



## Arquitectura de Computadores

Ano Lectivo de 2008/2009

2º Semestre

2º Teste

9 de Junho de 2009

Duração: 1h30+0h30

- O teste é sem consulta, apenas tem disponível o anexo que lhe deverá ter sido entregue com o teste. Por favor, não escreva nesse anexo e devolva-o no final do teste.
- Resolva o teste no próprio enunciado, o espaço reservado para cada pergunta é suficiente para a sua resposta. Tenha em atenção que cada grupo deve ficar em folhas separadas. Utilize as costas das folhas para rascunho.
- Identifique todas as folhas que entregar, **folhas não identificadas não serão cotadas!**
- Responda ao teste com calma. Se não sabe responder a uma pergunta, passe à seguinte e volte a ela no fim.

### I. (2 + 2 + 2,5 = 6,5 val.)

As perguntas deste grupo referem-se todas ao processador P3 e são independentes entre si.

1. Indique quais as micro-operações realizadas pela micro-instrução 76037799h.

|    |    |    |     |     |     |       |    |    |      |    |          |    |    |    |    |     |    |    |    |    |    |    |     |     |   |   |   |   |   |   |   |
|----|----|----|-----|-----|-----|-------|----|----|------|----|----------|----|----|----|----|-----|----|----|----|----|----|----|-----|-----|---|---|---|---|---|---|---|
| 31 | 30 | 29 | 28  | 27  | 26  | 25    | 24 | 23 | 22   | 21 | 20       | 19 | 18 | 17 | 16 | 15  | 14 | 13 | 12 | 11 | 10 | 9  | 8   | 7   | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| 0  | M5 |    | SR1 | SR2 | IAK | FM    |    |    | CALU |    |          |    | MA | MB | M2 | MRB | RB |    |    | WM | WR | MD | MAD | RAD |   |   |   |   |   |   |   |
| 1  |    |    |     |     | LS  | MCOND |    | CC | LI   | LF | CONST/NA |    |    |    |    |     |    |    |    |    |    |    |     |     |   |   |   |   |   |   |   |
|    |    |    |     |     |     |       |    |    |      |    |          |    |    |    |    |     |    |    |    |    |    |    |     |     |   |   |   |   |   |   |   |

#### Operações realizadas (em RTL):

|  |
|--|
|  |
|  |
|  |
|  |
|  |

#### NOTAS:

- a resposta é para ser dada na tabela à esquerda; utilize apenas as posições que considere necessárias
- a tabela em cima pode ser usada como rascunho

2. Considere que se pretende que as seguintes micro-operações se realizem num único ciclo de relógio:

$!c?CAR \leftarrow 100h, SBR \leftarrow CAR+1, SD \leftarrow 0F00h, RE \leftarrow SD$

Indique na tabela seguinte a codificação de uma micro-instrução que faz com que isso aconteça ou justifique que tal não é possível.

|    |    |    |     |     |     |       |    |    |      |    |          |    |    |    |    |     |    |    |    |    |    |    |     |     |   |   |   |   |   |   |   |
|----|----|----|-----|-----|-----|-------|----|----|------|----|----------|----|----|----|----|-----|----|----|----|----|----|----|-----|-----|---|---|---|---|---|---|---|
| 31 | 30 | 29 | 28  | 27  | 26  | 25    | 24 | 23 | 22   | 21 | 20       | 19 | 18 | 17 | 16 | 15  | 14 | 13 | 12 | 11 | 10 | 9  | 8   | 7   | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| 0  | M5 |    | SR1 | SR2 | IAK | FM    |    |    | CALU |    |          |    | MA | MB | M2 | MRB | RB |    |    | WM | WR | MD | MAD | RAD |   |   |   |   |   |   |   |
| 1  |    |    |     |     | LS  | MCOND |    | CC | LI   | LF | CONST/NA |    |    |    |    |     |    |    |    |    |    |    |     |     |   |   |   |   |   |   |   |
|    |    |    |     |     |     |       |    |    |      |    |          |    |    |    |    |     |    |    |    |    |    |    |     |     |   |   |   |   |   |   |   |

NOTA: Preencher com 0, 1 ou X para os sinais indiferentes.

Número: \_\_\_\_\_

Nome: \_\_\_\_\_

1/7

3. Pretende-se implementar no processador P3 a instrução *checksum*: **CHSUM op1, op2**. Esta instrução efectua a operação ou-exclusivo (XOR) sobre todas as palavras de 16 bits de um dado vector com **op2** elementos, armazenado em memória a partir do endereço dado por **op1**. Caso o número de elementos do vector (**op2**) seja positivo, a instrução deverá devolver, em **op1**, o valor complementar do *checksum* calculado. Caso contrário deverá devolver a constante **FFFFh**. Em qualquer dos casos, deverá ainda actualizar as flags Z e N, de acordo com o valor retornado em **op1**.

Ilustra-se abaixo uma implementação desta função na linguagem C, bem como uma possível invocação da rotina pretendida, na linguagem assembly:

```
short chsum (short addr, short length){
    short checksum=0;
    for(i=0; i<length; i++)
        checksum ^= MEM[addr+i]
    return(~checksum);
}
```

```
...
Data   STR  A2C4h, 2AC1h, 715Fh, 12ABh,
        4279h, 952Bh, FF02h, 21D5h
...
MOV    R2, DATA
CHSUM  R2, 8
```

Preencha a tabela com o micro-programa correspondente à fase de execução da instrução **CHSUM**. Deverá utilizar exclusivamente as micro-instruções disponibilizadas na tabela abaixo. Apenas deverá indicar na folha de resposta o número de cada uma das micro-instruções utilizadas.

|          |                 |
|----------|-----------------|
| CHSUM0:  |                 |
| CHSUM1:  | R8<-R0          |
| CHSUM2:  |                 |
| CHSUM3:  | c?CAR<-CHSUM11  |
| CHSUM4:  |                 |
| CHSUM5:  | z?CAR<-CHSUM11  |
| CHSUM6:  |                 |
| CHSUM7:  |                 |
| CHSUM8:  | RD<-RD+1        |
| CHSUM9:  |                 |
| CHSUM10: | !z?CAR<-CHSUM6  |
| CHSUM11: |                 |
| CHSUM12: | CAR<-WB, RD<-R8 |

NOTA: Nas posições livres indique o número da micro-instrução (de acordo com a lista abaixo).

| Nº | Micro-Instrução     |
|----|---------------------|
| 1  | CAR<-IF0            |
| 2  | CAR<-IF1            |
| 3  | CAR<-F1             |
| 4  | CAR<-F2             |
| 5  | SBR<-CAR+1, CAR<-F1 |
| 6  | SBR<-CAR+1, CAR<-F2 |
| 7  | SBR<-CAR+1, CAR<-WB |
| 8  | CAR<-IH0            |
| 9  | CAR<-WB             |
| 10 | CAR<-CAR+1          |
| 11 | R8<-R8 ⊕ R9         |
| 12 | R8<-R8 + R9         |
| 13 | R8<-R8 ∨ R9         |
| 14 | R8<-R8 ∧ R9         |
| 15 | R9<-R8 ⊕ R9         |
| 16 | R9<-R8 + R9         |
| 17 | R9<-R8 ∨ R9         |
| 18 | R9<-R8 ∧ R9         |

| Nº | Micro-Instrução     |
|----|---------------------|
| 19 | R8<-M[R9]           |
| 20 | R8<-M[EA]           |
| 21 | R8<-M[SD]           |
| 22 | R8<-M[RD]           |
| 23 | R9<-M[R8]           |
| 24 | R9<-M[EA]           |
| 25 | R9<-M[SD]           |
| 26 | R9<-M[RD]           |
| 27 | R8<- /R8            |
| 28 | R8<- /R9            |
| 29 | R9<- /R8            |
| 30 | R9<- /R9            |
| 31 | R8<- /R8, Flags Z,N |
| 32 | R8<- /R9, Flags Z,N |
| 33 | R9<- /R8, Flags Z,N |
| 34 | R9<- /R9, Flags Z,N |
| 35 | shl RD              |
| 36 | shr RD              |

| Nº | Micro-Instrução    |
|----|--------------------|
| 37 | shl SD             |
| 38 | shr SD             |
| 39 | shl SD, Flags C, Z |
| 40 | shl RD, Flags C, Z |
| 41 | RD<-shr RD         |
| 42 | RD<-shl RD         |
| 43 | SD<-shr SD         |
| 44 | SD<-shl SD         |
| 45 | RD<-R0             |
| 46 | RD<-RD+1           |
| 47 | RD<-RD-1           |
| 48 | RD<-RD+1, Flags Z  |
| 49 | RD<-RD-1, Flags Z  |
| 50 | SD<-SD+R0          |
| 51 | SD<-SD+1           |
| 52 | SD<-SD-1           |
| 53 | SD<-SD+1, Flags Z  |
| 54 | SD<-SD-1, Flags Z  |

II. (2,5<sub>1,5+0,3+0,7</sub> + 4<sub>0,5+1,5+1+1</sub> + 2<sub>0,4+0,4+0,4+0,4+0,4</sub> = 8,5 val.)

1. Considere um sistema com um processador que possui 16 linhas de endereço - de A15 (maior peso) a A0 (menor peso). Neste sistema existem 3 circuitos de memória:

- ROM M1 - Possui 16k palavras e ocupa a gama de endereços mais baixos.
- ROM M2 - Possui 8k palavras e ocupa a gama de endereços imediatamente a seguir à ocupada pela memória M1.
- RAM M3 - Possui 8k palavras e ocupa a gama de endereços com início em 8000h.

a) Indique quais as funções de descodificação destas memórias, MS - *Memory Select* (circule a letra correcta em cada).

| ROM M1                            | ROM M2                                       | RAM M3                                       |
|-----------------------------------|----------------------------------------------|----------------------------------------------|
| A: MS1 = $\overline{A15}.A14$     | A: MS2 = $A15.\overline{A14}.A13$            | A: MS3 = $\overline{A15}.A14.A13$            |
| B: MS1 = $A15.A14$                | B: MS2 = $\overline{A15}.A14.\overline{A13}$ | B: MS3 = $A15.\overline{A14}.\overline{A13}$ |
| C: MS1 = $\overline{A0}.A1$       | C: MS2 = $\overline{A0}.A1.\overline{A2}$    | C: MS3 = $A0.\overline{A1}.\overline{A2}$    |
| D: MS1 = $\overline{A15}.A14.A13$ | D: MS2 = $\overline{A15}.A14$                | D: MS3 = $A15.A14$                           |
| E: MS1 = $\overline{A0}.A1.A2$    | E: MS2 = $\overline{A0}.A1$                  | E: MS3 = $A0.A1$                             |
| F: Nenhuma destas                 | F: Nenhuma destas                            | F: Nenhuma destas                            |

b) Indique a dimensão do espaço de endereçamento não ocupado pelas memórias: \_\_\_\_\_

c) Considere um sistema desenvolvido com base no P3 que usa interrupções e que necessita de 7k palavras de memória para código e 3k palavras de memória para dados. Para realizar o sistema dispõe de memórias ROM de 4k palavras e memórias RAM de 16k palavras. Indique:

- O menor número de memórias ROM que necessita usar: \_\_\_\_\_
- O menor número de memórias RAM que necessita usar: \_\_\_\_\_
- A partir de que endereço colocaria a(s) memória(s) de código: \_\_\_\_\_
- A partir de que endereço colocaria a(s) memória(s) de dados: \_\_\_\_\_

2. Considere um processador como o P3, com um espaço de endereçamento de 64k palavras de 16 bits e uma cache associativa de 2 vias com capacidade de 2k palavras e blocos de 4 palavras. Na cache são usadas políticas de substituição LRU e de escrita *write-back* com *write-allocate*.

- a) Indique a largura em número de bits dos campos de etiqueta, índice e deslocamento da cache.

Etiqueta: \_\_\_\_\_ Índice: \_\_\_\_\_ Deslocamento: \_\_\_\_\_

- b) Considere que se executa o seguinte programa.

```

SIZE      EQU      1000h
MASK      EQU      000Fh

          ORIG      8000h
A         TAB      SIZE
B         TAB      SIZE

          ORIG      0000h
Loop:     MOV       R1, R0
          MOV       R2, M[R1+A]
          AND       R2, MASK
          ADD       M[R1+B], R2
          INC       R1
          CMP       R1, SIZE
          BR.NZ     Loop

```

Considere que a cache descrita acima é apenas usada para **dados** e que se executa o programa da alínea anterior. Indique qual é a taxa de sucesso (*hit rate*) na cache de dados.

|       |        |        |        |        |                   |
|-------|--------|--------|--------|--------|-------------------|
| A: 0% | B: 33% | C: 50% | D: 75% | E: 83% | F: Nenhuma destas |
|-------|--------|--------|--------|--------|-------------------|

- c) Analise o impacto da dimensão SIZE dos vectores A e B na taxa de sucesso na cache de dados e indique um valor de SIZE que permitiria melhorar esta taxa de sucesso.

|         |         |          |          |                              |                   |
|---------|---------|----------|----------|------------------------------|-------------------|
| A: 100h | B: 800h | C: 2000h | D: 4000h | E: SIZE não afecta esta taxa | F: Nenhuma destas |
|---------|---------|----------|----------|------------------------------|-------------------|

- d) Considere um sistema que possui memória primária e cache. O tempo de acesso à memória primária é de 100ns e o tempo de acesso à cache é de 20ns. Se a taxa de sucesso for de 75% indique o tempo médio para um acesso de leitura.

|         |         |         |         |         |                   |
|---------|---------|---------|---------|---------|-------------------|
| A: 95ns | B: 80ns | C: 75ns | D: 45ns | E: 20ns | F: Nenhuma destas |
|---------|---------|---------|---------|---------|-------------------|

3. Considere um sistema de memória virtual paginada. O espaço de endereçamento virtual é constituído por 16 páginas, cada uma com 32 palavras. O espaço de endereçamento físico é constituído por 8 páginas.

a) Indique o número de bits do campo de deslocamento de um endereço virtual: \_\_\_\_\_

b) Indique o espaço de endereçamento virtual: \_\_\_\_\_

c) Pretende-se aceder ao endereço virtual 13Ch.

Indique, em hexadecimal, o índice da tabela de páginas que será consultado: \_\_\_\_\_

d) Indique o número de bits guardados em cada entrada da tabela de páginas: \_\_\_\_\_  
(sem contabilizar os bits de controlo)

e) No acesso ao endereço virtual 13Ch foi lido da tabela de páginas o valor 3.

Indique, em hexadecimal, o endereço físico que será acedido: \_\_\_\_\_

**III. (1 + 1 + 1,5 + 1,5 = 5 val.)**

1. Numa transmissão de informação através de comunicação série assíncrona, primeiro é enviado o bit menos significativo de um valor de 8 bits, é usado 1 *stop* bit e paridade par. No receptor foi lida a seguinte sequência:

```
11111011110111111111000000000010101010111111
```

Indique, em hexadecimal, os valores recebidos e verifique se houve algum erro na transmissão. Justifique a sua resposta.

2. Numa transmissão série síncrona orientada ao bit, pretende-se transmitir a sequência

11111011110111111111100111111

Indique a sequência de bits a enviar (ignore a parte do cabeçalho e verificação de erros e assuma a *flag* habitual 01111110). (nota: utilize só as posições da tabela que considere necessárias)

[illegible]

3. Num sistema com DMA, indique a vantagem principal da existência de uma cache no sistema.

4. Tendo apenas em conta o ponto de vista do controlador DMA, indique justificando qual a política de escrita mais desejável para a cache do sistema.