

UNIOESTE
Ciência da Computação

Sistemas Digitais
Circuitos Combinacionais

Prof. Jorge Habib El Khouri
Prof. Antonio Marcos Hachisuca

2020/2021

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.

Sumário

1. Revisão – Sistemas de Numeração
2. Revisão – Representação de Dados
3. Revisão – Operações com Binários
4. Álgebra Booleana
5. Simplificação de Expressões
6. Mapa de Karnaugh
7. Elementos Lógicos Universais
8. Circuitos Combinacionais
9. Circuitos Sequenciais

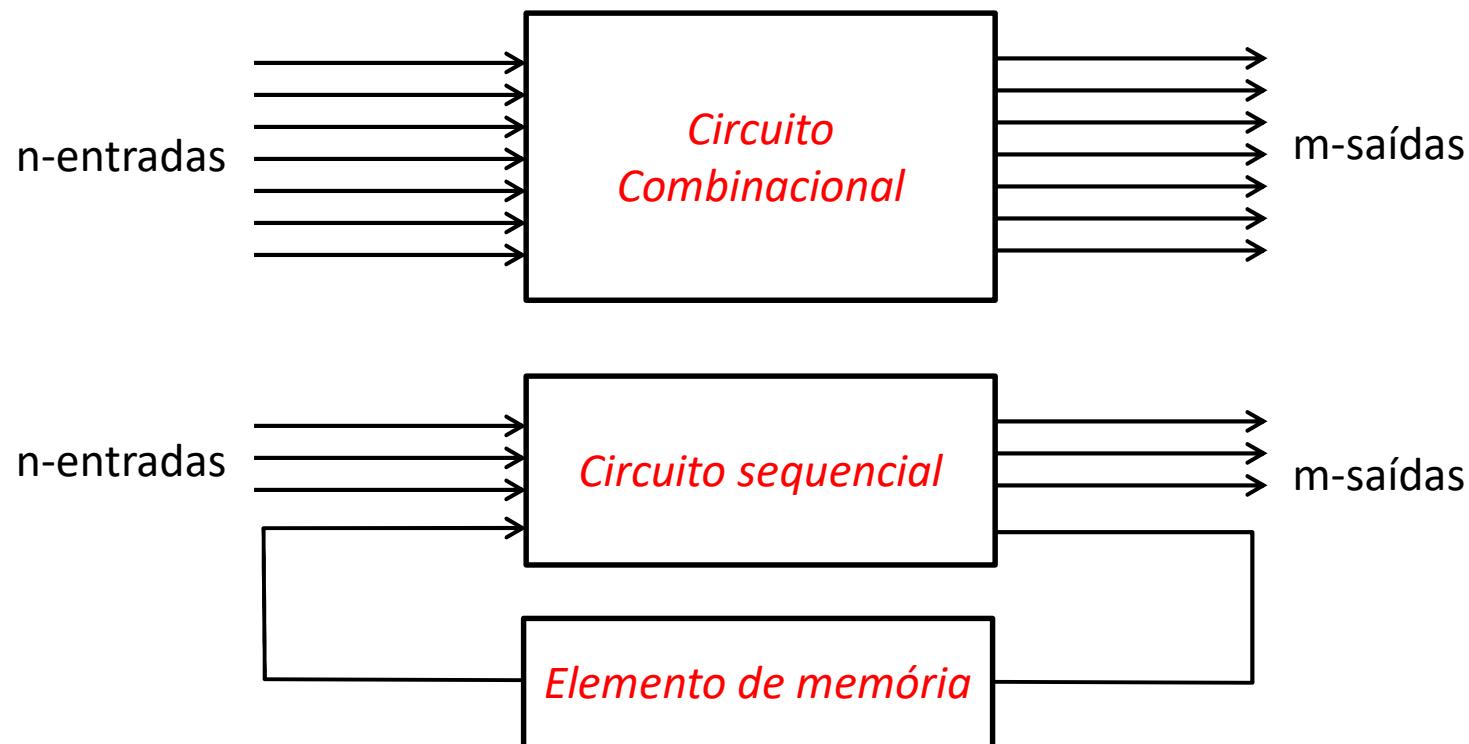
Circuitos Combinacionais

- 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



Sumário

- 1. Revisão – Sistemas de Numeração
- 2. Revisão – Representação de Dados
- 3. Revisão – Operações com Binários
- 4. Álgebra Booleana
- 5. Simplificação de Expressões
- 6. Mapa de Karnaugh
- 7. Elementos Lógicos Universais
- 8. Circuitos Combinacionais
- 9. Circuitos Sequenciais

- 1. Somador / Subtrator
- 2. Comparadores
- 3. Codificador/decodificador
- 4. Multiplexador/Demux
- 5. Geradores de paridade
- 6. ULA
- 7. Multiplicadores / Divisores
- 8. PLD/PLA/ROM

Regras Básicas da Álgebra Booleana

Comutativa

$$A + B = B + A$$

$$AB = BA$$

Associativa

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

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

Distributiva

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

$$A + 0 = A$$

$$A + 1 = 1$$

$$A + A = A$$

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

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

$$A \cdot 0 = 0$$

$$A \cdot 1 = A$$

$$A \cdot A = A$$

$$A \cdot \bar{A} = 0$$

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

$$A \oplus 0 = A$$

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

$$A \oplus A = 0$$

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

$$A + AB = A$$

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

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

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

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

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

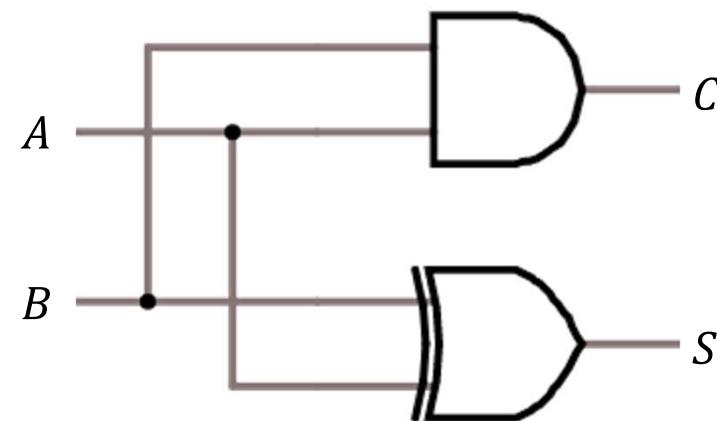
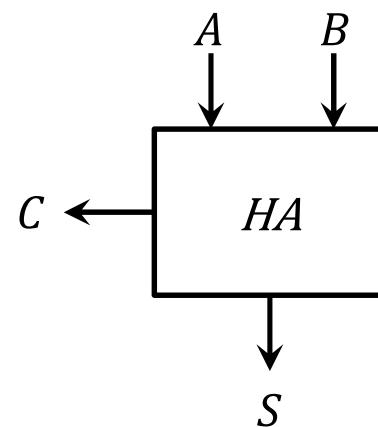
Circuito Somador *Half-Adder*

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

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

$$S = A \oplus B$$

$$C = AB$$



Circuito Somador Full-Adder

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

$$S = ??$$

$$C_{out} = ??$$

AB \ C

	0	1
00		1
01	1	
11		1
10	1	

$$S = \bar{A}\bar{B}C_{in} + \bar{A}B\bar{C}_{in} + A\bar{B}\bar{C}_{in} + ABC_{in}$$

$$S = \bar{A}(\bar{B}C_{in} + B\bar{C}_{in}) + A(\bar{B}\bar{C}_{in} + BC_{in})$$

$$S = \bar{A}(B \oplus C_{in}) + A(\bar{B} \oplus \bar{C}_{in})$$

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

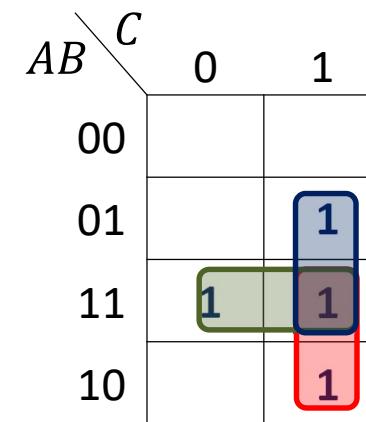
$$\begin{array}{r} C_{out} \\ + \\ \begin{array}{r} A \\ B \\ \hline S \end{array} \end{array}$$

Circuito Somador Full-Adder

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

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

$$C_{out} = ??$$



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

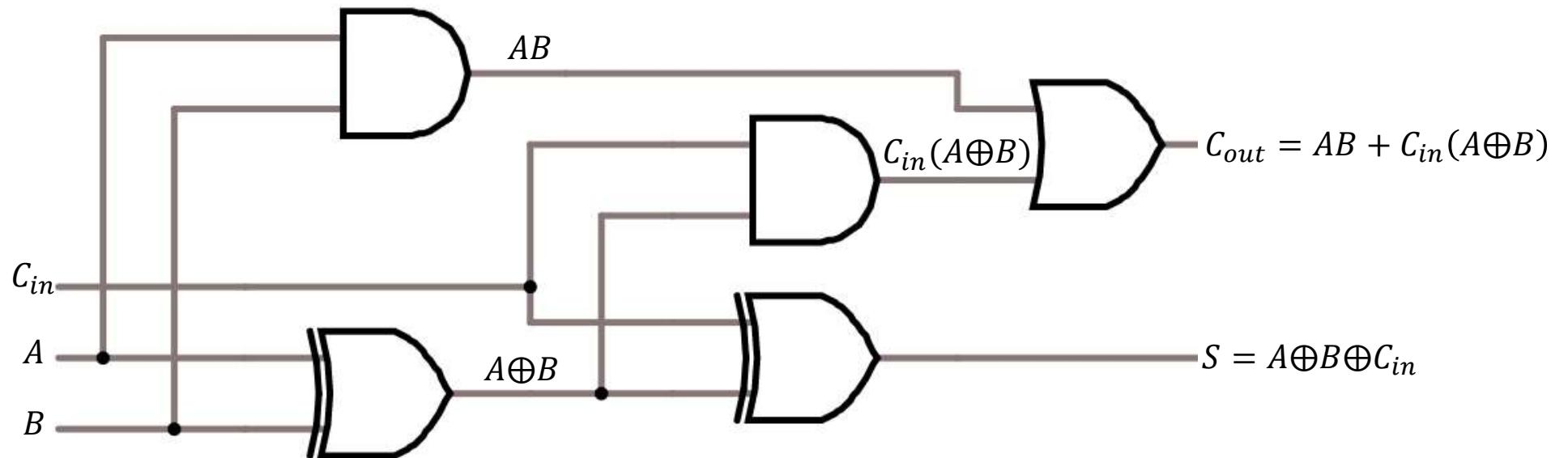
$$C_{out} = AB(\bar{C}_{in} + C_{in}) + C_{in}(\bar{A}B + A\bar{B})$$

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

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

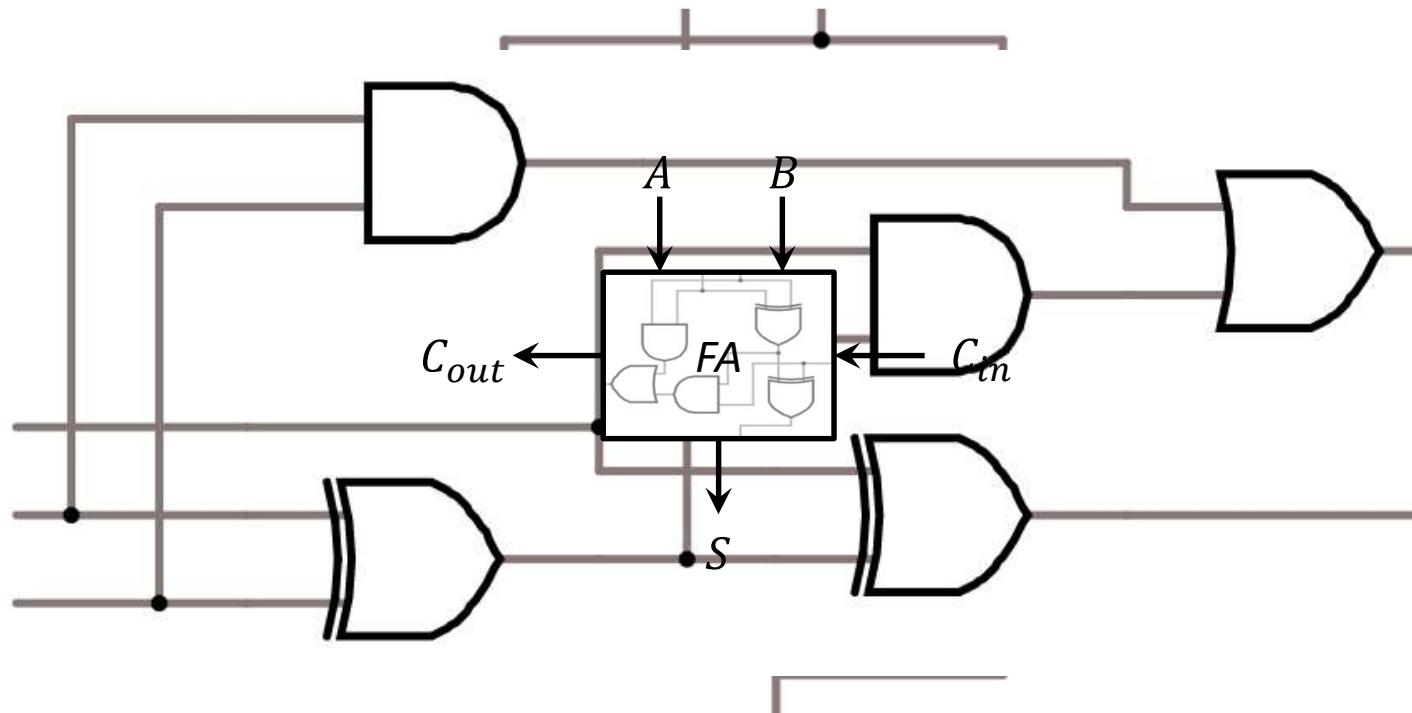
$$C_{out} = AB + C_{in}(A + B)$$

Circuito Somador *Full-Adder*



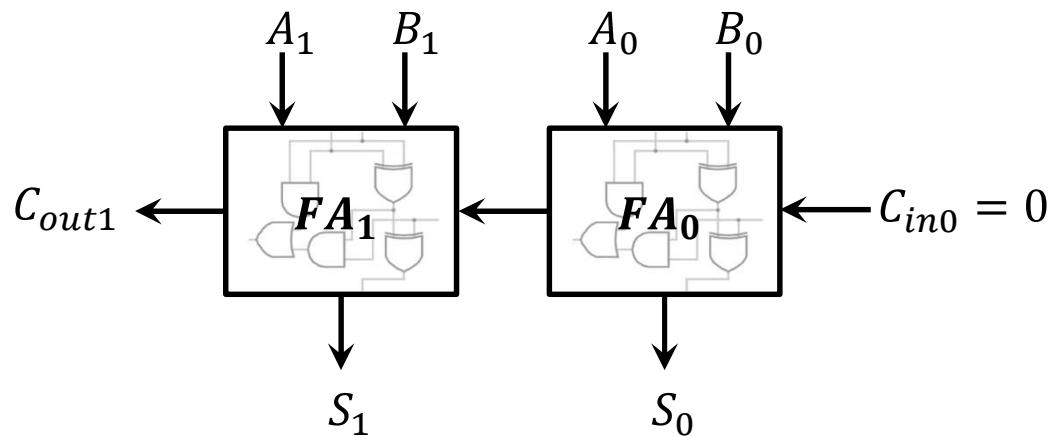
Circuito Somador *Full-Adder*

Somador de 1 bit:



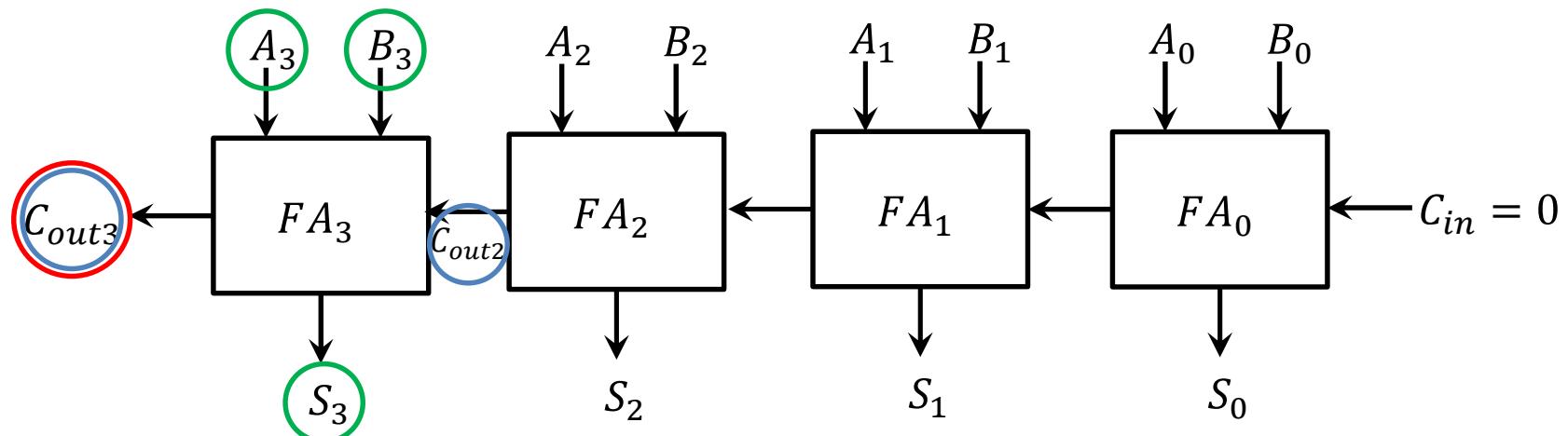
Circuito Somador *Full-Adder*

Somador de 2 bits:



Circuito Somador *Full-Adder*

Somador de 4 bits: $S_{0...3} = A_{0...3} + B_{0...3}$



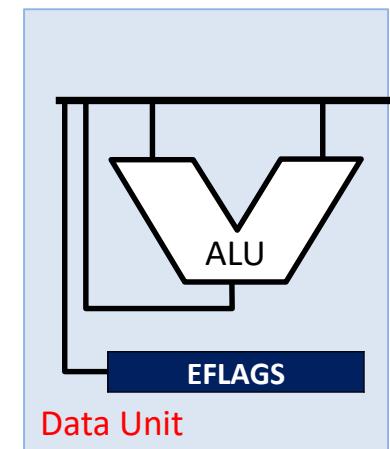
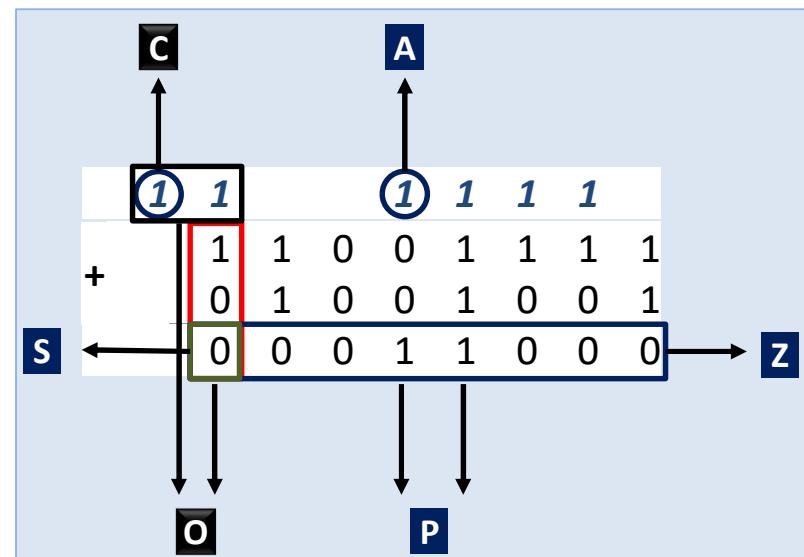
CARRY = flag para soma de inteiros sem sinal: se 1 então Erro

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

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

Círculo Somador Full-Adder

C	Valor do último Carry
P	Paridade par (#1 é par)
A	Carry Auxiliar do 4º para o 5º bit
Z	Resultado zero
S	Bit de Sinal
O	Qualidade para inteiro com sinal



EFLAGS																															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	I D	V I P	V I F	A C	V M	R F	0	N T	I O P L	O F	D F	I F	T F	S F	Z F	0	A F	0	P F	1	C F	

Círculo Somador

Exercício

Obter a expressão que fornece o flag de qualidade ($OVFL - O$) do resultado R em relação aos operandos A e B para soma de inteiros com sinal.

Circuito Somador *Full-Adder*

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

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

$$OVFL = S_a S_b \bar{S}_r + \bar{S}_a \bar{S}_b S_r$$

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

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

Circuito Somador Full-Adder

SINAIS			$C_3C_2C_1C_0$
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

$$\begin{array}{r}
 0001 \\
 0101 \\
 + 0001 \\
 \hline
 0110
 \end{array}$$

$$\begin{array}{r}
 1111 \\
 1111 \\
 + 0001 \\
 \hline
 0000
 \end{array}$$

$$\begin{array}{r}
 0111 \\
 0111 \\
 + 0001 \\
 \hline
 1000
 \end{array}$$

$$\begin{array}{r}
 0011 \\
 1011 \\
 + 0001 \\
 \hline
 1100
 \end{array}$$

$$\begin{array}{r}
 1101 \\
 0101 \\
 + 1101 \\
 \hline
 0010
 \end{array}$$

$$\begin{array}{r}
 1001 \\
 1101 \\
 + 1001 \\
 \hline
 0110
 \end{array}$$

$$\begin{array}{r}
 0001 \\
 0101 \\
 + 1001 \\
 \hline
 1110
 \end{array}$$

$$\begin{array}{r}
 1101 \\
 1101 \\
 + 1101 \\
 \hline
 1010
 \end{array}$$

$$OVFL = C_3 \oplus C_2$$

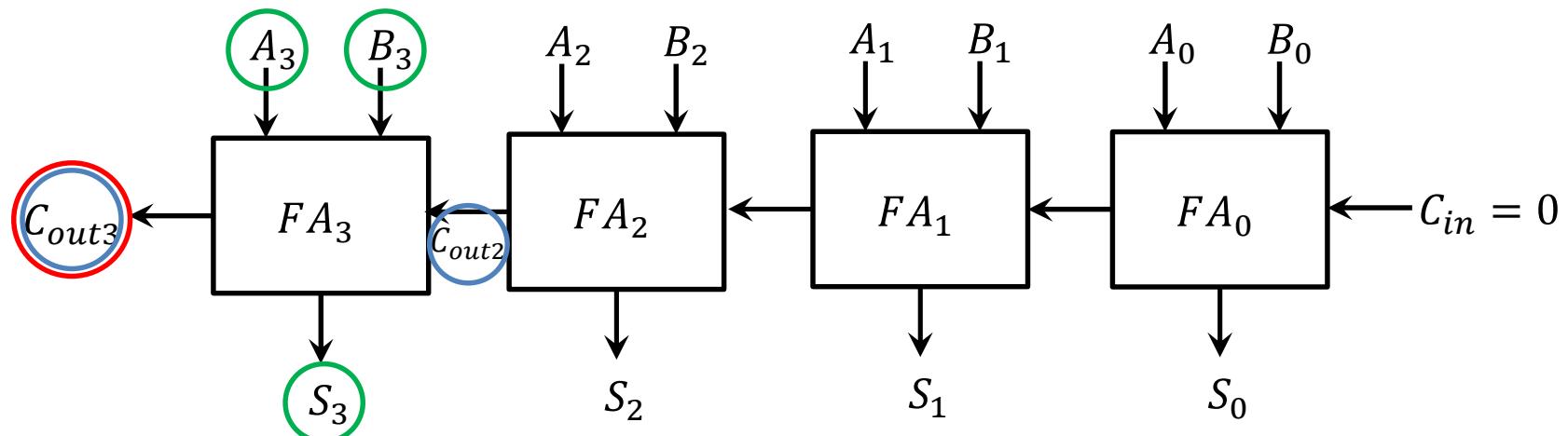
Círcuito Somador

Exercício

Implementar um somador de *4 bits* usando *1 Bit Full Adder*.

Circuito Somador *Full-Adder*

Somador de 4 bits: $S_{0...3} = A_{0...3} + B_{0...3}$

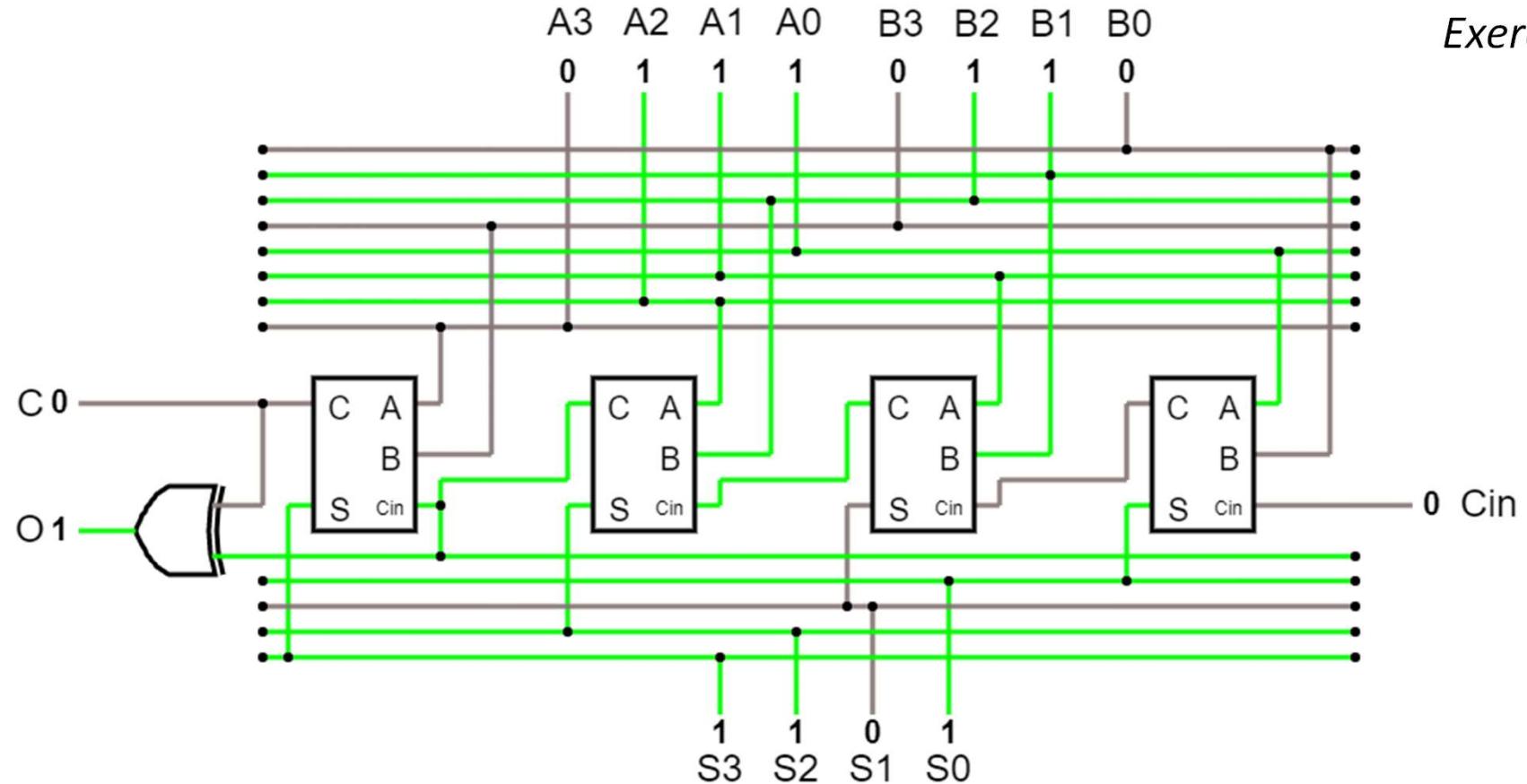


CARRY = flag para soma de inteiros sem sinal: se 1 então Erro

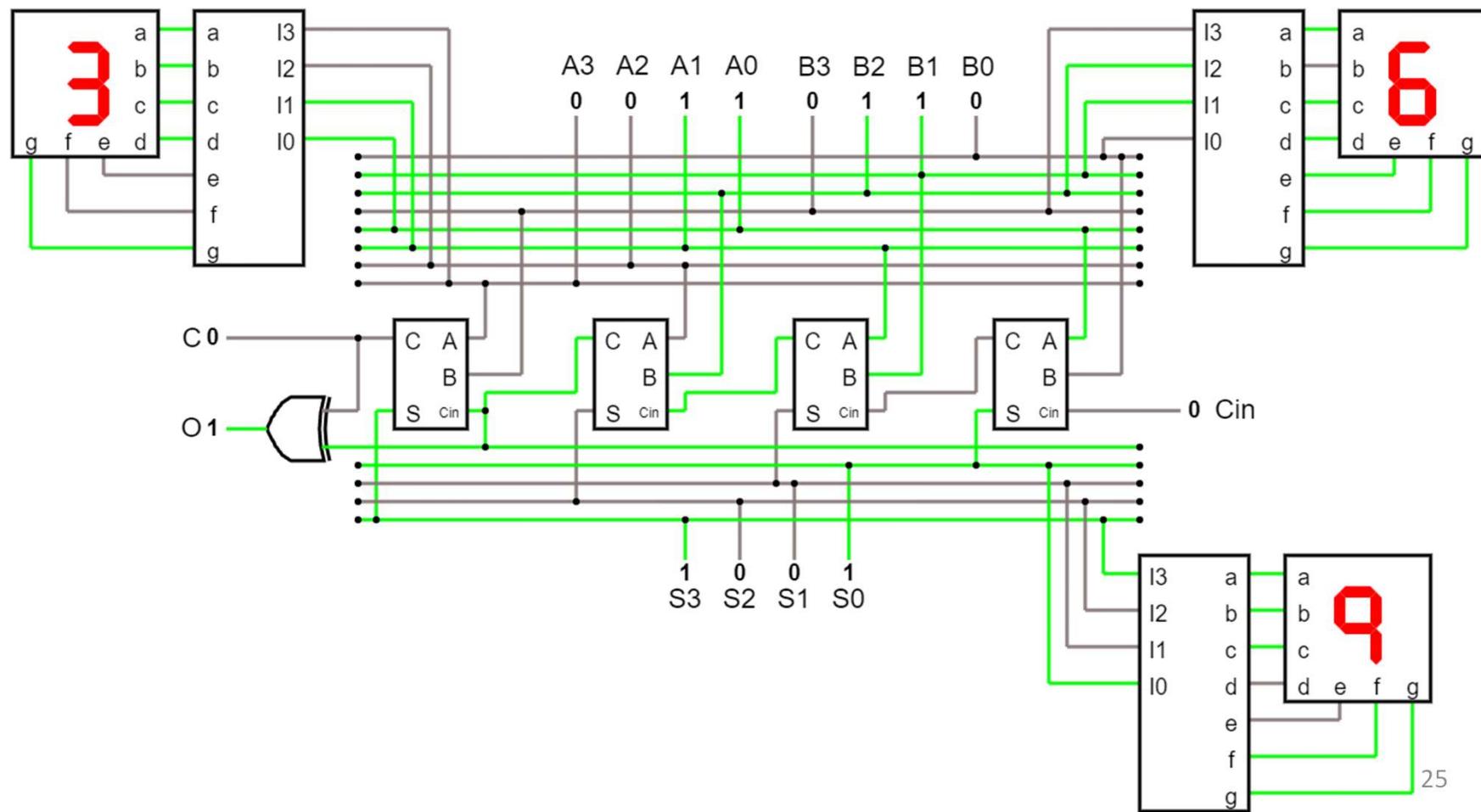
OVFL = flag para soma de inteiros com sinal: se 1 então Erro

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

Círculo Somador *Exercício*



Circuito Somador



Círculo Somador

Exercício

Implementar um sistema de votação do tipo SIM ou NÃO.

Este sistema será utilizado em uma plateia onde cada um dos participantes disporá de uma chave com três posições (SIM, NÃO, NEUTRO).

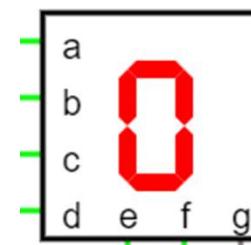
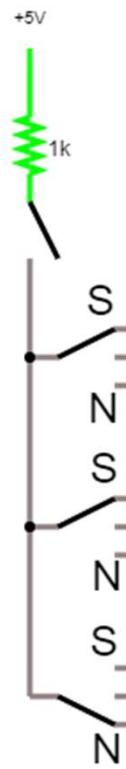
Ao ser solicitado o voto, cada um posicionará a chave conforme a sua intenção.

Um painel mostrará a quantidade de votos SIM e NÃO, tão logo o apresentador libere a apresentação;

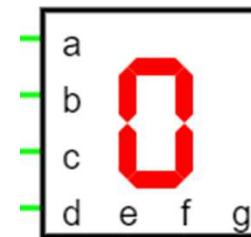
As peças disponíveis são Switches, Full Adder de 1 bit, Decoder/Display de sete segmentos, fontes e resistores. [FLOYD 2015].

O sistema deverá permitir até 3 votantes.

Círcuito Somador *Exercício*

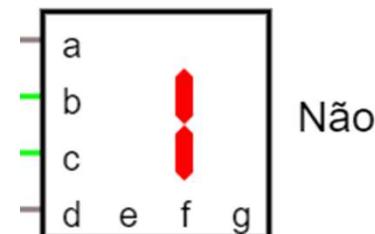
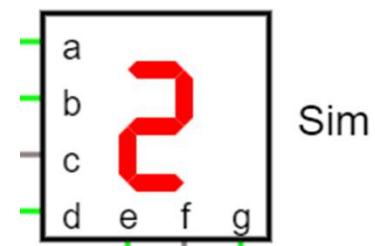
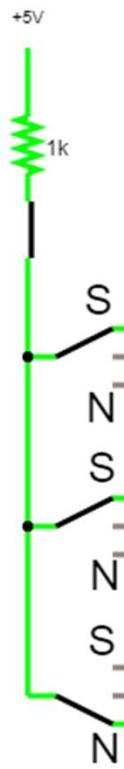


Sim

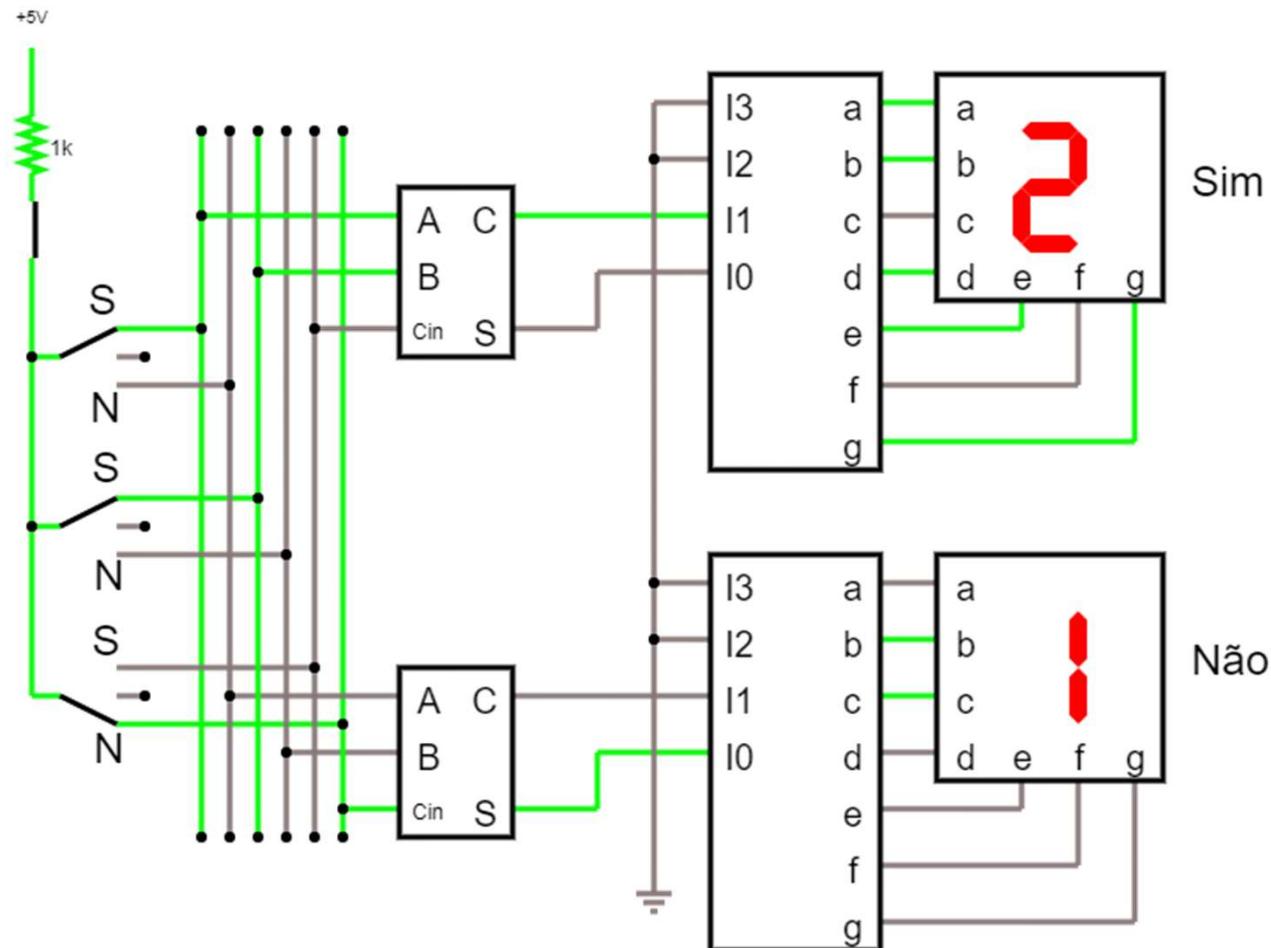


Não

Círcuito Somador *Exercício*



Círcuito Somador *Exercício*



Círculo Somador

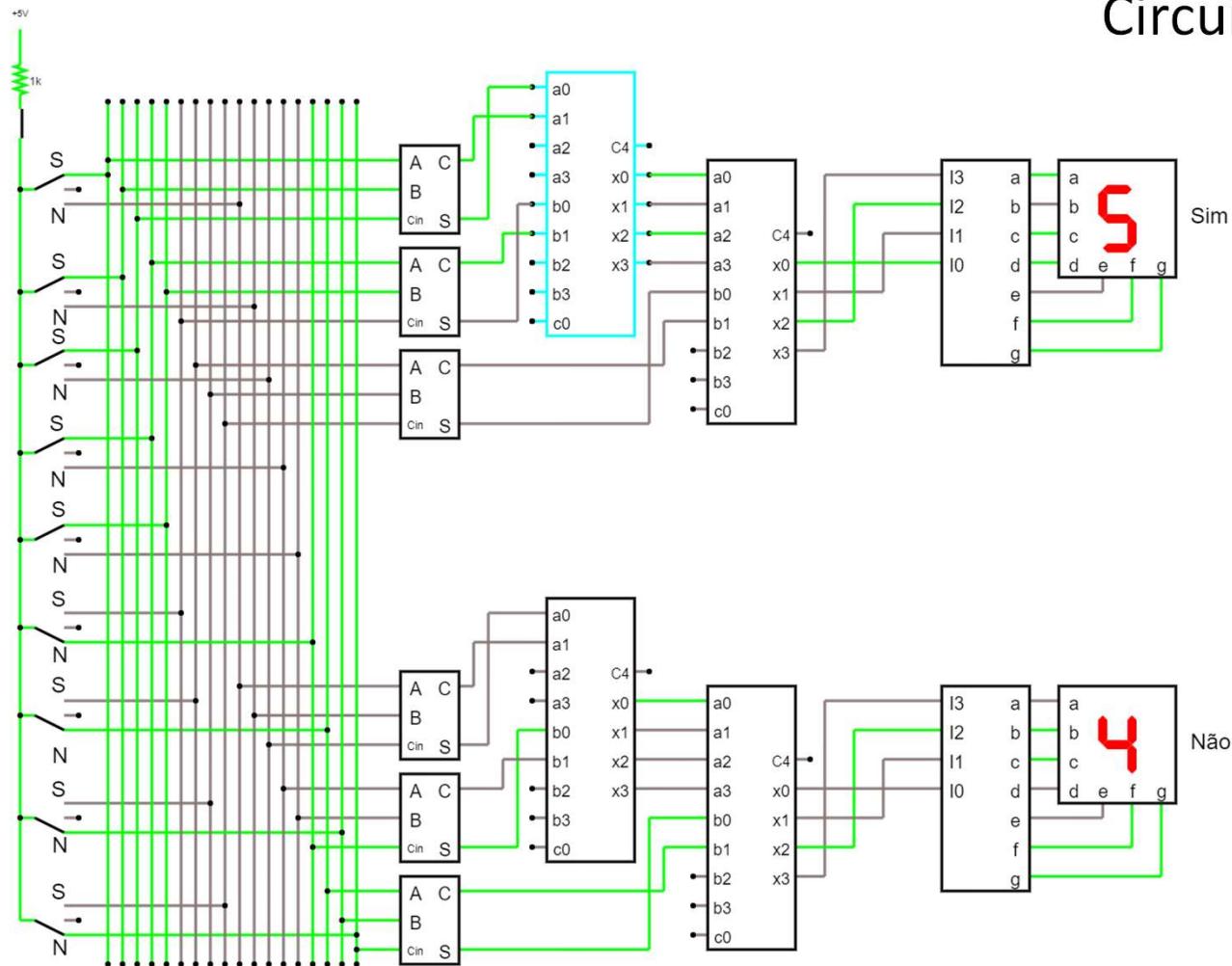
Exercício

Ampliar este sistema para permitir 9 votantes.

Além dos componentes elencados, pode ser usado somador de *4 bits*.

Círcuito Somador

Exercício



Sumário

- 1. Revisão – Sistemas de Numeração
- 2. Revisão – Representação de Dados
- 3. Revisão – Operações com Binários
- 4. Álgebra Booleana
- 5. Simplificação de Expressões
- 6. Mapa de Karnaugh
- 7. Elementos Lógicos Universais
- 8. Circuitos Combinacionais
- 9. Circuitos Sequenciais
- 1. Somador / Subtrator
- 2. Comparadores
- 3. Codificador/decodificador
- 4. Multiplexador/Demux
- 5. Geradores de paridade
- 6. ULA
- 7. Multiplicadores / Divisores
- 8. PLD/PLA/ROM

Circuito Subtrator

Full-Subtractor

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

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

$$B_{out} = ??$$

$$\begin{array}{r} -1 & 0 \\ -0 & \\ \hline 1 & \end{array} \quad \begin{array}{r} -1 & -1 \\ -0 & \\ \hline 1 & \end{array} \quad \begin{array}{r} -1 & -1 \\ -0 & \\ \hline 0 & \end{array}$$

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

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

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

\bar{B}_{in}	B_{in}
AB	C
00	0
01	1
11	1
10	5

$$\boxed{\bar{A}B} + \boxed{B_{in}} + \boxed{\bar{A}\bar{B}_{in}}$$

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

Circuito Subtrator

Full-Subtractor

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

$$\begin{array}{r} -1 \ 0 \\ -0 \\ \hline 1 \end{array} \quad \begin{array}{r} -1 \ -1 \\ -0 \\ \hline 1 \end{array} \quad \begin{array}{r} -1 \ -1 \\ -0 \\ \hline 0 \end{array}$$

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

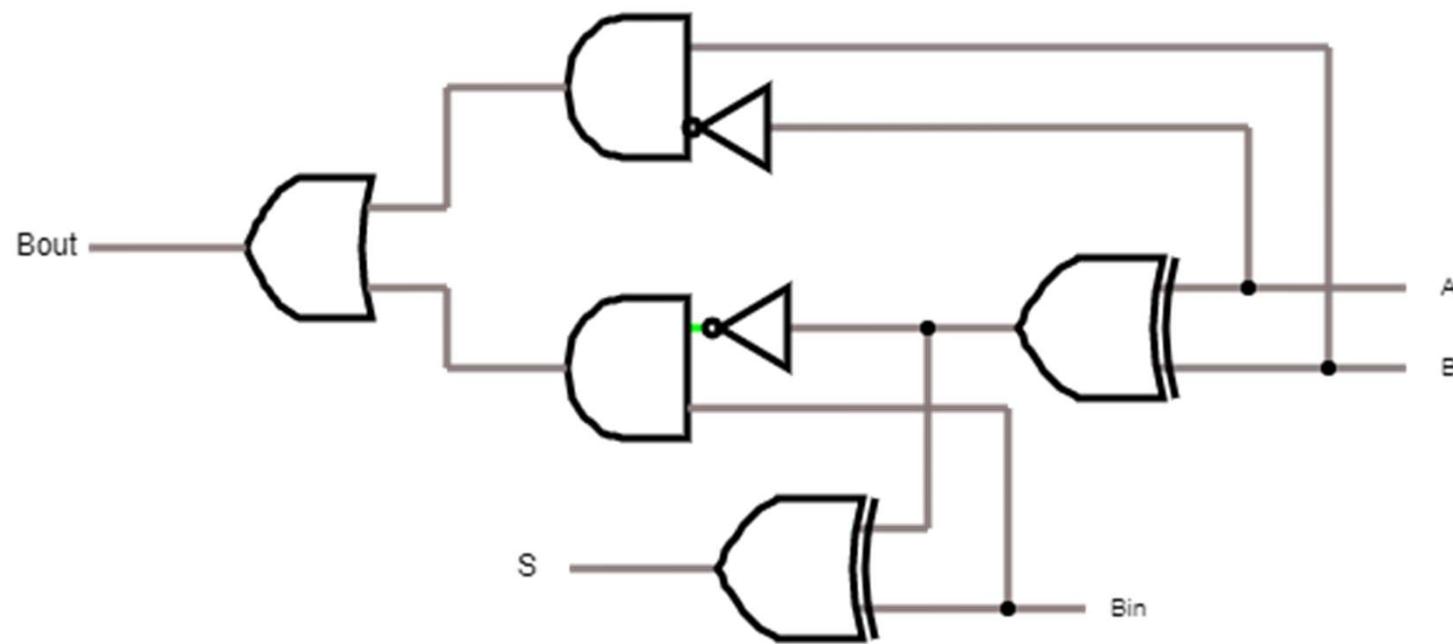
$$B_{out} = ??$$

	\bar{B}_{in}	B_{in}
AB	C	0 1
$\bar{A}\bar{B}$	00	0 1
$\bar{A}B$	01	1 1
AB	11	1 1
$A\bar{B}$	10	4 5

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

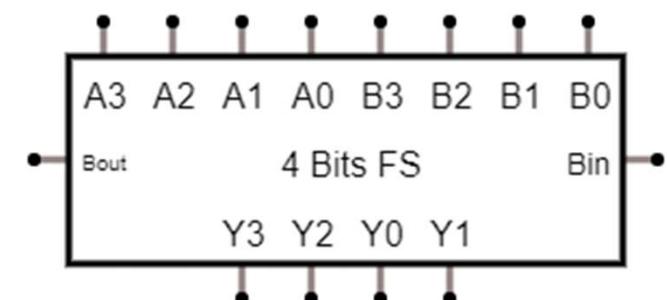
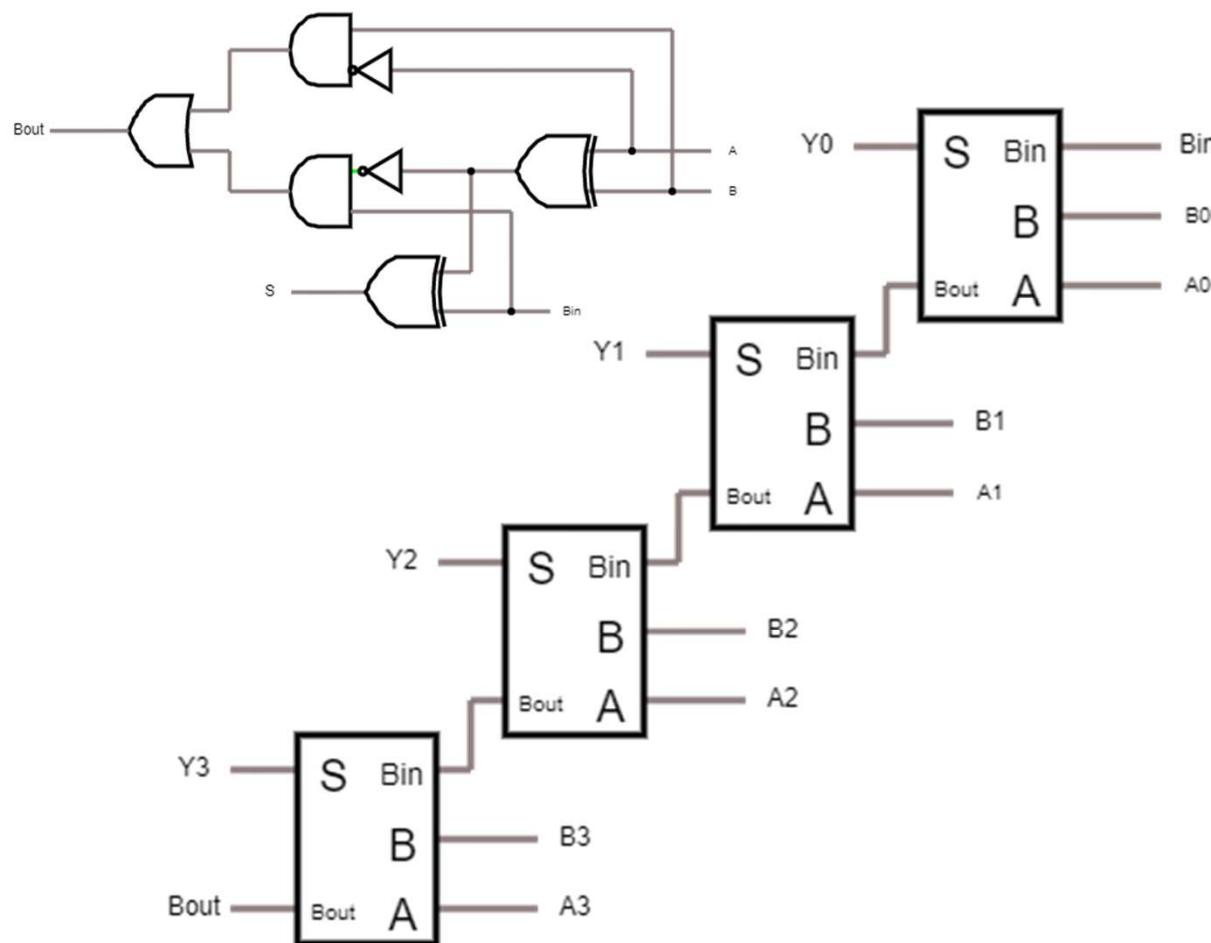
$$B_{out} = \bar{A}\bar{B} + B_{in}(AB + \bar{A}\bar{B}) \quad B_{out} = \bar{A}\bar{B} + B_{in}(\overline{A \oplus B})$$

Circuito Subtrator *Full-Subtractor*



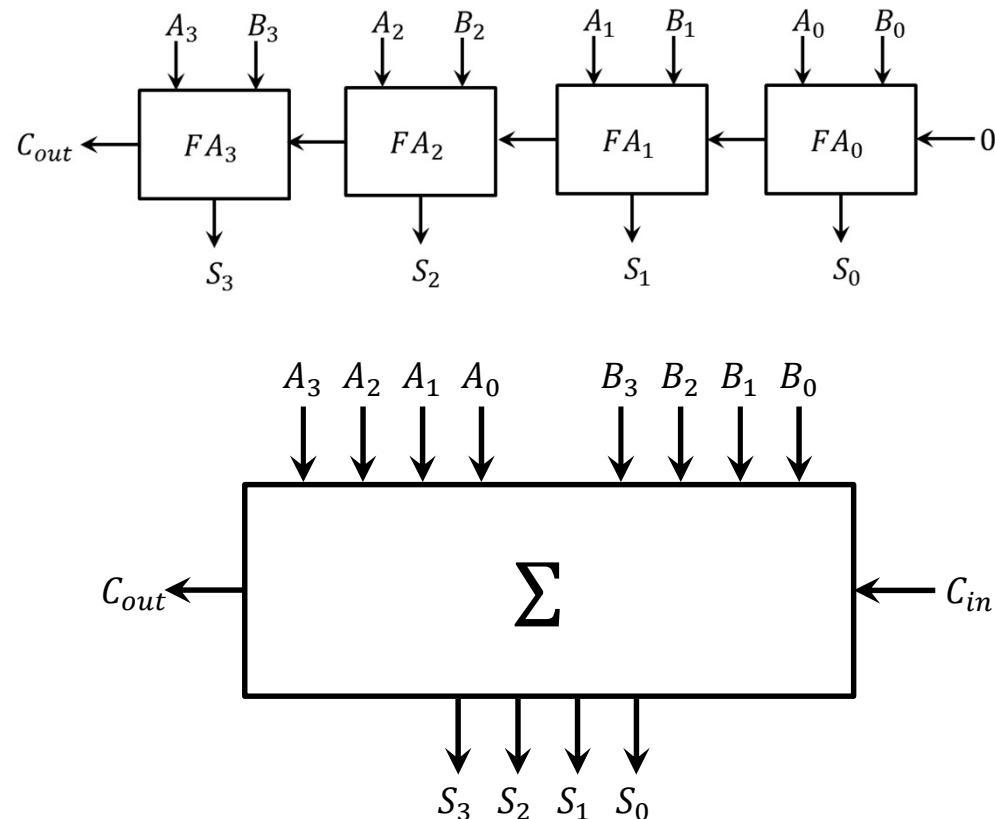
Circuito Subtrator

Full-Subtractor

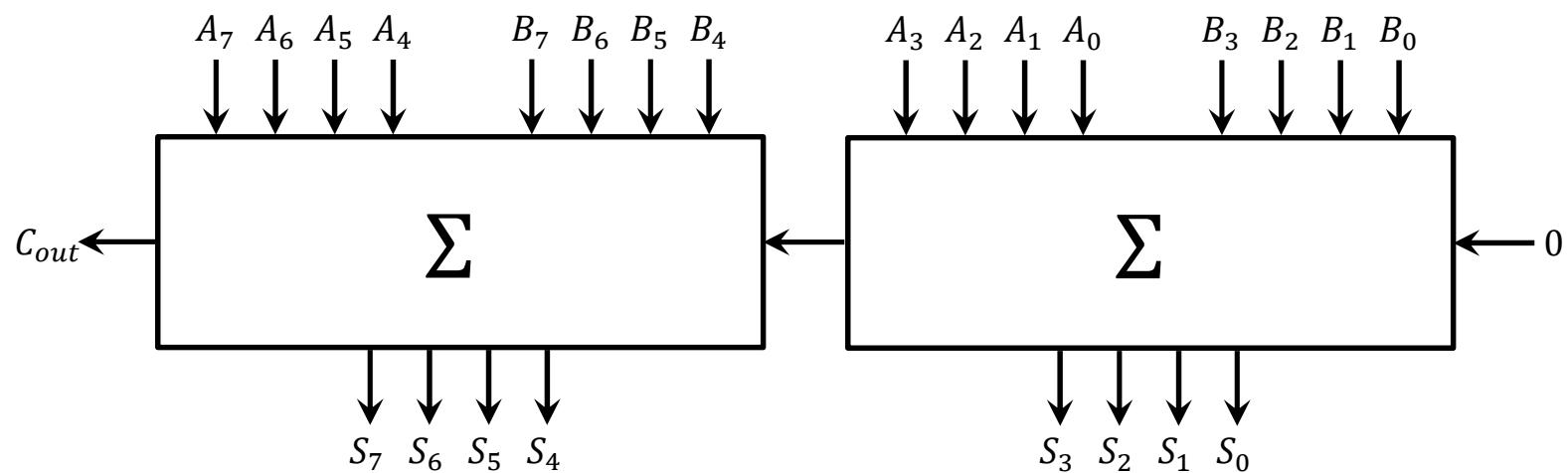


ENCADEANDO SOMADORES INTEGRANDO SOMADOR/SUBTRATOR

Circuito Somador *Somador de 4 Bits*



Circuito Somador *Somador de 8 Bits*



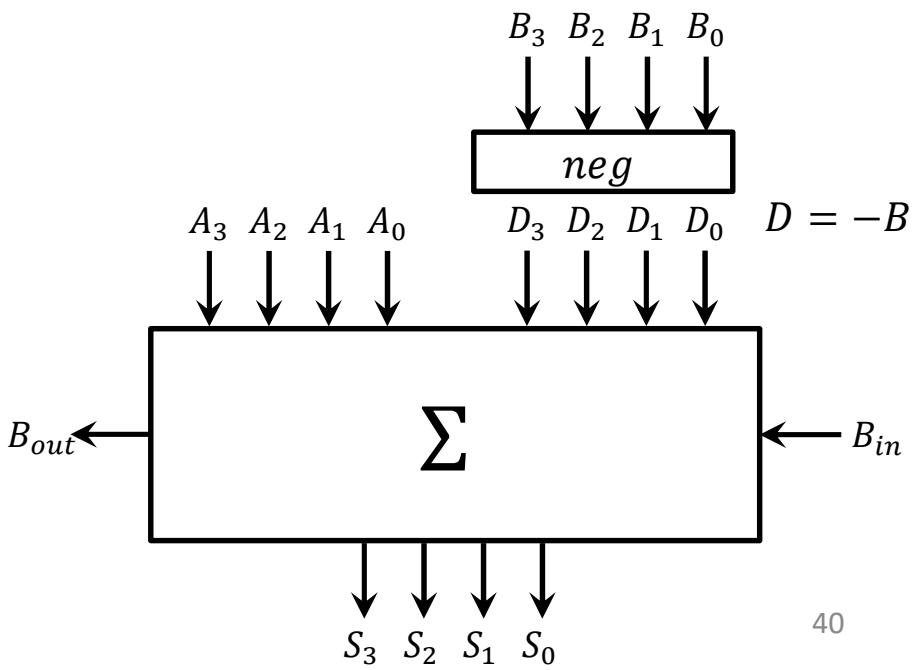
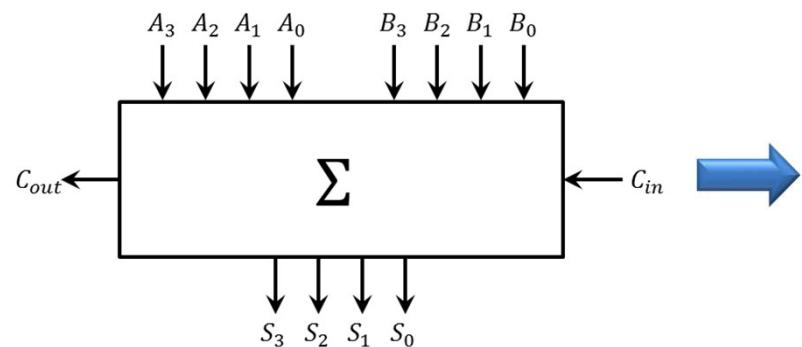
Integrando Somador/Subtrator Somador de 4 Bits

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

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



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



Circuito Subtrator

Subtrator de 4 Bits

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

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

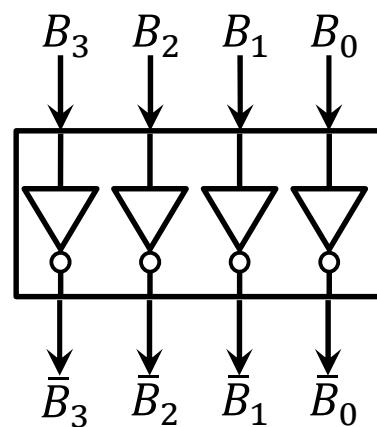
A transformação deverá converter um inteiro sem sinal em seu equivalente negativo em $C - 2$.

Exemplo: 5 em -5 .

$$B = 0101$$

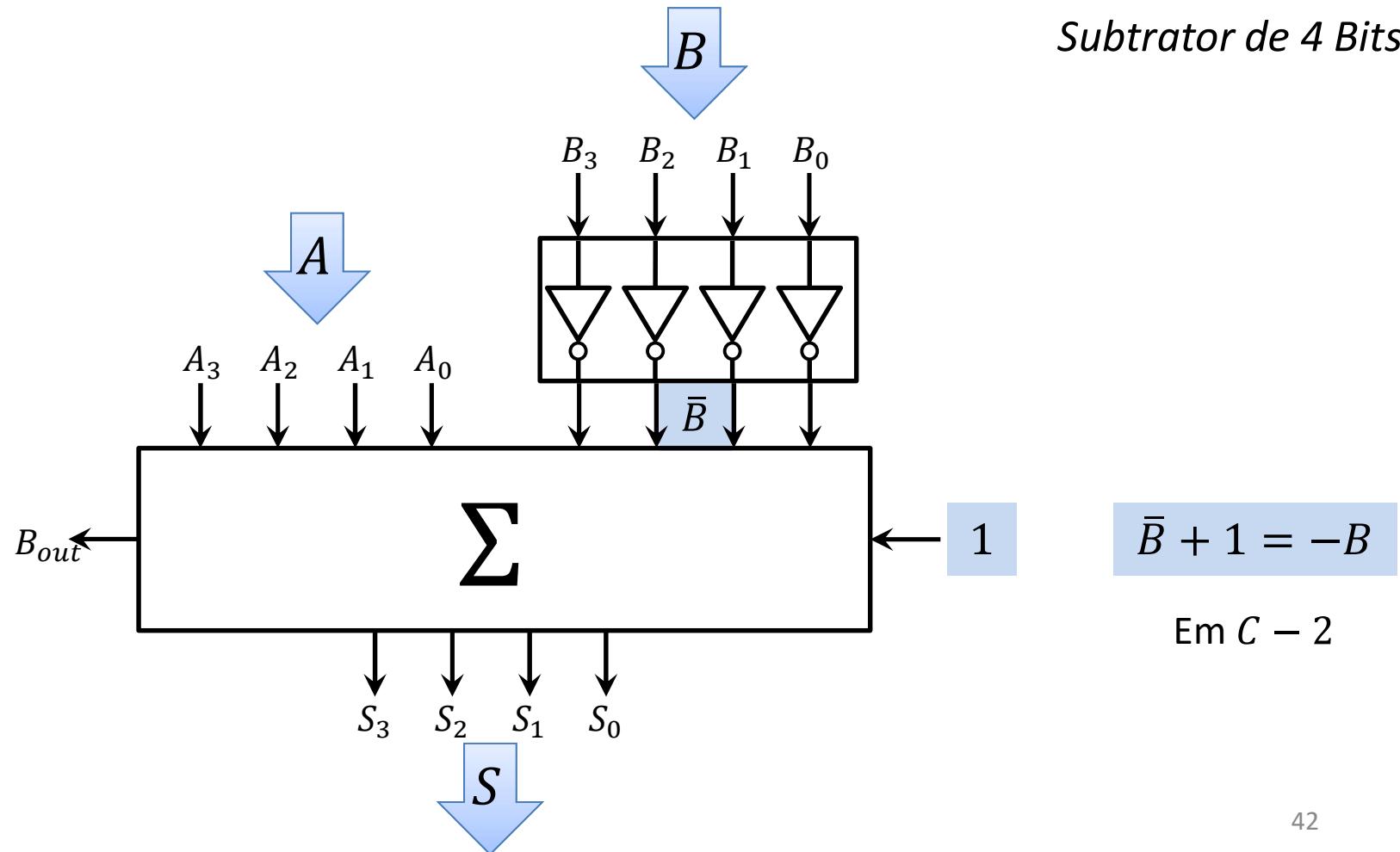
$$\bar{B} = 1010$$

$$\begin{array}{r} + \\ \hline -B = 1011 \end{array}$$



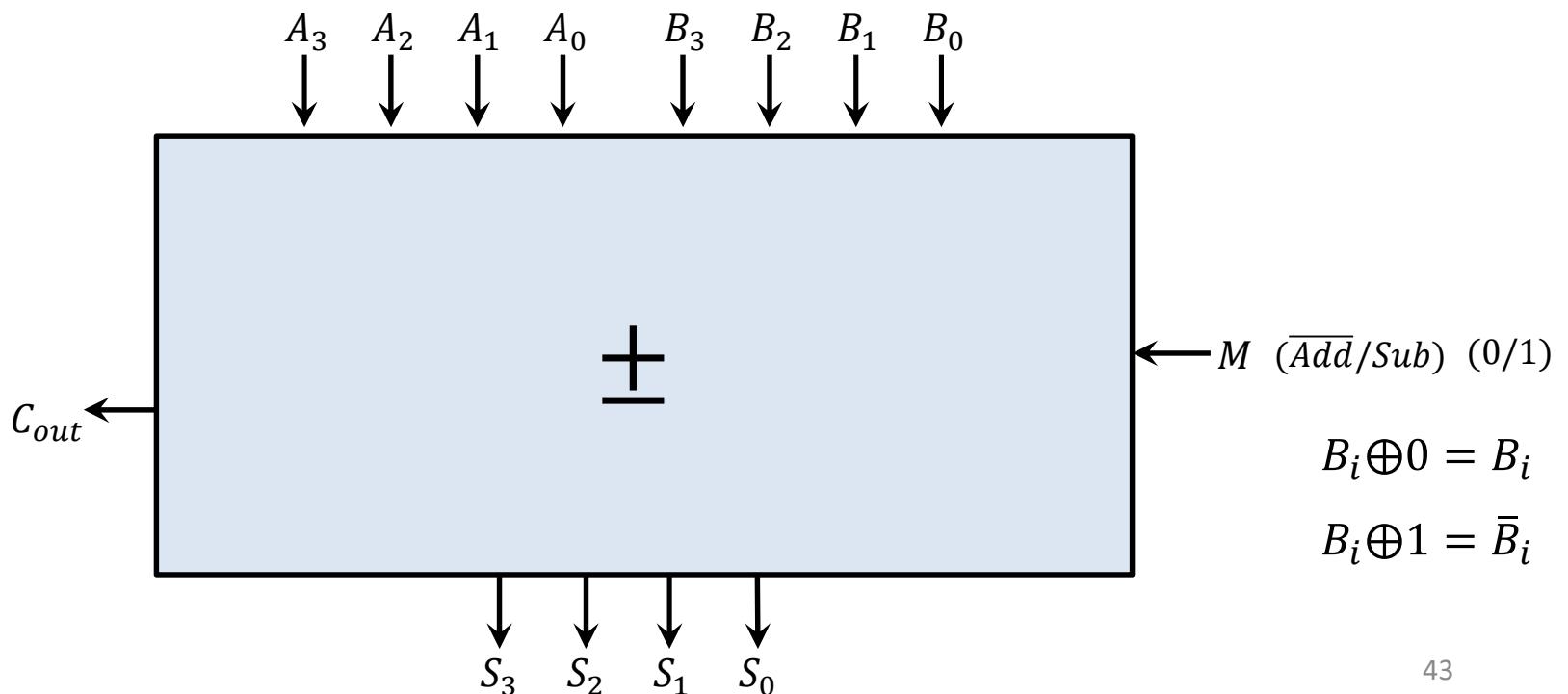
Circuito Subtrator

Subtrator de 4 Bits

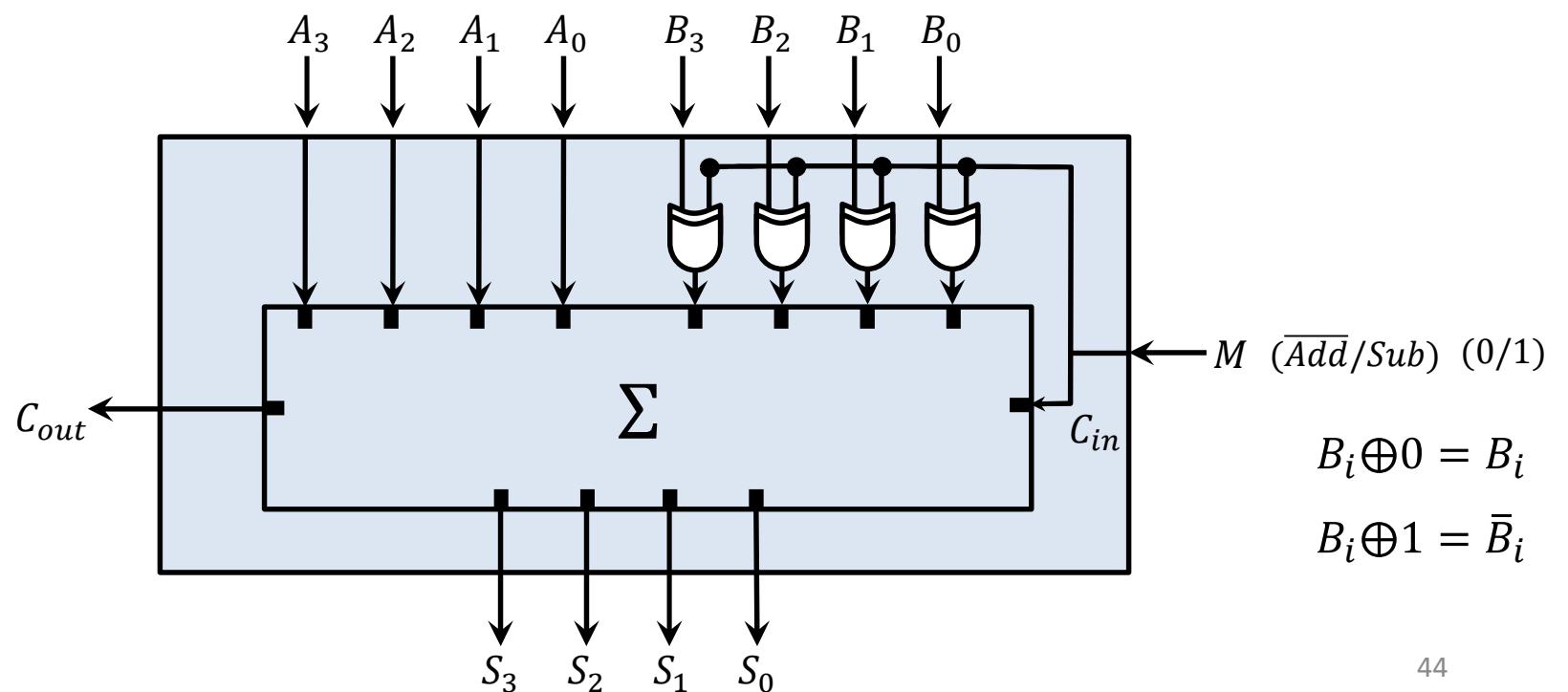


Circuito Somador/Subtrator

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



Circuito Somador/Subtrator



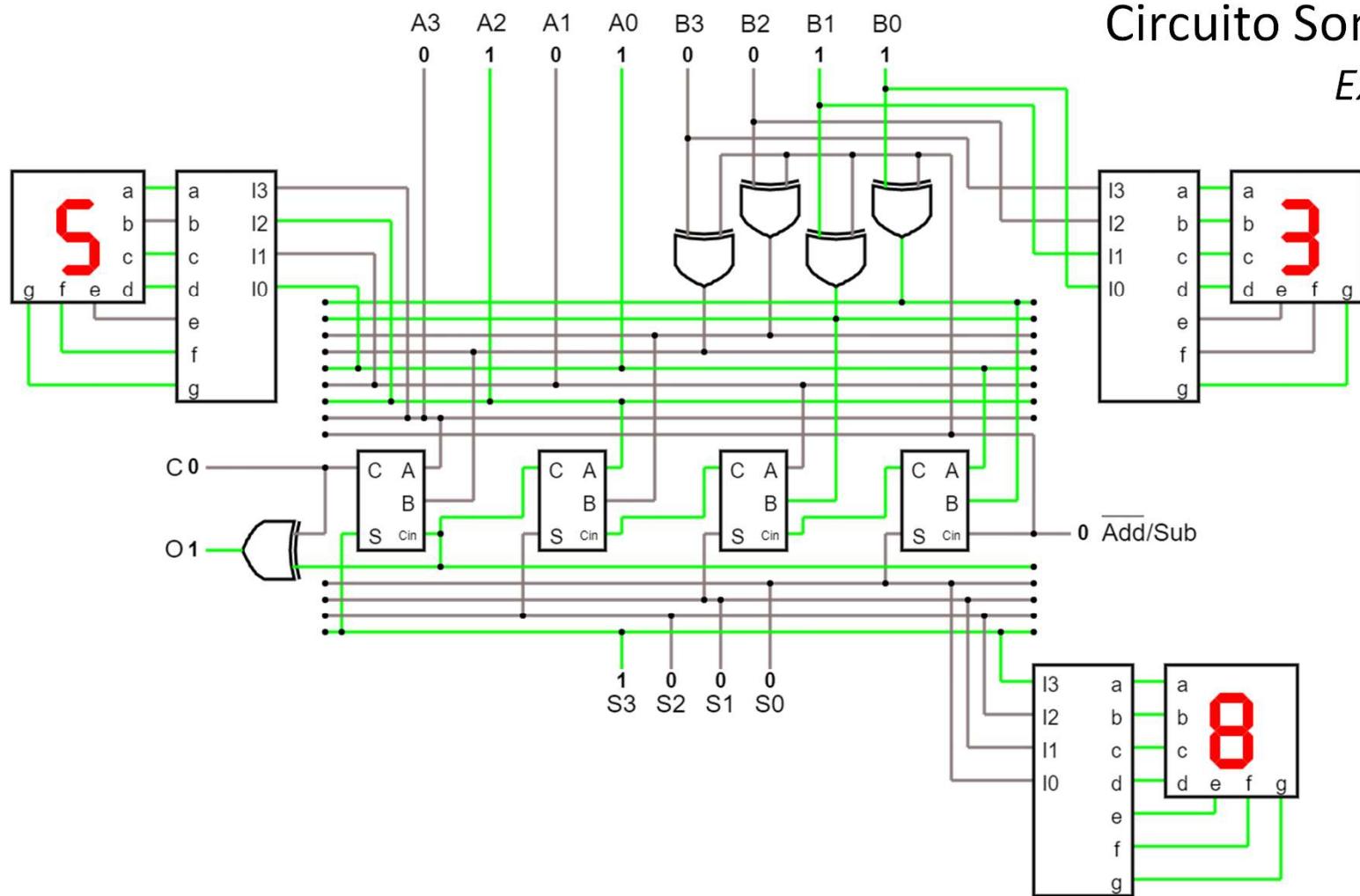
Círcuito Somador

Exercício

Implementar um somador/subtrator de *4 bits* usando *1 Bit Full Adder*.

Círcuito Somador

Exercício



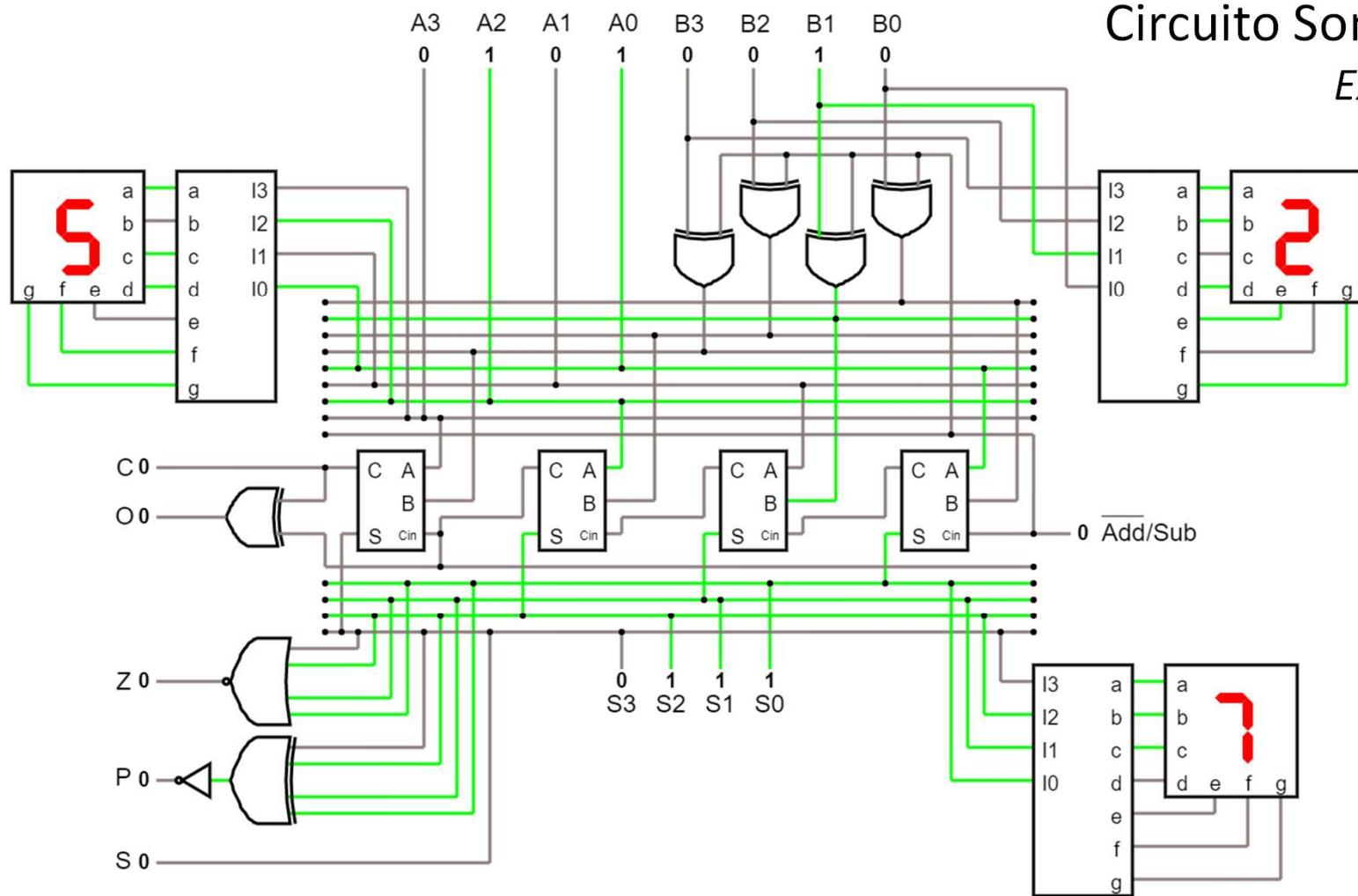
Círculo Somador

Exercício

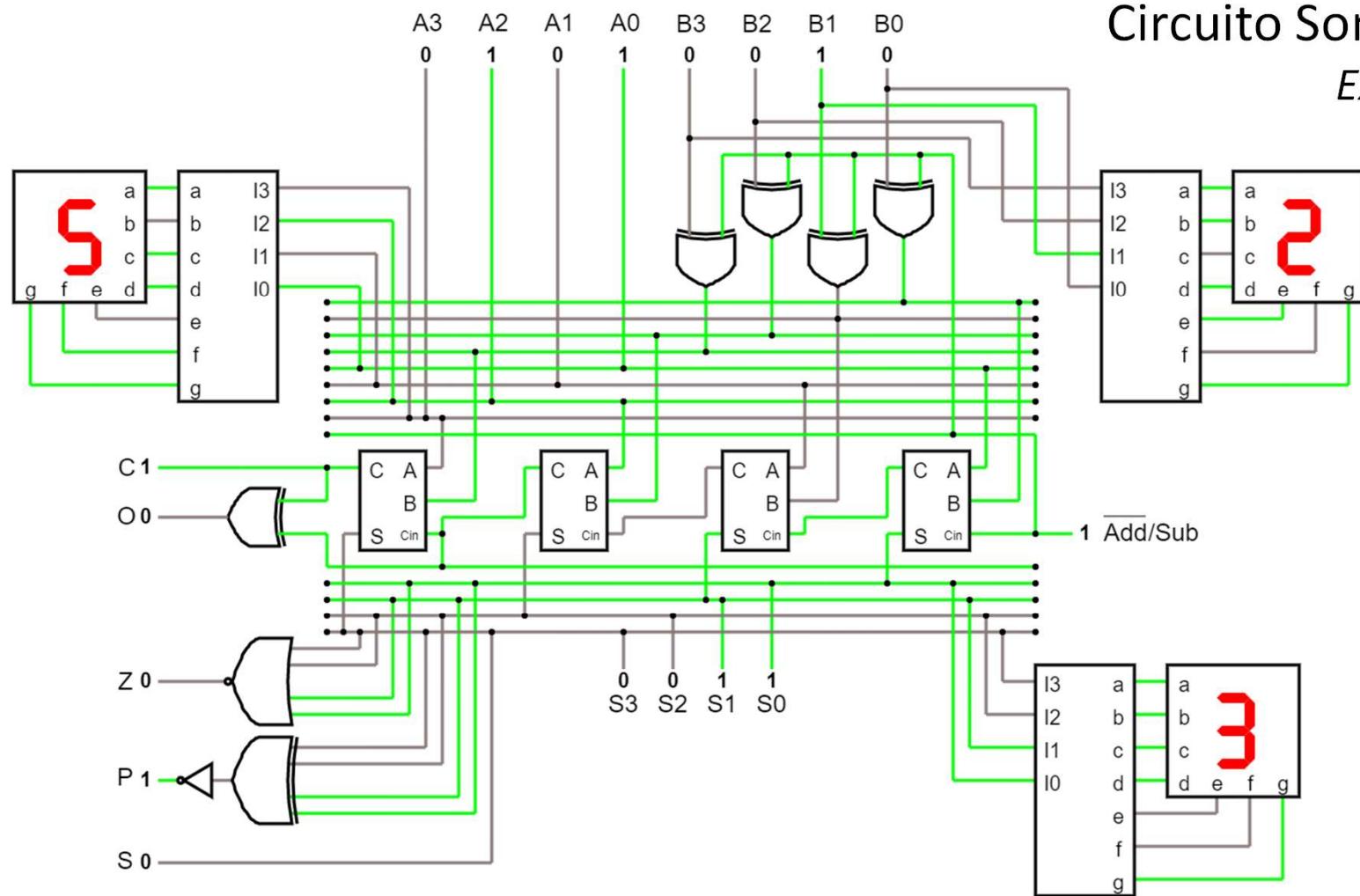
Implementar um somador/subtrator de 4 bits usando 1 Bit Full Adder com os $FLAGS = \{C, P, Z, S, O\}$.

C	Valor do último Carry
P	Paridade par (#1 é par)
Z	Resultado zero
S	Bit de Sinal
O	Qualidade para inteiro com sinal

Círculo Somador Exercício



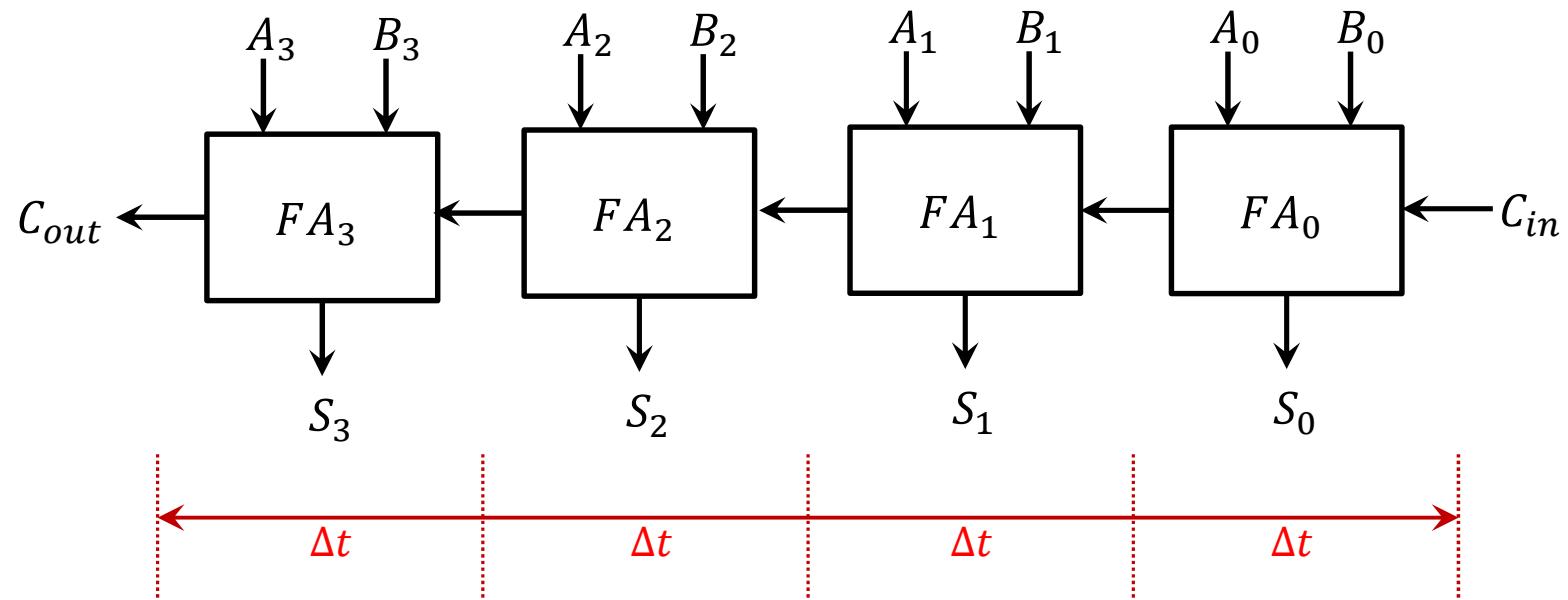
Círcuito Somador Exercício



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.



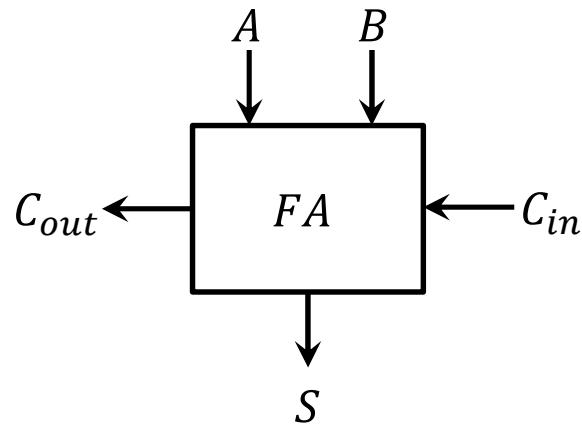
Círculo Somador/Subtrator

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 propagation:* $C_p = A + B$



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

Círculo Somador/Subtrator Optimização

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

- Estágio FA_0

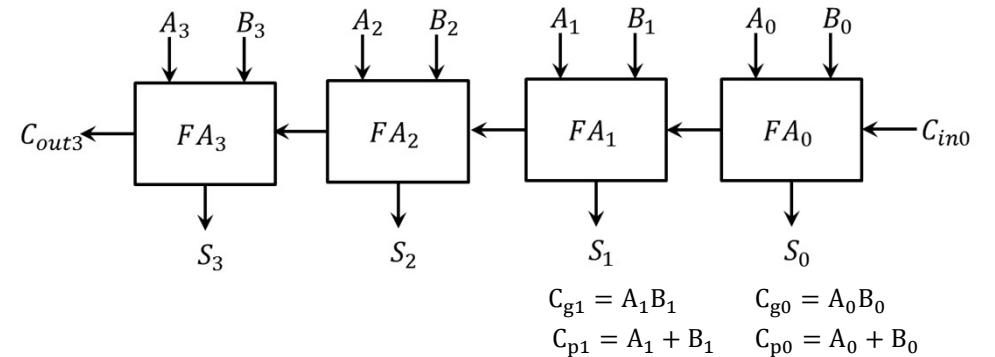
$$C_{out0} = C_{g0} + C_{p0}C_{in0}$$

- Estágio FA_1

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

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

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



Círculo Somador/Subtrator Optimização

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

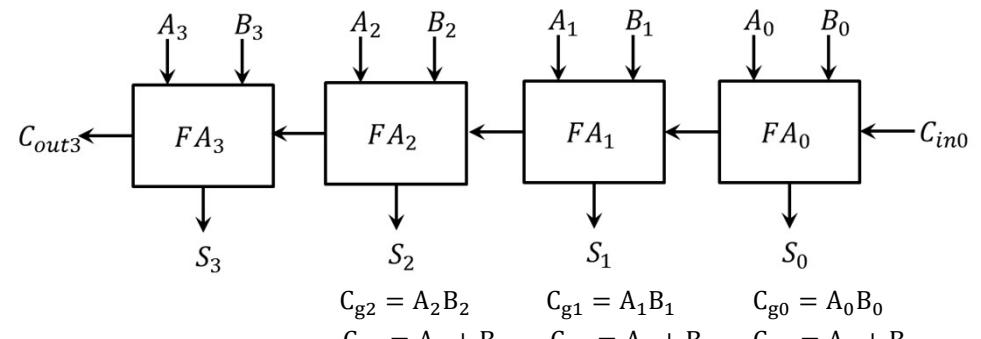
- Estágio FA_2

$$C_{in2} = C_{out1}$$

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

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

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



Círculo 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

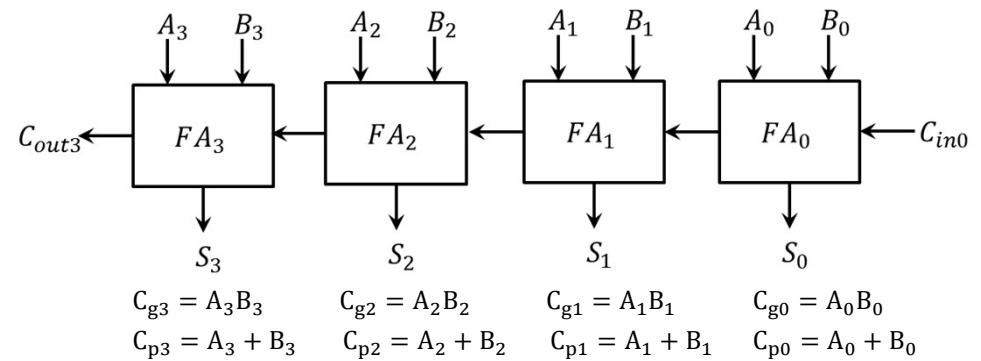
$$C_{in3} = C_{out2}$$

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

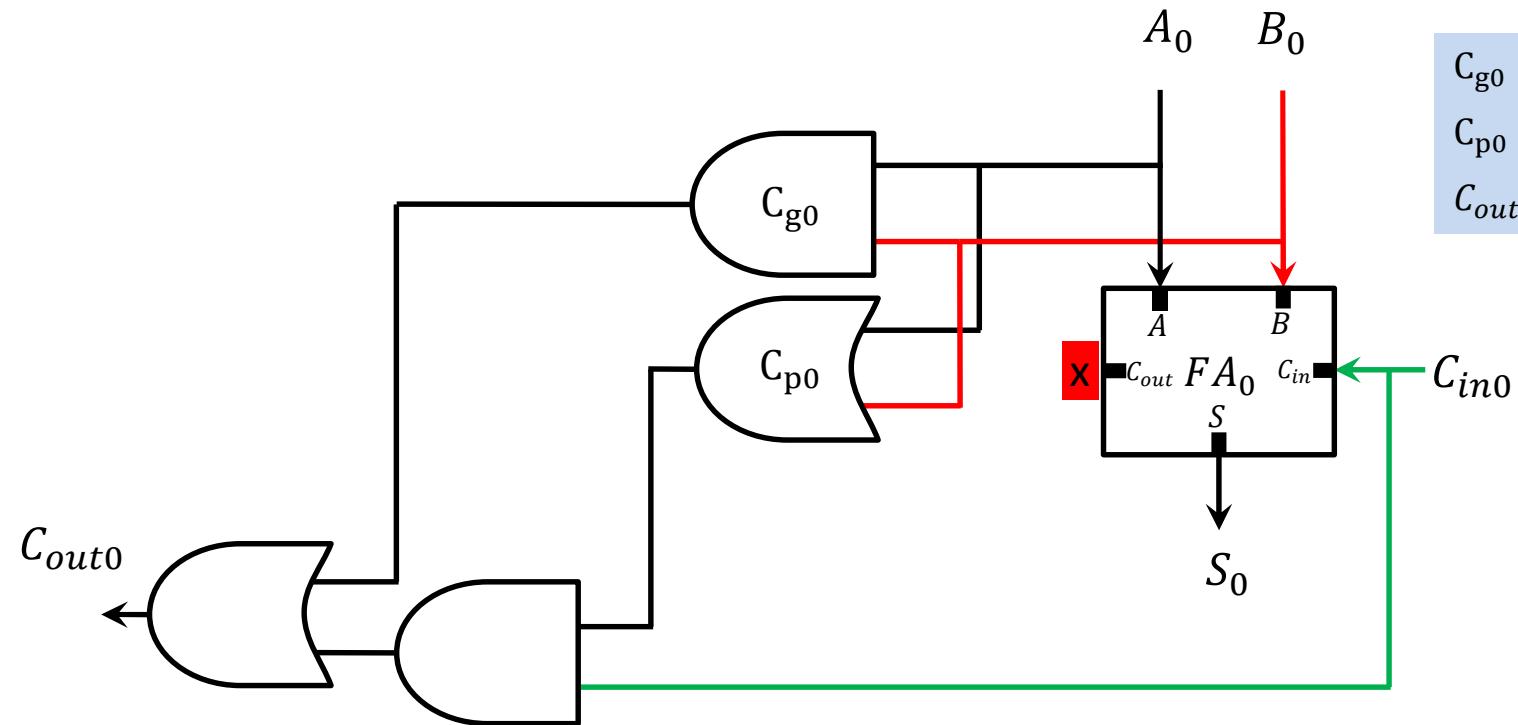
$$C_{out3} = C_{g3} + C_{p3} C_{in3} = C_{g3} + C_{p3} C_{out2}$$

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

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



Circuito Somador/Subtrator *Look-Ahead Carry*

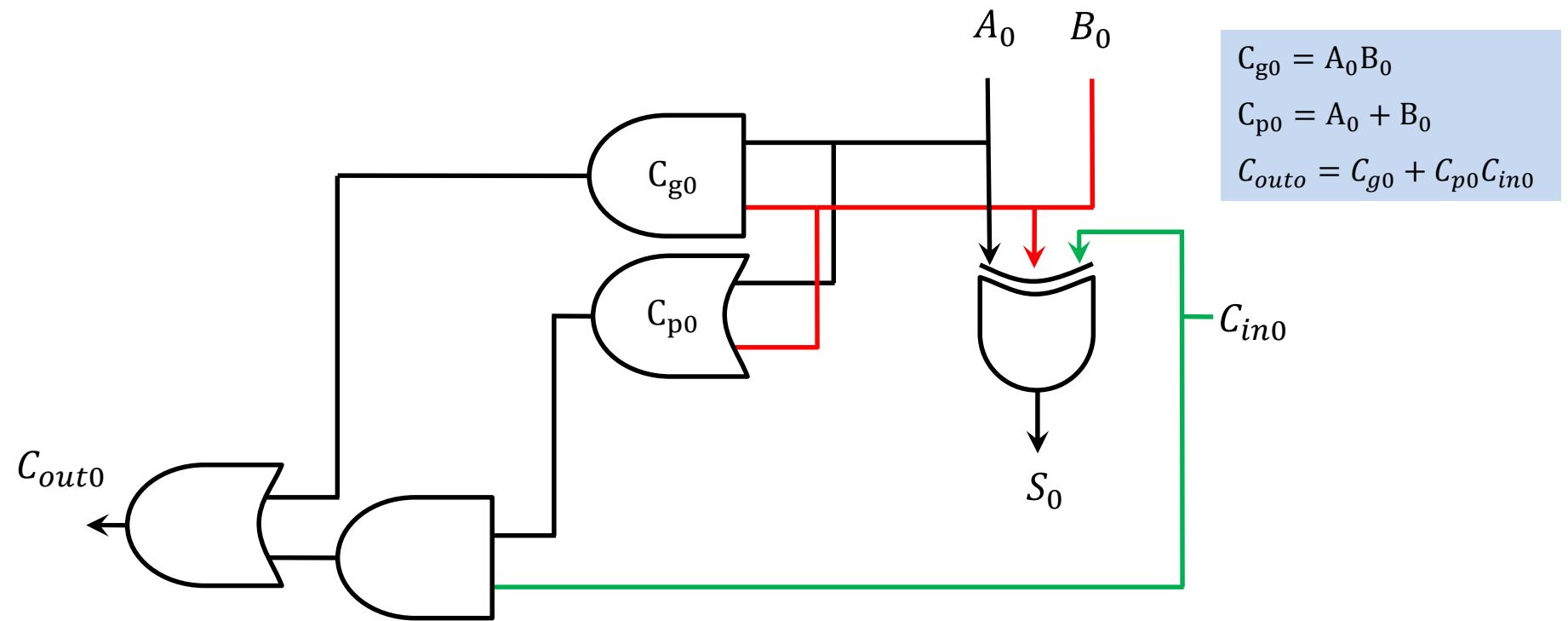


$$C_{g0} = A_0 B_0$$

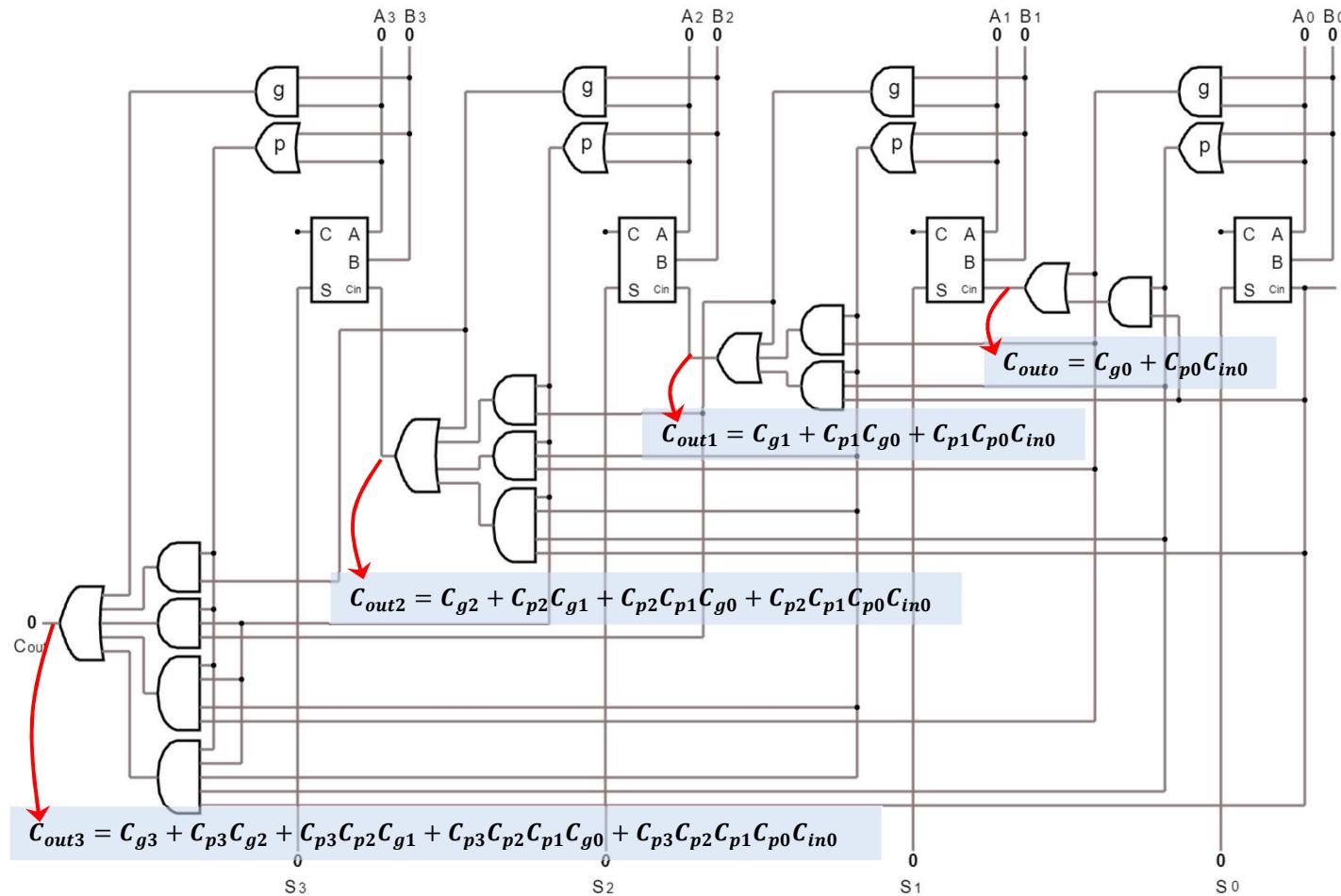
$$C_{p0} = A_0 + B_0$$

$$C_{out0} = C_{g0} + C_{p0} C_{in0}$$

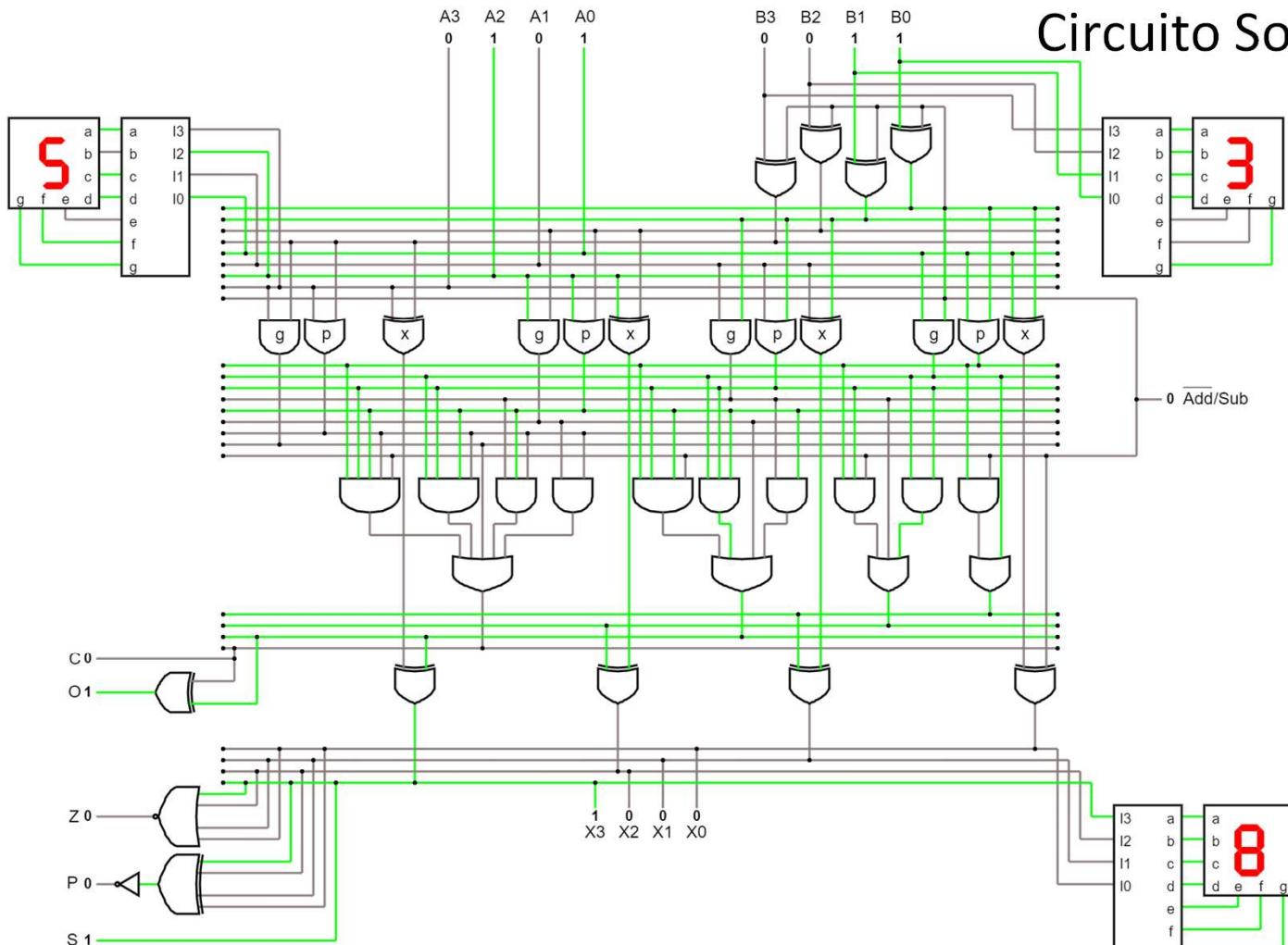
Circuito Somador/Subtrator *Look-Ahead Carry*



Circuito Somador/Subtrator Look-Ahead Carry



Circuito Somador/Subtrator Look-Ahead Carry



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

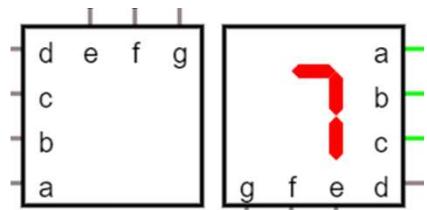
Círcuito Somador

Exercício

Implementar um circuito que apresenta um inteiro com sinal (codificado em $C - 2$ com 4 bits) no display de sete segmentos ($5, -5, 2, -1$):

Círculo Somador *Exercício*

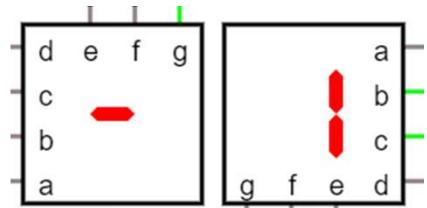
0 1 1 1



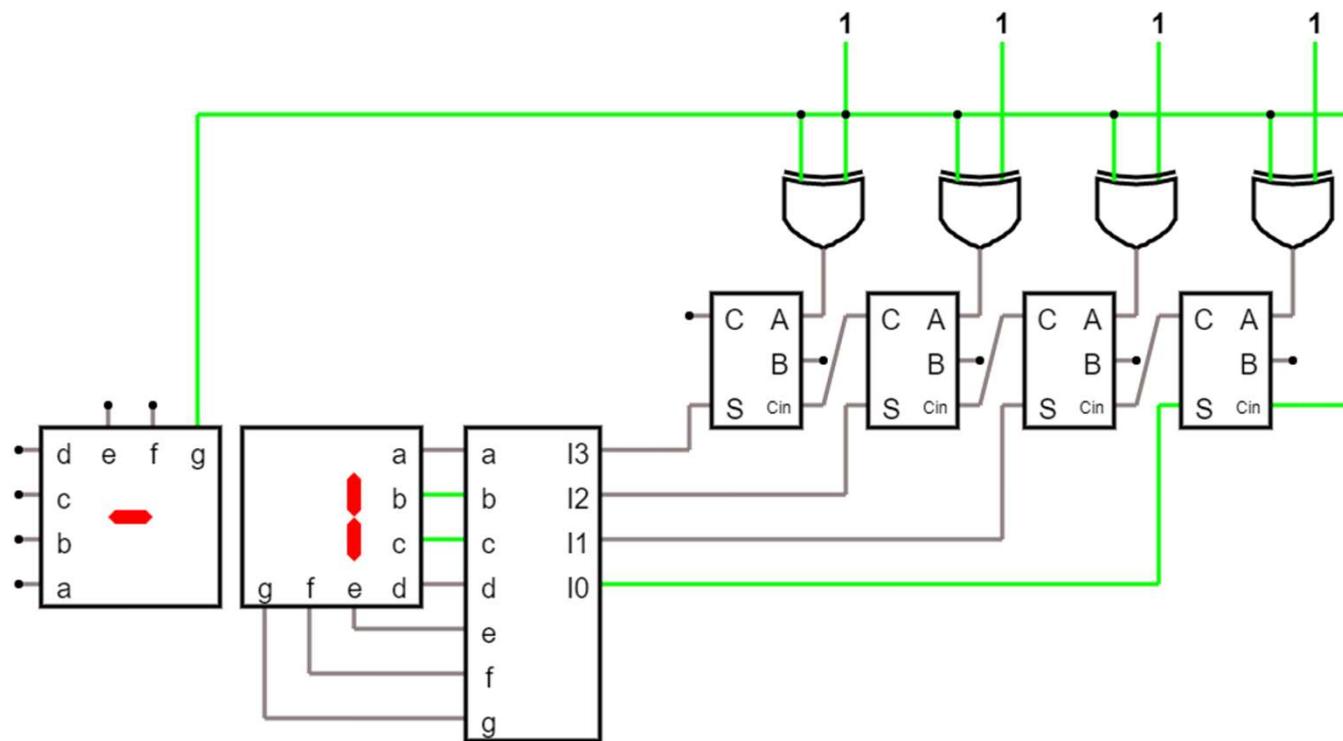
Círculo Somador

Exercício

1 1 1 1



Círculo Somador *Exercício*

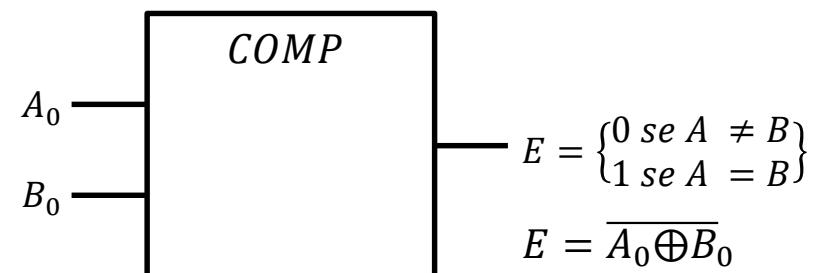


Sumário

- 1. Revisão – Sistemas de Numeração
- 2. Revisão – Representação de Dados
- 3. Revisão – Operações com Binários
- 4. Álgebra Booleana
- 5. Simplificação de Expressões
- 6. Mapa de Karnaugh
- 7. Elementos Lógicos Universais
- 8. Circuitos Combinacionais
- 9. Circuitos Sequenciais
- 1. Somador / Subtrator
- 2. Comparadores
- 3. Codificador/decodificador
- 4. Multiplexador/Demux
- 5. Geradores de paridade
- 6. ULA
- 7. Multiplicadores / Divisores
- 8. PLD/PLA/ROM

Círculo Comparador

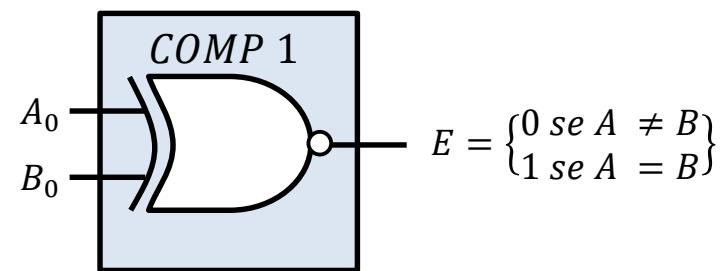
- A função de um círculo comparador é fornecer a relação entre duas quantidades binárias.
- A funcionalidade mais simples é determinar se dois números são iguais.
- Segue um comparador de igualdade de 1 bit:



XOR

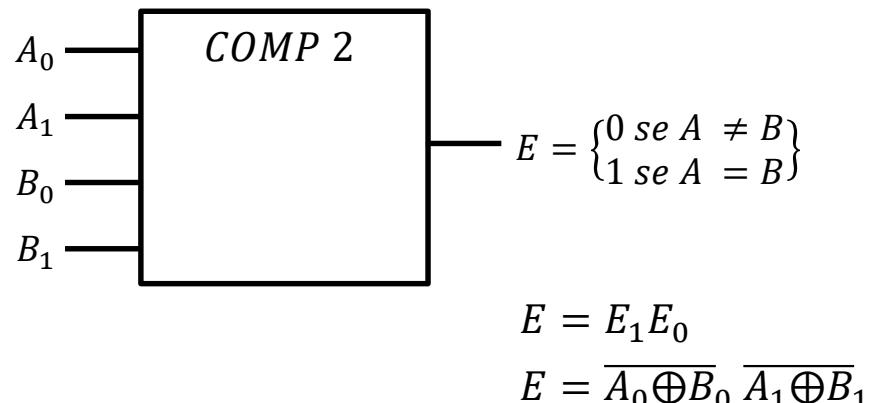
A	B	X	\bar{X}
0	0	0	1
0	1	1	0
1	0	1	0
1	1	0	1

Circuito Comparador *Igualdade - 1 bit*



Circuito Comparador

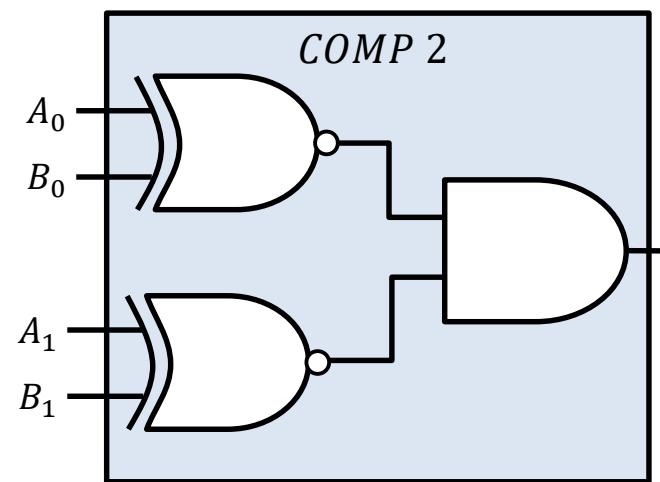
- Comparador de igualdade de 2 bits:



XOR

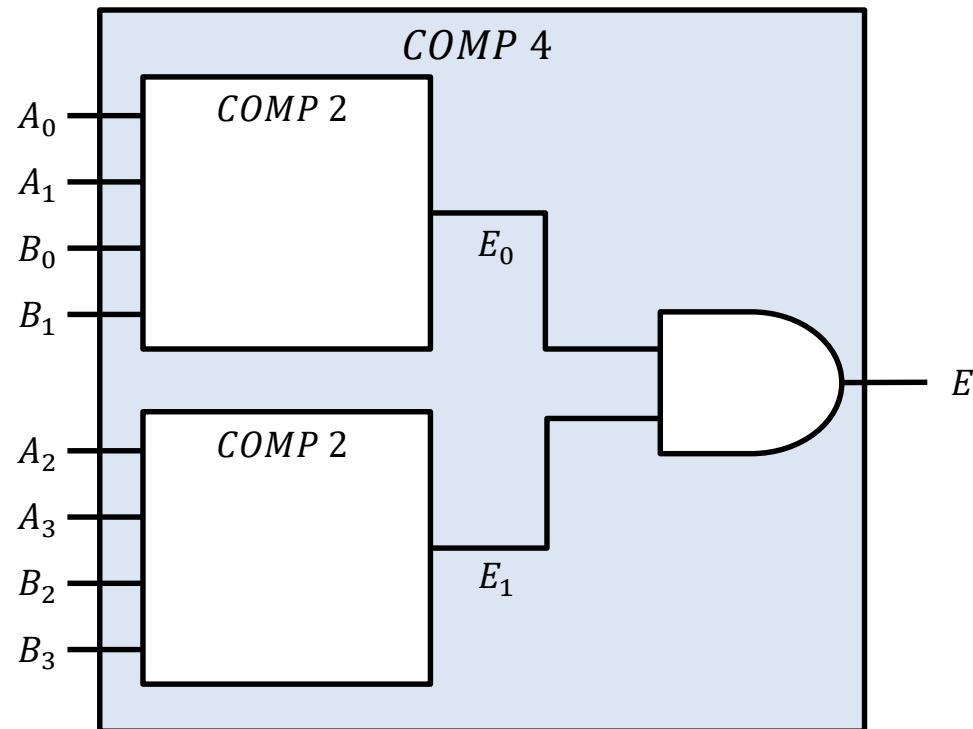
A	B	X	\bar{X}
0	0	0	1
0	1	1	0
1	0	1	0
1	1	0	1

Circuito Comparador *Igualdade - 2 bits*



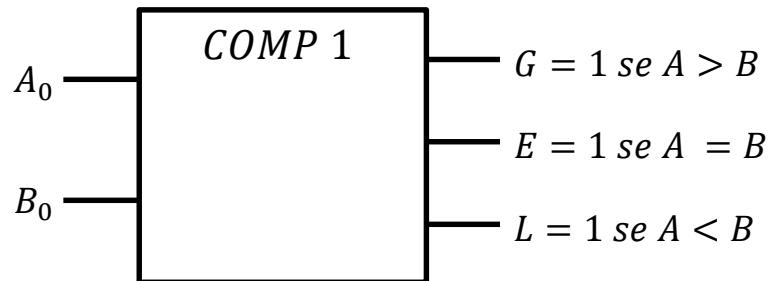
$$E = \begin{cases} 0 & \text{se } A \neq B \\ 1 & \text{se } A = B \end{cases}$$

Circuito Comparador
Igualdade - 4 bits



Circuito Comparador *Completo – 1 bit*

- Um comparador completo testa a relação entre dois inteiros A e B , informando se $A = B$, $A < B$ ou $A > B$.
- Segue um comparador completo de 1 bit:



$$G = A_0 \bar{B}_0$$

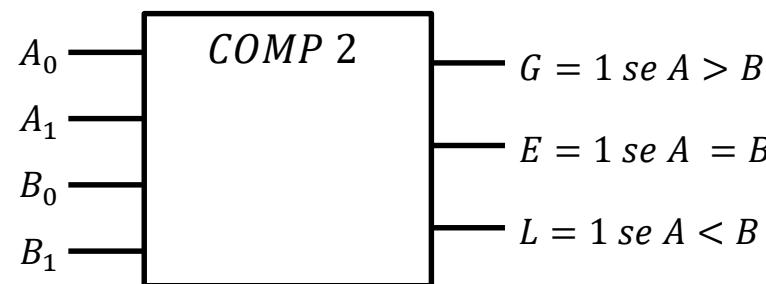
$$E = \overline{A_0 \oplus B_0}$$

$$L = \bar{A}_0 B_0 = \overline{G + E} = \bar{G} \bar{E}$$



Circuito Comparador Completo – 2 bits

- Comparador completo de 2 bits:



$$G = G_1 + E_1 G_0$$

$$G_1 = A_1 \bar{B}_1$$

$$E_1 = \overline{A_1 \oplus B_1}$$

$$G_0 = A_0 \bar{B}_0$$

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

Circuito Comparador Completo – 2 bits

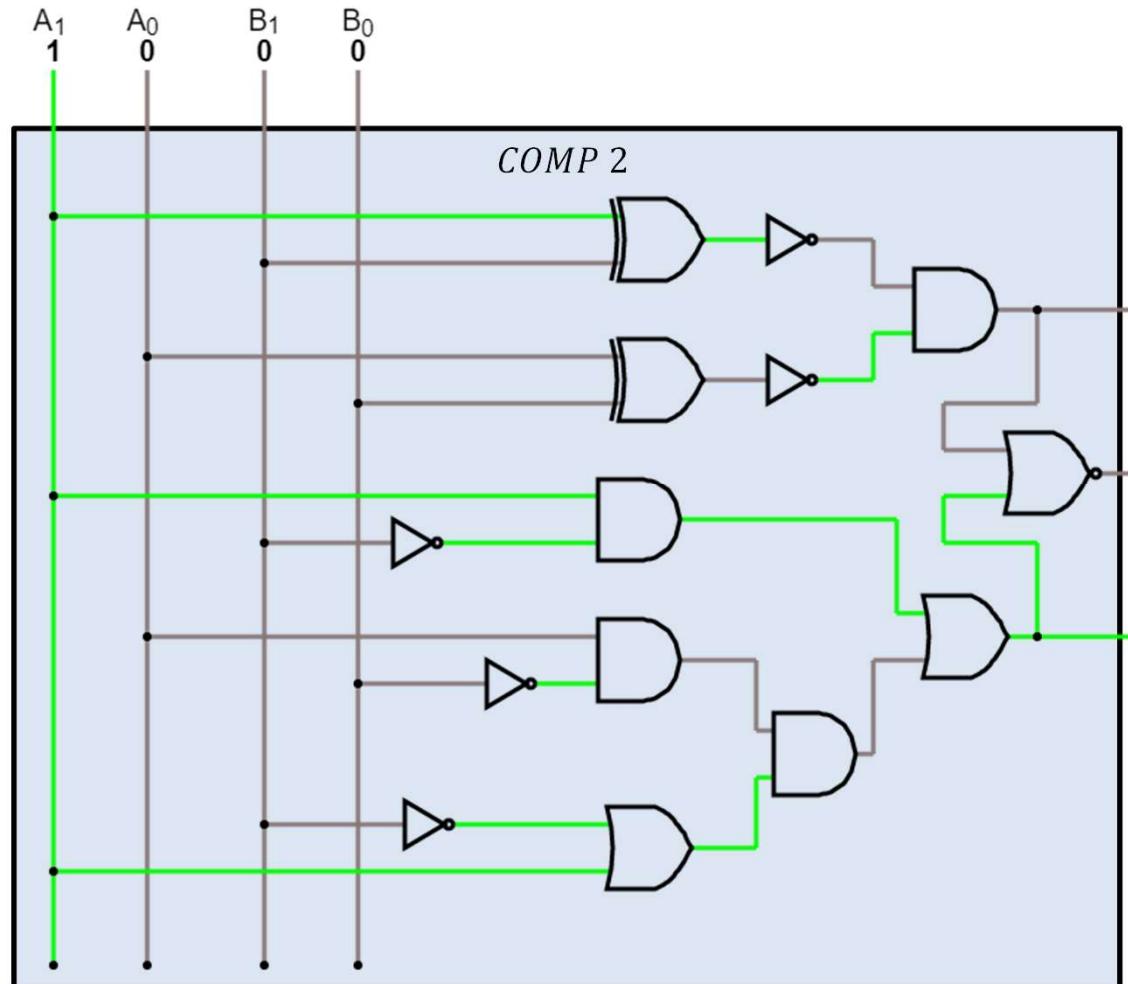
A_1	A_0	B_1	B_0	G	E	L
0	0	0	0		1	
0	0	0	1			1
0	0	1	0			1
0	0	1	1			1
0	1	0	0	1		
0	1	0	1		1	
0	1	1	0			1
0	1	1	1			1
1	0	0	0	1		
1	0	0	1	1		
1	0	1	0		1	
1	0	1	1			1
1	1	0	0	1		
1	1	0	1	1		
1	1	1	0	1		
1	1	1	1			

$\bar{B}_1 \bar{B}_0$	$\bar{B}_1 B_0$	$B_1 B_0$	$B_1 \bar{B}_0$	
$A_1 A_0$	$B_1 B_0$			
$\bar{A}_1 \bar{A}_0$	00	01	11	10
00	0	1	3	2
01	1	4	5	7
11	1	12	13	15
10	1	8	9	11

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

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

Circuito Comparador Completo – 2 bits



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

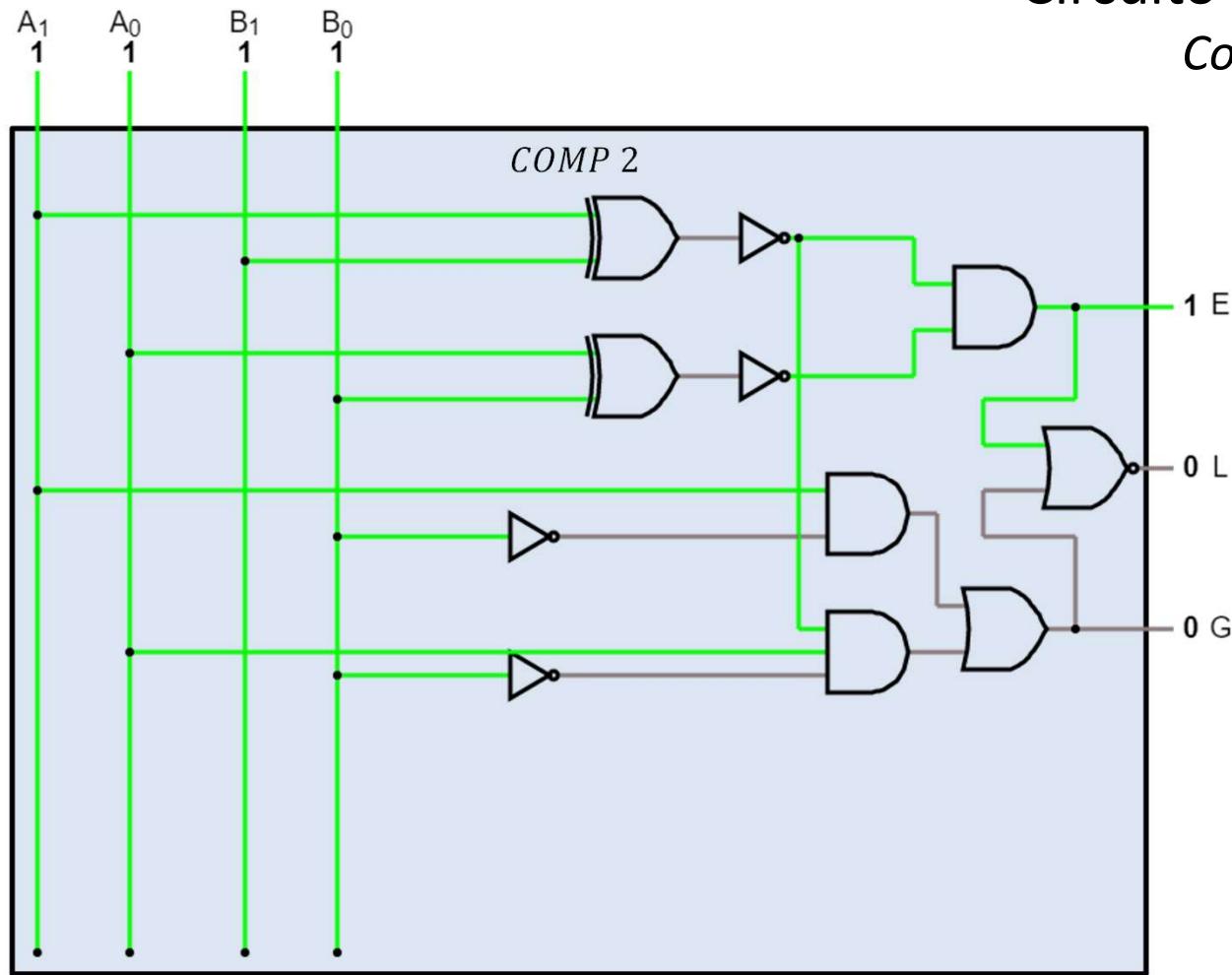
$\bar{B}_1 \bar{B}_0$	$\bar{B}_1 B_0$		$B_1 B_0$	$B_1 \bar{B}_0$	
$A_1 A_0$	$B_1 B_0$	00	01	11	10
$\bar{A}_1 \bar{A}_0$	00		0	1	3
$\bar{A}_1 A_0$	01	1		5	7
$A_1 A_0$	11	1	1	15	14
$A_1 \bar{A}_0$	10	1	1	9	11

$$A_1 \bar{B}_1 + \bar{A}_1 A_0 \bar{B}_1 \bar{B}_0 + A_1 A_0 B_1 \bar{B}_0$$

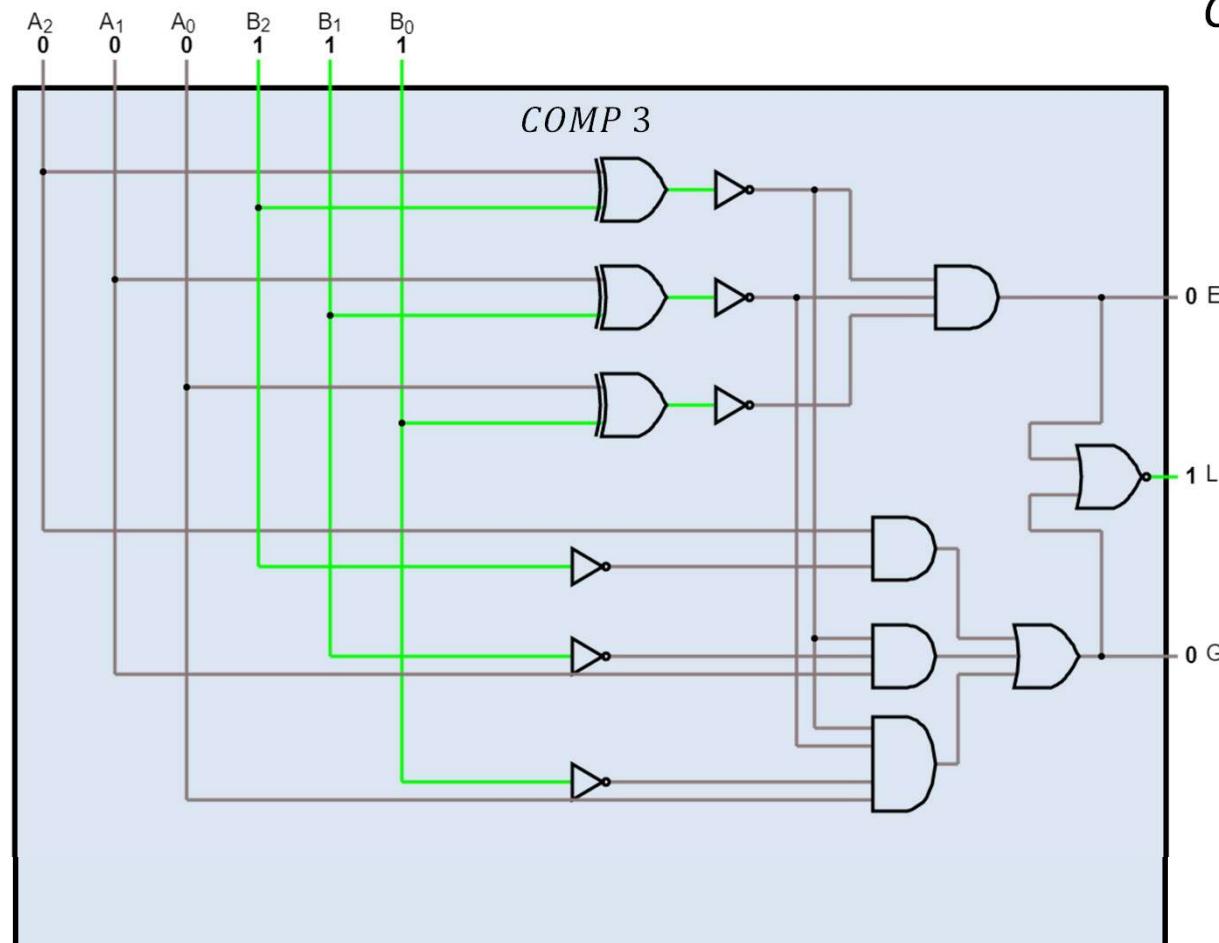
$$G = A_1 \bar{B}_1 + A_0 \bar{B}_0 (\bar{A}_1 \bar{B}_1 + A_1 B_1)$$

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

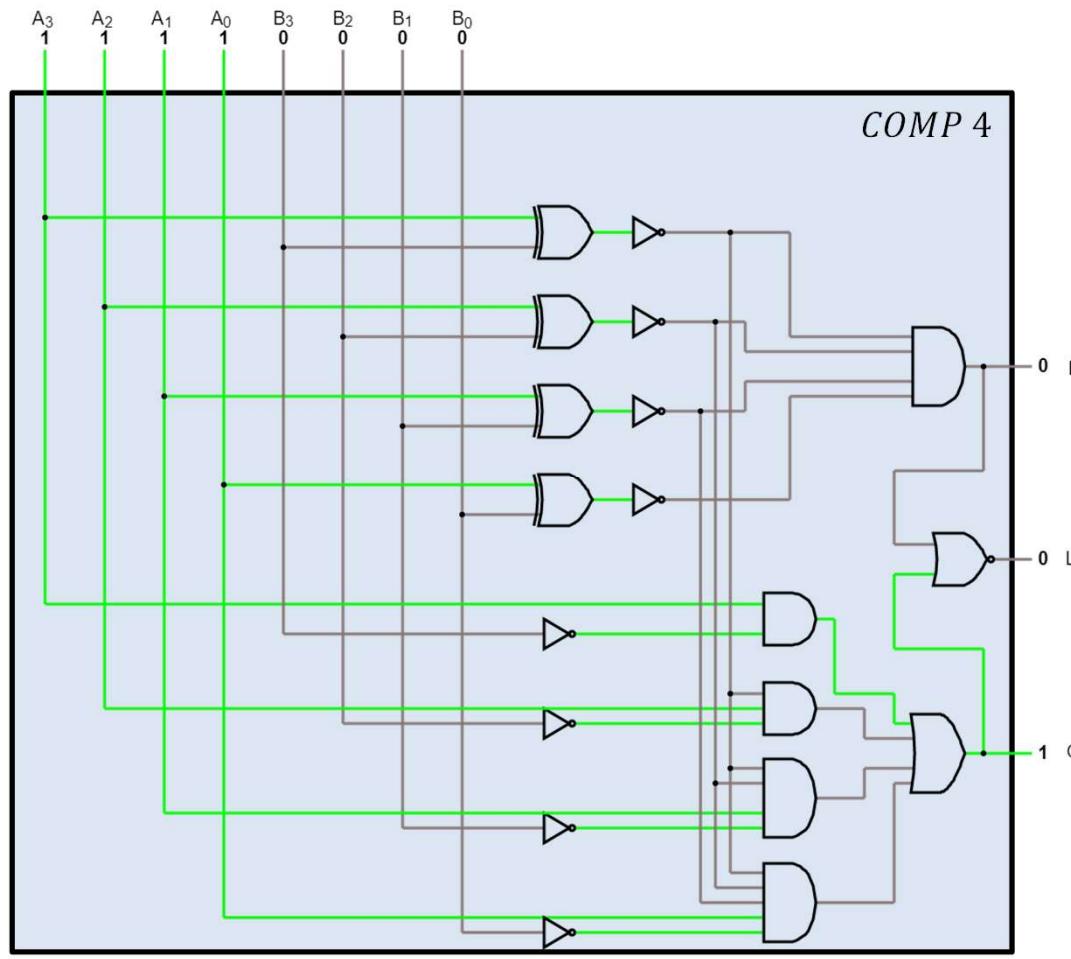
Circuito Comparador Completo – 2 bits



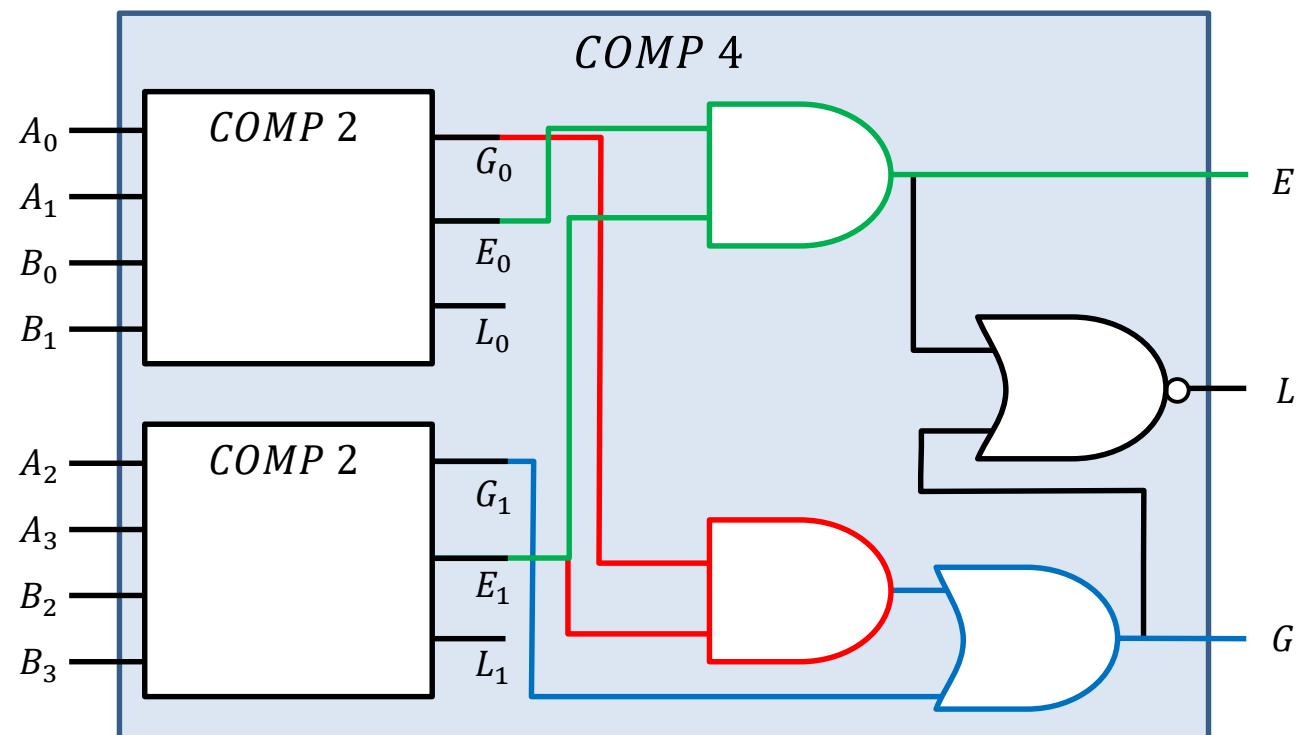
Circuito Comparador Completo – 3 bits



Circuito Comparador Completo – 4 bits

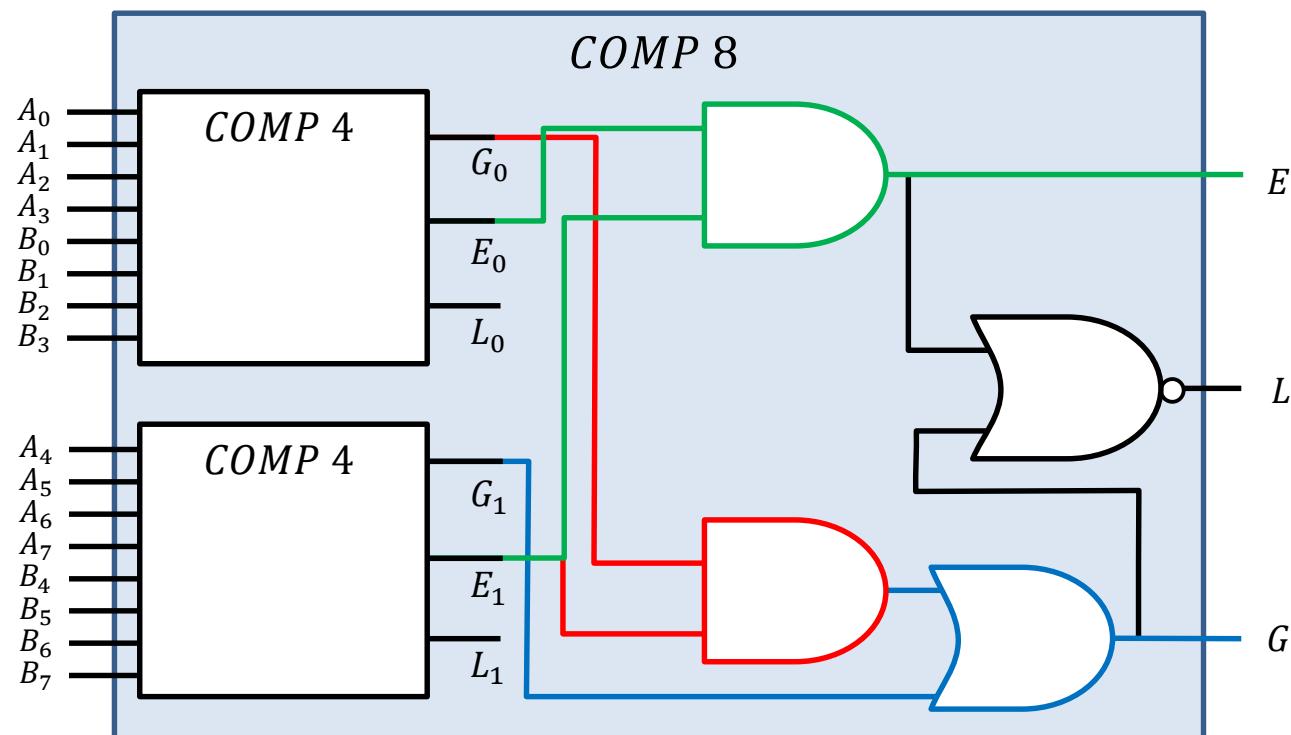


Circuito Comparador Completo – 4 bits



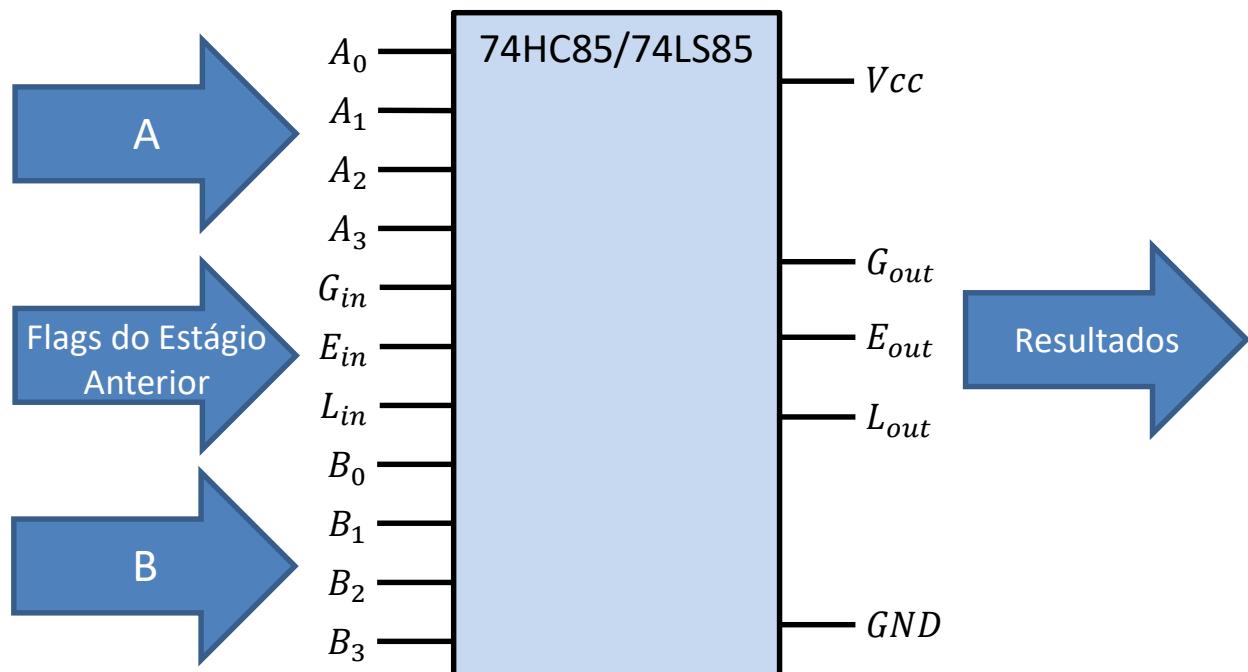
$$G = G_1 + E_1 G_0$$

Circuito Comparador Completo – 8 bits



$$G = G_1 + \underset{g_1}{\cancel{E_1}} G_0$$

Circuito Comparador Completo – 4 bits com Cascata



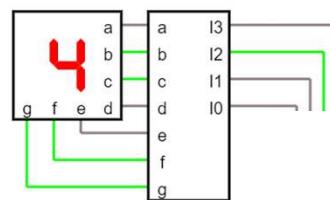
Círculo Comparador

Exercício

Implementar um círculo que entrega o maior entre dois números inteiros de 4 bits:

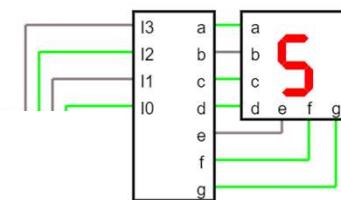
$$X = \text{maior}(A, B)$$

Círcuito Somador Exercício



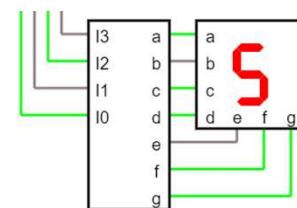
A3 A2 A1 A0 B3 B2 B1 B0

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

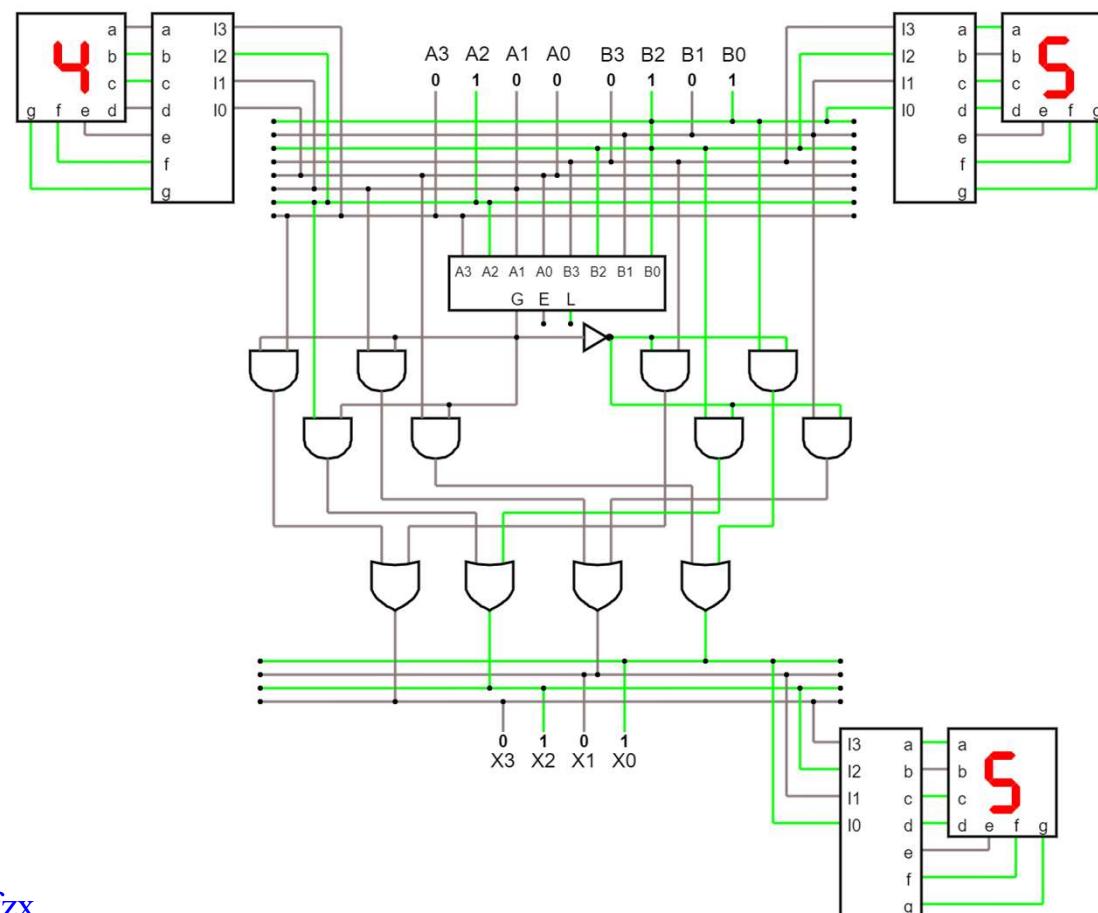


X3 X2 X1 X0

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

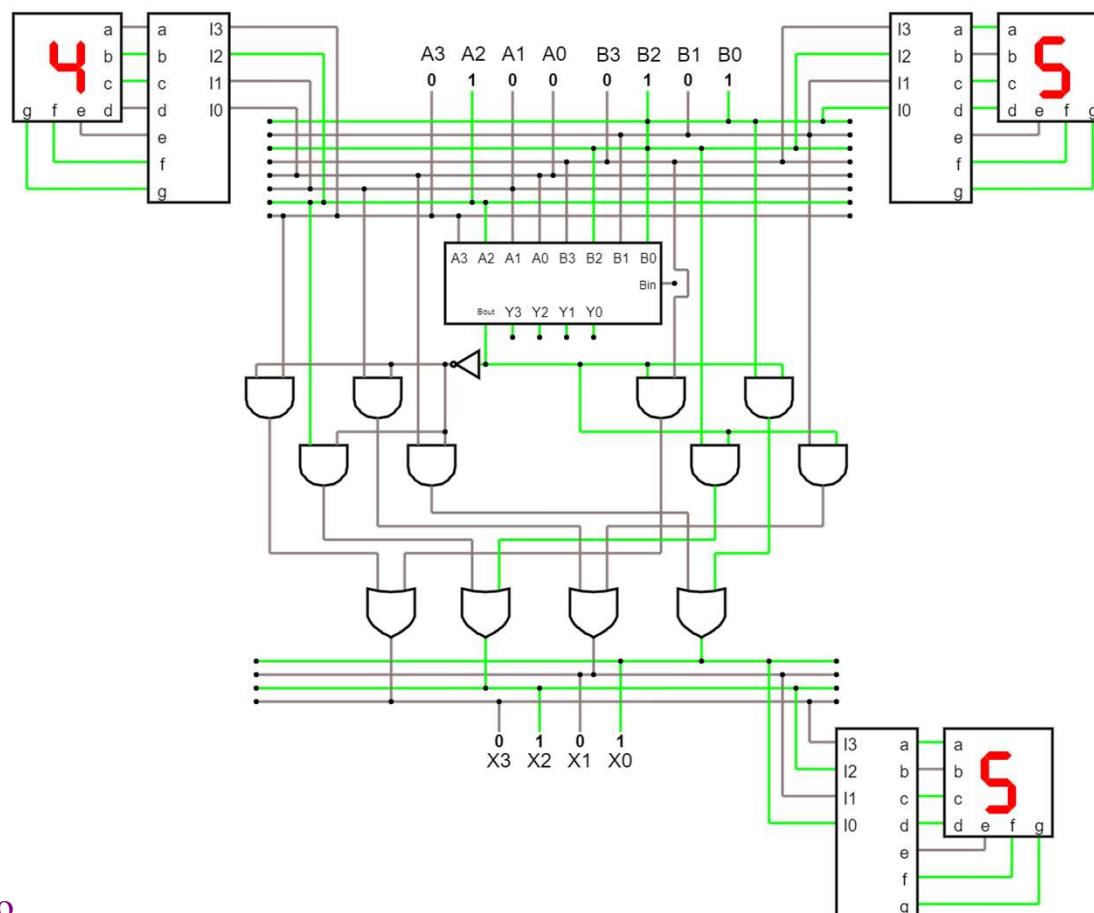


Círcuito Somador *Exercício*



<https://tinyurl.com/yhpwyfzx>

Círcuito Somador *Exercício*



<https://tinyurl.com/yz3pqsgo>

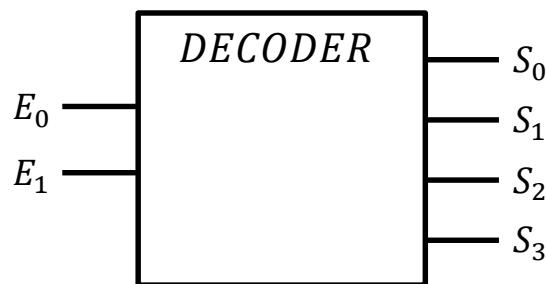
Sumário

- 1. Revisão – Sistemas de Numeração
- 2. Revisão – Representação de Dados
- 3. Revisão – Operações com Binários
- 4. Álgebra Booleana
- 5. Simplificação de Expressões
- 6. Mapa de Karnaugh
- 7. Elementos Lógicos Universais
- 8. Circuitos Combinacionais
- 9. Circuitos Sequenciais

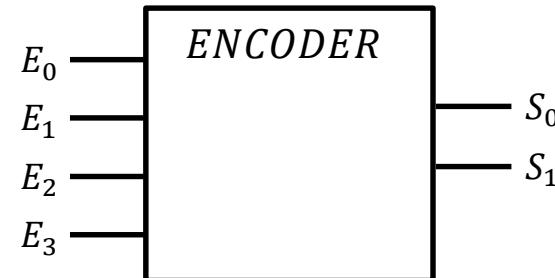
- 1. Somador / Subtrator
- 2. Comparadores
- 3. Codificador/decodificador
- 4. Multiplexador/Demux
- 5. Geradores de paridade
- 6. ULA
- 7. Multiplicadores / Divisores
- 8. PLD/PLA/ROM

Decoder/Encoder

- 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 $2^n \Rightarrow n$

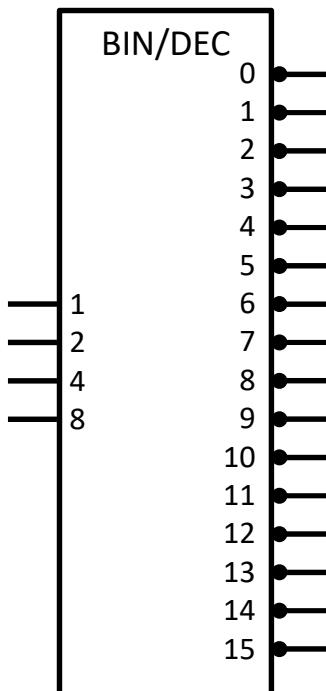


Formato Binário \Rightarrow Formato X



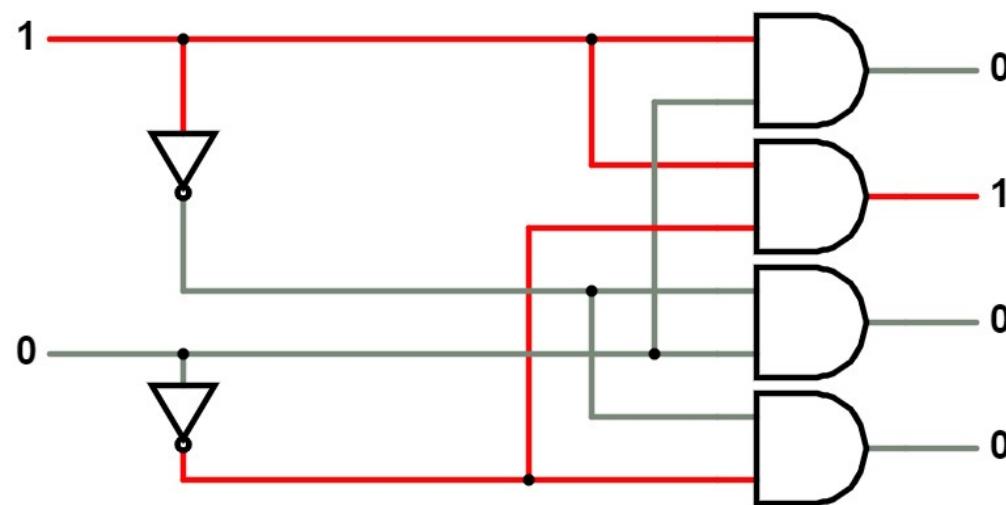
Formato X \Rightarrow Formato Binário

Decoder



Valor Decimal	ENTRADAS				Função de Decodificação	SAÍDAS														
	A_3	A_2	A_1	A_0		0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
0	0	0	0	0	$\bar{A}_3\bar{A}_2\bar{A}_1\bar{A}_0$	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	1	$\bar{A}_3\bar{A}_2\bar{A}_1A_0$	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
2	0	0	1	0	$\bar{A}_3\bar{A}_2A_1\bar{A}_0$	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
3	0	0	1	1	$\bar{A}_3\bar{A}_2A_1A_0$	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0
4	0	1	0	0	$\bar{A}_3A_2\bar{A}_1\bar{A}_0$	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
5	0	1	0	1	$\bar{A}_3A_2\bar{A}_1A_0$	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0
6	0	1	1	0	$\bar{A}_3A_2A_1\bar{A}_0$	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0
7	0	1	1	1	$\bar{A}_3A_2A_1A_0$	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0
8	1	0	0	0	$A_3\bar{A}_2\bar{A}_1\bar{A}_0$	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0
9	1	0	0	1	$A_3\bar{A}_2\bar{A}_1A_0$	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0
10	1	0	1	0	$A_3\bar{A}_2A_1\bar{A}_0$	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0
11	1	0	1	1	$A_3\bar{A}_2A_1A_0$	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0
12	1	1	0	0	$A_3A_2\bar{A}_1\bar{A}_0$	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0
13	1	1	0	1	$A_3A_2\bar{A}_1A_0$	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0
14	1	1	1	0	$A_3A_2A_1\bar{A}_0$	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
15	1	1	1	1	$A_3A_2A_1A_0$	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Decoder

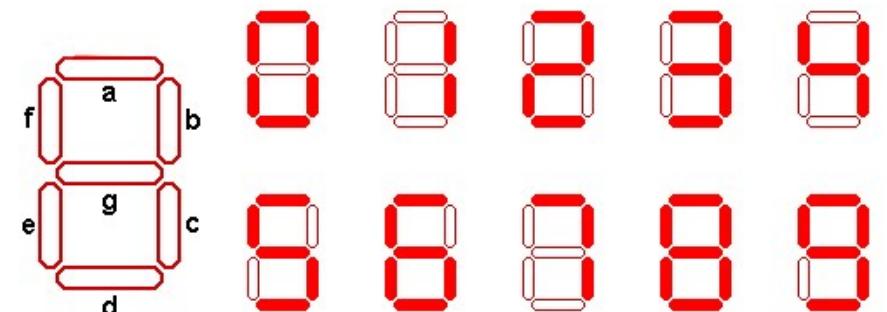
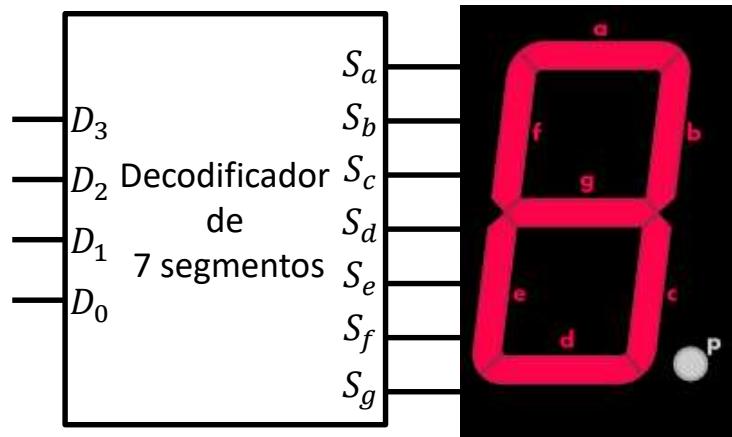


Decodificador $2: 2^2$

<https://tinyurl.com/yf9k8kx4>

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

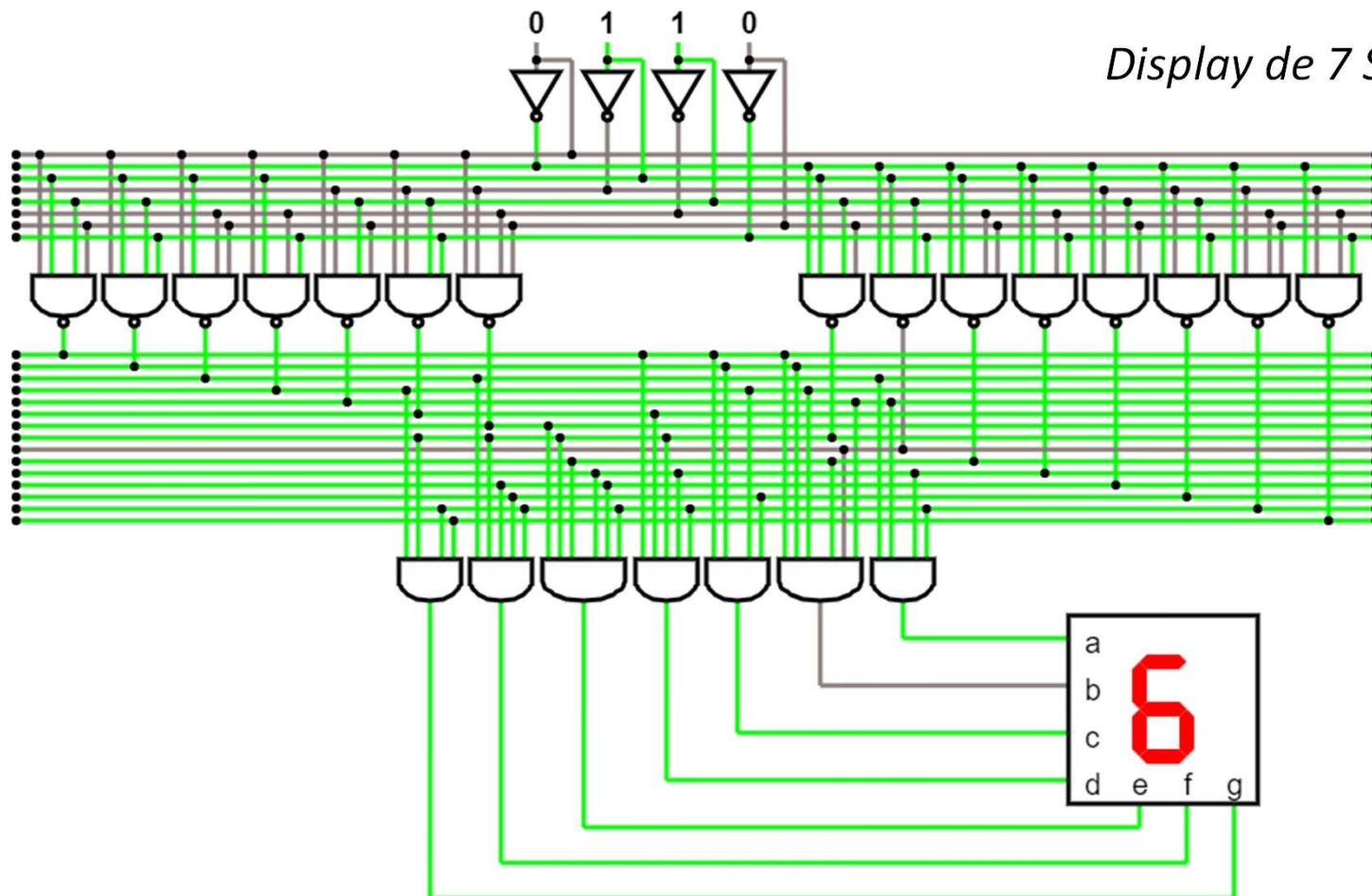
Exemplo: Como agrupar e qual a função? $f_a = \sum(0,2,3,5,6,7,8,9)$

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

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

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

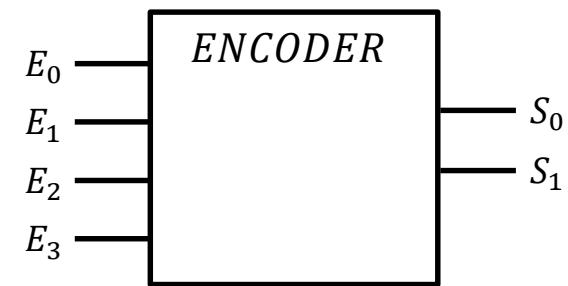
Decoder
Display de 7 Segmentos



Sumário

- 1. Revisão – Sistemas de Numeração
- 2. Revisão – Representação de Dados
- 3. Revisão – Operações com Binários
- 4. Álgebra Booleana
- 5. Simplificação de Expressões
- 6. Mapa de Karnaugh
- 7. Elementos Lógicos Universais
- 8. Circuitos Combinacionais
- 9. Circuitos Sequenciais
- 1. Somador / Subtrator
- 2. Comparadores
- 3. Codificador/decodificador
- 4. Multiplexador/Demux
- 5. Geradores de paridade
- 6. ULA
- 7. Multiplicadores / Divisores
- 8. PLD/PLA/ROM

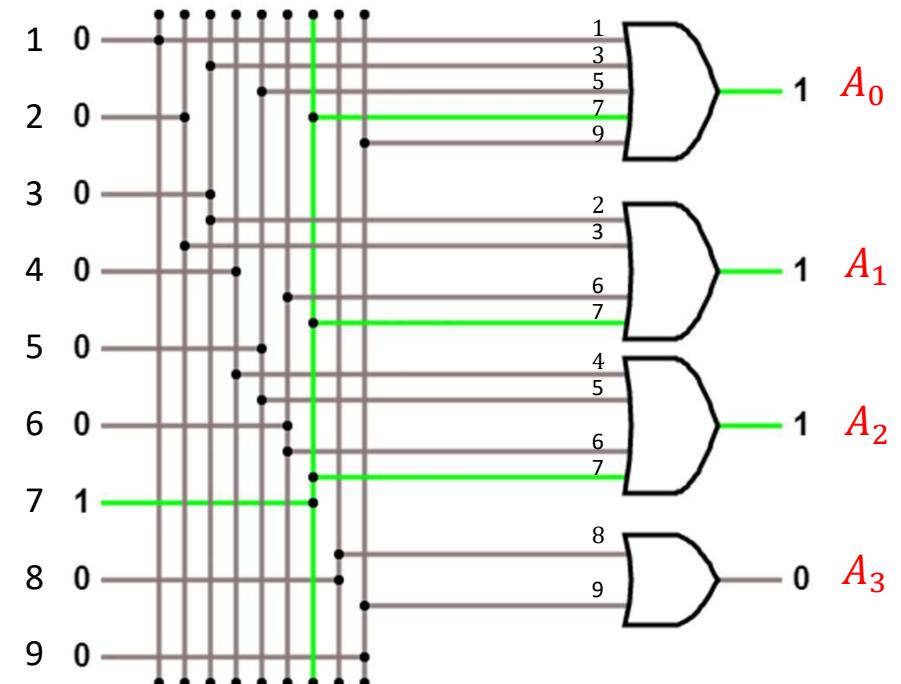
Encoder



Formato X \Rightarrow Formato Binário

Encoder

DEC/BIN	
$0 = 0$	0
$1 = 1$	1
$2 = 2$	2
$3 = 1 + 2$	3
$4 = 4$	1 2 4
$5 = 1 + 4$	1 2 5
$6 = 2 + 4$	1 2 4 6
$7 = 1 + 2 + 4$	1 2 4 7
$8 = 8$	8
$9 = 1 + 8$	1 8

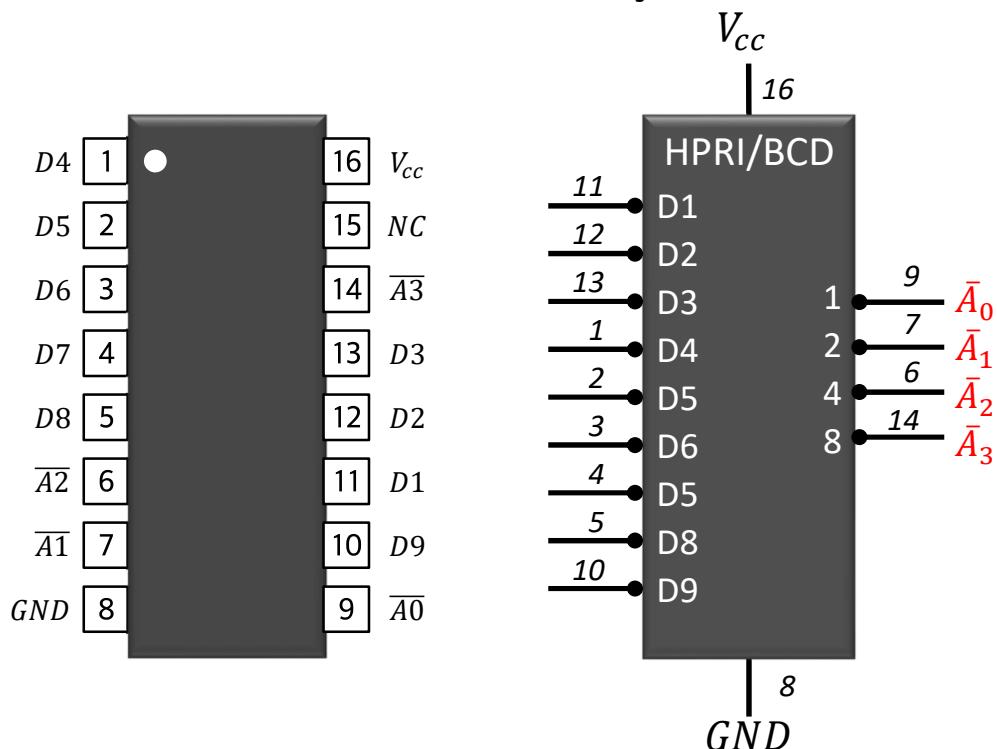


Considerando que apenas uma entrada será ativada.

<https://tinyurl.com/ydmd96ve>

Encoder

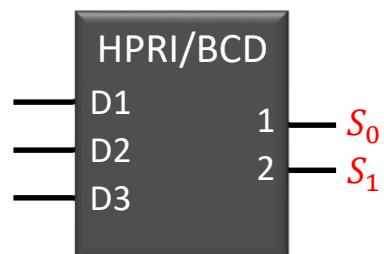
- O 74HC147 é um codificador com prioridade e entradas ativas em *LOW* (0).
- As saídas são codificações em *BCD*, ativas em *LOW*.



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

Encoder

- Obter a Tabela Verdade, Expressões e Circuito Lógico para um Encoder com prioridade para *BCD* de 3 bits. As entradas e saídas são ativas em *HIGH* (1).



Encoder

- Tabela Verdade para entradas e saídas ativas em *HIGH* (1). Encoder com prioridade para *BCD* de 3 bits:

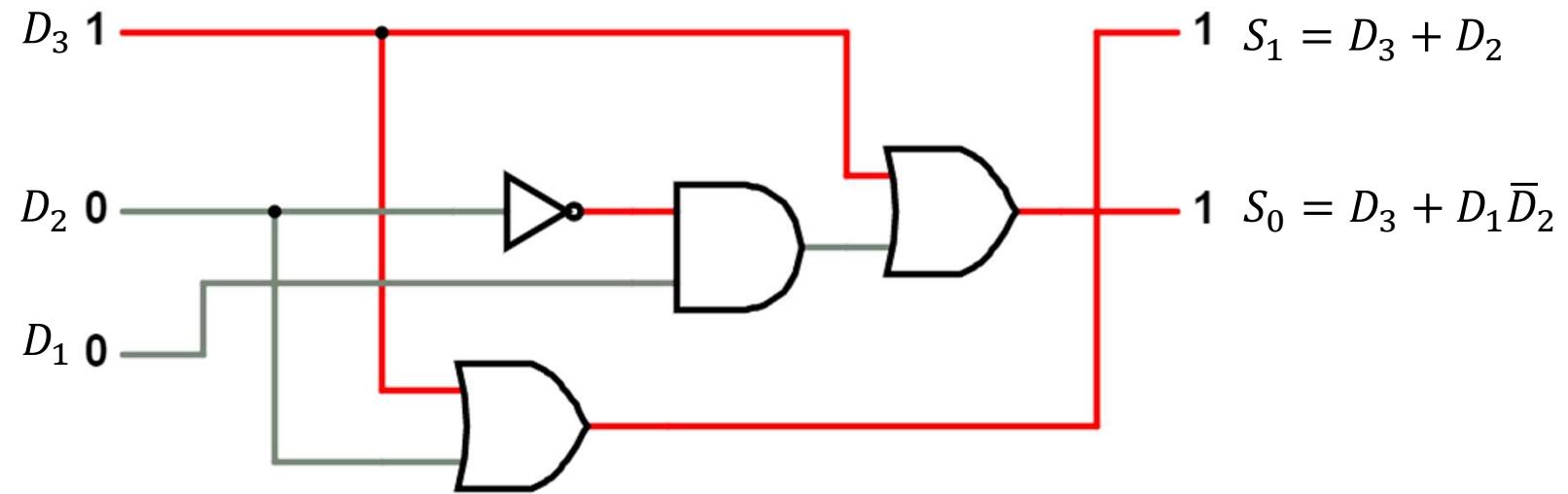
<i>D</i> 3	<i>D</i> 2	<i>D</i> 1	<i>S</i> 1	<i>S</i> 0
0	0	0	0	0
0	0	1	0	1
0	1	0	1	0
0	1	1	1	0
1	0	0	1	1
1	0	1	1	1
1	1	0	1	1
1	1	1	1	1

<i>D</i> 3	<i>D</i> 2	<i>D</i> 1	<i>S</i> 1	<i>S</i> 0
0	0	0	0	0
1	X	X	1	1
0	1	X	1	0
0	0	1	0	1

$$S_1 = D_3 + D_2 \bar{D}_3 = D_3 + D_2$$

$$S_0 = D_3 + D_1 \bar{D}_2 \bar{D}_3 = D_3 + D_1 \bar{D}_2$$

Encoder

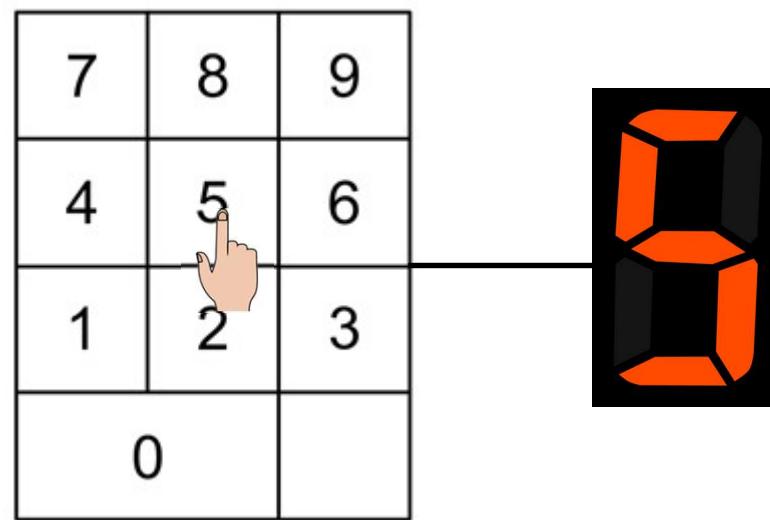


Codificador de Prioridade

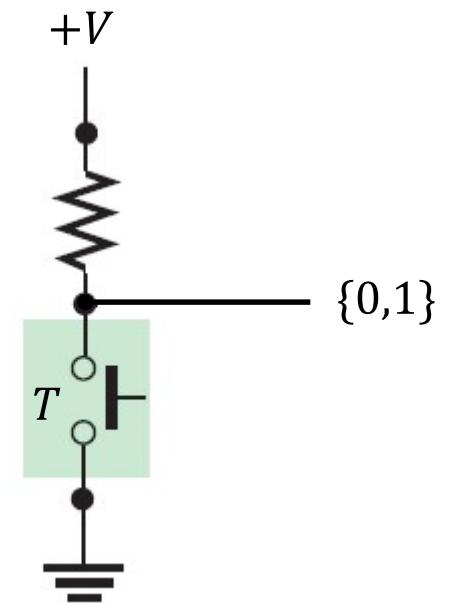
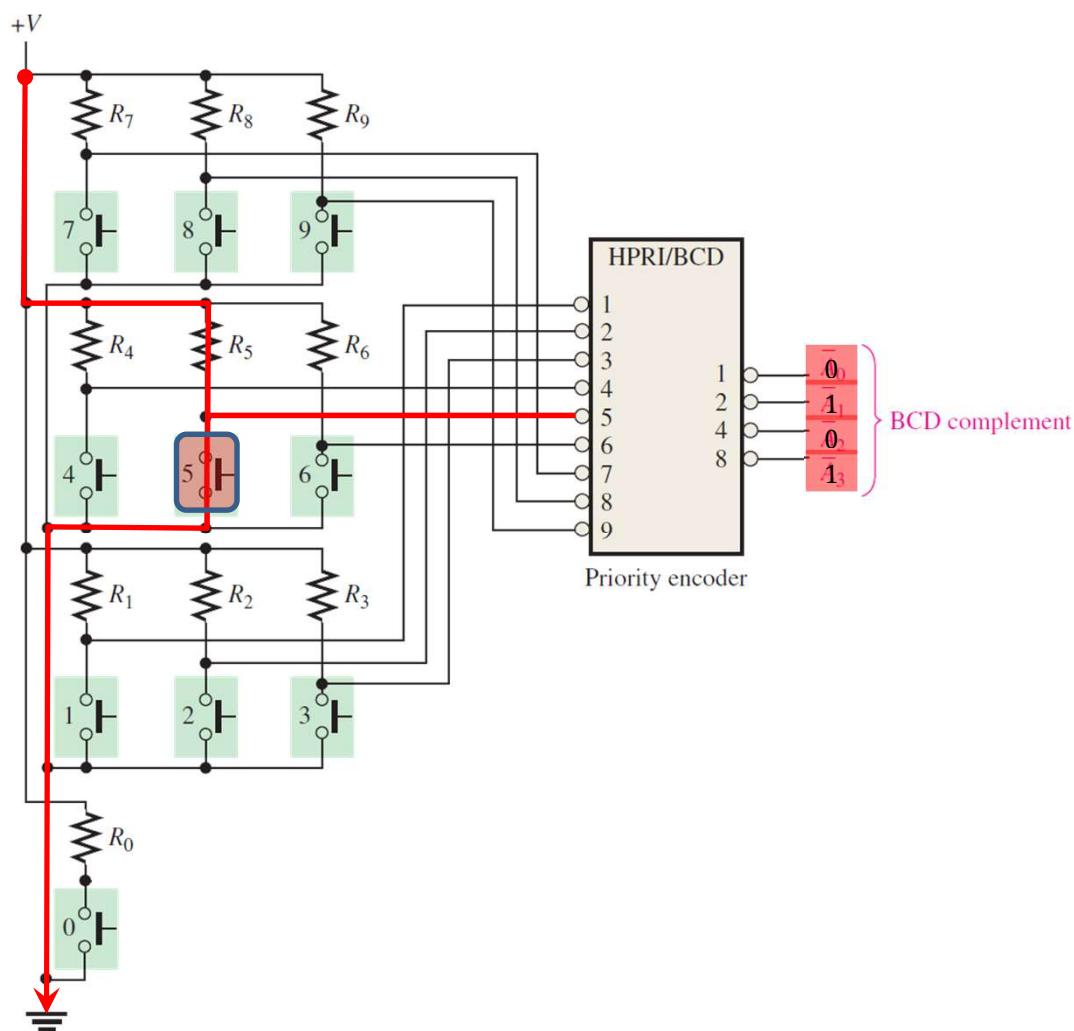
<https://tinyurl.com/yg7c6tnf>

Encoder

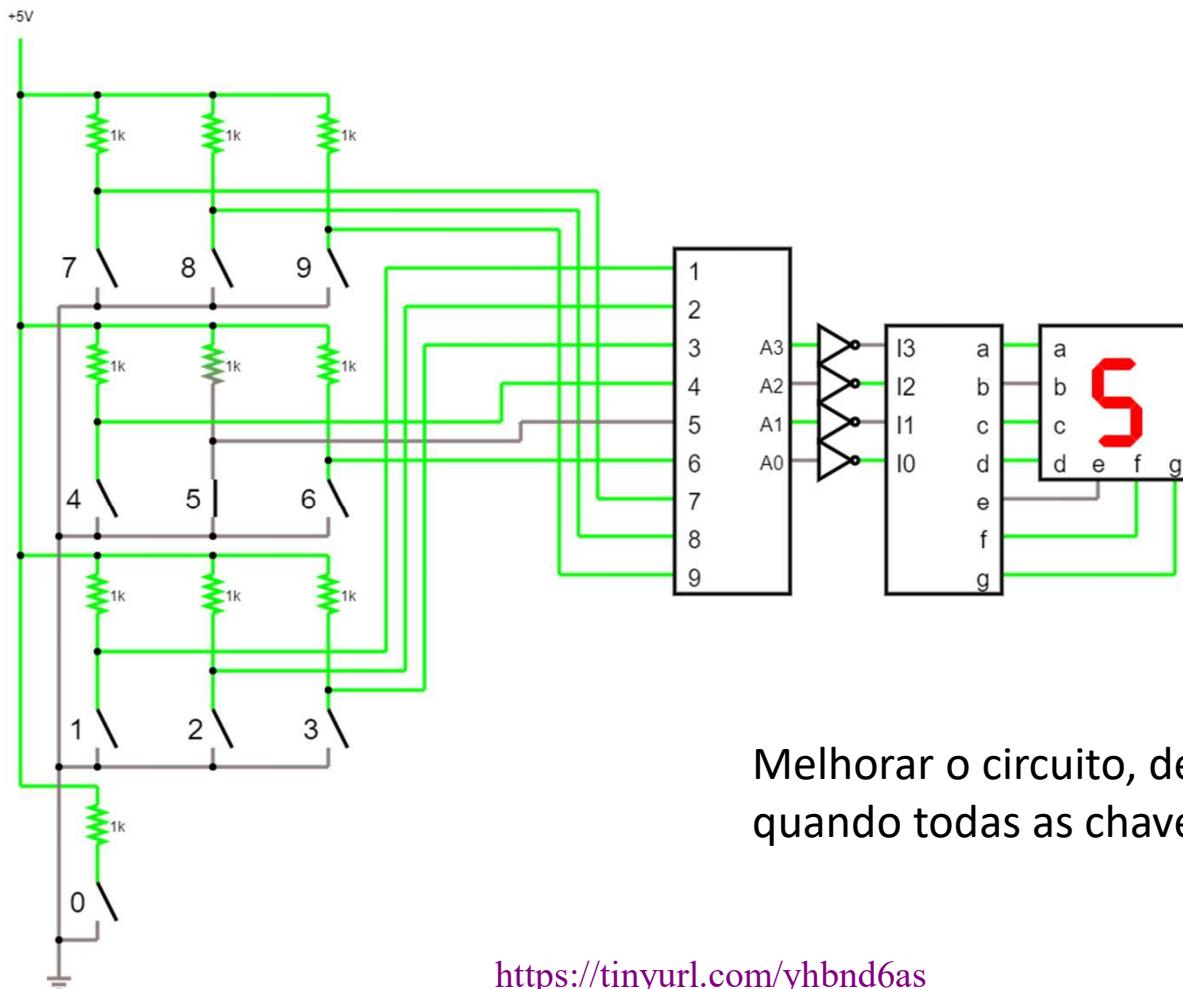
- Elaborar um circuito que emula um teclado numérico de 0 a 9.



Encoder



Encoder



Melhorar o circuito, desligando o display quando todas as chaves estiverem abertas.

<https://tinyurl.com/yhbnd6as>

Sumário

- 1. Revisão – Sistemas de Numeração
- 2. Revisão – Representação de Dados
- 3. Revisão – Operações com Binários
- 4. Álgebra Booleana
- 5. Simplificação de Expressões
- 6. Mapa de Karnaugh
- 7. Elementos Lógicos Universais
- 8. Circuitos Combinacionais
- 9. Circuitos Sequenciais
- 1. Somador / Subtrator
- 2. Comparadores
- 3. Codificador/decodificador
- 4. Multiplexador/Demux
- 5. Geradores de paridade
- 6. ULA
- 7. Multiplicadores / Divisores
- 8. PLD/PLA/ROM