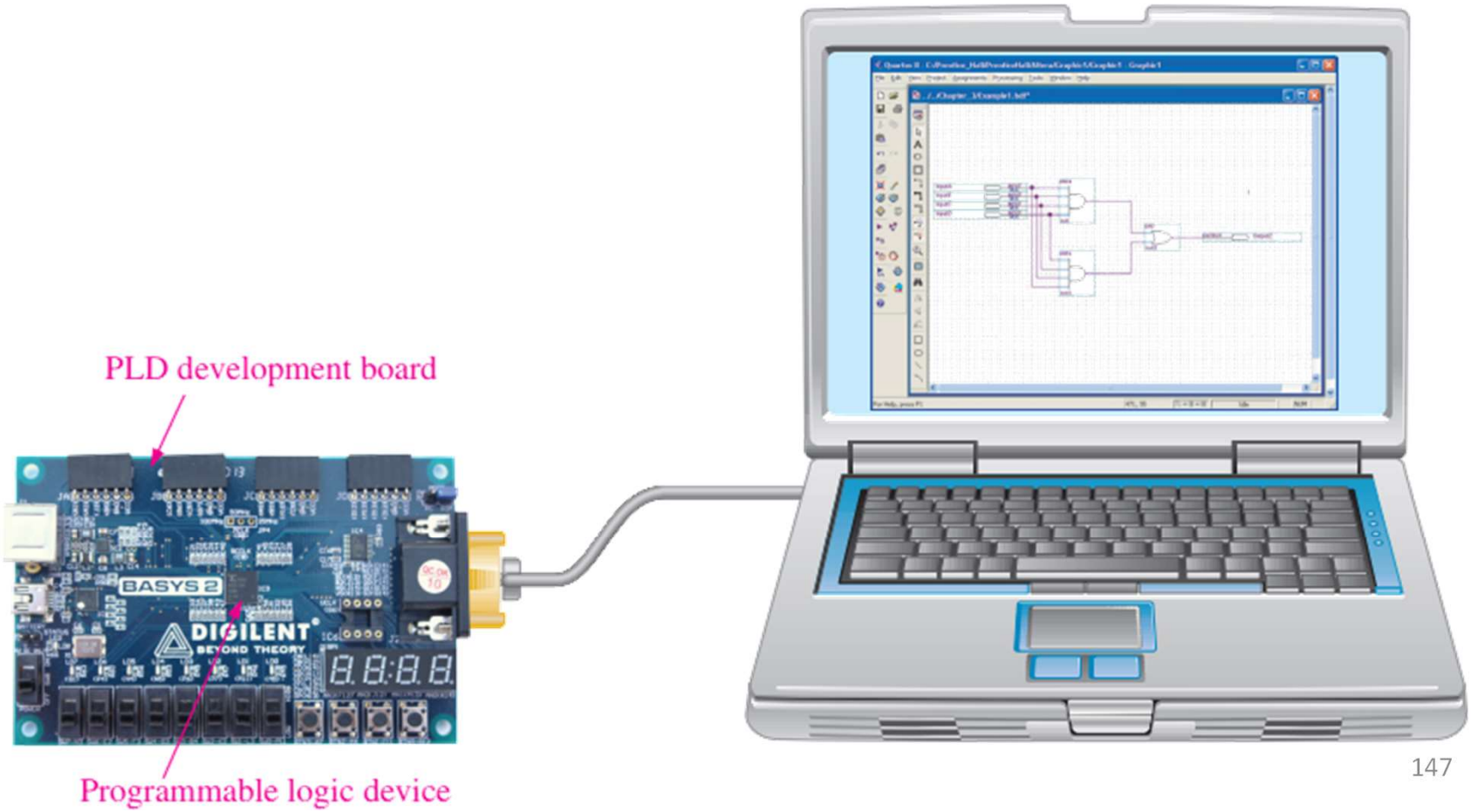
(a) Antifuse is open before programming.

(b) Programming voltage breaks down insulation layer to create contact.

(c) Antifuse is effectively shorted after programming.

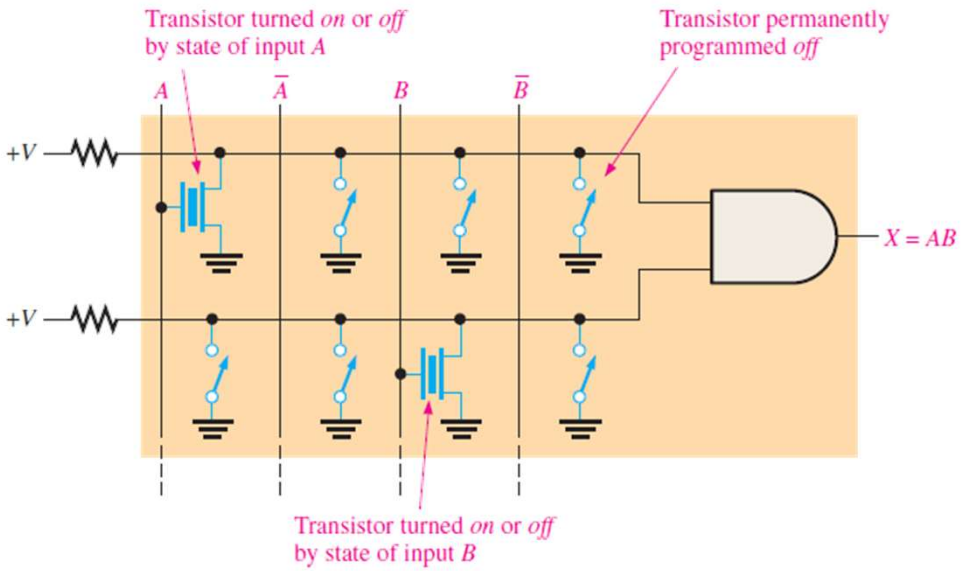
PLD

Programmable Logic Device



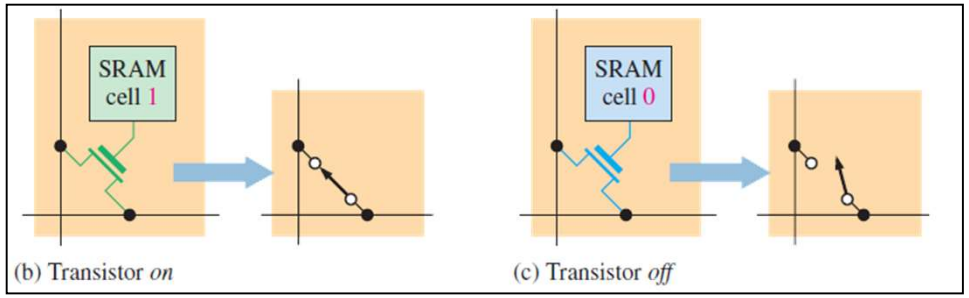
PLA
Programmable Logic Array

- Na **Tecnologia EPROM** (*Electrically Programmable Read-Only Memories*) um transistor faz o papel do link;
- A maioria dos dispositivos de lógica programável de tecnologia EPROM é programável uma única vez (OTP).
- No entanto, chips com janelas podem ser apagadas com luz ultravioleta e reprogramadas.
- A tecnologia EPROM utiliza um tipo especial de transistor MOS, conhecido como *floating gate transistor*, como link programável.



PLA
Programmable Logic Array

- Na **Tecnologia EEPROM** (*Erasable Electrically Programmable Read-Only Memories*) é semelhante à EPROM pois também utiliza um transistor de porta flutuante;
- A diferença é que a EEPROM pode ser apagada e reprogramada eletricamente sem a necessidade de luz UV.
- A **Tecnologia Flash** é baseada também em link de transistor, sendo não volátil e reprogramável.
- É semelhante à EEPROM, porém mais rápidos e disponíveis em dispositivos de maior densidade.
- A **Tecnologia SRAM** utiliza processo semelhante nas memórias RAM estáticas (*Static Random-Access Memories*).
- Uma célula de memória do tipo SRAM é usada para ativar ou desativar um transistor, que por sua vez conecta ou desconecta linhas e colunas.

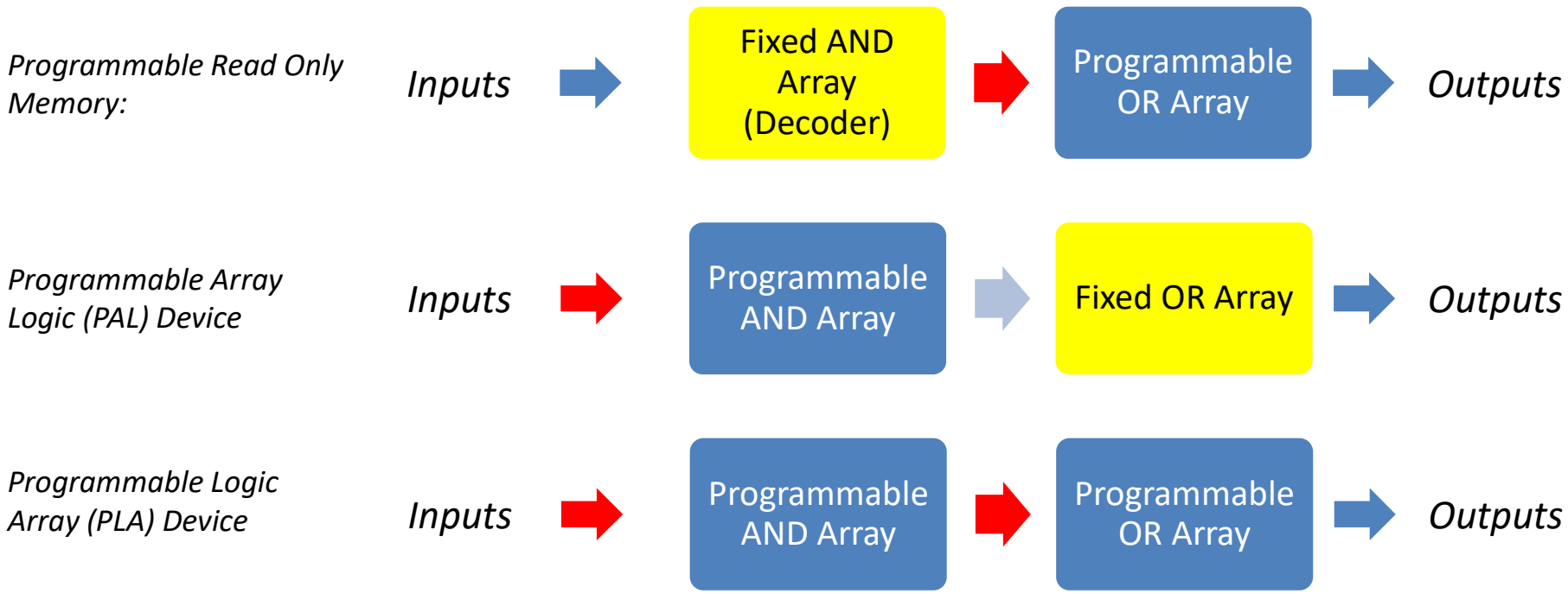


PLA
Programmable Logic Array

- A tecnologia SRAM difere das demais tecnologias por utilizar uma tecnologia volátil.
- Isso significa que uma célula SRAM não retém os dados quando a energia é interrompida.
- Quando energizado, os dados programados serão carregados na memória; e consequentemente configurarão o PLD baseado no conteúdo da SRAM.
- As tecnologias de processo de fusível, anti-fusível, EPROM, EEPROM e flash são não-voláteis, portanto, elas retêm sua programação quando a energia for interrompida.
- Um fusível fica permanentemente aberto, um anti-fusível fica permanentemente fechado e os transistores de porta flutuante usados em matrizes baseadas em EPROM e EEPROM podem reter seu estado ligado ou desligado indefinidamente.

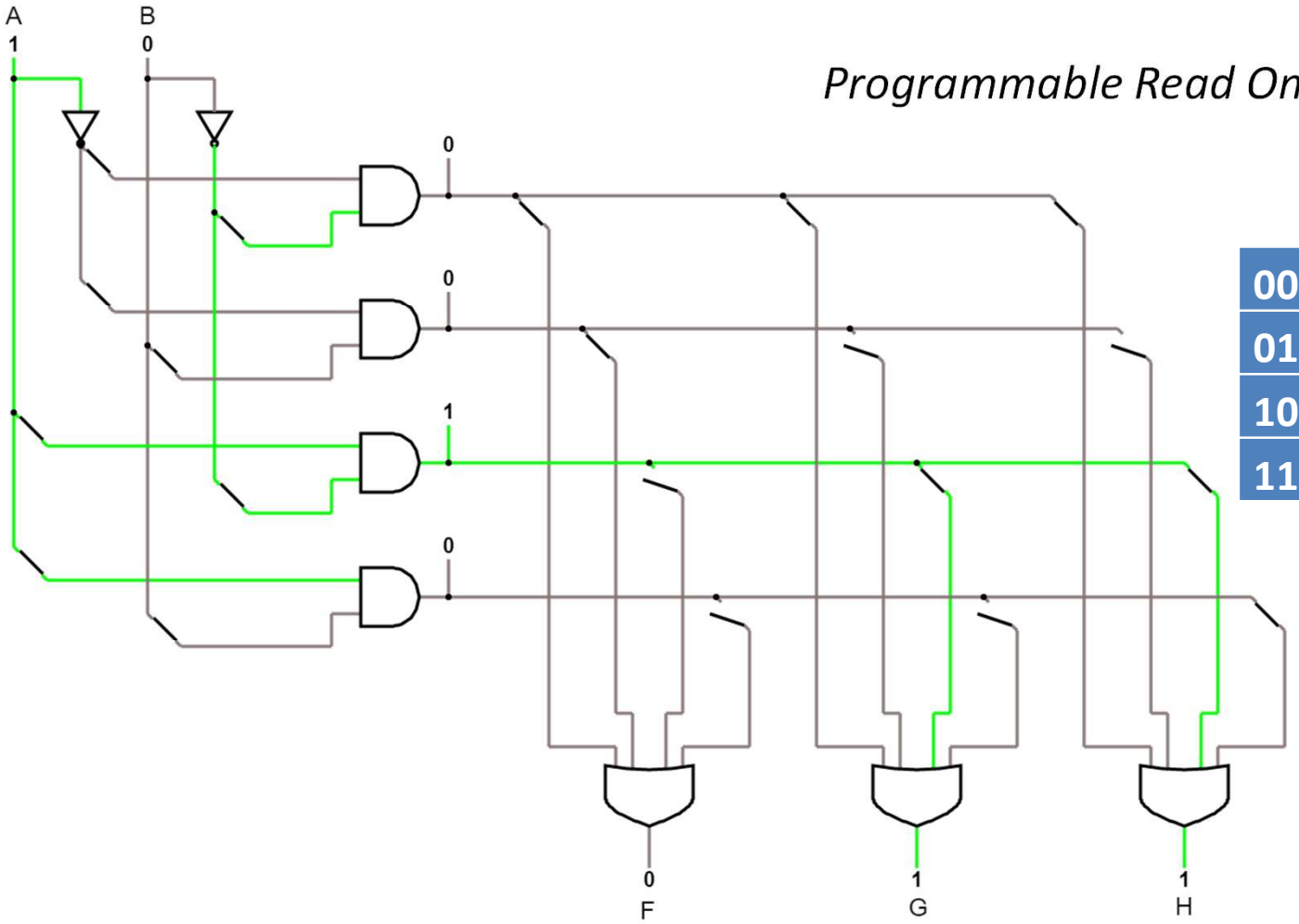
Disponíveis Programáveis

Tipos



PROM

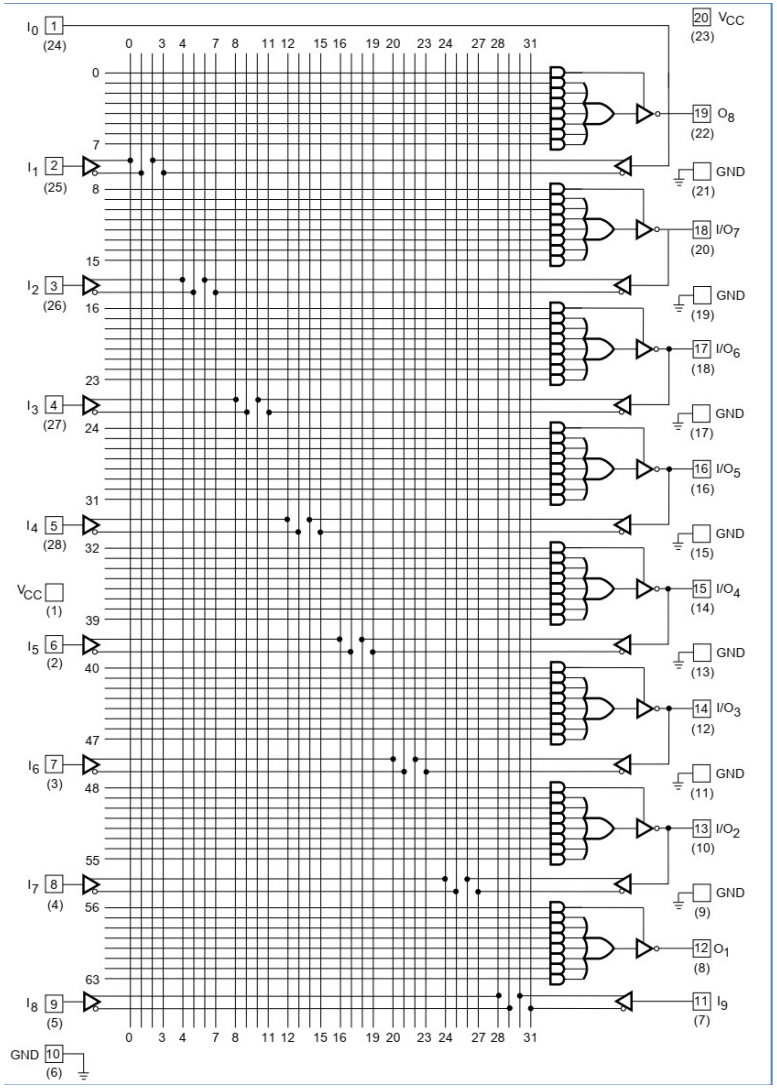
Programmable Read Only Memory

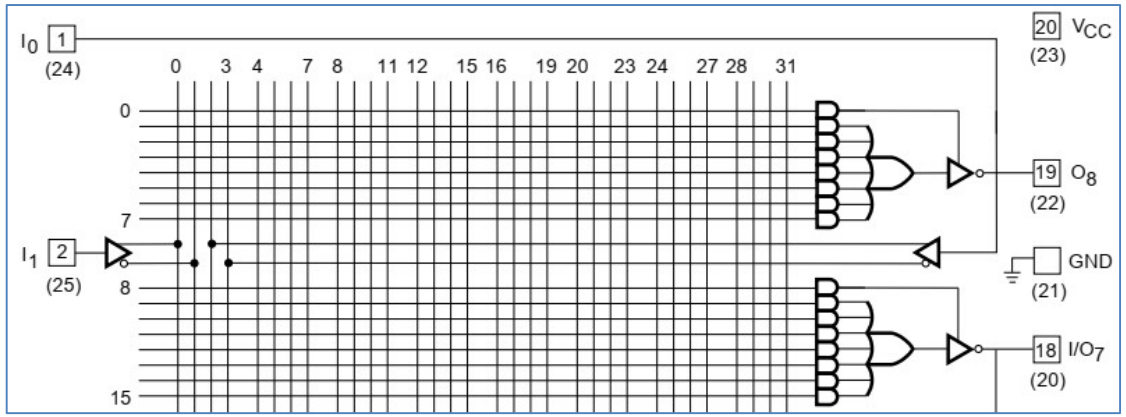


00	1	1	1
01	1	0	0
10	0	1	1
11	0	0	1

PAL

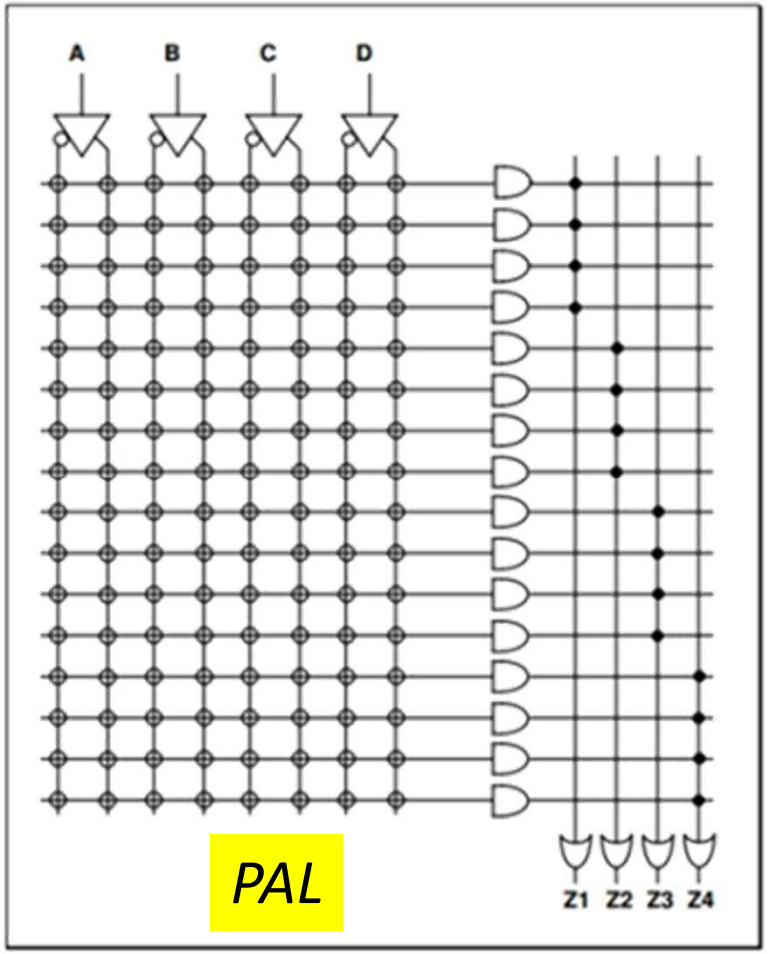
PAL16L8 Family

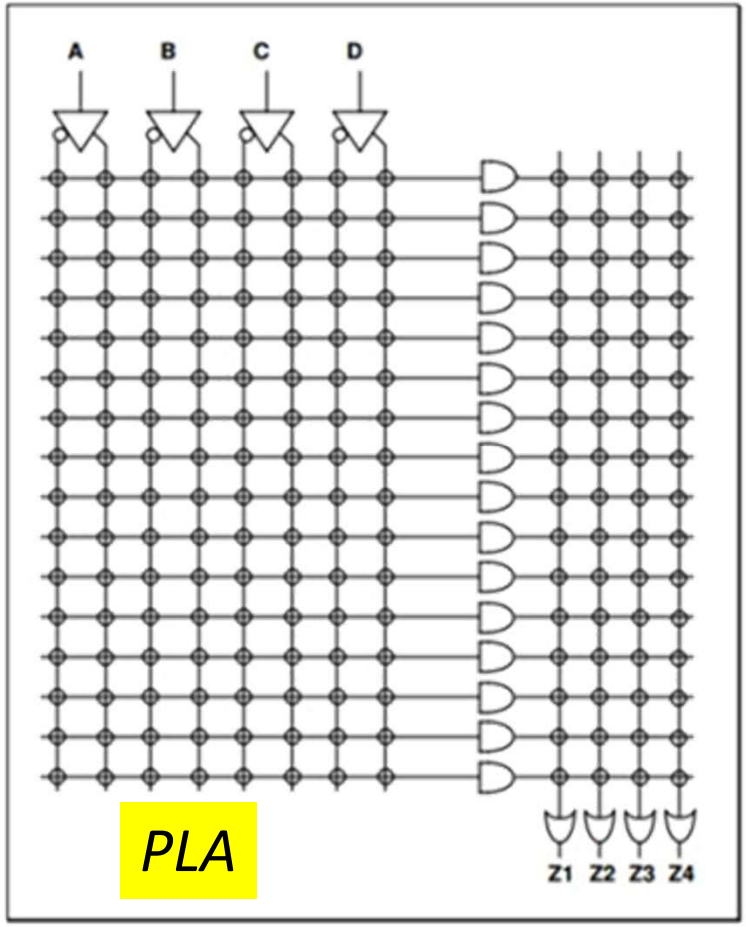




Disponíveis Programáveis

PAL versus PLA

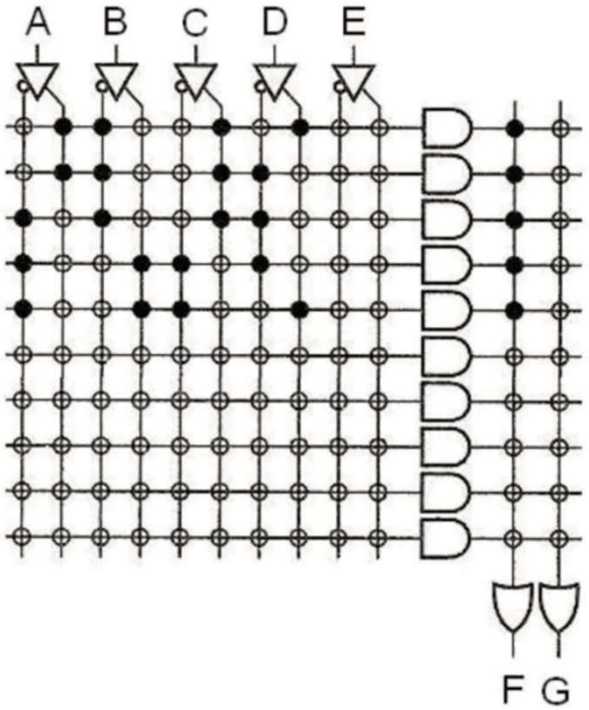




Programmable Logic Array

Exercício

41) [POSCOMP 2010] Considere o circuito digital apresentado no diagrama a seguir. Ressalte-se que, por convenção, chaves representadas por círculos escuros representam conexões fechadas e chaves representadas por círculos vazados representam conexões abertas.



Programmable Logic Array

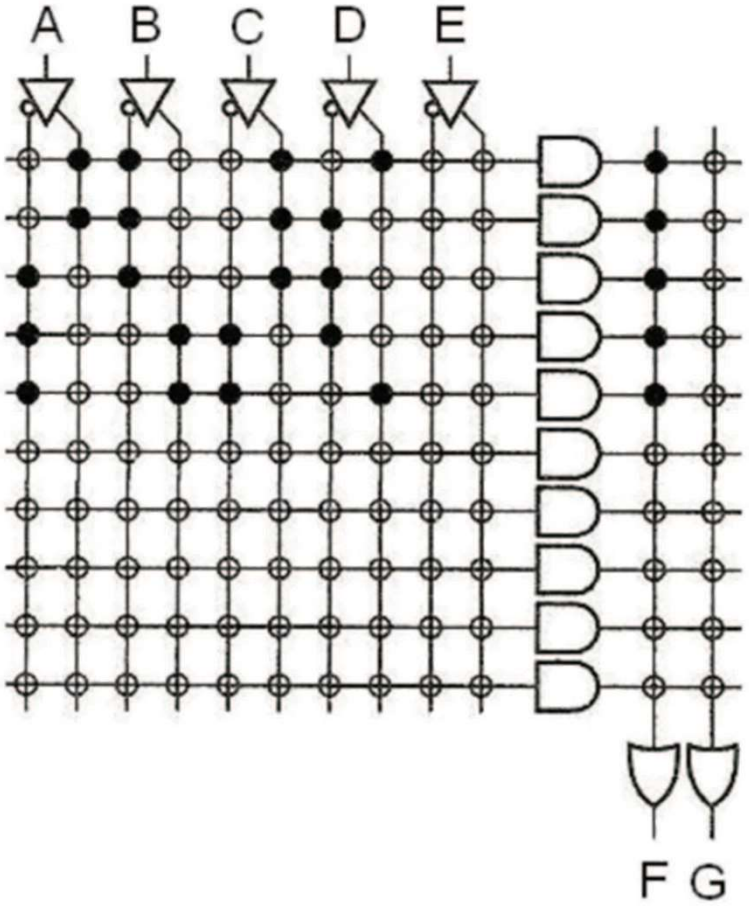
Exercício

Assinale a alternativa correta.

a) O circuito representa uma implementação em PAL da função $F = \bar{A}B\bar{C} + \mathrm{B}\bar{C}D + \mathrm{A}\bar{B}C$.
b) O circuito representa uma implementação em FPGA da função $F = \bar{A}B\bar{C} + \mathrm{B}\bar{C}D + \mathrm{A}\bar{B}$.
c) O circuito representa uma implementação em PLA da função $F = \bar{A}B\bar{C} + \mathrm{B}\bar{C}D + \mathrm{A}\bar{B}C$.
d) O circuito representa uma implementação em PAL da função $G = \bar{A}B\bar{C} + \mathrm{B}\bar{C}D + \mathrm{A}\bar{B}C$.
e) O circuito representa uma implementação em PLA da função $G = \bar{A}B\bar{C} + \mathrm{B}\bar{C}D + \mathrm{A}\bar{B}C$.

Programmable Logic Array

Exercício



$A\bar{B}CD$

$A\bar{B}C\overline{D}$

$\bar{A}\bar{B}C\overline{D}$

$\bar{A}B\bar{C}\overline{D}$

$\bar{A}B\bar{C}D$

$A\bar{B}C$

$\bar{B}C\overline{D}$

$\bar{A}B\bar{C}$

$$F = \bar{A}B\bar{C} + \bar{B}C\overline{D} + A\bar{B}C$$

Programmable Logic Array

Exercício

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

Assinale a alternativa correta.

a) O circuito representa uma implementação em PAL da função $F = \bar{A}B\bar{C} + \mathrm{B}\bar{C}D + \mathrm{A}\bar{B}C.$

b) O circuito representa uma implementação em FPGA da função $F = \bar{A}B\bar{C} + \mathrm{B}\bar{C}D + \mathrm{A}\bar{B}C.$

c) O circuito representa uma implementação em PLA da função $F = \bar{A}B\bar{C} + \mathrm{B}\bar{C}D + \mathrm{A}\bar{B}C.$ **?**

d) O circuito representa uma implementação em PAL da função $G = \bar{A}B\bar{C} + \mathrm{B}\bar{C}D + \mathrm{A}\bar{B}C.$

e) O circuito representa uma implementação em PLA da função $G = \bar{A}B\bar{C} + \mathrm{B}\bar{C}D + \mathrm{A}\bar{B}C.$