Arquitectura de Computadores

2011/2012 2º Semestre

1º Teste (A) - 11/04/2012

Número: _____ Nome: _____

INSTRUÇÕES:

- A duração da prova é de 2 horas.
- Preencha imediatamente o seu número e nome de forma bem legível, em todas as folhas.
- Responda apenas no presente enunciado; não serão aceites folhas adicionais. Seja sucinto.
- A cotação das perguntas está indicada entre parênteses à direita do texto.
- Nas perguntas Verdade/Falso ou de escolha múltipla as respostas erradas descontam.

I

1. Considere os seguintes valores para os registos do processador P3:

R1	R2	R3	R4	R5	R6	R7	PC	SP	RE
0001h	8FD9h	0370h	41F0h	805Ah	ABBAh	0000h	0210h	FADEh	0014h

Nas alíneas a), b) e c), indique qual o novo valor, em hexadecimal, para todos (**e apenas**) os registos que são escritos na execução de cada instrução. Use **?** caso não tenha informação suficiente para determinar o novo valor de um registo.

As alíneas são independentes, isto é, assuma como valores iniciais para cada alínea os indicados na tabela acima.

a) DEC R3 **(1)**

R1	R2	R3	R4	R5	R6	R7	PC	SP	RE
		036Fh					0211h		0010h

b) RORC R2, 1 **(1)**

R1	R2	R3	R4	R5	R6	R7	PC	SP	RE
	C7ECh						0211h		0016h

c) RTI **(1)**

R1	R2	R3	R4	R5	R6	R7	PC	SP	RE
							?	FAE0	?

d) Para o ciclo completo de execução da instrução PUSH M[R6+20h] indique na tabela seguinte a sequência de acessos à memória, especificando o endereço, dados e tipo de acesso (**L**eitura/**E**scrita). **(2)**

Nota 1: A tabela tem 5 linhas, utilize apenas as que considerar necessárias.
Nota 2: Utilize os valores iniciais dos registos indicados na tabela no cimo desta página.
Nota 3: Use **?** para indicar que não tem informação suficiente para determinar um dado valor.

	Endereço	Dados	L / E
1	0210h	5036h	L
2	0211h	0020h	L
3	ABDAh	?	L
4	FADEh	?	E
5			

(página propositadamente em branco; pode usar para rascunho)

II

2.1 Considere o programa que se segue, o qual está incompleto. A funcionalidade da rotina "*Procura*" é verificar se um vector contém um determinado elemento. Se o vector contiver o elemento, a rotina devolve a posição de memória onde esse elemento se encontra. Caso contrário devolve o valor -1.

```
            NUM_DADOS   EQU     8
            NUM         EQU     5

                        ORIG    8000h
            Vector      STR     1, 3, 7, 2, 13, 5, 8, 21
            Pos         WORD    0

                        ORIG    0000h
                        ...

1:                      PUSH    NUM
2:                      PUSH    Vector
3:                      PUSH    NUM_DADOS
4:                      CALL    Procura
5:                      POP     M[Pos]

                        ...


6:          Procura:    PUSH    R1
7:                      PUSH    R2
8:                      MOV     R1, M[SP+5]
9:                      ADD     R1, M[SP+4]
10:                     DEC     R1
11:                     MOV     R2, M[SP+6]
12:         Ciclo:      CMP     R2, M[R1]
13:                     BR.NZ   Continua
14:                     BR      Termina
15:         Continua:   DEC     R1
16:                     CMP     R1, M[SP+5]
17:                     BR.NN   Ciclo
18:                     MOV     R1, R0
19:                     DEC     R1
20:         Termina:    MOV     M[SP+6], R1
21:                     POP     R2
22:                     POP     R1
23:                     RETN    2
```

a) Complete as linhas 8, 9, 10 e 11 tendo em conta que a passagem de parâmetros é feita pela pilha (ver linhas 1 a 3) e que: **(1,5)**

- R1 será usado para percorrer o vector do fim para o início.
- R2 contém o valor a procurar.

b) Preencha a linha 16 de modo a que o programa cumpra a funcionalidade desejada. **(0,5)**

c) Complete as linhas 5, 20 e 23 tendo em conta que a saída deve ser feita pela pilha e que após a execução da rotina o resultado deverá ser guardado na variável *Pos*. **(1,5)**

d) Indique o que teria de fazer entre as linhas 13 e 14 para que, em vez da posição de memória, a rotina devolvesse o índice do elemento procurado (isto é, a posição dentro do array – valor entre 0 e NUM_DADOS - 1). Apresente o código correspondente. **(1)**

SUB R1, M[SP+5]

e) Considere o programa tal como apresentado na página anterior e analise as linhas 13 e 14. Indique que código usaria para obter a mesma funcionalidade de forma mais eficiente. **(0,5)**

Linhas 13 e 14 → BR.Z Termina

III

3.1 Considere o seguinte programa em Assembly do P3. Considere que a rotina EscCont escreve na janela de texto o número que se encontra na posição de memória Contador.

```
SP_INICIAL      EQU     FDFFh
INT_MASK_ADDR   EQU     FFFAh
IO_WRITE        EQU     FFFEh

                ...

                ORIG    8000h
Contador        WORD    0000h

                ORIG    0000h
                JMP     Inicio

1:  Event:      PUSH    R1
2:              MOV     R1, 18h
3:              MOV     M[SP+3], R1
4:              POP     R1
5:              RTI

    ContHex:    INC     M[Contador]
                RET

                ...

    Inicio:     MOV     R7, SP_INICIAL
                MOV     SP, R7
                MOV     R7, INT_MASK
                MOV     M[INT_MASK_ADDR], R7
                ENI
    CicloCont:  CALL    EscCont
                INC     R0
    Espera:     BR.NZ   Espera
                CALL    ContHex
                BR      CicloCont
```

a) Indique como deve ser definida a tabela de vectores de interrupção por modo a que quando se carrega no botão I5 ("Janela Placa" do simulador) a rotina Event seja executada. **(1)**

```
                ORIG    FE05h
INT5            WORD    Event
```

b) Para as mesmas condições da alínea anterior, indique qual o valor, em hexadecimal, que deverá ter INT_MASK. **(0,5)**

0020 h

c) Descreva como funciona a rotina de interrupção Event e como ela afecta a execução do programa. **(1,5)**

Activa a flag Z (Z=1) o que faz com que o programa saia do ciclo de espera e incremente o contador e escreva o seu valor. _____

_____

_____

d) Pretende-se alterar o programa para que a rotina Event seja executada a cada 400ms.

i) Completar os valores a atribuir às seguintes constantes: **(0,5)**

```
TIMER_COUNT     EQU     FFF6h
TIMER_CTRL      EQU     FFF7h
TIMER_ENABLE    EQU     1
TIMER_VALUE     EQU     4
INT_MASK        EQU     8000h
```

ii) Indique como inicializava a tabela de vectores de interrupção **(0,5)**

```
                ORIG    FE0Fh
INT15           WORD    Event
```

iii) Indique que instruções acrescentaria na rotina Event. Use as linhas numeradas (de 1 a 5) para indicar entre que linhas colocaria as novas instruções. **(1)**

(Nota: Ignore outras alterações que porventura possam ser necessárias no programa)

Entre as linhas 1 e 2:

```
                MOV     R1, TIMER_VALUE
                MOV     M[TIMER_COUNT], R1
                MOV     R1, TIMER_ENABLE
                MOV     M[TIMER_CTRL], R1
```

IV

4.1 Considere o seguinte troço de código em Assembly do P3:

```
                ORIG    0010h

                XCH     R1, M[R2+15]
Wait:           JMP.N   Wait
```

Traduza estas instruções Assembly para o código objecto correspondente, indicando o endereço de memória e o respectivo conteúdo em hexadecimal. **(2)**

Nota 1: Tenha em atenção a base de representação das constantes.
Nota 2: Preencha apenas as linhas da tabela que considerar necessárias.

Endereço	Conteúdo
0010h	BA72h
0011h	000Fh
0012h	C520h
0013h	0012h

4.2 Considere um programa escrito em Assembly do P3 em que estão a ser usados números positivos e negativos (representados em complemento para dois) e em que se convencionou trabalhar em vírgula fixa com 4 casas decimais. Indique, caso seja possível, como representaria em hexadecimal o número **-12,125**. **(1)**

FF3Eh

4.3 Indique se as seguintes afirmações são verdadeiras ou falsas. **(2)**

Nota: Cada pergunta certa vale 0,5 valores; respostas erradas descontam 0,25 valores.
A classificação mínima desta pergunta é 0.

a) Se uma rotina de interrupção for terminada, por engano, com a instrução RET, verifica-se que a rotina retorna para a instrução correcta, tal como se tivesse sido usado RTI.

X Verdadeiro ☐ Falso

b) A instrução RETN 1 tem exactamente o mesmo efeito que RTI.

☐ Verdadeiro X Falso

c) A instrução JMP ocupa sempre duas palavras de memória.

☐ Verdadeiro X Falso

d) Após a instrução INC R0 é indiferente usar ADD ou ADDC.

X Verdadeiro ☐ Falso

(página propositadamente em branco; pode usar para rascunho)