

Departamento de Eletrônica e Computação ELC 1011– Organização de Computadores

Nome:			

3ª Prova (Gabarito)

1 – (1,0 ponto, 10 min.) Escreva os números +15,125 e +22,75 em ponto fixo sem sinal. Use 5 bits para a parte inteira e 3 bits para a parte fracionária.

Solução: Separamos a parte inteira da parte fracionária do número decimal. Convertemos a parte inteira para binário usando o método da divisão longa. Ajustamos o número de bits, neste exemplo, para ter 5 bits. A parte fracionária é convertida para binário usando o método da multiplicação longa. Em seguida ajustamos o número de bits.

2 – (1,0 ponto, 10 min.) Escreva os números +26,625 e -26,625 em ponto fixo, representados em complemento de 2. Use 5 bits para a parte inteira e 3 bits para a parte fracionária.

Solução: Retiramos os sinais dos números. Realizamos a conversão para binário como no problema anterior. O número positivo teve ter o bit mais significativo igual a zero. Se for um, significa que não é possível representar o número com este número de bits. Para obter a representação do número negativo, realizamos o complemento de 2 do número. O bit mais significativo deve ser igual a 1.

(a) + 26,625

 $26,625 = 11010101_2$. O bit mais significativo é um: não é possível representar este número usando 5 bits para a parte inteira. Com 6 bits (011010101₂) ou mais bits na parte inteira, podemos representar este número com sinal, em complemento de dois.

(a) -26,625

Encontramos a representação de +26,625 = 011010101₂. São necessários seis bits na parte inteira, para representar corretamente o valor positivo. Complementamos o número (complemento de 2): -26,625 = 100101011₂. O bit mais significativo é um, indicando que o número é negativo. Todavia, são necessários 6 bits para representar o número corretamente.

	Sem sinal							
+26,625 =								
-26,625 =								

Resposta: Não é possível representar os valores +26,625 e -26,625 em complemento de 2, usando 5 bits para a parte inteira e 3 bits para a parte fracionária.

3 – (1,5 pontos, 10 min.) Sejam os números binários $A=1000\ 0111_2$ e $B=1111\ 1111_2$, representados em complemento de 2. Calcule A+B e A-B. Apresente os vem uns das operações. Indique se houve overflow na operação. Por quê? O resultado está correto?

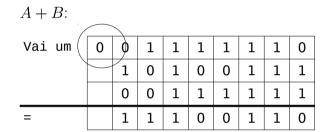
A-B:

A+B:									
Vai um	I	1	1	1	1	1	1	1	0
		1	0	0	0	0	1	1	1
		1	1	1	1	1	1	1	1
=		1	0	0	0	0	1	1	0

		_							
Vai um	(8)	0	0	0	0	1	1	1	1
		1	0	0	0	0	1	1	1
		0	0	0	0	0	0	0	0
=		1	0	0	0	1	0	0	0

O vem um e vai um dos bits mais significativos são iguais. Não há overflow nas operações. Os resultados estão corretos.

4 – (1,5 pontos, 10 min.) Sejam os números binários $A=1010\ 0111_2$ e $B=0011\ 1111_2$, representados como números sem sinal. Calcule A+B e A-B. Apresente os vem uns das operações. Indique se houve *overflow* na operação. Por quê? O resultado está correto?



A-B:									
. ($\overline{}$							
Vai um(1	p	0	0	0	1	1	1	1
		1	0	1	0	0	1	1	1
		1	1	0	0	0	0	0	0
=		0	1	1	0	1	0	0	0

Na adição, o vai um dos bits mais significativos é igual a 0. Não há overflow. O resultado da operação está correto.

Na subtração, há um vai um dos bits mais significativos. Não há overflow. O resultado da operação está correto.

5 – (1,5 pontos, 20 min.) Faça a multiplicação dos números binários, com 5 bits e sem sinal, 10110_2 e 11101_2 . Use o segundo algoritmo da multiplicação. Apresente todas as iterações.

Iteração	Passo	Multiplicando	Produto
0	Valores iniciais	10110	0 00000 11101
1	(a) Bit menos significativo do produto igual a 1: soma multiplicando. (b) Desloca o produto	10110	0 00000 11101 + 0 10110 0 10110 11101 0 01011 01110
2	(a) Bit menos significativo do produto igual a 0: nenhuma operação.(b) Desloca o produto.	10110	0 01011 01110 0 00101 10111
3	(a) Bit menos significativo do produto igual a 1: soma multiplicando.	10110	0 00101 10111

	(b) Desloca o produto		0 1	.0110 .1011 10111
4	(a) Bit menos significativo do produto igual a 1: soma multiplicando. (b) Desloca o produto	10110	+ 0 1 1 0	1101 11011 .0110 .0011 11011
5	(a) Bit menos significativo do produto igual a 1: soma multiplicando. (b) Desloca o produto	10110	+ 0 1 1 0	.0001 11101 .0110 .0111 11101 .0011 11110

Resposta: O resultado da multiplicação é 1001111110₂.

 $6-(1,0\ ponto,\ 10\ min.)$ Mostre a representação binária do número -35,375 em precisão simples, no padrão IEEE 754 (Use 32 bits: 1 bit de sinal, 8 bits para o expoente e 23 bits para a fração. O peso é 127_{10} .).

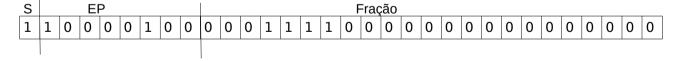
$$(-1)^s \cdot (1 + fracao) \cdot 2^{expoente-peso}$$

Retiramos o sinal do número e convertemos para binário:

$$35,375 = 100011,11_2$$

O expoente polarizado é igual ao expoente somado ao peso, que neste caso é 127.

O bit de sinal S é igual a 1 porque o número é negativo.



7 – (1,0 ponto, 10 min.) Qual o valor decimal do seguinte número, representado em ponto flutuante, precisão simples: 11000010 10100100 01000000 00000000? Resposta: –82,125

Sinal = 1. O número é negativo.

Expoente polarizado = 100001012 = 133

Subtraímos o peso para encontrarmos o expoente.

Expoente = 133-127 = 6

Fração = 010010001

Significando = 1, Fração = 1, 010010001_2 = 1, 283203125

O valor do número = $-1,283203125 \cdot 2^6$ = -82,125

8 – (1,5 pontos, 10 min.) Seja a seguinte instrução de adição no processador MIPS: add \$t0, \$t1, \$t2. Após a execução desta instrução, como você verifica o overflow, considerando que: (a) \$t1 e \$t2 são números sem sinal e (b) \$t1 e \$t2 são números com sinal, representados em complemento de 2. Explique.

Resposta:

(a) \$t1 e \$t2 são números sem sinal.

O maior valor possível de ser armazenado em \$t0 = 0xFFFF FFFF ou $2^{32}-1$. Haverá overflow se a seguinte condição for verdadeira:

$$\$t1 + \$t2 > 2^{32} - 1$$

$$\$t1 > 2^{32} - 1 - \$t2$$

$$\$t1 > \$t2$$
 \$t2 \(\epsilon\) complementado bit a bit (complemento de 1).

Haverá *overflow* na operação de adição quando a desigualdade acima for verdadeira. Na verificação da desigualdade, os valores devem ser tratados sem sinal.

- (b) \$t1 e \$t2 são números com sinal (complemento de 2).
 - 1. Verificamos os sinais de \$t1 e \$t2 (bits mais significativos dos números).
 - (a) Se forem diferentes, não haverá overflow na operação.
 - (b) Se forem iguais, poderá haver overflow na operação. Neste caso, comparamos os bits de sinal de um dos operandos (\$t1 ou \$t2) com o bit de sinal do resultado (\$t0).
 - i. Se forem diferentes, houve overflow na operação de adição.
 - ii. Se forem iguais, não houve overflow.

Tabela 1 - Exemplo do 2º algoritmo da multiplicação.

Iteração	Passo	Multiplicando		Produto
0 -	(0) valores Iniciais	1011	0	0000 1101
	1a) somar os bits mais significativos do produto com o multiplicando	1011	0	0000 1101 1011
	(1b) deslocamento lógico do produto e o vai-um do bit MSB para a direita	1011	0	1011 1101 0101 1110
2 - ('2a) nenhuma operação ◀	1011	0	0101 1110
	(2b) deslocamento lógico do produto e o vai-um do bit MSB para a direita	1011	0	0010 1111
	3a) somar os bits mais significativos o produto com o multiplicando	1011	+	0010 1111 1011
	(3b) deslocamento lógico do produto e o vai-um do bit MSB para a direita	1011	0	1101 1111 0110 1111
	4a) somar os bits mais significativos do produto com o multiplicando	1011	<u>0</u> +	0110 1111 1011
	(4b) deslocamento lógico do produto e o vai-um do bit MSB para a direita	1011	0	0001 11 <u>1</u> 1 1000 1111
				Resultado