



Organização e Arquitetura de Computadores

Representações Numéricas **REPRESENTAÇÃO EM PONTO FIXO**

Prof. Msc. Luiz Carlos Reis

- Quando há uma operação aritmética entre dois números de n algarismos e o resultado é um valor de $n+1$ algarismos
- Este fato ocorre pois em computadores com circuitos digitais seus elementos e/ou recursos são finitos
 - Diferente de um papel e lápis, onde o papel possui espaço suficiente para colocar mais um algarismo
- Memórias, registradores possuem tamanho fixo e finito
- Não há espaço para um bit extra
 - Ex.: vai 1

- Não é possível criar, na máquina, um número que contenha uma quantidade de algarismos acima de seu limite
- O que deve ocorrer quando uma operação aritmética produzir um resultado cujo o valor é superior ao limites da máquina
 - Deve acusar erro
- O *overflow* ocorre quando há uma soma entre dois números com o **mesmo sinal e o resultado é um valor com sinal diferente.**

Overflow

- Exemplo para números de 4 bits:

$$\begin{array}{r} 0101 \\ + 0110 \\ \hline 11011 \end{array}$$

Overflow, mudou o sinal.

$$\begin{array}{r} 1010 \\ - 0011 \\ \hline 1010 \\ + 1101 \\ \hline 10111 \end{array}$$

Overflow, mudou o sinal

Complemento bit a bit

Soma um

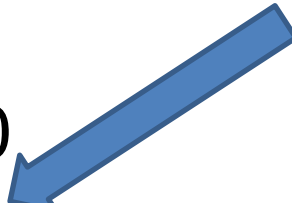
Devemos inverter o valor para complemento de 2 para realizar soma (Lembrando que o computador só realiza soma)

- Realize as seguintes operações com os números representados em complemento de 2 indicando se há *overflow* ou não em uma estrutura de 6 bits.
 - a) $110100 - 001101$
 - b) $111011 - 010010$
 - c) $001011 + 011100$
 - d) $010001 - 100101$
 - e) $111011 + 010010$
 - f) $010010 - 111011$

Overflow

Só aplicamos o complemento de 2 devido a operação ser subtração e o computador só realiza soma.

a) $110100 - 001101 = (-12 - (+13))$ Lembrando que temos 5 bits disponíveis e $2^5 = 32$ números


$$\begin{array}{r} 110100 \\ + 110011 \\ \hline 1100111 \end{array}$$

(Complemento de 2, pois o computador só soma)

1100111 (-25)

O *overflow* ocorre quando há uma soma entre dois números com o **mesmo sinal** e o **resultado é um valor com sinal diferente**.

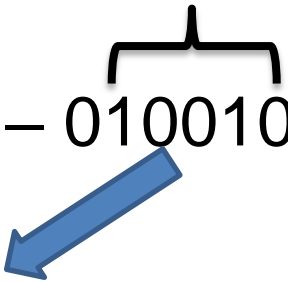
(Não mudou o sinal, portanto **NÃO** houve overflow)

Obs.: Zeros a esquerda não tem valor para números positivos e 1 (Uns) a esquerda também não tem valor para números negativos

Overflow

Lembrando que temos 5 bits disponíveis e $2^5 = 32$ números

b) $111011 - 010010$ (-5 - (+18))


$$\begin{array}{r} 111011 \\ + 101110 \text{ (Complemento de 2, pois o computador só soma)} \\ \hline 1101001 \text{ (-23)} \end{array}$$

O *overflow* ocorre quando há uma soma entre dois números com o **mesmo sinal** e o **resultado é um valor com sinal diferente**.

(Não mudou o sinal, portanto **NÃO** houve overflow)

Overflow

Não foi necessário realizar complemento de 2 pois a operação já é soma.

c) 001011 + 011100 (11 + 28) Lembrando que temos 5 bits disponíveis e $2^5 = 32$ números

$$\begin{array}{r} 001011 \\ + 011100 \\ \hline 100111 \end{array} \quad \begin{array}{l} \text{(Complemento de 2, pois o computador só soma)} \\ \text{(-25)} \end{array}$$

O *overflow* ocorre quando há uma soma entre dois números com o mesmo sinal e o resultado é um valor com sinal diferente.

(Mudou o sinal, portanto houve **overflow**)

Overflow

Só aplicamos o complemento de 2 devido a operação ser subtração e o computador só realiza soma.

d) $010001 - 100101 \quad (17 - (-27))$

Lembrando que temos 5 bits disponíveis e $2^5 = 32$ números

$$\begin{array}{r} 010001 \\ + 011011 \quad (\text{Complemento de 2, pois o computador só soma}) \\ \hline 101100 \quad (-20) \end{array}$$

O *overflow* ocorre quando há uma soma entre dois números com o **mesmo sinal** e o **resultado é um valor com sinal diferente**.

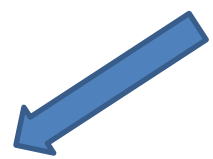
(Mudou o sinal, portanto houve **overflow**)

OBS.: Mesmo o número já ser negavito, temos que aplicar a regra de complemento de 2 devido a operação ser subtração e o computador só realiza soma, onde menos com menos irá dar soma.

Overflow

Não foi necessário
realizar complemento
de 2 pois a operação
já é soma.

e) $111011 + 010010$ ($-5 + (+18)$)

$$\begin{array}{r} 111011 \\ + 010010 \\ \hline 1001101 \end{array} \quad (13)$$


Lembrando que temos 5 bits
disponíveis e $2^5 = 32$ números

Neste caso temos uma exceção.

Como há uma soma entre dois números com **sinais diferentes, devemos comparar o resultado com o valor de maior número.**

Portanto temos que comparar o número 18 (010010) com o resultado 13 (001101) como não houve mudança de sinal **NÃO** houve overflow.

Overflow

Só aplicamos o complemento de 2 devido a operação ser subtração e o computador só realiza soma.

f) $010010 - 000101$ ($18 - (+5)$)

Lembrando que temos 5 bits disponíveis e $2^5 = 32$ números

$$\begin{array}{r} 010010 \\ + 111011 \\ \hline 1001101 \end{array} \quad \begin{array}{l} \text{(Complemento de 2, pois o computador só soma)} \\ \text{(13)} \end{array}$$

Neste caso temos uma exceção.

Como há uma soma entre dois números com **sinais diferentes, devemos comparar o resultado com o valor de maior número.**

Portanto temos que comparar o número 18 (010010) com o resultado 13 (001101) como não houve mudança de sinal **NÃO** houve overflow.

- Números inteiros, positivos e negativos, são representados na forma de ponto fixo;
- A grande vantagem dos números de ponto fixo é a possibilidade de realizar somas e subtrações diretamente;
- Para que as somas de números positivos e negativos tenham resultados corretos em sua faixa de validade, os números negativos devem ser representados em complemento de dois.

Prof. Msc. Luiz Carlos Reis

luiz.reis@cruzeirosul.edu.br