Spina

PCS 3115 Sistemas Digitais I

Módulo 03 – Aritmética Binária

Prof. Dr. Marcos A. Simplício Jr.
Prof. Dr. Edison Spina

versão: 5 (Fev/2018)

Spina

Conteúdo

Aritmética Binária

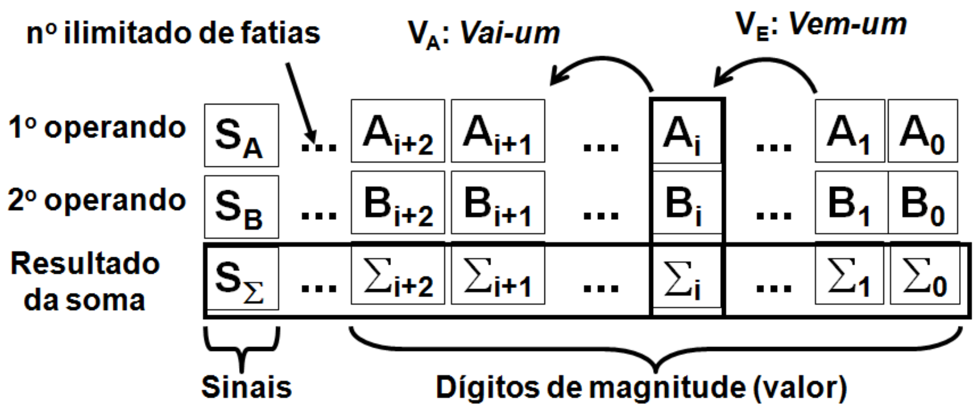
- Soma e Subtração com Números Decimais e Binários
  - Aritmética Modular
- Representação de números negativos
  - Sinal e magnitude
  - Complemento de base ➔ complemento de 2
  - Complemento de base diminuída ➔ complemento de 1
- Soma e Subtração com complemento de 1 e 2
  - Overflow

Spina

1. Soma e Subtração com Números Decimais e Binários

Operações com números decimais (*no papel*):

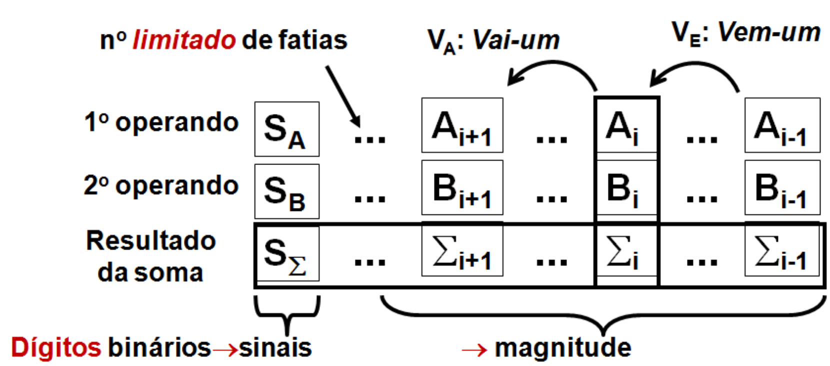
O resultado da operação (ex.: **soma**) pode ser decidido calculando-se os resultados parciais em ***fatias*** individuais:



Spina

1. Soma e Subtração com Números Decimais e Binários

Operações com números binários (*no computador*): O resultado da operação (ex.: **soma**) ***também*** pode ser decidido calculando-se os resultados parciais em ***fatias*** individuais, ***porém*** ....:

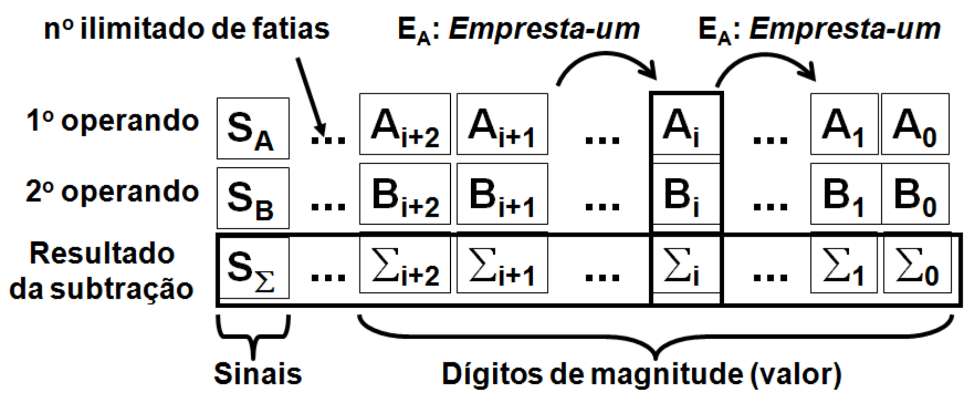


Spina

1. Soma e Subtração com Números Decimais e Binários

Operações com números decimais (*no papel*):

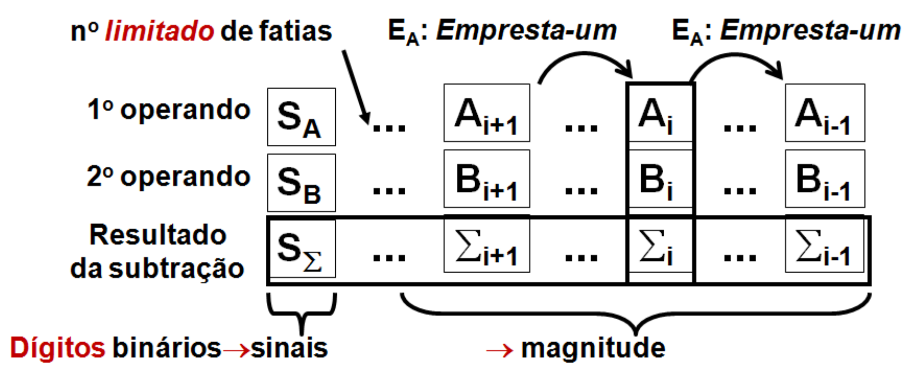
O resultado da operação (ex.: **subtração**) pode ser decidido calculando-se os resultados parciais em ***fatias*** individuais:



Spina

1. Soma e Subtração com Números Decimais e Binários

Operações com números binários (*no computador*): O resultado da operação (ex.: **subtração**) ***também*** pode ser decidido calculando-se os resultados parciais em ***fatias*** individuais, ***porém*** ....:



1. Soma e Subtração com Números Decimais e Binários

Tabela 1.1.: Resultado de operações de soma (adição) e subtração (diferença) de uma fatia de números em binário:

Vem-um, (*Carry-in*; c_{IN}); ***Vai-um*** (*Carry-out*; c_{OUT});

Empresta-um (*Borrow-in*; b_{IN}) – A ser subtraído desta fatia, que é mais significativa, e somada na anterior, que precisou pegar emprestado;

Empresta-um (*Borrow-out*; b_{OUT}) – A ser subtraído da fatia seguinte, que é mais significativa que esta, e somado nesta, que precisou pegar emprestado.

entradas			saídas: +		saídas: –	
c_{IN} [b_{IN}]	x	y	Σ (soma)	c_{OUT}	d (diferença)	b_{OUT}
0	0	0	0	0	0	0
0	0	1	1	0	1	1
0	1	0	1	0	1	0
0	1	1	0	1	0	0
1	0	0	1	0	1	1
1	0	1	0	1	0	1
1	1	0	0	1	0	0
1	1	1	1	1	1	1

Spina

1. Soma e Subtração com Números Decimais e Binários

Desafio:

Desenvolver técnicas de **realização** de **operações**: elas fundamentalmente estarão **baseadas** na escolha do **tipo** de **representação** que se utilizará para os números binários, a fim de **resolver** os empecilhos e **problemas** das **operações de soma e subtração em binário**.

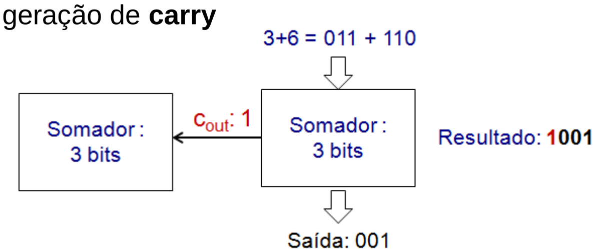
Spina

Spina

1. Soma e Subtração com Números Decimais e Binários

Problema 1: limitação no número de fatias

- Nem todos os números podem ser representados: existem **faixas de representações** possíveis
  - Ex.: 000 a 999 com 3 dígitos decimais;
  - Ex.: 00000000 a 11111111 com 8 dígitos binários (bits)
- Podem ocorrer **overflows**: resultado de operação aritmética não cabe na representação
  - Ex.: 100 * 10 = 1000 (não representável com 3 dígitos decimais)
  - Problema deve ser tratado: geração de **carry**

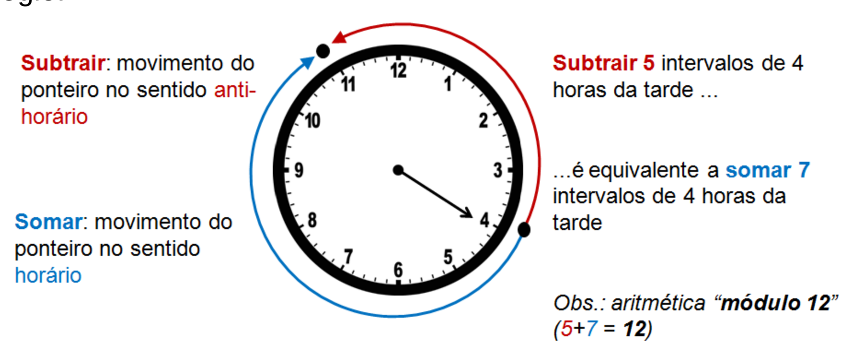


Spina

1. Soma e Subtração com Números Decimais e Binários

Aritmética modular:

- Leva a equivalências entre algumas operações de adição e subtração: subtrair x é equivalente a somar (n – x)
- Formalmente: (a – x) mod n = (a + n – x) mod n, pois (n mod n) = 0
- Ex: relógio:



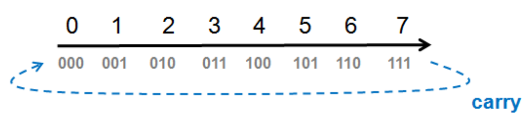
Spina

1. Soma e Subtração com Números Decimais e Binários

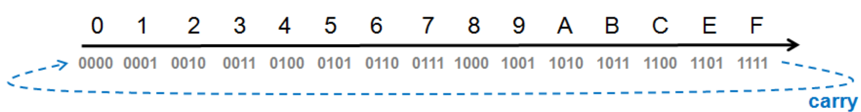
Aritmética modular:

Módulos para soma binária operam com aritmética modular: somadores de n bits realizam somas módulo 2^n

Somador de 3 bits: somas módulo 8



Somador de 4 bits: somas módulo 16



Spina

1. Soma e Subtração com Números Decimais e Binários

Problema 1: limitação no número de fatias

- Nem todos os números podem ser representados: existem **faixas de representações** possíveis
- Podem ocorrer **overflows**: resultado de operação aritmética não cabe na representação (gera-se carry)

Problema 2: **números negativos**

- Deve-se adotar alguma forma eficiente de **representá-los** e de fazer operações aritméticas com eles
- **Exercício:** propor uma solução para representação de números negativos em binário

Spina

2. Representação de números negativos

Representação em **sinal e magnitude**:

1 bit (mais significativo) para o sinal ➔

0: positivo; 1: negativo

Bits restantes para a magnitude

Ex.: **01010101** = **+ 85** ; **11010101** = **– 85**

Faixa de representação com **n bits**: $[-(2^{n-1} - 1) , +(2^{n-1} - 1)]$

Spina

2. Representação de números negativos

Simples para humanos entenderem, mas...

Desperdício: duas representações para o número zero

00000000 = **+ 0** ; **10000000** = **– 0**

Operações de soma e subtração (= soma com inversão do sinal do segundo operando) pouco eficientes em hardware:

- Ex. (decimal):

+ 123	– 123	+ 123	+ 123
+ 333	– 333	– 333	– 111
--------	-------	-------	-------
+456	– 456	– 210	+ 012

Spina

2. Representação de números negativos

Algoritmo da soma:

Comparar sinal dos operandos:

if(iguais) {mantém sinal}

else {**comparar** números e usar sinal do de maior magnitude}

Circuitos de lógica extras...
<u>É possível fazer melhor?</u>

Spina

2. Representação de números negativos

- Representação em **sinal e magnitude:**
  - Quando observada em binário, não é muito "natural": sinal inverte a sequência usual encontrada em números positivos

-7	-6	-5	-4	-3	-2	-1	0	1	2	3	4	5	6	7
1111	1110	1101	1100	1011	1010	1001	0000 1000	0001	0010	0011	0100	0101	0110	0111

- Algo um pouco mais "natural" (ainda com desperdício):

-7	-6	-5	-4	-3	-2	-1	0	1	2	3	4	5	6	7
1000	1001	1010	1011	1100	1101	1110	0000 1111	0001	0010	0011	0100	0101	0110	0111

"Complemento de 1": bits de x e –x são invertidos

- Algo mais "natural" e sem desperdício:

-8	-7	-6	-5	-4	-3	-2	-1	0	1	2	3	4	5	6	7
1000	1001	1010	1011	1100	1101	1110	1111	0000	0001	0010	0011	0100	0101	0110	0111

"Complemento de 2": complemento de 1, mais 1

Spina

2. Representação de números negativos

Representação em **complemento de base:**

Aplica-se a ideia de aritmética modular: a representação de número negativo é dada pelo seu **complemento no espaço de valores possíveis**, ou

Número D representado com n dígitos (notação posicional) :

$$\mathbf{D = d_{n-1}d_{n-2}...d_1d_0}$$

Complemento na base r (do inglês, *radix*) do número D: obtido como $r^n - D$

Nota: r^n tem n+1 dígitos ➔ se D = 0, então exclui-se o dígito extra, de modo que 0 é representado simplesmente como n zeros

Decimal: complemento de 10; binário: complemento de 2

Ex: Base r = 10 (decimal); n = 3 (3 dígitos); D = 345:

- $r^n - D = 10^3 - 345 =$; $1000 - 345 = 655$;
- Logo, o complemento na base 10 de 345 é 655!

18

Spina

2. Representação de números negativos

- Representação em **complemento de base:**

Dígito	Complemento Binário	Octal	Decimal	Hexa
0	1	7	9	F
1	0	6	8	E
2	-	5	7	D
3	-	4	6	C
4	-	3	5	B
5	-	2	4	A
6	-	1	3	9
7	-	0	2	8
8	-	-	1	7
9	-	-	0	6
A	-	-	-	5
B	-	-	-	4
C	-	-	-	3
D	-	-	-	2
E	-	-	-	1
F	-	-	-	0

- Exemplos
  - $Comp(1849_{10}) = 8150 + 1$
    $= 8151_{10}$
  - $Comp(0F36_{16}) = F0C9 + 1$
    $= F0CA_{16}$
  - $Comp(1010_2) = 0101 + 1$
    $= 0110_2$

Complemento de 2: inverter bits e somar 1 ao resultado

Spina

2.1. Números binários: Complemento de 2

- **Números positivos**: idem a notação sinal-módulo;
- Para **inverter** o **sinal**:

  1) Invertem-se todos os bits (equivale a complementar de 1 cada um dos bits) e

  2) Soma-se 1 ao resultado.

Exemplo: $+27_{10}$ = **0 0 0 1 1 0 1 1**

↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ bits são complementados

1 1 1 0 0 1 0 0

+ 1 soma-se 1 ao resultado

-27_{10} = **1 1 1 0 0 1 0 1**

Não é fácil de ler mas
é fácil de tratar no computador

22

Spina

2.1. Números binários: Complemento de 2

Extensão de sinal (*sign extension*):

Ao aumentar o número de bits de D, deve-se tomar cuidado para manter o sinal correto!

Regra prática

Se D é **positivo**: adicionar **0s** à esquerda

Se D é **negativo**: adicionar **1s** à esquerda

Truncagem

Diminuir o número de bits de D, cortando bits "sobrando" à esquerda: 0s se D é positivo; 1s se D é negativo

Resultado só é válido se o sinal do número se mantém

$+1_{10} = 1_2 = 00000001_2$ $\quad -8_{10} = 1000_2 = 11111000_2$

$-7_{10} = 1001_2 = 11111001_2$ $\quad -9_{10} = 10111_2 = 11110111_2$

23

Spina

2. Representação de números negativos

Complemento de Base Diminuída (Base Menos Um):

- Equivalente ao complemento de base sem o "mais 1"

Mais formalmente:

- Número D representado com n dígitos (notação posicional) :

$$D = d_{n-1}d_{n-2}\ldots d_1 d_0$$

- Complemento na base r de D: obtido como $(r^n - 1) - D$

Regra prática

- Números **positivos**: idem a notação sinal-módulo;
- Para **inverter o sinal**: Complementar todos os dígitos d_i com relação a $(r - 1)$

Mais fácil de ler mas ...
Difícil de tratar no computador
Dois zeros, etc

Spina

2. Representação de números negativos

Representação em **complemento de base __diminuída__:**

	Complemento			
Dígito	**Binário**	**Octal**	**Decimal**	**Hexa**
0	1	7	9	F
1	0	6	8	E
2	-	5	7	D
3	-	4	6	C
4	-	3	5	B
5	-	2	4	A
6	-	1	3	9
7	-	0	2	8
8	-	-	1	7
9	-	-	0	6
A	-	-	-	5
B	-	-	-	4
C	-	-	-	3
D	-	-	-	2
E	-	-	-	1
F	-	-	-	0

- Exemplos
  - $Comp(1849_{10}) = 8150_{10}$
  - $Comp(0F36_{16}) = F0C9$
  - $Comp(1010_2) = 0101$

Complemento de 1:
só inverter bits
Mas aritmética mais difícil de tratar

Spina

2. Representação de números negativos

- **Complemento de base diminuída:**
  - Faixa de representação na base **r**: $\left[-\left(\left\lfloor \frac{r^n}{2} \right\rfloor_{CHÃO} - 1\right), +\left(\left\lceil \frac{r^n}{2} \right\rceil^{TETO} - 1\right)\right]$
    - Há tantos números negativos quanto números positivos
    - Duas representações para o número zero:
  - Faixa de representação com **n bits**: $[-(2^{n-1} - 1), +(2^{n-1} - 1)]$
    - Também conhecido como **complemento de 1**

-7	-6	-5	-4	-3	-2	-1	0	1	2	3	4	5	6	7
1000	1001	1010	1011	1100	1101	1110	0000	0001	0010	0011	0100	0101	0110	0111
							1111							

"Complemento de 1": bits de x e –x são invertidos

Spina

3. Aritmética: Adição e Subtração

<u>Complemento de 2</u>: contagem "natural"

Adição *n* + *m*: equivale a contar *m* a partir de *n* no sentido da esquerda para a **direita**

Ex.: $(-5 + 6)_{10} = 1011_2 + 0110_2 =$ "1011_2 + 6 p/ direita" $= 0001_2 = 1_{10}$

Subtração *n* – *m*: equivale a contar *m* a partir de *n* no sentido da direita para a **esquerda**

Ex.: $(4 - 2)_{10} = 0100_2 - 0010_2 =$ "0100_2 + 2 p/ esquerda" $= 0010_2 = 2_{10}$

Overflow: operação ultrapassa fronteira entre +7 e –8,

Ex.: $(6 + 4)_{10} = 0110_2 + 0100_2 =$ "0110_2 + 4 p/ direita" $= 1010_2 =$ **–6**

+

–

-8	-7	-6	-5	-4	-3	-2	-1	0	1	2	3	4	5	6	7
1000	1001	1010	1011	1100	1101	1110	1111	0000	0001	0010	0011	0100	0101	0110	0111

Overflow se cruzar fronteira em qualquer direção

27

Spina

3. Aritmética: Adição e Subtração

Complemento de 2: **adição**

Pode-se usar as regras usuais da soma, ignorando o "vai-um" no bit mais significativo, se houver

	+3	0011		-2	1110
+	+4	+ 0100	+	-6	+ 1010
	+7	0111		-8	^{1} 1000
	+6	0110		+4	0100
+	-3	+ 1101	+	-7	+1001
	+3	^{1} 0011		-3	1101

Ignorar "vai-um"

Spina

3. Aritmética: Adição e Subtração

Complemento de 2: **overflow na adição**

Nunca ocorre se sinais dos operandos são diferentes

Detecção de ocorrência: duas regras **equivalentes**

- Soma de dois números com o mesmo sinal produz resultado de sinal diferente
- "Vem-um" (c_{IN}) que chega na posição de sinal é diferente do "*vai-um"* (c_{OUT}) que sai da posição do sinal.

entradas			saídas: +	
c_{IN}	x	y	Σ (soma)	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

regra 1

regra 2

Spina

3. Aritmética: Adição e Subtração

Complemento de 2: **subtração**

Pode ser feita como se fosse uma adição, e depois verificam-se os sinais para detectar overflow

ou: nega-se subtraendo e faz-se soma normal, verificando overflow usando as regras da adição

Operação *m* – *n* equivale a: 1) Complementar bit-a-bit *n*

2) Somar *m* e *n* com c_{IN} = 1

Exemplos:

		c_{IN} = 1			c_{IN} = 1
+4	0100	0100	+3	0011	0011
– +3	– 0011	+ 1100	– +4	– 0100	+ 1011
+1		10001	–1		1111
		c_{IN} = 1			c_{IN} = 1
+4	0100	0100	+3	0011	0011
– –3	– 1101	+ 0010	– –4	– 1100	+ 0011
+7		0111	+7		0111

30

Spina

3. Aritmética: Adição e Subtração

- Exercícios: Soma e subtração em **complemento de 2**

–3	+5
+ –6	+ +6
–9	+11
+4	–3
– +4	– –4
0	+1
–4	–2
– +5	+ –6
–9	–8

3. Aritmética: Adição e Subtração

- Exercícios: Soma e subtração em **complemento de 2**

–3	1101		
+ –6	+ 1010	overflow	
–9	**1** 0111	+7	

+5	0101		
+ +6	+ 0110	overflow	
+11	1011	–5	

		$\mathbf{c_{IN} = 1}$
+4	0100	0100
– +4	– 0100	+ 1011
0		**1** 0000

		$\mathbf{c_{IN} = 1}$
–3	1101	1101
– –4	– 1100	+ 0011
+1		**1** 0001

		$\mathbf{c_{IN} = 1}$	
–4	1100	1100	
– +5	– 0101	+ 1010	overflow
–9		**1** 0111	+7

–2	1110
+ –6	+ 1010
–8	**1** 1000

Spina

32

3. Aritmética: Adição e Subtração

Complemento de 1: contagem “natural”, exceto pelo zero extra (1111)

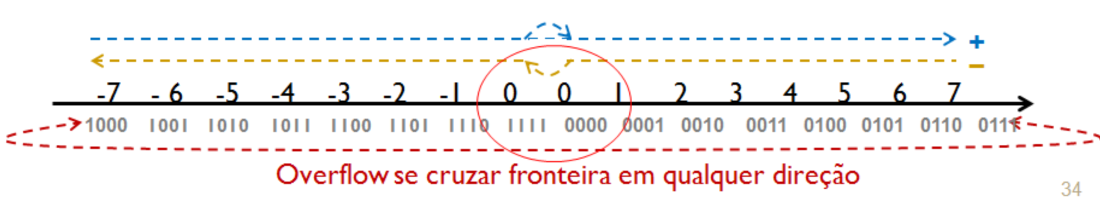
Adição *n* + *m*: contar *m* a partir de *n* para a **direita**, somando 1 se for feita a transição de 1111 para 0000

Ex.: $(–5 + 6)_{10} = 1010_2 + 0110_2 =$ “1010_2 + 6+1 p/ direita” $= 0001_2 = 1_{10}$

Subtração *n* – *m*: contar *m* a partir de *n* para a **esquerda,** subtraindo 1 se for feita a transição de 0000 para 1111

Ex.: $(4 – 6)_{10} = 0100_2 – 0110_2 =$ “0100_2 + 6+1 p/ esquerda” $= 1101_2 = –2_{10}$

Overflow: mesmas regras do complemento de 2



Spina

34

Spina

3. Aritmética: Adição e Subtração

Complemento de 1: adição

Pode-se usar as regras usuais da soma, **somando-se o “vai-um”** no bit mais significativo **ao resultado**

Exemplos:

+3	0011	-2	1101
+ +4	+ 0100	+ -5	+ 1010
+7	0111	-7	1 0111 +
			1000
+6	0110	+4	0100
+ -3	+ 1100	+ -7	+1000
+3	1 0010 +	-3	1100
	0011		

Reciclar
“vai-um”

Spina

3. Aritmética: Adição e Subtração

- **Complemento de 1: subtração**
  - Complementar parcela sendo subtraída e realizar a soma
- **Exemplos/Exercícios:**

+4	+3
– +3	– +4
+1	–1
–2	+6
– +5	– –3
+7	+9

Spina

3. Aritmética: Adição e Subtração

- **Complemento de 1**: **subtração**
  - Complementar parcela sendo subtraída e realizar a soma
- **Exemplos/Exercícios:**

+4	0100	0100	+3	0011	0011
– +3	– 0011	+ 1100	– +4	– 0100	+ 1011
+1		**1** 0000	–1		1110
		0001			

–2	1101	1101	+6	0110	0110	
– +5	– 0101	+ 1010	– –3	– 1100	+ 0011	overflow
- 7		**1** 0111	+9		1001	-6
		0001				
		1000				

Spina

3. Aritmética: Adição e Subtração

- Exercício 10.8.1. Faça as operações com 6 bits (incluindo o bit de sinal) em Complemento de 2. Indique a ocorrência de Transbordo:

a) + 19 + (–12) b) – 19 + (–12)

c) + 19 + (+12) d) – 19 + (+12)

e) + 21 + (–11) f) – 21 + (–11)

g) + 21 + (+11) h) – 21 + (+11)

- Exercício 10.8.2. Idem anterior para a notação Complemento de 10 (base 10) usando 3 dígitos.

37

3. Aritmética: Adição e Subtração

- Exercício 11.2.1. Faça as operações com 6 bits (inclui o bit de sinal) em Complemento de 1. Indique a ocorrência de Transbordo:

a) + 19 + (–12)
b) – 19 + (–12)
c) + 19 + (+12)
d) – 19 + (+12)
e) + 21 + (–11)
f) – 21 + (–11)
g) + 21 + (+11)
h) – 21 + (+11)

- Exercício 11.2.2. Idem anterior para a notação Complemento de 9 (base 10) usando 3 dígitos.

Spina

38

Lição de Casa

Leitura Obrigatória:

Capítulo 2 do Livro Texto.

Exercícios Obrigatórios:

Capítulo 2 do Livro Texto;

Lista de Exercícios do Módulo 3.

Spina

Livro Texto

- Wakerly, J.F.; *Digital Design – Principles & Practices;* Fourth Edition, ISBN: 0-13-186389-4, Pearson & Prentice-Hall, Upper Saddle, River, New Jersey, 07458, 2006.

Spina