Insper

Elementos de Sistemas

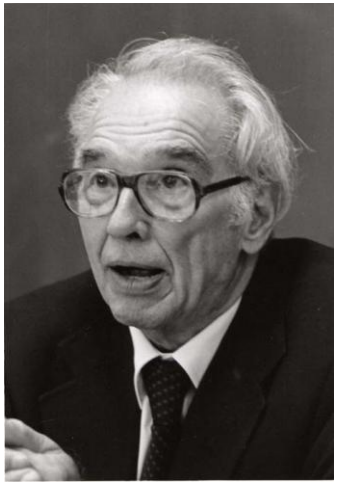
Unidade Lógica Aritmética

"Contar é a religião desta geração, sua esperança e salvação."
"Counting is the religion of this generation, its hope and salvation."
Gertrude Stein (1874–1946) **escritora americana**
apud Nisan, N. & Schocken, S. 2005. Elements of Computing Systems

Ciência da Computação - Luciano Soares <lpsoares@insper.edu.br>

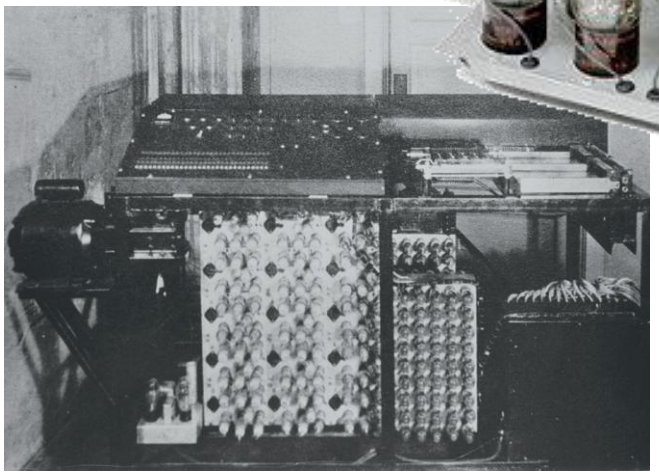
John Vincent Atanasoff & Clifford Berry

John Atanasoff e Clifford Berry implementaram na Iowa State University em 1939 o primeiro somador de 16-bits a válvula, e em 1941 fizeram uma máquina para resolver equações lineares, chamado de ABC (Atanasoff-Berry Computer), com 60 posições de memória de 50-bit feitas de capacitores, rodando a 60Hz. O julgamento Honeywell v. Sperry Rand declarou inválida a patente do ENIAC e reconheceu que Mauchly havia obtido ideias a partir do trabalho de Atanasoff.



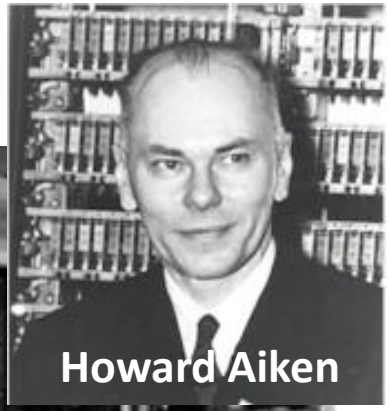
John Atanasoff

Clifford Berry



Harvard Mark 1

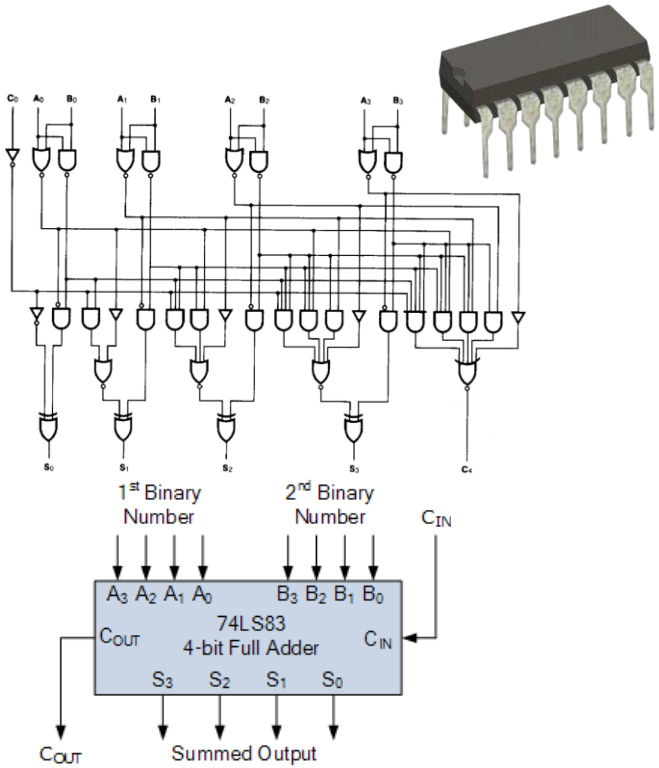
O Harvard Mark 1 (IBM automatic sequence controlled calculator), foi criado em 1943, sendo um sistema eletromecânico à base de relés. Howard Aiken fez o design do hardware do Mark 1, baseado em uma representação e armazenamento numérico decimal.

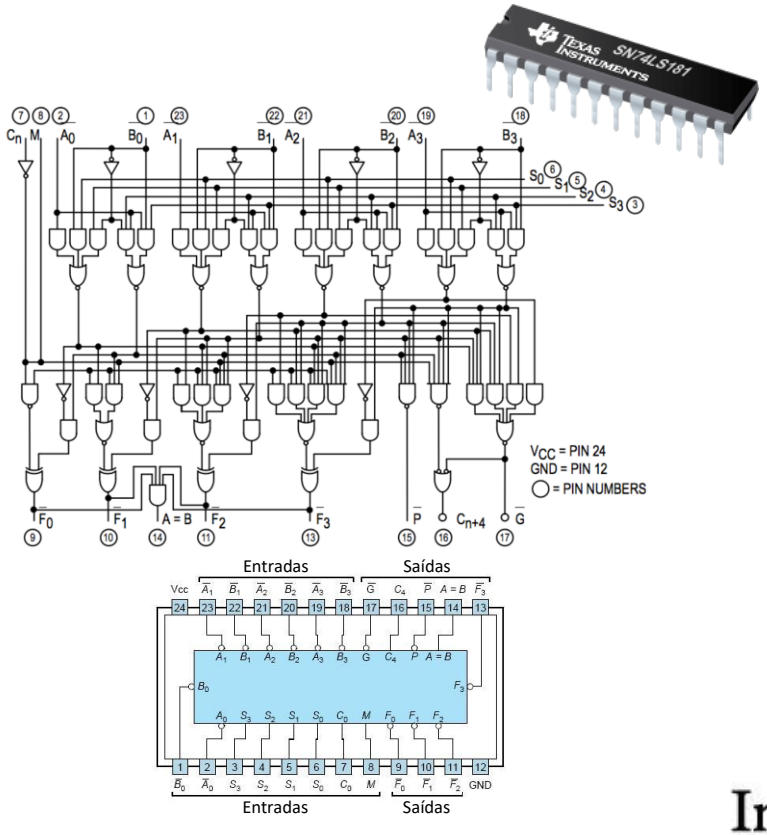


O complemento de dois é a representação predominante de inteiros com sinal nos computadores atuais.

7483 e 74181

O 7483 e o 74181 são circuitos integrados de lógica aritmética de 4 bits que foram usados em alguns dos primeiros computadores baseados em circuitos integrados.



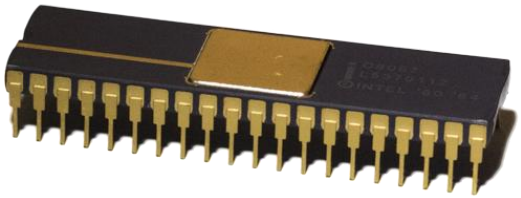


Insper

Unidade de Ponto Flutuante

Uma unidade de ponto flutuante, ou FPU (*Floating-Point Unit*), é um circuito especializado na execução de operações com números representados em ponto flutuante. Ela pode existir como um coprocessador separado ou estar integrada ao processador principal.

AMD AM9511



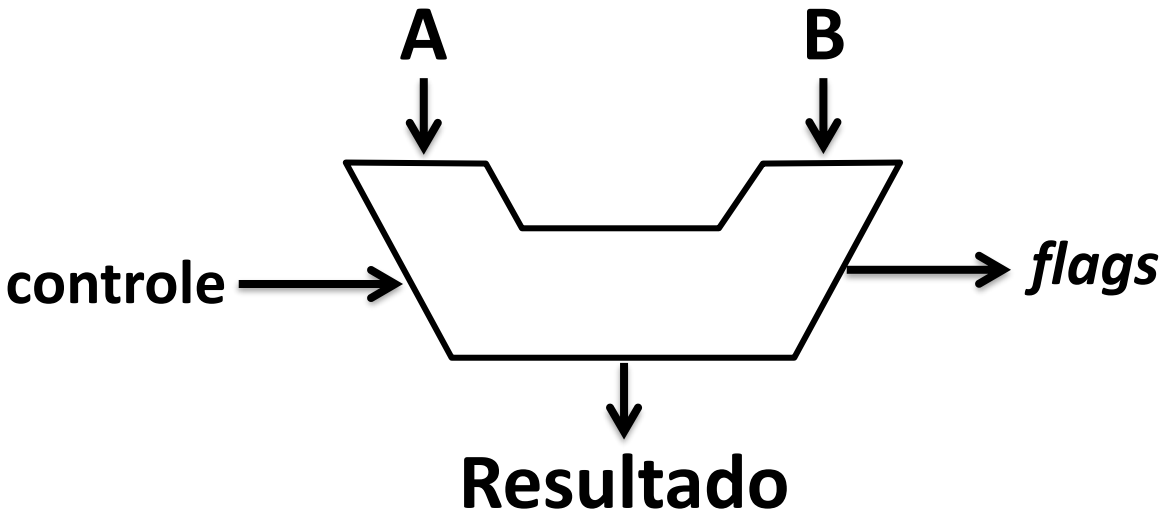
Intel 8087

Motorola 68881

Insper

Unidade Lógica Aritmética (ULA)

ULA ou Arithmetic Logic Unit (**ALU**), é um componente central do caminho de dados da CPU. A **ALU** consiste de um circuito digital capaz de fazer operações aritméticas e lógicas .



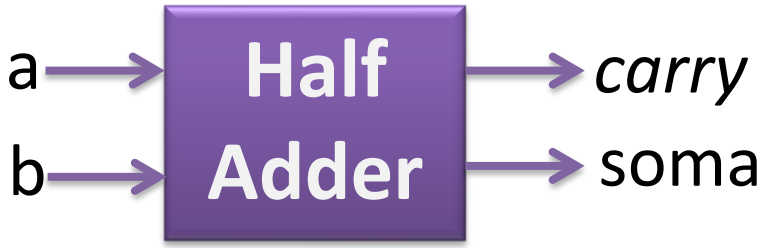
Meio-somador (Half Adder)

Adição binária é uma operação fundamental para quase todas as computações feitas em uma CPU:

Entradas			*Saídas*	
a	**+**	**b**	**soma**	**carry**
0	+	0	0	0
0	+	1	1	0
1	+	0	1	0
1	+	1	$^{1}0$	1

Operador aritmético de soma.
Não é o operador lógico OR.

S = a⊕b
C = a·b

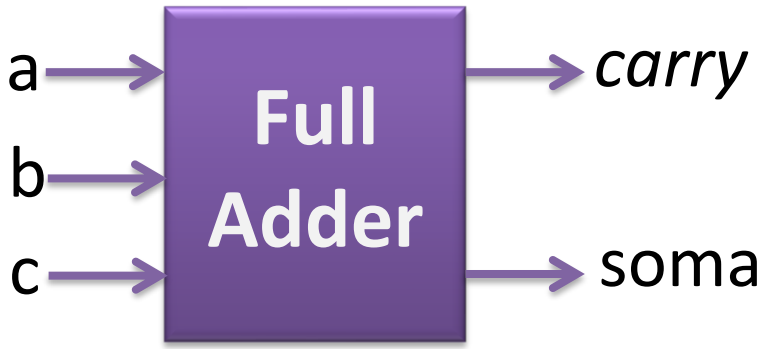


Insper

Circuito Aritmético – Adders (Full)

Faz a adição de três bits, retornando a soma e o *carry*.

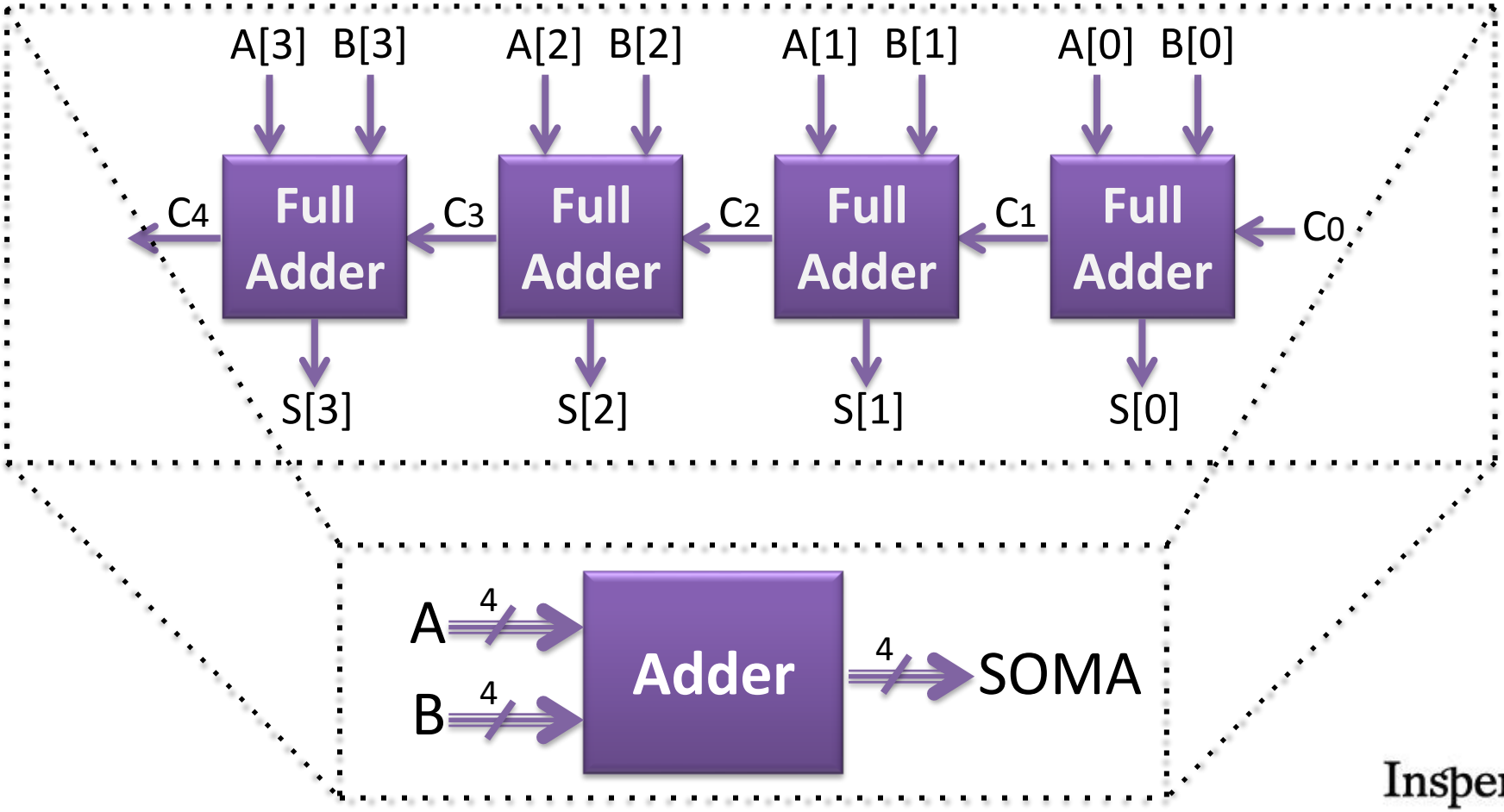
Entradas					*Saídas*	
a	**+**	**b**	**+**	**c**	**soma**	**carry**
0	+	0	+	0	0	0
0	+	0	+	1	1	0
0	+	1	+	0	1	0
0	+	1	+	1	0	1
1	+	0	+	0	1	0
1	+	0	+	1	0	1
1	+	1	+	0	0	1
1	+	1	+	1	1	1



Insper

Circuito Aritmético – Adder

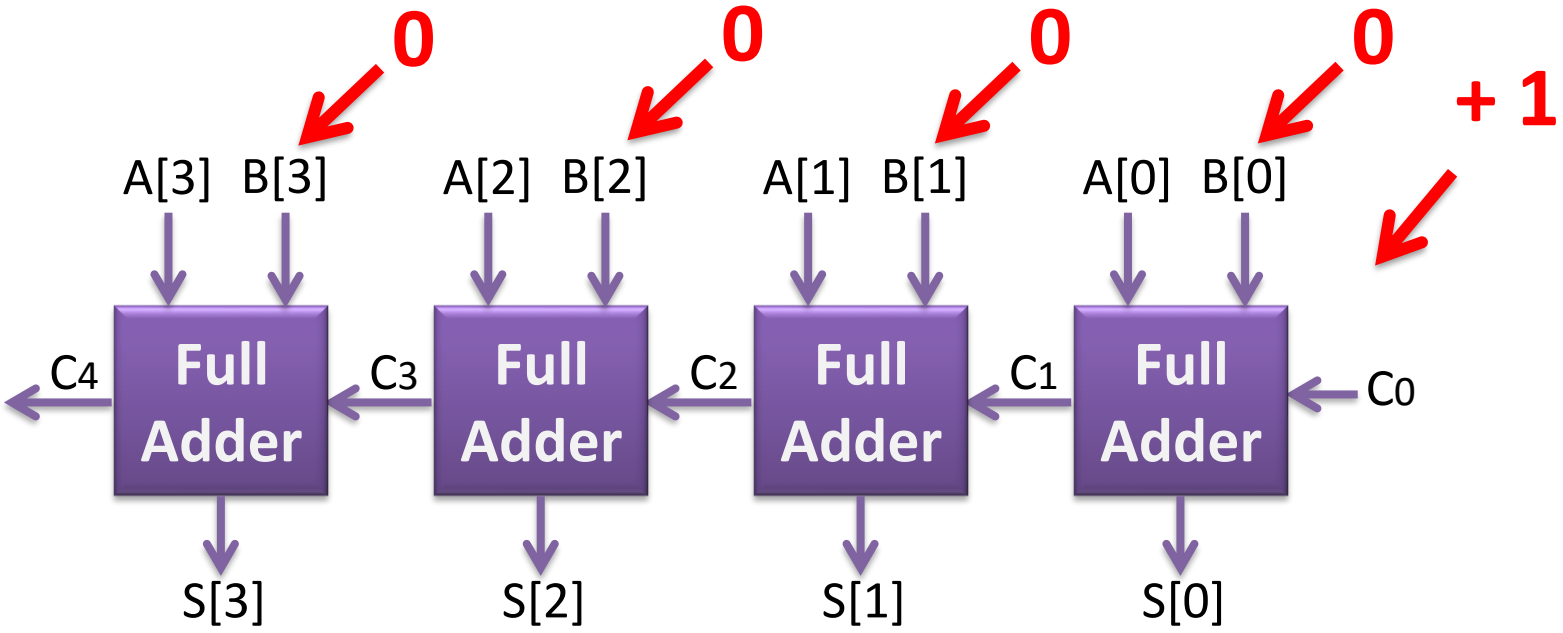
Um circuito para adicionar dois valores de n bits pode ser construído encadeando n somadores completos (Full Adders).



Insper

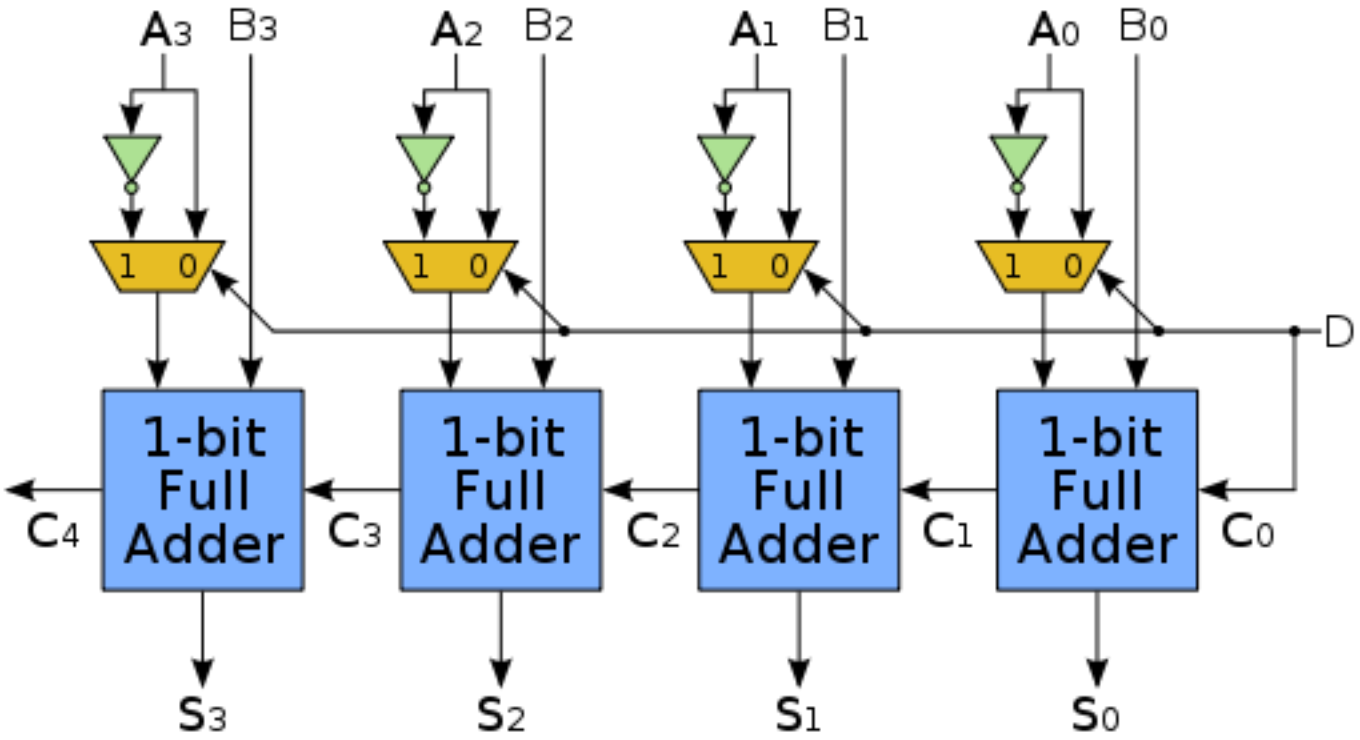
Circuito Aritmético – Incrementador

Um incrementador, ou seja, um circuito que soma 1 a um valor binário, pode ser usado por exemplo para um contador.



Insper

Somador/Subtrator de 4 bits

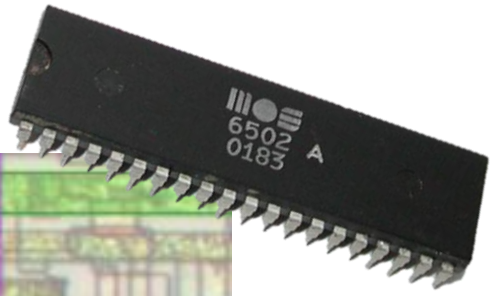


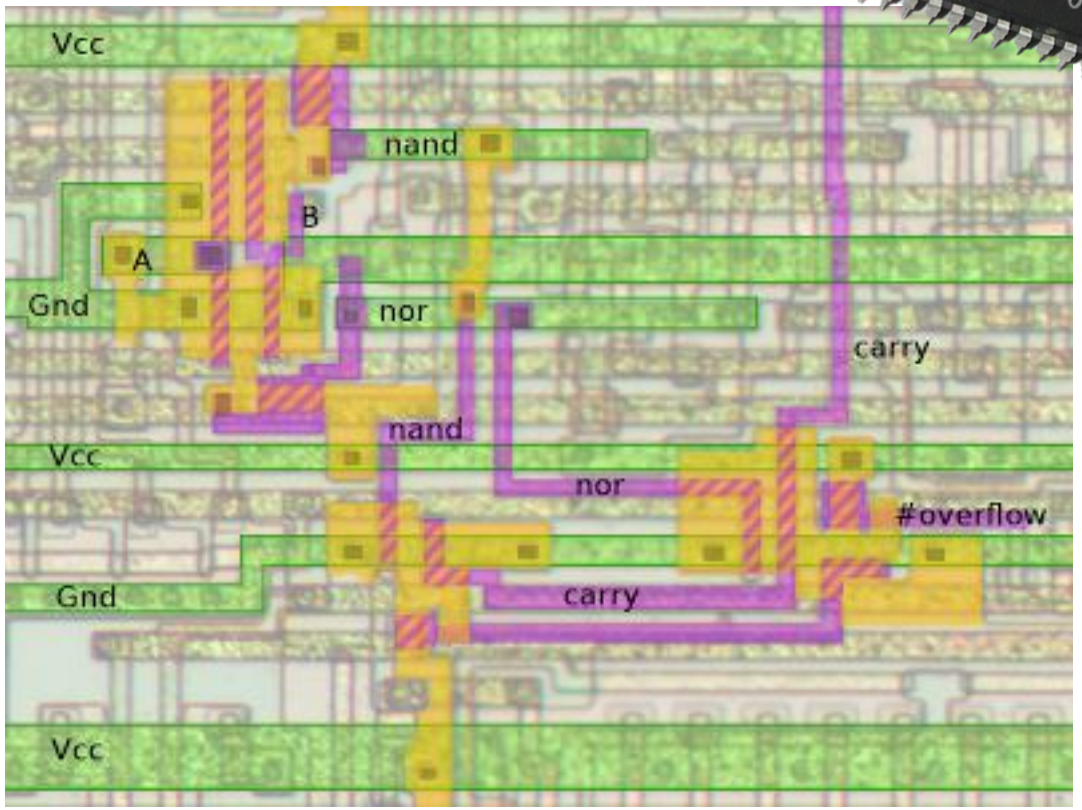
somador/subtrator que calcula **`B+A e B-A`**

Insper

Somador em CPU

Diversas otimizações podem ser realizadas, mas o princípio de circuito para soma é sempre muito parecido.





CPU 6502

Ken Shirriff's blog

Insper

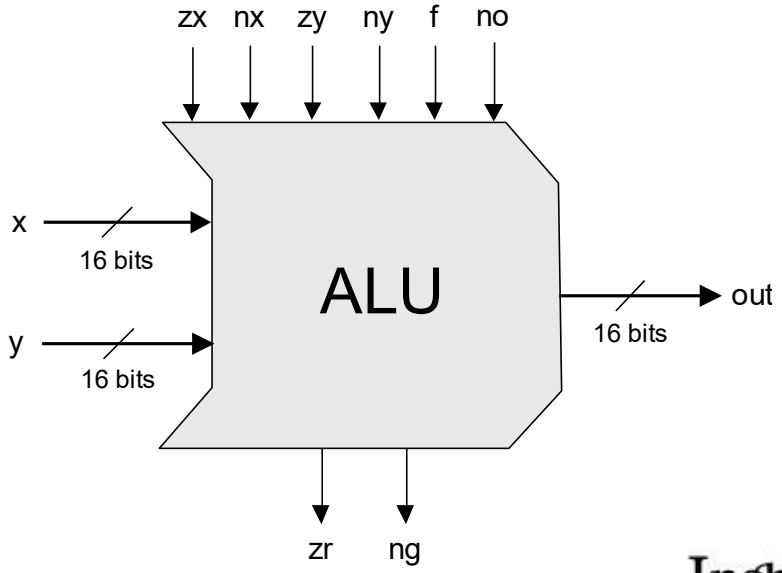
ALU do Z0 (baseada no Hack)

A arquitetura do Z0 (Hack) é baseada em 16 bits. Ou seja, ele tem de fazer operações com entradas de 16 bits e sair com o valor da operação também em 16 bits. Existem 6 bits de controle e mais dois bits de resultado (flags).

As operações desejadas devem retornar as seguintes saídas:

```
x+y, x-y, y-x,
0, 1, -1,
x, y, -x, -y,
!x, !y,
x+1, y+1, x-1, y-1,
x&y, x|y
```

Para isso os seguintes controles podem ser usados:



Insper

Negar e Inverter

A negação é representada com o símbolo – (menos aritmético)

Por exemplo $A = 5_{10} = 00000101_2$

$-A = -5_{10} = 11111011_2$

A inversão é representada com o símbolo ! (operador lógico not)

Por exemplo $A = 5_{10} = 00000101_2$

$!A = 11111010_2 = -6_{10}$

Também conhecido como **bitwise not**

Tabela de controle da ALU do Z0 (Hack)

se zx: x=0	se nx: x=!x	se zy: y=0	se ny: y=!y	se f: x+y senão: x&y	se no: out=!out	out
1	0	1	0	1	0	0
1	1	1	1	1	1	1
1	1	1	0	1	0	-1
0	0	1	1	0	0	x
1	1	0	0	0	0	y
0	0	1	1	0	1	!x
1	1	0	0	0	1	!y
0	0	1	1	1	1	-x
1	1	0	0	1	1	-y
0	1	1	1	1	1	x+1
1	1	0	1	1	1	y+1
0	0	1	1	1	0	x-1
1	1	0	0	1	0	y-1
0	0	0	0	1	0	x+y
0	0	0	1	1	1	y-x
0	0	0	0	0	0	x&y
0	1	0	1	0	1	x|y

Exemplos de comportamento da ALU

Exemplo 1

controle	zx	nx	zy	ny	f ? + : &	no
	0	0	0	0	1	0

} x+y

x	x[7]	x[6]	x[5]	x[4]	x[3]	x[2]	x[1]	x[0]
	0	0	0	0	1	0	1	0

} 10

y	y[7]	y[6]	y[5]	y[4]	y[3]	y[2]	y[1]	y[0]
	1	1	1	1	1	1	1	0

} -2

saída	out[7]	out[6]	out[5]	out[4]	out[3]	out[2]	out[1]	out[0]	zr	ng
	0	0	0	0	1	0	0	0	0	0

8

Obs: Exemplo com 8 bits

Exemplos de comportamento da ALU

Exemplo 2

controle	zx	nx	zy	ny	f ? + : &	no
	0	0	1	1	1	0

} x-1

x	x[7]	x[6]	x[5]	x[4]	x[3]	x[2]	x[1]	x[0]
	0	0	0	0	1	0	1	0

} 10

y	y[7]	y[6]	y[5]	y[4]	y[3]	y[2]	y[1]	y[0]
	1	1	1	1	1	1	1	0

} -2

saída	out[7]	out[6]	out[5]	out[4]	out[3]	out[2]	out[1]	out[0]	zr	ng
	0	0	0	0	1	0	0	1	0	0

9

Obs: Exemplo com 8 bits

Exemplos de comportamento da ALU

Exemplo 3

controle	zx	nx	zy	ny	f ? + : &	no
	0	0	1	1	1	1

} -x

x	x[7]	x[6]	x[5]	x[4]	x[3]	x[2]	x[1]	x[0]
	0	0	0	0	1	0	1	0

} 10

y	y[7]	y[6]	y[5]	y[4]	y[3]	y[2]	y[1]	y[0]
	1	1	1	1	1	1	1	0

} -2

saída	out[7]	out[6]	out[5]	out[4]	out[3]	out[2]	out[1]	out[0]	zr	ng
	1	1	1	1	0	1	1	0	0	1

-10

Obs: Exemplo com 8 bits

Exemplos de comportamento da ALU

Exemplo 4

controle	zx	nx	zy	ny	f ? + : &	no
	1	0	1	0	1	0

} 0

x	x[7]	x[6]	x[5]	x[4]	x[3]	x[2]	x[1]	x[0]
	0	0	0	0	1	0	1	0

} 10

y	y[7]	y[6]	y[5]	y[4]	y[3]	y[2]	y[1]	y[0]
	1	1	1	1	1	1	1	0

} -2

saída	out[7]	out[6]	out[5]	out[4]	out[3]	out[2]	out[1]	out[0]	zr	ng
	0	0	0	0	0	0	0	0	1	0

0

Obs: Exemplo com 8 bits

Insper

www.insper.edu.br