Universidade Federal de Campina Grande Centro de Engenharia Elétrica e Informática Unidade Acadêmica de Sistemas e Computação
Curso de Bacharelado em Ciência da Computação

Introdução à Computação

A Informação e sua Representação (Parte III – Adicional)

Prof.a Joseana Macêdo Fechine Régis de Araújo
joseana@computacao.ufcg.edu.br

Carga Horária: 60 horas

A Informação e sua Representação

- Aritmética em Sinal e Magnitude
- Aritmética em Ponto Flutuante
- Aritmética em BCD (informação adicional)

Aritmética em Sinal e Magnitude

Algoritmo para operação aritmética de adição

1. Verificam-se os sinais dos números e efetua-se uma comparação entre eles.

- Se ambos os números têm o mesmo sinal, somam-se as magnitudes; o sinal do resultado é o mesmo das parcelas.
- Se os números têm sinais diferentes:

  a) identifica-se a maior das magnitudes e registra-se o seu sinal;

  b) subtrai-se a magnitude menor da maior (apenas as magnitudes);

  c) sinal do resultado é igual ao sinal da maior magnitude.

Aritmética em Sinal e Magnitude

- **Exemplo**: Realize as operações aritméticas a seguir (em sinal e magnitude). Considere a palavra de dados com 6 bits.

a) $(+13)_{10} + (+12)_{10}$

b) $(+18)_{10} + (-11)_{10}$

c) $(-21)_{10} + (+10)_{10}$

d) $(-17)_{10} + (-9)_{10}$

e) $(+17)_{10} + (+19)_{10}$

f) $(-17)_{10} + (-19)_{10}$

Aritmética em Sinal e Magnitude

a) $(+13)_{10} + (+12)_{10}$

+13	001101
+12	001100
+25	**011001**

b) $(+18)_{10} + (-11)_{10}$

+18	010010
-11	101011
+ 7	**000111**

c) $(-21)_{10} + (+10)_{10}$

-21	110101
+10	001010
- 11	**101011**

d) $(-17)_{10} + (-9)_{10}$

-17	110001
-9	101001
-26	**111010**

Aritmética em Sinal e Magnitude

a) $(+17)_{10} + (+19)_{10}$

"vai 1"

+17	010001
+19	010011
+36	**100100**

overflow

b) $(-17)_{10} + (-19)_{10}$

"vai 1"

-17	110001
-19	110011
-36	**100100**

Estouro (***overflow***) - existência de um "vai 1" para o bit de sinal.

Faixa de representação de valores para 6 bits (em sinal e magnitude) $\Rightarrow$ -31 a + 31.

Aritmética em Sinal e Magnitude

Subtração (Minuendo - Subtraendo = Resultado)

Algoritmo para operação aritmética de subtração

1. Troca-se o sinal do subtraendo.

2. Procede-se como no algoritmo da adição.

Aritmética em Sinal e Magnitude

- **Exemplo**: Realize as operações aritméticas a seguir (em sinal e magnitude). Considere a palavra de dados com 6 bits.

  a) $(-18)_{10} - (+12)_{10}$

  b) $(-27)_{10} - (-14)_{10}$

  c) $(+27)_{10} - (+31)_{10}$

  d) $(+19)_{10} - (-25)_{10}$

Aritmética em Sinal e Magnitude

a) $(-18)_{10} - (+12)_{10}$ $= -12_{10}$

-18	110010
-12	101100
-30	**111110**

b) $(-27)_{10} - (-14)_{10}$ $= +14_{10}$

-27	111011
+14	001110
-13	**101101**

c) $(+27)_{10} - (+31)_{10}$ $= -31_{10}$

+27	011011
-31	111111
- 4	**100100**

d) $(+19)_{10} - (-25)_{10}$ $= +25_{10}$

vai 1

+19	010011
+25	011001
+44	**101100**

overflow

Aritmética em Sinal e Magnitude

- O problema encontrado pelos fabricantes de computadores na implementação da **ULA** (Unidade Lógica e Aritmética) que efetuasse operações aritméticas com valores representados em sinal e magnitude residiu, principalmente, em dois fatores: **custo** e **velocidade**.
  - **Custo** - necessidade de construção de dois elementos, um para efetuar somas e outro para efetuar subtração (dois componentes eletrônicos).
  - **Velocidade** - ocasionada pela perda de tempo gasto na manipulação dos sinais, de modo a determinar o tipo de operação e o sinal do resultado.

Aritmética em Sinal e Magnitude

- **Outro inconveniente:** dupla representação para o zero, o que requer um circuito lógico específico para evitar erros de má interpretação.

- Sistemas modernos não empregam aritmética em sinal e magnitude, a qual foi definitivamente substituída pela **aritmética em complemento de 2** (no caso de representação em **ponto fixo**).

Aritmética em Ponto flutuante

- Representação de números em ponto flutuante (notação científica)

$$n = \pm m \times b^{\pm e}$$

n - número

m – mantissa (ou significando)

b - base (2)

e - expoente

- As operações aritméticas devem ser realizadas considerando o produto; as mantissas e os expoentes devem ser manipulados separadamente.
- Ao fazer a operação dos expoentes, verificar qual a sua representação.

Aritmética em Ponto flutuante

Adição e Subtração

$$(m_1 \times b^{e1}) + (m_2 \times b^{e2}) = (m_1 \times b^{e1}) + (m_3 \times b^{e1})$$

$$= (m_1 + m_3) \times b^{e1}$$

$$(m_1 \times b^{e1}) - (m_2 \times b^{e2}) = (m_1 \times b^{e1}) - (m_3 \times b^{e1})$$

$$= (m_1 - m_3) \times b^{e1}$$

- São operações mais complexas em ponto flutuante do que em aritmética de ponto fixo, devido à **necessidade de alinhamento da vírgula** (ponto) **fracionária**.

Aritmética em Ponto flutuante

- Processo usual de ajuste da igualdade dos expoentes:o sistema efetua a subtração entre os valores dos expoentes. Em seguida, a mantissa de menor valor é dividida (deslocamento à direita - *shift right*) pelo valor da base$^{\text{diferença}}$.

- A operação de soma ou subtração das mantissas é uma simples operação em ponto fixo; o resultado da operação deve ser normalizado.

Aritmética em Ponto flutuante

Exemplo: Realize a operação aritmética $25,5_{10}$ (A) + $3,75_{10}$ (B), em ponto flutuante, utilizando 32 bits.

$25,5_{10}$ = 0 10000011 10011000000000000000000
= **1,10011 x 2^4** = **0,110011 x 2^5**

$3,75_{10}$ = 0 10000000 11100000000000000000000
= 1,111 x 2^1 = 0,1111 x 2^2 = **0,0001111 x 2^5**

A + B = 0,1110101 x 2^5 = $29,25_{10}$

$29,25_{10}$ = 1,110101 x 2^4
= 0 10000011 11010100000000000000000 (IEEE 754)

Aritmética em Ponto flutuante

Multiplicação e Divisão

- As operações de multiplicação e divisão com números representados em ponto flutuante também requerem a manipulação separada da mantissa e do expoente, não havendo porém a necessidade da operação de alinhamento da vírgula (que é uma operação demorada).

$$(m_1 \times b^{e1}) \times (m_2 \times b^{e2}) = (m_1 \times m_2)\, b^{(e1+e2)}$$

$$(m_1 \times b^{e1}) / (m_2 \times b^{e2}) = (m_1 / m_2)\, b^{(e1-e2)}$$

Aritmética em BCD

- Os algoritmos da aritmética em BCD são os mesmos que os correspondentes à aritmética em decimal, à qual sempre estivemos acostumados.

- Por exemplo, na soma em BCD, soma-se os algarismos individualmente, em decimal; quando a soma ultrapassa dez, "vai um" para o dígito de maior ordem (a próxima "casa") e, assim, por diante.

Aritmética em BCD

Algoritmo da operação de adição em BCD (A+B)

1. Decompor os números A e B em grupos de 4 bits, cada grupo representando um dos algarismos decimais dos números.

2. Somar os novos números A e B (aritmética binária), como se fossem valores binários puros (bit a bit).

3. Se o resultado da soma dos quatro primeiros algarismos for igual ou menor que 1001 (9_{10}) ele está correto. Retornar ao item 1.

Aritmética em BCD

Algoritmo da operação de adição em BCD (A+B)

4. Se o resultado for superior a 1001 e menor (ou igual) que 1111, soma-se 0110 (6_{10}) àquele valor; o novo resultado é o algarismo desejado; o "vai 1" após o último bit à esquerda é ainda acrescentado à soma da parcela seguinte. Retornar ao item 1.

5. Se o resultado for superior a 1111, somar 0110 ao resultado; o "vai 1" obtido com essa última soma é transferido para a parcela seguinte. Retornar ao item 1.

6. O processo acaba após a adição do último grupo de 4 bits (algarismo decimal mais significativo).

Aritmética em BCD

Exemplo: Adicionar $A=74_{10}$ e $B=65_{10}$ (em BCD)

0111	0100	74
0110	0101	65
1101	1001	
0110		
1 0011	1001	
1	3 9	

Aritmética em BCD

- **Exemplo**: Adicionar $A=1734_{10}$ e $B=4985_{10}$

```
  0001 0111 0011 0100        +1734
  0100 1001 1000 0101        +4985
  ------------------         -----
                             +6719

                    1001 (1)
               ´1 1011   (2)
                    0110
  ------------------------
     1  0000  0001 1001
  0101  0110
  ------------------------
  0110  0111  0001 1001
   6     7     1    9
```

Aritmética em BCD

Adição Rápida

1. Soma-se 6 a todos os dígitos da parcela A
2. A seguir somam-se as parcelas levando em conta o "vai 1" normalmente.
3. Nos dígitos em que não houver "vai 1" para a casa seguinte, soma-se dez, sendo que os "vai 1“ desta última operação serão desprezados.

A vantagem deste método é que ao se corrigir os dígitos não é necessário transportar o "vai um", ele é simplesmente abandonado.

Aritmética em BCD

- **Exemplo**: Adicionar $A=1734_{10}$ e $B=4985_{10}$

0001 0111 0011 0100	A
0110 0110 0110 0110	+ 6
0111 1101 1001 1010	
1 1	
0111 1101 1001 1010	(A+6)
0100 1001 1000 0101	+ B
1100 0111 0001 1111	
1010 1010	(+10)
0110 0111 0001 1001	
6 7 1 9	

Aritmética em BCD

Operação de subtração em aritmética BCD

- A operação de subtração de números representados na forma decimal é pouco simples e prática, devido ao fato de não ser possível converter de modo rápido os números negativos para sua representação de complemento.

Aritmética em BCD

Algoritmo da operação de subtração em BCD (A-B)

A subtração é dividida em 3 partes

1. Complementar cada dígito do subtraendo
2. Após o complemento fazer a soma de mais 1. (A + Bc + 1)
3. Nos dígitos em que não houver "vai um" deve-se somar 10, a fim de corrigi-los.

Aritmética em BCD

Exemplo: Realize a subtração 74_{10} - 65_{10} (em BCD)

1	0	
0111	0100	A
1001	1010	Bc
	1	+ 1
0000	1111	
	+ 1010	
0000	1001	
0	9	

Aritmética em BCD

- Em face da complexidade dos processos e a pouca utilidade prática em computação, não serão apresentados detalhes nem exemplos de métodos para realização de operações de multiplicação e divisão com valores representados em BCD.

Pode-se efetuar a multiplicação através de sucessivas somas e a divisão a partir de sucessivas subtrações (processo lento!).