

Regras:

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

R1	R2	R3	R4	R5	R6	R7	PC	SP	RE
0010h	EFD9h	0037h	1AFCh	A1FFh	0001h	EC3Fh	0093h	FB5Ah	000Ch

Para as perguntas a), b) e c), indique quais são os novos valores, em hexadecimal, de todos (e apenas) os registos que são escritos na execução de cada instrução. Use ? para indicar que não tem informação suficiente para determinar o novo valor de um registo.

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

1 Val.

(a) COM R4

R1	R2	R3	R4	R5	R6	R7	PC	SP	RE

1 Val.

(b) DIV M[R5], R1

R1	R2	R3	R4	R5	R6	R7	PC	SP	RE

1 Val.

(c) RTI

R1	R2	R3	R4	R5	R6	R7	PC	SP	RE

2 Vals.

(d) Para o ciclo completo de execução da instrução `RORC M[R2+20h],4` indique na tabela seguinte a sequência de acessos à memória, especificando o endereço, dados e tipo de acesso (Leitura/Escrita).

Nota 1: A tabela tem 5 posições, 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	Leitura/Escrita
1			
2			
3			
4			
5			

(Página propositadamente em branco, para rascunho)

II. Considere o programa que se segue, o qual está incompleto.

```

NUM_DADOS    ORIG    8000h
DADOS_IN     WORD    16
DADOS_OUT    STR     2, 7, -2, 0, -5, 4, 1, -1, 21, 8, 11, -3, 7, -2, 10, 5
SOMA         TAB     16
RESULT       TAB     1

                                ORIG    0000h
                                ...
1              MOV     M[SOMA], R0
2              MOV     R1, -----
3              MOV     R2, -----
4              MOV     R3, -----
5  CICLO:      MOV     R4, M[R2]
6              CMP     R4, R0
7              -----
8              NEG     R4
9  SKIP:       ADD     M[SOMA], R4
10             MOV     M[R3], R4
11             INC     R2
12             -----
13             -----
14             BR.NZ   CICLO
15             MOV     R4, M[SOMA]
16             -----
17             MOV     M[RESULT], R4
18  STOP:      BR      STOP

```

$\frac{1}{2}$ Val.

(a) Complete as linhas 2, 3 e 4 do programa tendo em conta que:

- R1 contém o número de dados a tratar (NUM_DADOS);
- R2 será usado como apontador para o vector DADOS_IN;
- R3 será usado como apontador para o vector DADOS_OUT.

$\frac{1}{2}$ Val.

(b) Preencha a linha 7 do programa considerando que se pretende saltar para a etiqueta SKIP caso o valor de R4 seja maior ou igual a 0.

1 Val.

(c) Indique que instruções deverão estar presentes nas linhas 12 e 13 do programa para cumprir a funcionalidade desejada.

1 Val.

(d) Considerando o conteúdo de DADOS_IN indique qual o conteúdo das primeiras 5 posições de DADOS_OUT quando o programa termina.

DADOS_OUT[0]	DADOS_OUT[1]	DADOS_OUT[2]	DADOS_OUT[3]	DADOS_OUT[4]

1 Val.

(e) Indique qual o conteúdo da variável SOMA quando o programa termina.

SOMA=-----

$\frac{1}{2}$ Val.

(f) Pretende-se que a variável RESULT contenha a média dos valores de DADOS_OUT. Preencha a linha 16 do programa para conseguir esse objectivo.

(Página propositadamente em branco, para rascunho)

III. Considere o seguinte excerto de um programa para o processador P3.

```

Basetime      EQU    0300
Timer_Units   EQU    FFF6h
Timer_Ctrl    EQU    FFF7h
Leds_Ctrl     EQU    FFF8h
Mask_Addr     EQU    FFFAh
Sp_Init       EQU    FFD0h

Var1          ORIG   8000h
              WORD   0000h

              ORIG   0000h
              JMP    Main

Timer:        ...
              MOV    R1, Basetime
              MOV    M[Timer_Units], R1
              MOV    R1, 1h
              MOV    M[Timer_Ctrl], R1
              CALL   PiscaLed
              RTI

TrataIO:      MOV    R1, 0001h
              XOR    M[Timer_Ctrl], R1
              RTI

Main:         MOV    R1, Sp_Init
              MOV    SP, R1
              CALL   InitInts      ; Interrupts Initialization
              CALL   Timer         ; Timer Initialization
              ENI

              ...
Halt:         BR     Halt

```

1½ Vals.

- (a) Considere que a rotina `InitInts` efectua a activação das interrupções 0 e 15. Desenvolva a respectiva rotina em código assembly, comentando as respectivas instruções. Efectue também as alterações necessárias ao código por forma a adicionar as rotinas de interrupção à tabela de vectores de interrupção.

1½ Vals.

- (b) A inicialização do temporizador é feita de modo incorrecto. Justifique a afirmação e indique eventuais consequências deste procedimento. Apresente uma pequena rotina de inicialização e as respectivas correcções na rotina **Main**.

1 Val.

- (c) Indique, justificando, qual a funcionalidade da rotina **TrataIO**.

1½ Vals.

- (d) Considere a rotina **PiscaLed**. Desenvolva em código Assembly a rotina que permita fazer piscar o LED menos significativo da placa de IO do processador P3.
Sugestão: pode utilizar a variável **Var1** para auxiliar o desenvolvimento desta rotina.

IV.

1½ Vals.

- (a) Considere o seguinte troço de código em Assembly do P3:

TABLEDIM	EQU	0020
ProcData:	ORIG	01A0h
	SHRA	M[R3+TABLEDIM], 12
	CALL.C	ProcData
	...	

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

Endereço	Conteúdo

Nota 1: Tenha em atenção a base de representação das constantes.

Nota 2: Preencha apenas as linhas da tabela que considerar necessárias.

1 Val.

- (b) Numa futura evolução do processador P3 foi considerada a hipótese de estender o banco de registos para 32 registos (em vez dos actuais 16 registos), mantendo-se a codificação das instruções com palavras de 16 bits. Enumere um exemplo concreto que ilustre bem uma vantagem desta opção e outro exemplo que demonstre uma desvantagem desta decisão.

- (c) Qual dos programas em *Assembly* corresponde à melhor compilação de cada um dos seguintes troços de programa em C. Considere que não ocorre *overflow* nas operações e que o tipo de dados `short int` é representado com 16-bits.

1/2 Val.

```
#define N 3
short int a, b, c, m;

m=(a+b+c)*N
```

A:	B:	C:	D:	E:
MOV R1, M[a] ADD R1, M[b] ADD R1, M[c] MUL R1, 3 MOV M[m], R1	MOV M[m], M[a] ADD M[m], M[b] ADD M[m], M[c] MUL M[m], 3	MOV R1, M[a] ADD R1, M[b] ADD R1, M[c] MOV M[m], R1 ADD R1, R1 ADD M[m], R1	ADD R1, M[a] ADD R1, M[b] ADD R1, M[c] MOV M[m], R1 ADD R1, R1 ADD M[m], R1	Nenhuma das anteriores

1/2 Val.

```
#define N 20
register short int s, i;
short int a[N];

s=0;
for (i=0; i<N; i++)
    s=s+a[i];
```

A:	B:	C:	D:	E:
MOV M[s], R0 MOV R1, R0 E1: CMP R1, 20 BR.Z E2 ADD M[s], M[R1+a] INC R1 BR E1 E2:	MOV R1, R0 MOV R2, R0 E1: CMP R2, 20 BR.NN E2 ADD R1, M[R2+a] INC R2 BR E1 E2:	MOV R1, R0 MOV R2, R0 E1: CMP R2, 20 BR.P E2 ADD R2, M[R1+a] INC R2 BR E1 E2:	MOV s, R0 MOV i, R0 E1: CMP i, 20 BR.NN E2 ADD s, a[i] INC i BR E1 E2:	Nenhuma das anteriores

1 1/2 Vals.

- (d) Indique se as seguintes afirmações são verdadeiras ou falsas.

Tenha em atenção que:

- Cada pergunta certa vale +0,5 valores;
- Cada pergunta errada vale -0,25 valores;
- A cotação mínima da pergunta é 0.

- i. No processador P3, a activação da flag C na sequência de uma operação aritmética indica que o respectivo resultado ultrapassou a gama de representação permitida.

☐

Verdadeiro

☐

Falso

- ii. Logo após a invocação de uma rotina, a posição de memória M[SP] armazena o endereço de retorno para onde o programa deve retornar após a execução da instrução RET.

☐

Verdadeiro

☐

Falso

- iii. Se o registo R7 tiver o valor 8000h, então a instrução XOR R1, R7 é equivalente à instrução NEG R1.

☐

Verdadeiro

☐

Falso