Facens

Prof. André Breda Carneiro
Prof. Rafael R. da Paz

Organização Básica de Computadores

FACENS
Sorocaba/2020

Sistemas de numeração

Revisão dos Sistemas Numéricos

Decimal:

baseado em 10 dígitos (0,1,2,3,4,5,6,7,8,9)

Exemplo:

$$5264_{(10)} = (5x1000) + (2x100) + (6x10) + 4x1$$

$$5264_{(10)} = (5x\boxed{10}^3) + (2x10^2) + (6x10^1) + (4x10^0)$$

Base ou raíz

Decimal:

Valores fracionários:

$$75{,}32 = (7x10^{1}) + (5x10^{0}) + (3x10^{-1}) + (2x10^{-2})$$

De uma forma geral:

$$X = \sum_{i} x_i 10^{i}$$

onde: $x_i \in (0,1,2,3,4,5,6,7,8,9)$ e i corresponde a posição do dígito.

Hexadecimal:

baseado em 16 dígitos (0,1,2,3,4,5,6,7,8,9,A,B,C,D,E,F)

Onde "A" tem valor 10... e "F" valor 15

Exemplo:

$7C1_{(16)} = (7x16^2) + (12x16^1) + (1x16^0)$

$7C1_{(16)} = (7x256) + (12x16) + (1x1)$

$7C1_{(16)} = 1985$

Ou seja conversão de hexadecimal em decimal

Binário:

baseado em 2 dígitos (0,1)

Exemplo:

$1101_{(2)} = (1x2^3) + (1x2^2) + (0x2^1) + (1x2^0)$

$1101_{(2)} = (8) + (4) + (0) + (1)$

$1101_{(2)} = 13$

Ou seja conversão de binário em decimal

Conversão de decimal para binário

```
25 |_2___
 1   12 |_2___
      0    6 |_2___
           0    3 |_2___
                1    1
```

$$25 = 1 x\ 2^4 + 1\ x\ 2^3 + 0\ x\ 2^2 + 0\ x\ 2^1 + 1\ x\ 2^0$$

1 1 0 0 1

Conversão de decimal para Hexadecimal

$$
\begin{array}{lll|l}
1985 & & & 16 \\
\hline
1 & 124 & & 16 \\
\cline{4-4}
 & 12 & & 7
\end{array}
$$

7C 1 , pois 12 em hexa é "C"

$$7C1_{(16)} = (7x16^2) + (12x16^1) + (1x16^0)$$

$$7C1_{(16)} = 1985_{(10)}$$

Conversão de decimal para Hexadecimal

1985	16		
1	124	16	
	12	7	

7C 1 , pois 12 em hexa é "C"

$$7C1_{(16)} = (7x16^{2}) + (12x16^{1}) + (1x16^{0})$$

$$7C1_{(16)} = 1985_{(10)}$$

Representação de Inteiros

Na representação de binária podem ser representados com 0 e 1, o sinal de negativo e ponto:

-101.01010

Ao se trabalhar com números binários no computador não é possível se usar o sinal de negativo e o ponto. Com números positivos e inteiros a representação é direta:

00110001 = 49

00010101 = 21

usando-se 8 bits

Representação de sinal

O bit de maior significância (mais a esquerda) é tratado com o bit de sinal. Se o bit de sinal for 0, o número é positivo, se for 1 é negativo.

Representação Sinal-Magnitude

A forma mais simples é tal que os n-1 bits representam a magnitude do número. Assim:

$$A = \begin{cases} \sum_{i=0}^{n-2} a_i 2^i & \text{se } a_{n-1} = 0 \\ -\sum_{i=0}^{n-2} a_i 2^i & \text{se } a_{n-1} = 1 \end{cases}$$

Exemplo:

+18 = 00010010

- 18 = 10010010

Desvantagens

- As operações aritméticas se tornam mais complicadas;
- Existem duas representações para o zero:

+0 = 00000000

-0 = 10000000

Representação de Complemento de Dois

- Números inteiros:

$$A = \sum_{i=0}^{n-2} a_i 2^i \qquad (1)$$

- Números negativos:

$$A = -a_{n-1} 2^{n-1} + \sum_{i=0}^{n-2} a_i 2^i \qquad (2)$$

Note que (1) está contido em (2) uma vez que para números positivos, $a_{n-1} = 0$

O zero é identificado como sendo positivo assim em sua representação o bit de sinal é 0.

Arquitetura de computadores

7	6	5	4	3	2	1	0

-2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0

-128	64	32	16	8	4	2	1

1	0	0	1	0	1	0	1

-128			16		4		1	= -107

1	1	1	0	1	0	0	0

-128	64	32		8				= -24

Conversão do número de bits de representação

- representação sinal-magnitude:

  +18 = 00010010 (8 bits)

  +18 = 0000000000010010 (16 bits)

  -18 = 10010010 (8 bits)

  -18 = 1000000000010010 (16 bits)

- representação de complemento de 2, forma correta de aumentar o número de bits:

+18 = 00010010 (8 bits)

+18 = 0000000000010010 (16 bits)

-18 = 11101110 (8 bits)

-18 = $1\underbrace{11111111}_{\text{completa-se com o mesmo valor do bit de sinal}}1101110$ (16 bits)

Aritmética de Inteiros (Representação de Complemento de Dois)

- Negação

1. Tome o complemento booleano de cada bit (inclusive o bit de sinal).
2. Tratando o resultado como um binário inteiro não sinalizado, acrescente 1

Exemplo

$$
\begin{array}{rr}
+18 = & 00010010 \\
\text{complemento} = & 11101101 \\
+ & 1 \\
\hline
 & 11101110 = -18
\end{array}
$$

Mostre que: (-18) = +18

Arquitetura de computadores

$$
\begin{array}{rl}
+0 = & 00000000 \\
\text{complemento} = & 11111111 \\
+ & 1 \\
\hline
\boxed{1} & 00000000 = 0
\end{array}
$$

“carry in” ou
“vai um”

$$
\begin{array}{rl}
-128 = & 10000000 \\
\text{complemento} = & 01111111 \\
+ & 1 \\
\hline
& 10000000 = \boxed{-128}
\end{array}
$$

não existe uma representação para 2^n

Portanto para 8 bits temos:

7	6	5	4	3	2	1	0
-2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
-128	64	32	16	8	4	2	1

O menor número –128 passado por zero e até 127

1	0	0	0	0	0	0	0

-128 = -128

0	1	1	1	1	1	1	1

0 64 32 16 8 4 2 1 = 127

Arquitetura de computadores

Tipo da ling. C	Número bits	<limits.h> Constant	Valores Faixa
		SCHAR_MIN	-128
signed char	8	SCHAR_MAX	127
			0
unsigned char	8	UCHAR_MAX	255
		SHRT_MIN	-32768
signed short	16	SHRT_MAX	32767
			0
unsigned short	16	USHRT_MAX	65535
		INT_MIN	-32768
signed int	16	INT_MAX	32767
			0
unsigned int	16	UNIT_MAX	65535
		LONG_MIN	-2147483647
signed long	32	LONG_MAX	2147483647
			0
unsigned long	32	ULONG_MAX	4294967295
		LLONG_MIN	-9,22337E+18
signed long long	64	LLONG_MAX	9223372036854775807
			0
unsigned long long	64	ULLONG_MAX	1,84467E+19

Considere um número inteiro de 8 bits

a) Faixa do número para inteiro não sinalizado

$2^8 = 256$.: Faixa de número é de 0 .. 255

b) Faixa do número para inteiro sinalizado (perco bit para sinal)

$2^7 = 128$.: Faixa de número é de -128 .. 127

Adição

Quando ocorre um overflow a ALU deve sinalizar para que este resultado não seja usado.

Um overflow pode ocorrer mesmo que não exista um carry.

```
 1001              1100
+0101             +0100
 1110 = -2        10000 = 0
(a) (-7) + (+5)   (b) (-4) + (+4)

 0011              1100
+0100             +1111
 0111 = 7         11011 = -5
(c) (+3) + (+4)   (d) (-4) + (-1)

 0101              1001
+0100             +1010
 1001 = Overflow  10011 = Overflow
(e) (+5) + (+4)   (f) (-7) + (-6)
```

ure 8.4 Addition of Numbers in Twos Comp
Representation

Regra: se dois números são somados, e têm o mesmo sinal, ocorre um overflow se o resultado tiver sinal oposto.1

Subtração

Para subtrair um número (subtraendo) de outro (minuendo), deve-se tomar o complemento de dois (negação) do subtraendo e soma-lo ao minuendo.

Exemplo:

$$\boxed{0100} - \boxed{0010}$$

$$\begin{array}{r} 0100 \\ +\ 1110 \\ \hline 1\ 0010 \end{array} \qquad\qquad \begin{array}{r} 1101 \\ +\quad 1 \\ \hline \boxed{1110} \end{array}$$

```
  0010
 +1001
  1011 = -5

(a) M = 2 = 0010
    S = 7 = 0111
   -S =     1001
```

```
  0101
 +1110
 10011 = 3

(b) M = 5 = 0101
    S = 2 = 0010
   -S =     1110
```

```
  1011
 +1110
 11001 = -7

(c) M =-5 = 1011
    S = 2 = 0010
   -S =     1110
```

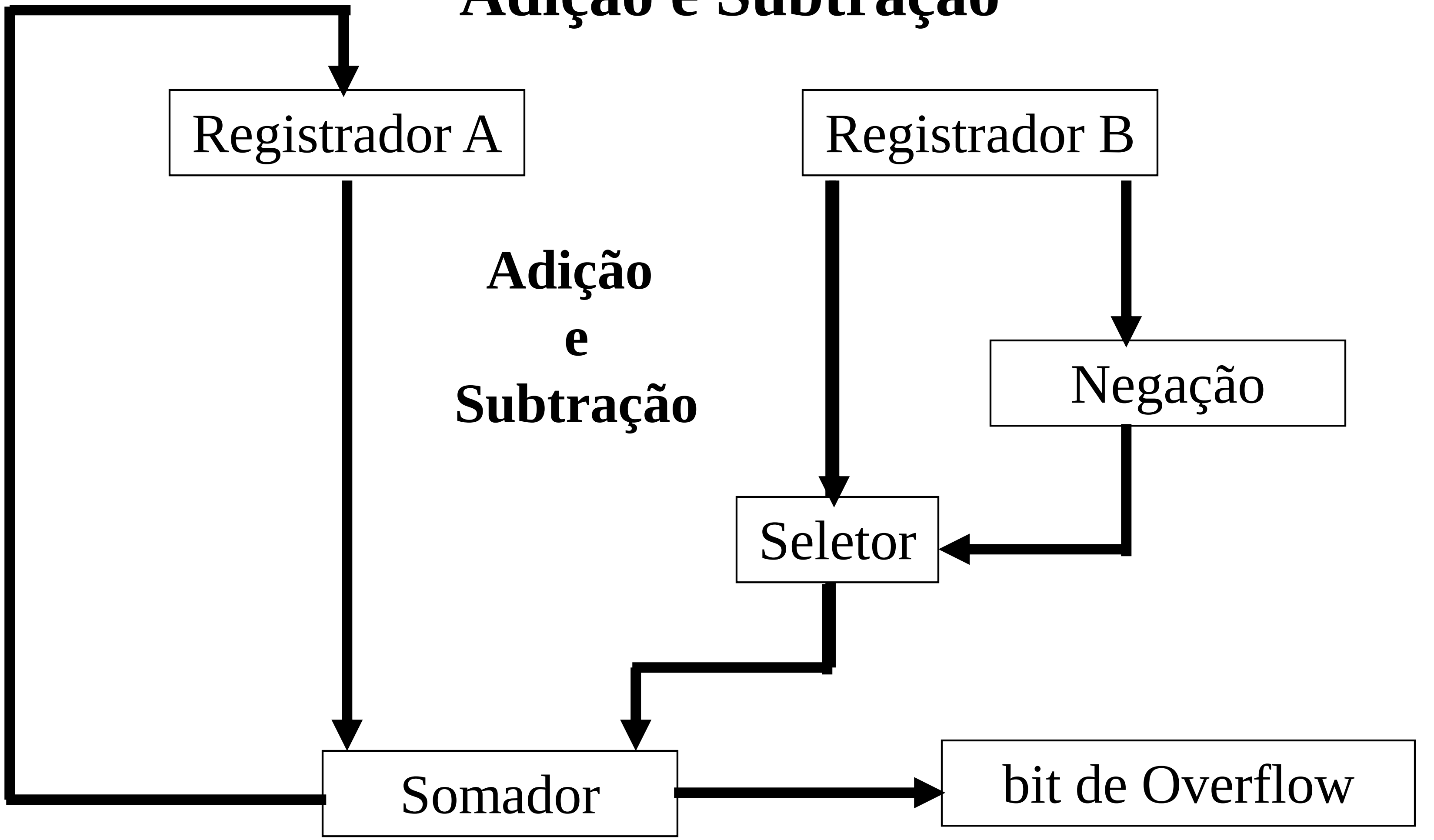
```
  0101
 +0010
  0111 = 7

(d) M = 5 = 0101
    S =-2 = 1110
   -S =     0010
```

```
  0111
 +0111
  1110 = Overflow

(e) M = 7 = 0111
    S =-7 = 1001
   -S =     0111
```

```
  1010
 +1100
 10110 = Overflow

(f) M =-6 = 1010
    S = 4 = 0100
   -S =     1100
```

Exemplos de subtração em completo de dois (m-s)

Diagrama de blocos de um Hardware para Adição e Subtração



Arquitetura de computadores

Exercícios

1) Converta os números de binário para decimal:
   a)10011   b)111011   c)1110 1111   d)0110 1110 1111

2) Qual é a melhor forma de representação números inteiros: sinal de magnitude ou complemento de dois.

3) Demonstre a faixa de funcionamento dos números inteiros sendo sinalizado e não sinalizado com:
   a) 12 bits   b)20 bits   c)24 bits

4) Demonstre os complemento de dois dos números:
   a) 23 (para 8 bits)
   b) 127 (para 8 bits)
   c) 0 (para 8 bits)
   d) 128 (para 8 bits)
   e) 3000(para 16 bits)

Arquitetura de computadores

Exercícios

5) Converta os números da base decimal para: hexa e binário.

a)10 b)64 b)121 c)1255 d)512 e)497

6) Converta os números da base hexadecimal para decimal e binário.

a) 36 b)2000 c)ABCD d) 1204 e)3333

7) Considere o números decimal apresentados nas letras abaixo:

a) 36 e 40 b)20 e 20 c)123 e 100 d) 240 e 204

Efetue a soma em binário e indique carry e overflow. Usar operações em 8 bits

8) Considere o números decimal apresentados nas letras abaixo:

a) 36 e 40 b)20 e 20 c)123 e 100 d) 240 e 204

Efetue a subtração em binário e indique carry e overflow. Usar operações em 8 bits