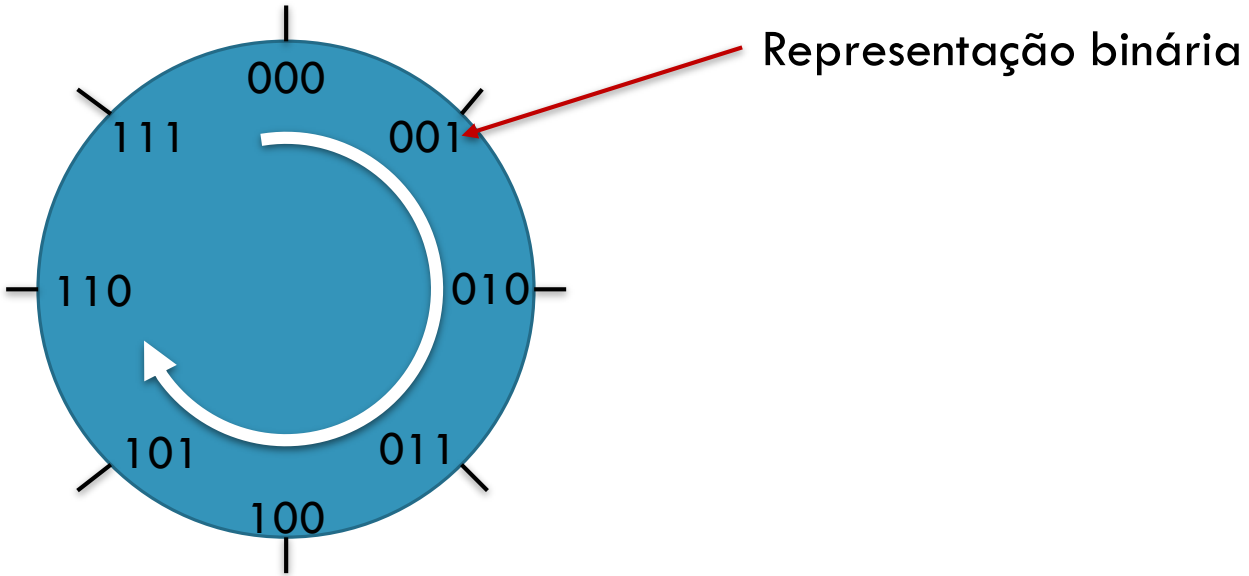
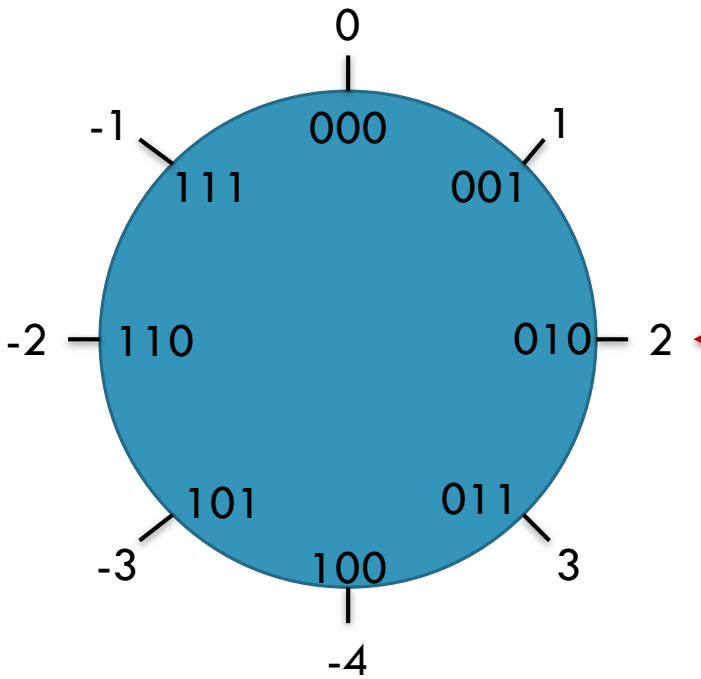
PROJETOS DIGITAIS E MICROPROCESSADORES
UNIDADE LÓGICA E ARITMÉTICA

Marco A. Zanata Alves

REPRESENTAÇÃO BINÁRIA COMPLEMENTO DE 2



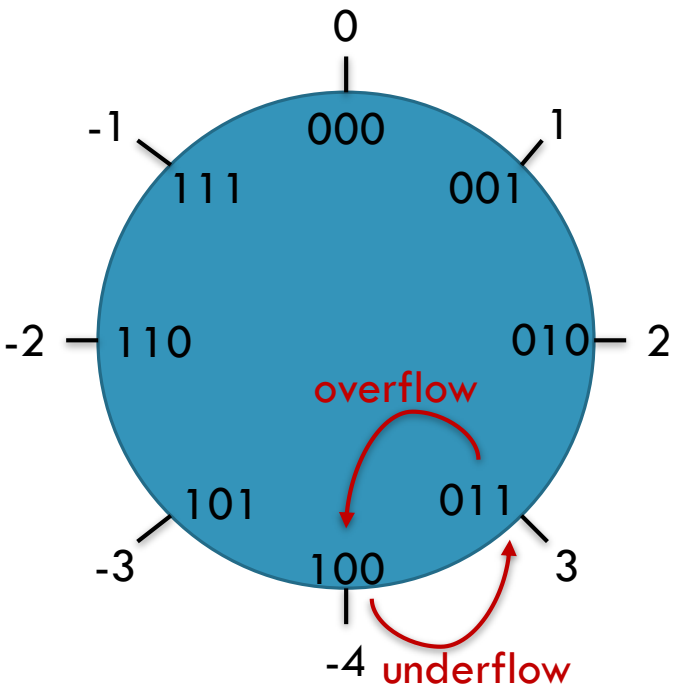
REPRESENTAÇÃO BINÁRIA COMPLEMENTO DE 2



-4	2	1

Representação decimal

REPRESENTAÇÃO BINÁRIA COMPLEMENTO DE 2



Overflow, acontece quando o valor armazenado é maior (em magnitude) do que o registrador pode armazenar ou representar

Underflow, acontece quando tentamos armazenar um valor menor que o menor valor absoluto que o registrador é capaz de armazenar ou representar

OVERFLOW DURANTE SOMAS

Se X e Y tiverem sinais opostos, nunca ocorrerá overflow

Overflow, poderá ocorrer apenas se X e Y tiverem sinais iguais

Overflow acontece quando X e Y tem sinais iguais, e o resultado tiver um sinal diferente

Pos + Pos = Neg ← Overflow

Neg + Neg = Pos ← Overflow

REPRESENTAÇÃO BINÁRIA COMPLEMENTO DE 2

-128	64	32	16	8	4	2	1

Converta os números:

$1100\ 0000_2$

$0100\ 0000_2$

$1111\ 1111_2$

$1000\ 0000_2$

$1000\ 1111_2$

Qual o bit **mais** significativo?

INTRODUÇÃO A UNIDADE LÓGICA E ARITMÉTICA (ULA)

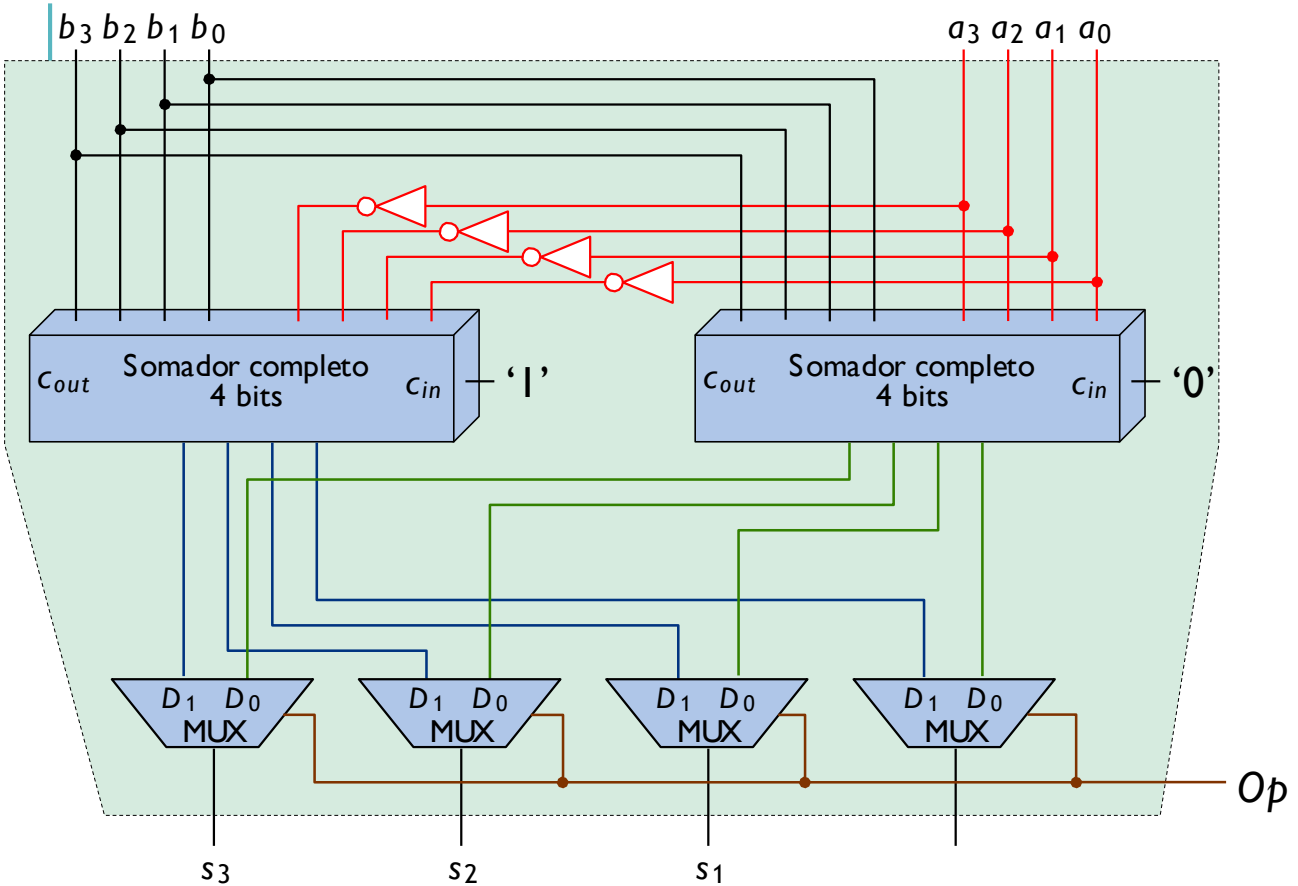
Construa um circuito com:

- 8 entradas de dados $b_3, b_2, b_1, b_0, a_3, a_2, a_1, a_0$
- 1 entrada de seleção Op
- 4 saídas s_3, s_2, s_1, s_0

tal que:

$(S_3S_2S_1S_0) = A + B$ se Op=0, $A - B$ se Op=1

Todas as operações são com números sem sinal. Desconsidere os casos em que há overflow.


b3 b2 b1 b0
a3 a2 a1 a0
Cout
Somador completo
4 bits
Cin
'1'
Cout
Somador completo
4 bits
Cin
'0'
D1 D0
MUX
D1 D0
MUX
D1 D0
MUX
D1 D0
MUX
Op
s3
s2
s1

UNIDADE LÓGICA-ARITMÉTICA

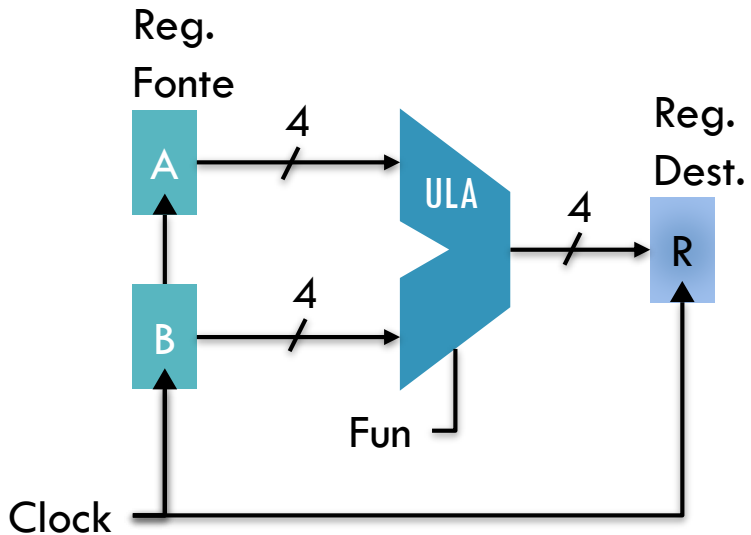
Unidade Lógica-Aritmética (ULA): circuito digital que faz operações lógicas e aritméticas. A operação a ser feita é selecionada pelos bits de seleção de operação/função

A ULA do exercício anterior só possui 1 bit de operação, para escolher entre soma e subtração.

Representado por um trapézio chanfrado

UNIDADE LÓGICA-ARITMÉTICA

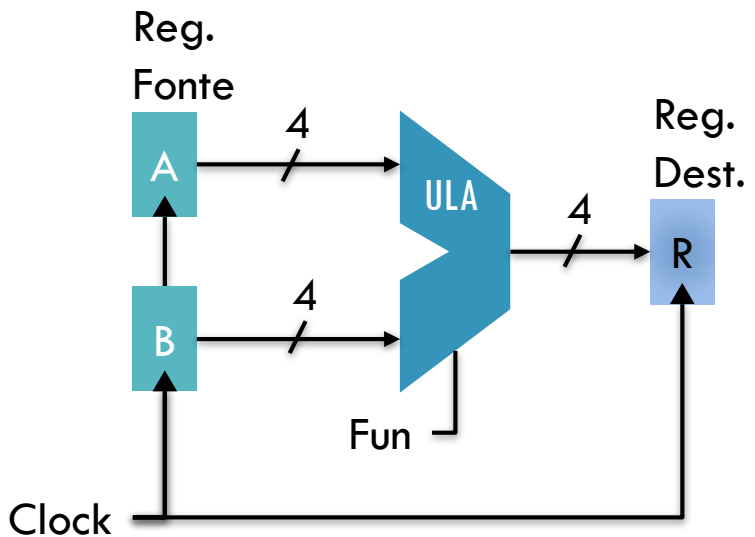
$$RegDest <= ULA(Fun, regA, regB \mid Fun \in \{+, -, \ll, \gg, \wedge, \vee, \underline{\vee}\}$$



Qtos. bits para Fun?

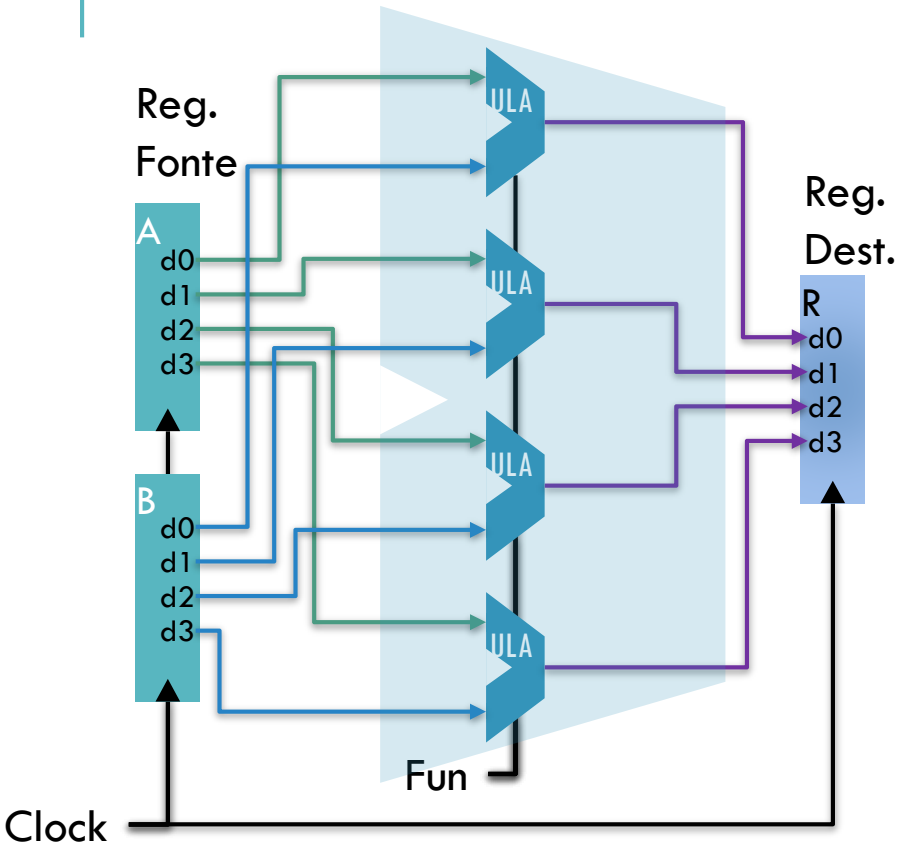
UNIDADE LÓGICA-ARITMÉTICA

$$RegDest <= ULA(Fun, regA, regB \mid Fun \in \{+, -, \ll, \gg, \wedge, \vee, \underline{\vee}, \neg\}$$



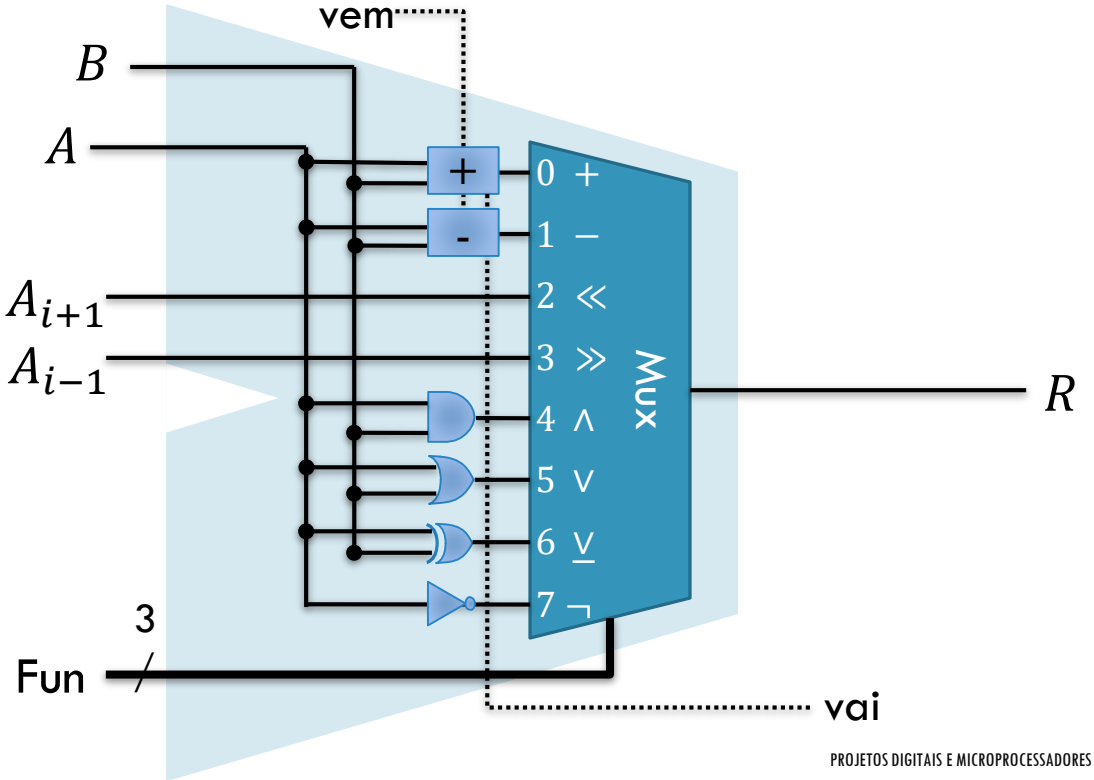
$$A = 0101$$
$$B = 0001$$
$$A + B = 0101 + 0001 = 1011$$
$$A - B = 0101 - 0001 = 0100$$
$$A \ll 1 = 0101 \ll 1 = 1010$$
$$A \gg 1 = 0101 \gg 1 = 0010$$
$$A \wedge B = 0101 \wedge 0001 = 0001$$
$$A \vee B = 0101 \vee 0001 = 0101$$
$$A \underline{\vee} B = 0101 \underline{\vee} 0001 = 0100$$
$$\neg A = \neg 0101 = 1010$$

UNIDADE LÓGICA-ARITMÉTICA



O que temos dentro de uma ULA?

DENTRO DA UNIDADE LÓGICA-ARITMÉTICA

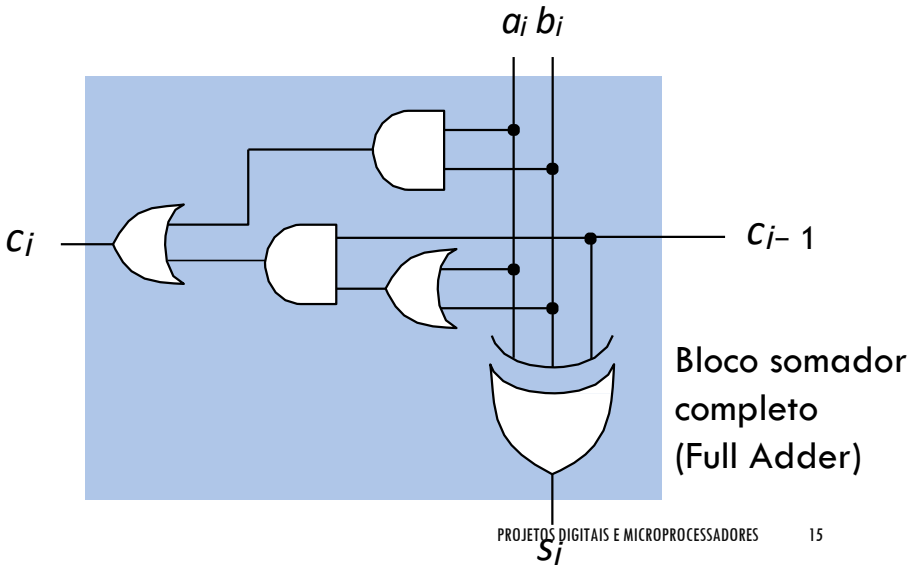


OTIMIZANDO SOMADORES

BLOCOS SOMADORES BINÁRIOS

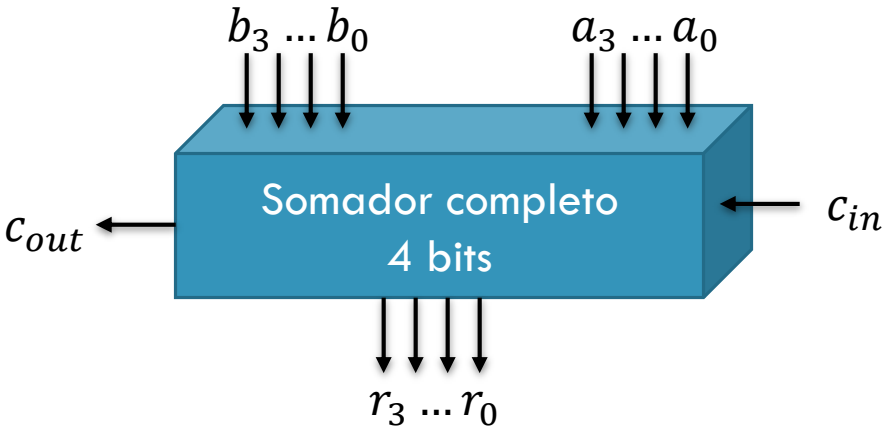
$$s_i = a_i \oplus b_i \oplus c_{i-1}$$

$$c_i = a_i b_i + (a_i + b_i) \cdot c_{i-1}$$

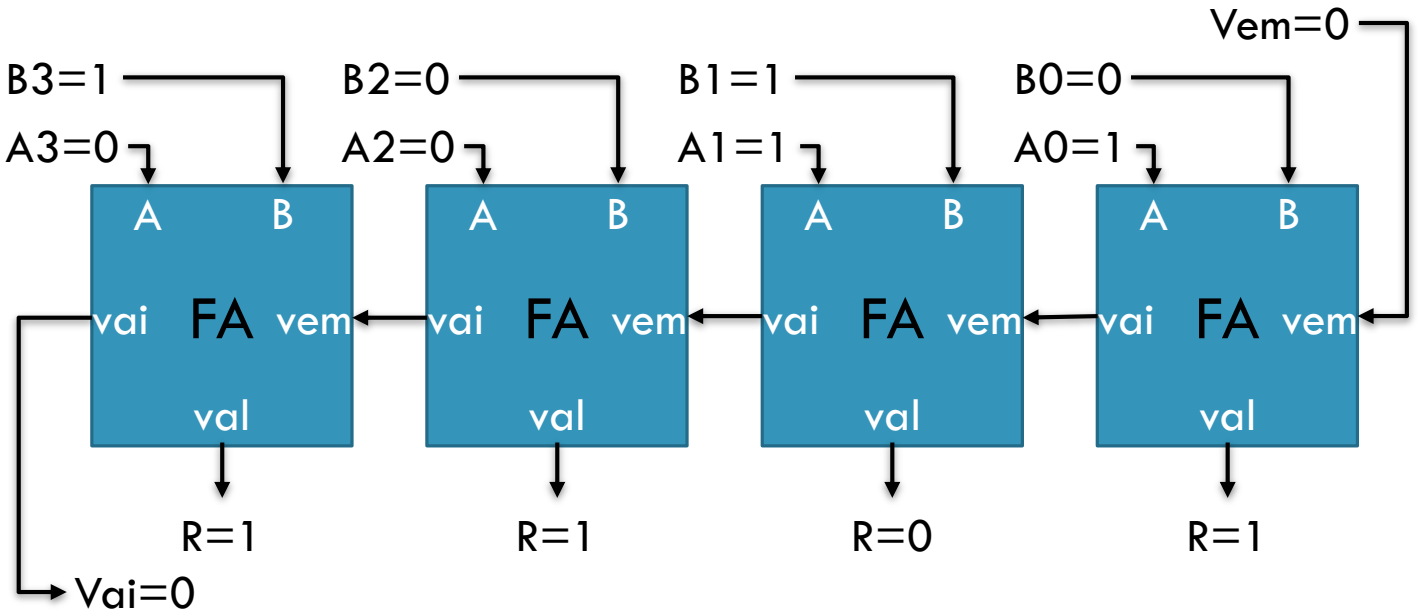


SOMADOR DE 4 BITS

Construa um somador de 4 bits, com FullAdders



SOMADOR DE 4 BITS

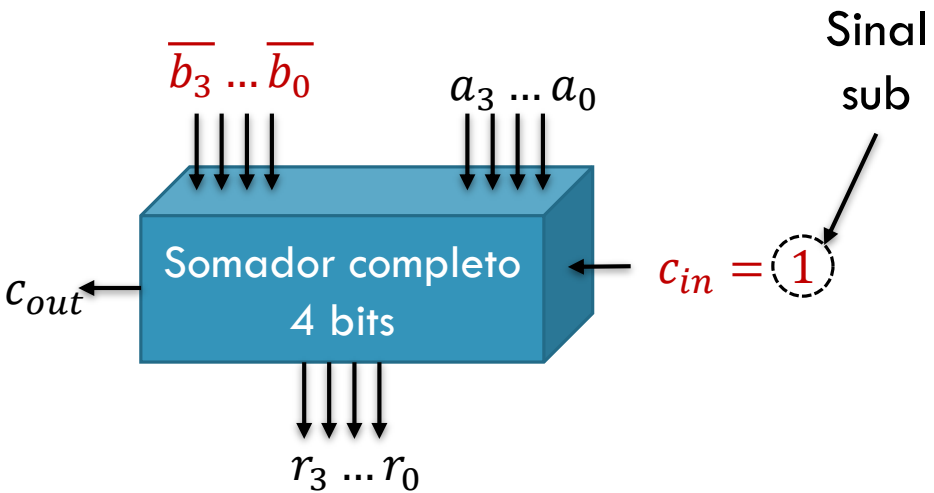


SUBTRATOR DE 4 BITS

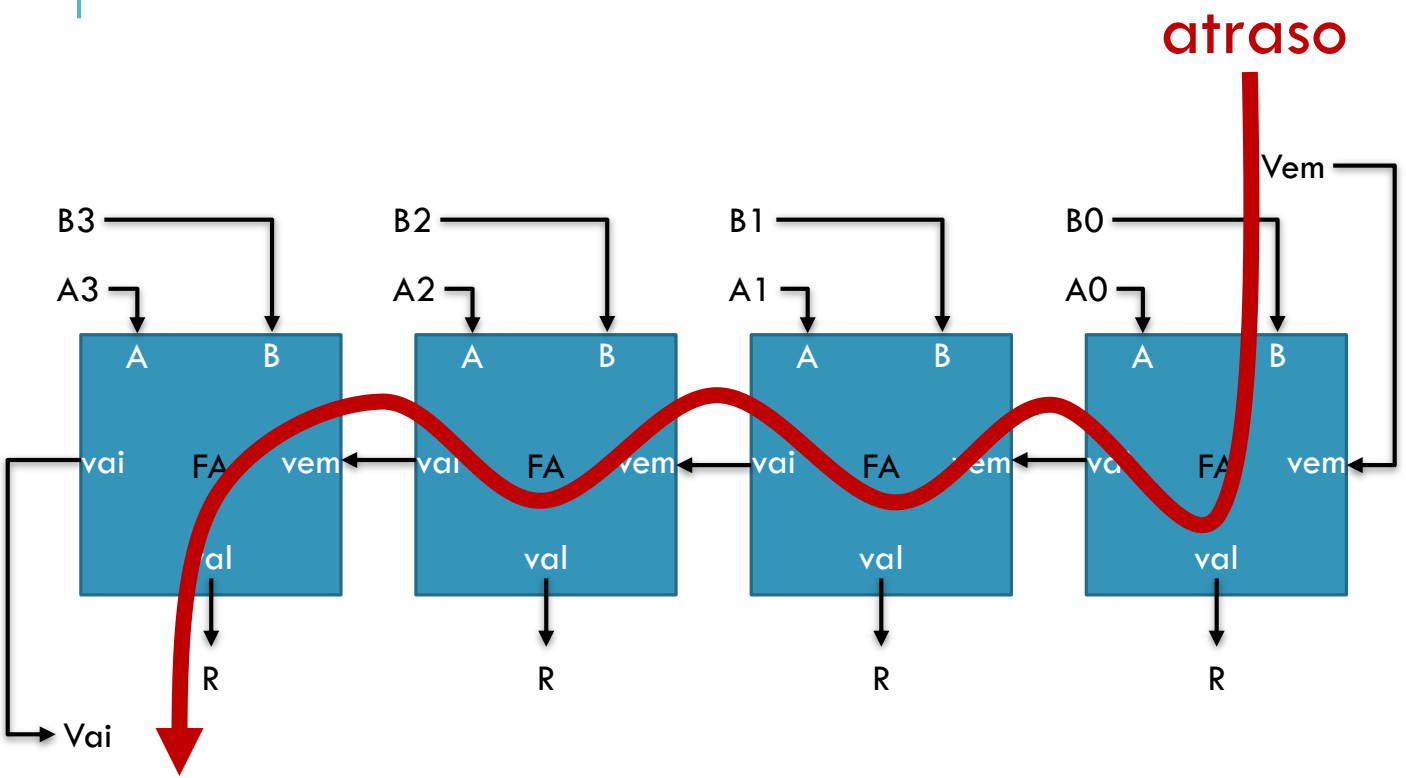
Construa um subtrator de 4 bits

SUBTRATOR DE 4 BITS

Construa um subtrator de 4 bits



MELHORA NO SOMADOR



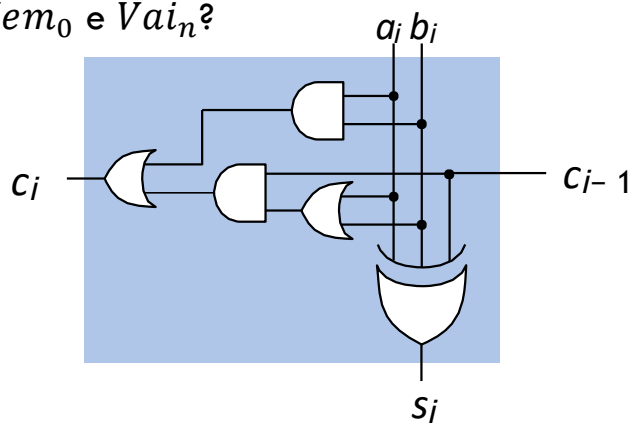
MELHORA NO SOMADOR

Os somadores com propagação de Vai-um (ripple carry) são muito lentos

O resultado (Vai_n) só será conhecido após todos os $(Vem_m, m < n)$ serem computados, um após o outro

Qual o número de portas lógicas entre Vem_0 e Vai_n?

Ideia: adiantar o sinal de Vai-um



SOMADOR CARRY-LOOKAHEAD

Esse somador utiliza uma técnica para aceleração do cálculo do carry

Para cada soma, nós vamos adiantar o sinal de carry a partir das entradas já disponíveis, sem esperar a propagação das informações sendo calculadas

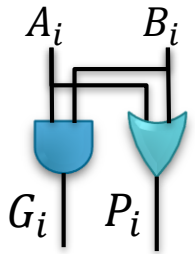
Para isso iremos precisar de mais portas lógicas para transformar trabalho sequencial em trabalho paralelo

Com o aumento da lógica, normalmente fazemos blocos somadores de 4 bits. Onde os blocos são ligados em série

SOMADOR CARRY-LOOKAHEAD

Iremos então antecipar o calculo do vai-um em função dos bits menos significativos

A	B	V_0	R	V_1
0	0	0	0	0
0	1	0	1	0
1	0	0	1	0
1	1	0	0	1
0	0	1	1	0
0	1	1	0	1
1	0	1	0	1
1	1	1	1	1



$G_i = a_i \cdot b_i$ Gera vai-um

$P_i = a_i + b_i$ Propaga vai-um

SOMADOR CARRY-LOOKAHEAD

$G_i = a_i \cdot b_i$ Gera vai-um

$P_i = a_i + b_i$ Propaga vai-um

Assim calculamos 4 vai-uns em paralelo!

Evita-se calcular 5 vai-uns em paralelo devido ao custo em portas lógicas, além do atraso não compensar

$$V_1 = G_0 + (P_0 \cdot V_0)$$

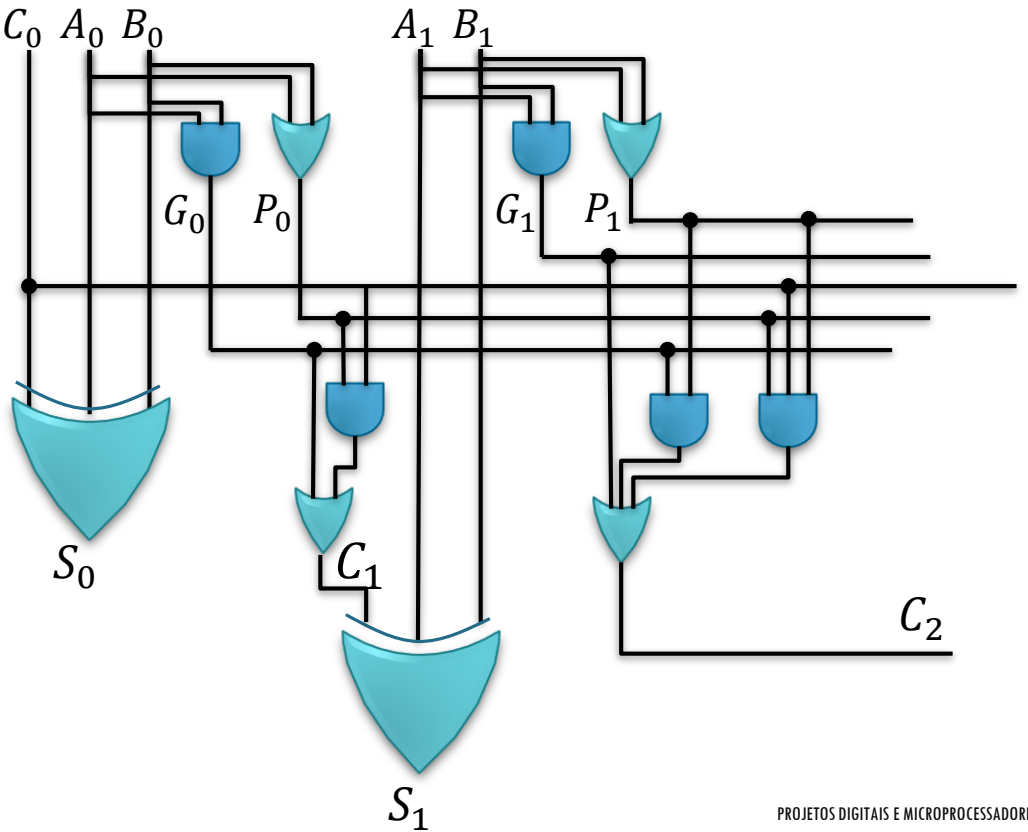
$$V_2 = G_1 + (P_1 \cdot G_0) + (P_1 \cdot P_0 \cdot V_0)$$

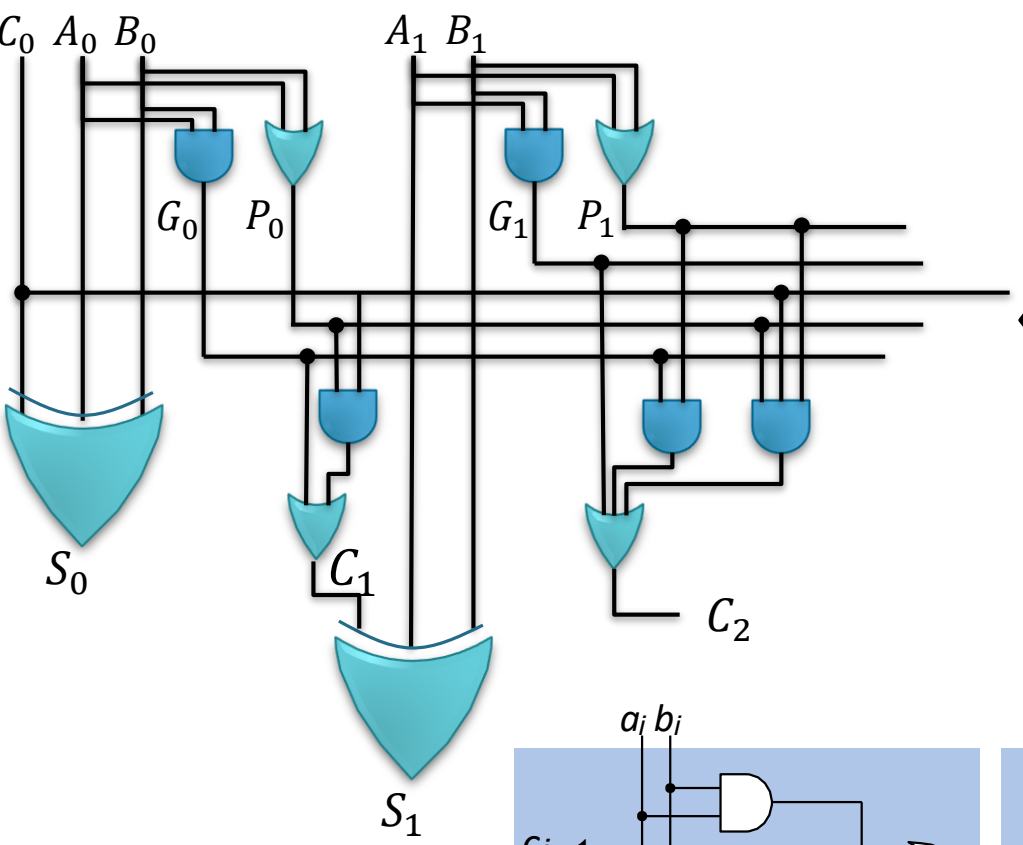
$$V_3 = G_2 + (P_2 \cdot G_1) + (P_2 \cdot P_1 \cdot G_0) + (P_2 \cdot P_1 \cdot P_0 \cdot V_0)$$

$$V_4 = G_3 + (P_3 \cdot G_2) + (P_3 \cdot P_2 \cdot G_1) + (P_3 \cdot P_2 \cdot P_1 \cdot G_0) + (P_3 \cdot P_2 \cdot P_1 \cdot P_0 \cdot V_0)$$

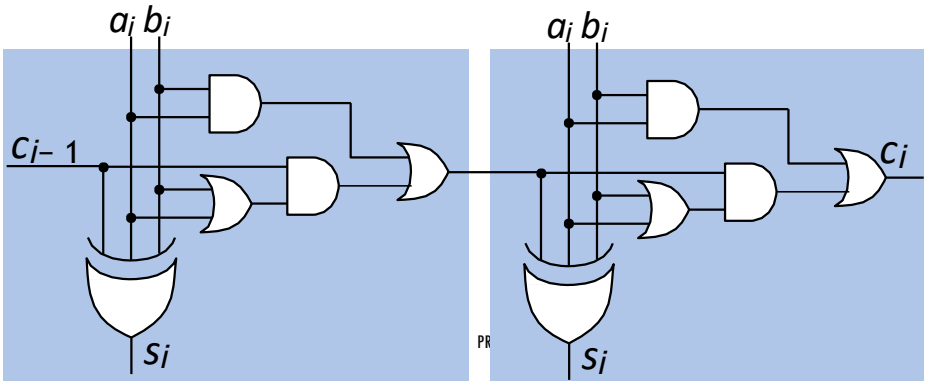
...

SOMADOR CARRY-LOOKAHEAD 2BITS





Qual o caminho crítico?



EXEMPLO COMPLETO