Unidade 4: Sistemas de Numeração

Representação de Dados em Ponto Fixo

Prof. Daniel Caetano

Objetivo: Apresentar as representações mais utilizadas para ponto fixo.

Bibliografia STALLINGS, 2003; MURDOCCA e HEURING, 2000.

INTRODUÇÃO

Na aula anterior foram apresentados os métodos para conversão entre diversas bases e também como realizar cálculos diretamente nas bases de potências de dois. Nesta aula serão apresentadas maneiras de representar numerais de ponto fixo com sinal usando a representação binária, de maneira compatível com os cálculos apresentados anteriormente.

1. PONTO FIXO

Chamamos de representação em ponto fixo aquela que o numeral é guardado como um valor inteiro na memória, sem a especificação da vírgula. Esta última, por sua vez, é considerada **fixa** em uma posição (daí o nome, ponto fixo) no momento da interpretação do valor. Considere o número binário abaixo, por exemplo:

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
1	1	0	1	1	0	0	1

Considerando esta representação sem sinal, em ponto fixo, ele pode ser interpretado como sendo um número inteiro: 11011001b, que é o mesmo que 217 em decimal. Se, por outro lado, considerarmos que a vírgula está entre o bit 3 e o bit 4, o número pode ser interpretaod como 1101,1001b, que é o mesmo que 13,5625.

Usualmente, o computador trata os números de ponto fixo como sendo inteiros, isto é, a vírgula situa-se à direita do bit 0, daí muitas vezes nos referirmos aos números inteiros como sendo números de ponto fixo (eles, de fato, o são!). Nesta aula, trataremos especificamente dos números inteiros.

2. REPRESENTAÇÃO DE SINAL

É muito comum a necessidade de representar números negativos. De fato, em especial quando fazemos contas, é frequente o surgimento dos números negativos. A primeira idéia

para representar números negativos é reservar o bit mais significativo (o de "número mais alto") como sendo o bit de sinal, não sendo considerado para a interpretação do valor. Considerando que se o bit de sinal contiver um valor 0 o número é positivo, se ele contiver o valor 1, o número será positivo.

Por exemplo: vamos representar o número 97:

Bit 7 (SINAL)	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0	1	1	0	0	0	0	1

Agora, vamos representar o número -97:

Bit 7 (SINAL)	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
1	1	1	0	0	0	0	1

Observe que a diferença está apenas no bit do sinal.

Como sobraram apenas 7 bits (0 a 6) para indicar o número, não é mais possível fazer números entre 0 e 255; com 7 bits é possível apenas gerar números entre 0 e 127, inclusive. Considerando o bit do sinal, porém, temos duas faixas representáveis: de -127 a 0 e de 0 a 127.

Ou seja: com 1 byte podemos representar um número de 0 a 255 sem sinal ou, usando a representação indicada assim, de -127 a 127, considerando o sinal. Os bits que representam a parte numérica recebem o nome de **magnitude**.

Entretanto, essa representação não é comumente usada. Há dois problemas "graves" com ela. O primeiro é que há dois zeros na sequência de -127 a 127: o 0 e o -0, que, obviamente, deveriam ter a mesma representação (zero é sempre zero!):

Bit 7 (SINAL)	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0	0	0	0	0	0	0	0

Bit 7 (SINAL)	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
1	0	0	0	0	0	0	0

O segundo problema é nas operações. Vejamos:

Decimal	Binário	Convertendo para Decimal
	1 1 1 1 1 1	
2	00000010b	2
-5	-00000101b	-5
-3	11111101b	-125

Opa, algo deu BEM errado! Vamos tentar repetir a operação, mas fazendo como 2 + (-5) e ver o resultado.

Binário	Convertendo para Decimal
1 1 1 1 1 1	
00000010b	2
+10000101b	-5
10000111b	-7

É, definitivamente essa representação não é muito boa!

2.1. Representação de Sinal com Complemento de Um

Os efeitos apresentados nas operações anteriores são advindos do fato que o bit de sinal está atrapalhando a continuidade da contagem binária: se tivermos o número 127 e somarmos 1, deveríamos ter uma contagem cíclica que levasse ao -127... Mas não é isso que ocorre:

Binário:	...	01111110b	01111111b	10000000b	10000001b	...
Decimal:	...	126	127	-0	-1	

Ainda: observe que ao **somarmos** 1 unidade positiva ao valor negativo -1, ele se torna -2, o que é totalmente inadequado. Para remediar essa solução, é usada a **representação em complemento**. O que significa isso? Significa que números negativos são obtidos invertendo **todos os bits** do número positivo.

Assim, se 97 positivo é assim:

Bit 7 (SINAL)	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0	1	1	0	0	0	0	1

O número -97 fica assim:

Bit 7 (SINAL)	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
1	0	0	1	1	1	1	0

Observe que os bits da magnitude foram completamente invertidos! Observe que, neste caso, agora há continuidade:

Binário:	...	01111110b	01111111b	10000000b	10000001b	...
Decimal:	...	126	127	-127	-126	

Binário:	...	11111110b	11111111b	00000000b	00000001b	...
Decimal:	...	-1	-0	0	1	

Adicionalmente, se somarmos 1 unidad positiva a um número negativo, ele terá o comportamento adequado. Por exemplo: somando 1 (00000001b) a -1 (11111110b) o resultado é -0 (11111111b) = 0 (00000000b).

Esta representação funciona **muito** melhor que a representação simplificada com bit de sinal, mas ainda tem um problema: há duas representações para zero. Isso significa que há uma descontinuidade na conta: se eu somar -1 com 2, ao invés de obter 1, eu obterei...

Decimal	Binário (comp. de 1)	Binário para Decimal
-1	11111110b	-1
+2	-00000010b	+2
+1	00000000b	0

Como resolver este problema?

2.2. Representação de Sinal com Complemento de Dois

Para corrigir os efeitos das representações anteriores, ao invés de realizar o complemento de 1 (inverter os bits), realiza-se o complemento de 2, que é o seguinte:

"Para converter um número positivo para negativo, calcula-se seu complemento e soma-se 1".

Em outras palavras, calcula-se o complemento de 1 e soma-se um ao resultado.

Assim, se 97 é representado como abaixo:

Bit 7 (SINAL)	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0	1	1	0	0	0	0	1

O número -97 em complemento de **um** fica assim:

Bit 7 (SINAL)	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
1	0	0	1	1	1	1	0

O número -97 em complemento de **dois** fica assim:

Bit 7 (SINAL)	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
1	0	0	1	1	1	1	1

Para desinverter, isto é, transformar um negativo em um positivo, o processo é inverso: subtrai-se 1 e, em seguida, realiza-se a operação de complemento de 1.

Com isso, todos os problemas anteriores ficam solucionados. A continuidade existe:

Binário:	...	01111110b	01111111b	10000000b	10000001b	...
Decimal:	...	126	127	-128	-127	

Binário:	...	11111110b	11111111b	00000000b	00000001b	...
Decimal:	...	-2	-1	0	1	

Além de ser corrigido o problema anterior - repetição do zero -, agora há um número negativo adicional a ser representado: -128.

3. BIBLIOGRAFIA

STALLINGS, W. **Arquitetura e organização de computadores**. 5ed. São Paulo: Ed. Pearson Prentice Hall, 2003.

MURDOCCA, M. J; HEURING, V.P. **Introdução à Arquitetura de Computadores**. S.I.: Ed. Campus, 2000.