UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
INSTITUTO DE INFORMÁTICA - DEPARTAMENTO DE INFORMÁTICA APLICADA

INF01107 - Introdução à Arquitetura de Computadores –2011/1

Trabalho Prático 1 - Simulador NEANDER

Escrever um programa para o simulador Neander que implemente as operações aritméticas básicas de multiplicação e divisão sobre números inteiros positivos, representados como inteiros sem sinal, isto é, os operados estão na faixa entre 0 e 255, inclusive.

Estas operações devem ser utilizadas para calcular o valor da expressão

$$x = (a/b) * c$$

Para a definição das variáveis devem ser obrigatoriamente utilizadas as seguintes posições de memória:

Endereço 128 – variável *a*
Endereço 129 – variável *b* (garantidamente $b > 0$)
Endereço 130 – variável *c*
Endereço 131 – variável *x*, byte mais significativo
Endereço 132 – variável *x*, byte menos significativo

A operação de divisão a ser programada é uma divisão inteira, ou seja, o seu resultado é o quociente da divisão, e o resto é ignorado. Assim, por exemplo, 9/4 = 2, 17/5 = 3 e 127/8 = 15. Como as variáveis *a* e *b* são números representados em um byte, o resultado também sempre poderá ser representado em um byte, sem problemas de estouro de representação.

A operação de multiplicação deverá ser programada de tal maneira que forneça o resultado em dois bytes (16 bits), para que não ocorram problemas de estouro de representação. Se fosse utilizado um único byte para armazenar o resultado, estouros seriam bem freqüentes.

Os valores fornecidos nas posições de memória 128, 129 e 130 não devem ser alterados pelo programa.

Dicas:

1. O Neander não possui operação de divisão. Entretanto, o quociente de uma divisão pode ser calculado através de uma série de subtrações sucessivas. Note que as subtrações são condicionais, ou seja, devem ser realizadas até que o valor do dividendo atual seja zero (e neste caso o resto é zero) ou até que o valor do dividendo atual seja menor que o divisor (e neste caso o dividendo atual é o resto da divisão).
2. O Neander não possui operação de subtração. Entretanto, uma subtração pode ser transformada em uma soma através do complemento do subtraendo.
3. O Neander não possui operação de multiplicação. Entretanto, uma multiplicação pode ser transformada em uma série de somas sucessivas. Note que, uma vez conhecido o valor do multiplicador, sabem-se quantas somas deverão ser realizadas.
4. Para que as somas sucessivas de uma multiplicação forneçam o resultado em dois bytes, basta realizar a soma no byte menos significativo, e incrementar o byte mais significativo toda a vez que a soma produzir um “vai-um” (carry).
5. Como o Neander não fornece diretamente a informação de “vai-um” de uma soma, ela deve ser calculada indiretamente, a partir das informações disponíveis. Este cálculo é um pouco complexo,

mas inicia com a constatação que a informação de "negativo" é simplesmente o bit mais significativo do byte sendo operado.

Os trabalhos serão corrigidos de forma automática, com 20 grupos de valores diferentes. Portanto, devem ser observadas rigorosamente as seguintes especificações:

- o código do programa deve iniciar na posição 0 da memória.
- a primeira instrução executável deve estar na posição 0 da memória.
- os endereços das variáveis de entrada devem ser exatamente os especificados acima.
- para constantes e variáveis adicionais, ou para trechos extras de programa, usar as posições de memória de 133 em diante.
- o programa será executado 20 vezes de forma consecutiva (sem ser carregado de disco a cada vez); portanto é necessário inicializar todas as variáveis utilizadas.

O trabalho deverá ser entregue no Moodle, na área de "Entrega do Primeiro Trabalho", na forma de um arquivo compactado (formato Zip ou Rar) composto por:

- um arquivo de memória do Neander (.mem), contendo o programa executável.
- um arquivo texto (.txt), com documentação contendo uma breve descrição do método utilizado e uma listagem do programa fonte usando mnemônicos simbólicos para as instruções (LDA, STA, etc ...) e indicando em decimal os endereços de memória nos quais as instruções ficam armazenadas e os endereços dos operandos. Esse programa fonte deve ser comentado. Não se esqueça de incluir seu nome **completo** e seu número de cartão nas primeiras linhas deste arquivo.

Para nomear os arquivos, utilize todo o seu nome, usando maiúsculas e minúsculas, sem espaços em branco e sem acentos. Assim, por exemplo, o aluno Um de Três Quatro deve denominar os seus arquivos de **`UmDeTresQuatro.MEM`**, **`UmDeTresQuatro.TXT`** e **`UmDeTresQuatro.ZIP`** (ou **`RAR`**).

Data de Entrega: 12/05/2011 via http://moodle.inf.ufrgs.br

Exemplos de casos de teste (todos os valores estão indicados no sistema decimal)

Endereço	128	129	130	131	132	**valor de x**
	a	**b**	**c**	**x, mais sign.**	**x, menos sign.**	
Caso 1	3	2	7	0	7	7
Caso 2	17	5	16	0	48	48
Caso 3	20	1	120	9	96	2400
Caso 4	127	250	12	0	0	0
Caso 5	250	128	24	0	24	24
Caso 6	255	130	250	0	250	250
Caso 7	180	170	255	0	255	255
Caso 8	190	19	12	0	120	120
Caso 9	250	1	255	249	6	63750
Caso 10	128	8	16	1	0	256
Caso 11	100	25	50	0	200	200
Caso 12	254	255	128	0	0	0
Caso 13	180	100	0	0	0	0
Caso 14	255	127	5	0	10	10
Caso 15	180	1	127	89	76	22860
Caso 16	160	2	13	4	16	1040
Caso 17	255	5	6	1	50	306
Caso 18	64	8	17	0	136	136
Caso 19	185	12	17	0	255	255
Caso 20	201	19	42	1	164	420