EA 869 – Turma U – 25/10/2011
Prova 2.2 – Sem consulta – Duração: 100 minutos

NOME: **R.A.:**

Questão 1. (3,0) Considere o processador micro-programado (folha anexa). Mostre o micro-programa para executar a instrução: "**Se (ACC) = 0, end (R2) + (R1), Acc (R2) + (R1); caso contrário, end (R1) – 1, Acc (R1) – 1**".

Esta instrução tem comprimento de 16 bits: **4** bits para o **CO** (**1101**) e **12** bits para **end**.

a) Explique as micro-operações das posições 0 e 1 da micro-memória;
b) Mostre o mapeamento para o micro-programa da instrução (início na posição **57H**);
c) Inicie o micro-programa a partir da posição indicada em b. Após a execução deve-se retornar para execução da próxima instrução de máquina (SC8 – subtração; **sugestão**: carregar **end** em REM na mesma microinstrução do teste).

Observe que cada linha da tabela corresponde a um endereço da micro-memória e a fase de busca já está preenchida.

Endereço da micro-memória (hexadecimal)	**Sinais de controle**	**Micro-operações/Comentários**
0	1, 15, 16, 19, 23	
1	1, 5, 10, 18, 22, 23	

Questão 2. **(1,5)** Para as instruções a seguir determine o conteúdo das posições de memória e registradores afetados por suas execuções, considerando que antes de suas execuções tinha-se os seguintes conteúdos: SP = 80, R2 = 280, R1 = 700, RIX= 700, RB = 1000, end.105 = 47, end.115 = 7, end.160 = 18, end.280 = 105, end.700 = 24, end.710 = 9, end.800 = 115, end.1000 = 32, end.1050 = 30.

Importante: considere as execuções das instruções independentes entre si, ou seja, os valores valem para a execução de cada uma das instruções.

O valor do PC e os modos de endereçamento para cada instrução são dados juntos com a instrução; o formato das instruções é: **operação operando-fonte, operando-destino.**

PC	**Instrução**	**Modos de endereçamento**
3000	MOVE R2, (R1)	registrador direto, registrador indireto
3100	ADD RIX(10), 160	indexado, absoluto direto
3200	MOVE #10, 9(PC)	imediato, relativo
3300	ADD (800), 50(RB)	absoluto indireto, baseado
3400	MOVE 105, -(R2)	absoluto direto, auto-decremento
3500	PUSH R1	registrador direto, pilha

Questão 3. (1,0) Na definição da macro ADSUB abaixo são usadas as pseudo-instruções AIF e AGO para implementar a montagem condicional.

```
MACRO     ADSUB N, PAR1, PAR2, RES      AGO .FIM
          MOVE PAR2, R1                 .ADD  ADD PAR1, R1
          AIF (N = 1) .ADD              .FIM  MOVE R1, RES
          SUB PAR1, R1                  ENDMACRO
```

Fazer a expansão das seguintes chamadas de ADSUB:

a) ADSUB 1, A, B, C

b) ADSUB 2, X, Y, Z

Questão 4. (2,5) Seja o programa abaixo:

```
DADO1: DW 16                    (7)   SUB:  POP R6
DADO2: DW 32                    (8)         POP R4
(1)    MOVE #160, SP            (9)         POP R5
(2)    MOVE #DADO2, R1          (10)        ADD R4, (R5)
(3)    MOVE DADO1, R2           (11)        MOVE (R5), R4
(4)    PUSH R1                  (12)        PUSH R4
(5)    PUSH R2                  (13)        PUSH R6
(6)    CALL SUB                 (14)        RTS
(15)   POP R1
```

Considere que: o formato da instrução é **operação fonte, destino; DW n** é a pseudo-instrução para reservar uma palavra com valor **n; DS m** é a pseudo-instrução para reservar **m** palavras sem valores definidos; **#NOME** é o endereço da variável NOME; cada instrução ocupa uma palavra da memória; a instrução (1) está na posição 500 da memória.

a) Mostrar o conteúdo de SP, R1 e R2 após a execução de (3)

b) Mostrar o conteúdo de SP e da pilha após a execução de (6)

c) Mostrar o conteúdo de SP e da pilha após a execução de (8)

d) Mostrar os valores de DADO1 e DADO2 após a execução de (11)

e) Mostrar o conteúdo de SP e da pilha após a execução de (14)

f) Mostrar o conteúdo de SP, da pilha e de R1, e os valores de DADO1 e de DADO2 após a execução de (15).

g) Qual é o mecanismo de passagem de parâmetros e quais são os tipos de passagem de parâmetros para os parâmetros da subrotina SUB?

Questão 5. (3,0) Seja uma arquitetura com 3 linhas de interrupção: **L1**, **L2** e **L3**. O programa principal, **PP**, está armazenado a partir da posição 400H da memória; as rotinas de serviço associadas às linhas – **ROT1**, **ROT2** e **ROT3** – estão armazenadas a partir das posições de memória, respectivamente, **600H**, **700H** e **800H**. A máscara de interrupção está nos três bits menos significativos do **PSW** (**X X X X X L3 L2 L1**). O vetor de interrupção ocupa posições contíguas da memória, a partir da posição **14H** da memória.

i) **PP** só pode ser interrompido por sinal de interrupção em **L2** ou **L3**;
ii) **ROT2** só pode ser interrompida por sinal de interrupção em **L1**;
iii) **ROT1** só pode ser interrompida por sinal de interrupção em **L3**;
iv) **ROT3** pode ser interrompida por sinal de interrupção em **L1** ou **L2**, mas não por sinal de interrupção na própria linha.

a) Explique o funcionamento do esquema de interrupção a partir da execução de **PP** e um sinal de interrupção em **L2** (indique as possíveis seqüências de ativação das rotinas com sinais de interrupção);

b) Explique o funcionamento do esquema de interrupção a partir da execução de **PP** e um sinal de interrupção em **L3** (indique as possíveis seqüências de ativação das rotinas com sinais de interrupção);

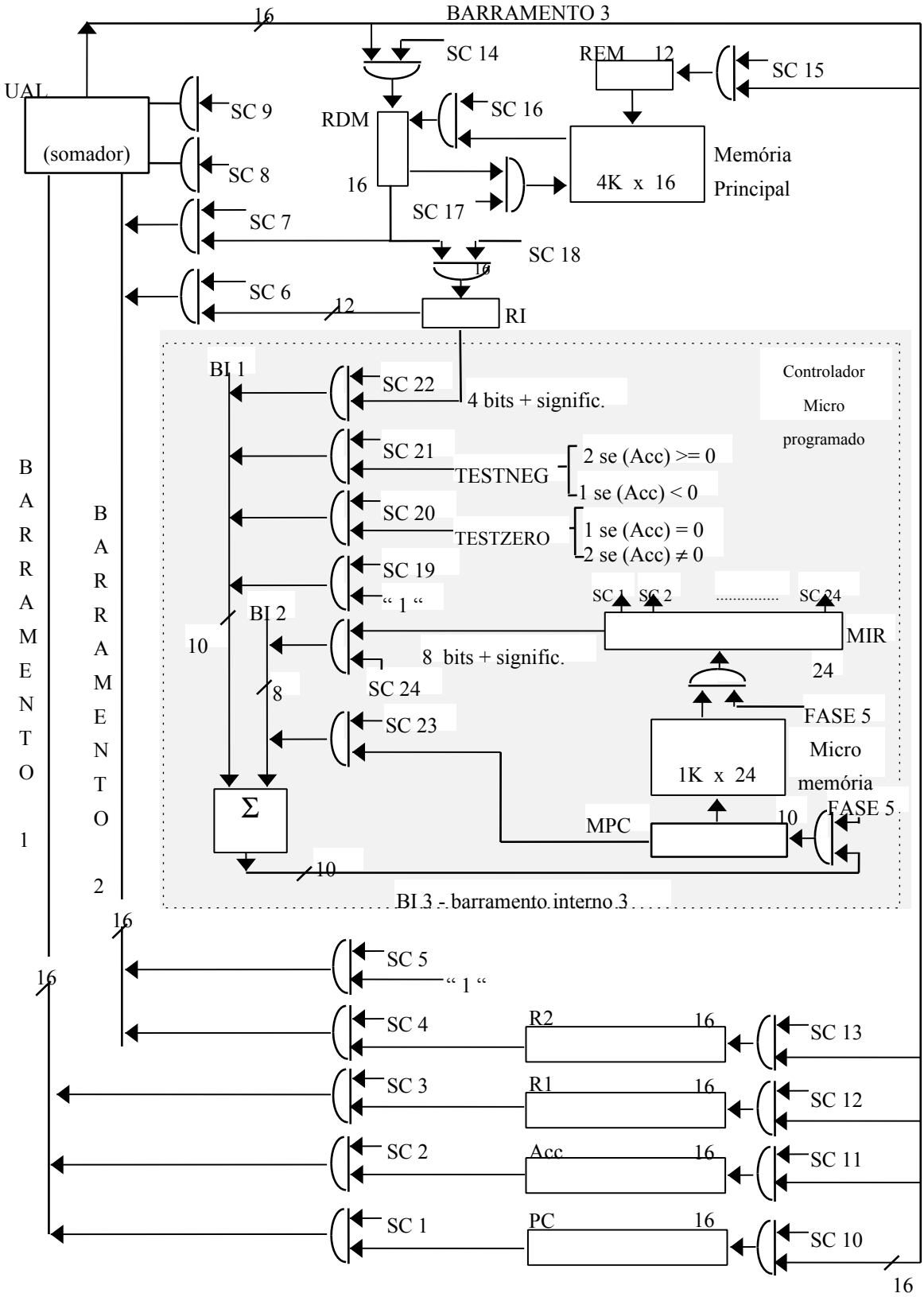
c) Mostre os valores das máscaras de interrupção para **PP**, **ROT1**, **ROT2** e **ROT3**:

mask: mask1:

mask2: mask3:

d) Mostre o mapa de memória (vetor de interrupção, **PP**, **ROT1**, **ROT2** e **ROT3**) com toda a informação relevante para o caso em que a arquitetura implementa o mecanismo de interrupção sem PSW no vetor de interrupção: **PUSH PC; PUSH PSW; JUMP (endi).** Usar psw, rti, move, etc. para indicar que são os códigos binários correspondentes a PSW, RTI, MOVE, etc.

e) Mostre o mapa de memória com toda a informação relevante para o caso em que a arquitetura implementa o mecanismo de interrupção com PSW no vetor de interrupção: **PUSH PC; PUSH PSW; MOVE endi, PSW; JUMP (endi + 1).** Usar psw, rti, move, etc. para indicar que são os códigos binários correspondentes a PSW, RTI, MOVE, etc.


16
BARRAMENTO 3
SC 14
REM 12
SC 15
UAL
SC 9
RDM
SC 16
(somador)
SC 8
16
4K x 16
Memória
Principal
SC 7
SC 17
SC 18
16
SC 6
12
RI
BI 1
SC 22
4 bits + signific.
Controlador
Micro
programado
SC 21
TESTNEG
2 se (Acc) >= 0
1 se (Acc) < 0
SC 20
TESTZERO
1 se (Acc) = 0
2 se (Acc) ≠ 0
SC 19
" 1 "
SC 1
SC 2
.....
SC 24
BI 2
MIR
10
8 bits + signific.
SC 24
8
24
FASE 5
SC 23
Micro
1K x 24
memória
FASE 5
Σ
MPC
10
10
BI 3 - barramento interno 3
B A R R A M E N T O 1
B A R R A M E N T O 2
16
16
SC 5
" 1 "
SC 4
R2
16
SC 13
SC 3
R1
16
SC 12
SC 2
Acc
16
SC 11
SC 1
PC
16
SC 10
16