ARQUITETURA DE COMPUTADORES 21010

Pretende-se um programa que faça procuras num conjunto de dados. O conjunto de dados deve ser colocado na zona inicial do programa, de modo a poder facilmente ser trocado por outro:

```
; Conjunto de dados
ORIG            8000h
Input           STR     60385, 8536, 7627, 24376, 49543, 13210, 9866, 53798
linha2          STR     60526, 42549, 27769, 25234, 60984, 49533, 45468, 26128
linha3          STR     56370, 34983, 13129, 62576, 62248, 63856, 57035, 20429
Tamanho         EQU     24
```

Neste caso o conjunto de dados é constituído por 24 números de 16 bits cada.

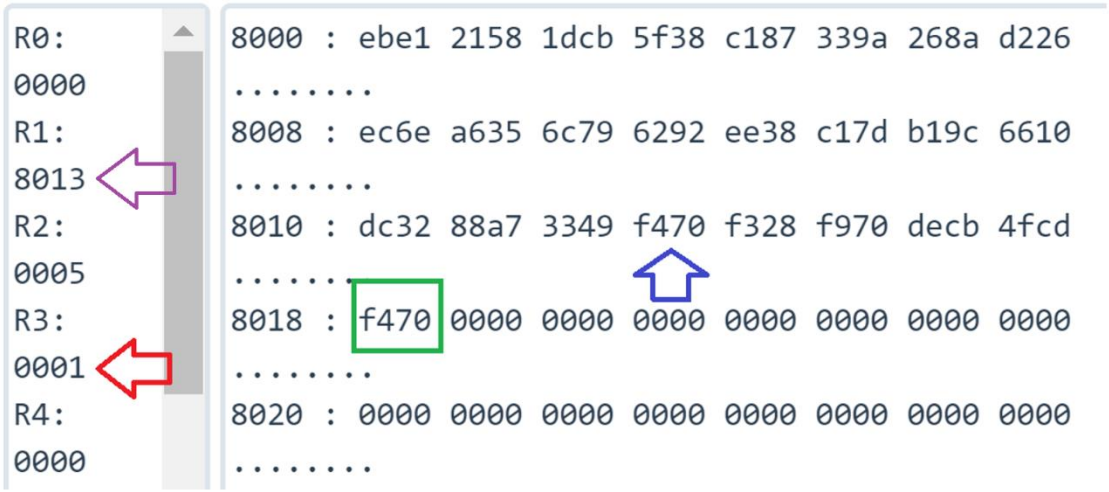
a) **[1 valor]** Faça um programa no Assembly do P3 que indique se existe um número na sequência de dados, devendo deixar no registo R3 o valor 1 no caso positivo, e o valor 0 caso contrário. Caso o resultado seja positivo, deve retornar em R1 o endereço onde foi localizado o número. O número a localizar deve ser colocado no início do programa, após o conjunto de dados, da seguinte forma:

```
Localizar       WORD    62576
```

Execute o programa para o caso anterior, e também para os números 2355, 18933 e 27765.

No relatório, nesta e nas alíneas seguintes, deve executar o programa após colocar um *breakpoint* no final deste, e indicar para cada uma das execuções o número de ciclos de relógio, e número de instruções executadas, bem como o resultado da execução. No caso desta alínea o resultado da execução é o valor do registo R3, e no caso positivo, no registo R1 ficará o endereço do número localizado. Pode efetuar uma captura de ecrã do simulador de modo a reportar o resultado final.

Revela-se o resultado de um caso de teste por alínea, de modo a facilitar a compreensão do que é solicitado. Na figura é possível ver a vermelho o resultado do registo R3, a roxo o resultado do registo R1, a verde a posição de memória com o valor a localizar, e a azul o local no conjunto de dados onde se encontra o valor. Neste caso o resultado é 1.



Alerta-se para o interesse em utilizar rotinas, logo nesta alínea e principalmente nas seguintes, em que a elaboração de uma ou duas rotinas simplifica consideravelmente o trabalho, e é valorizado no critério de avaliação "Simplicidade e Modularidade".

b) **[1 valor]** Pretende-se agora que calcule quantos bits são distintos entre o número a localizar (todos os casos: 62576, 2355, 18933 e 27765), e todos os restantes números do conjunto de dados. O resultado desta operação deve substituir os próprios números em memória.

Pode-se ver na imagem o resultado para a posição 8010h, onde se encontrava o valor dc32h. A diferença de número de bits para f470h é de 4 bits, um bit de diferença em cada dígito hexadecimal (f/d, c/4, 7/3, 2/0).

```
8010 : 0004
........
8018 : f470
........
```

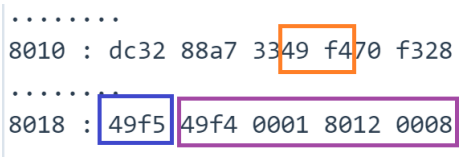
c) **[1 valor]** Pretende-se nesta alínea que identifique uma sequência de 16 bits seguidos do conjunto de dados, os quais podem estar em duas posições de memória (ex. sequência a iniciar 4 bits após o início da posição 8001h, e utilizando os

primeiros 4 bits da posição seguinte, a 8002h). Esta sequência de bits tem a particularidade de ter menos bits distintos para a palavra a localizar. Utilize aqui os casos de teste das alíneas anteriores: 62576, 2355, 18933 e 27765. Deve colocar os resultados na memória, após a palavra "Localizar":

```
Melhor          WORD    0
MelhorV         WORD    17
Posicao         WORD    Input
BitsPosicao     WORD    0
```

Na palavra "Melhor" deve ser registada a sequência de 16 bits que mais se aproxima da palavra "Localizar", em termos de bits distintos. Em "MelhorV" deve constar o número de bits distintos. Em "Posicao" deve ser indicada a posição de memória onde a sequência se inicia. Em "BitsPosicao" deve ser indicado o número de bits para o início da sequência.

Na imagem é visível o valor utilizado 18933 (49f5h) a azul, ficando a laranja a sequência de bits mais próxima (distância de 1 bit), que para felicidade nossa é visível em hexadecimal dado que a sequência inicia-se num bit múltiplo de 4. Na zona a roxo encontram-se as variáveis declaradas por ordem, primeiramente a própria sequência de 16 bits, 49f4h, seguida do número de bits de diferença, 1, a posição onde se encontra, 8012h, e número de bits em que se inicia nessa posição de memória, 8 bits. Significa que há exatamente dois dígitos hexadecimais que se têm de saltar, iniciando-se a sequência no bit seguinte.



d) **[1 valor]** Generalize agora a palavra a localizar para um conjunto de dados, como indicado de seguida:

```
Localizar       STR     4531h, ABC2h, 23ADh, 1010h, A531h, 2BC2h, 27ADh, 1610h
TamanhoL        EQU     8
```

Pretende-se que, para cada número K de 16 bits na string "Localizar", identifique os 16 bits de acordo com a alínea C, que mais se aproximam de K, e substitua o número K por esse valor.

Como resultado, pode-se ver numa imagem de parte da memória o vetor "Localizar" carregado para a memória:

```
........
8018 : 4531 abc2 23ad 1010 a531 2bc2 27ad 1610
```

Após a execução do programa, pode-se ver a primeira posição do vetor substituída pela sequência de bits do conjunto de dados mais próxima:

```
........
8018 : 4539
```

Neste caso é visível que a sequência encontrada difere apenas 1 bit (9/1).

BOM TRABALHO!

Regras para entrega do e-fólio B:

Forma de entrega:

Um ficheiro zip (extensão .zip e não outra como .rar) com o nome correspondente ao número de aluno.

O ficheiro zipado deverá por sua vez conter um ficheiro de Assembly do P3 por alínea, cujo nome será o número de aluno mais a letra da alínea ex: 999999a.as.

Deverá ainda ser entregue um pequeno relatório em formato pdf de até 5 páginas A4, com todos os cálculos e todas as opções tomadas na construção dos programas.

Regras relativas à escrita dos programas de Assembly:

Todos os programas devem acabar com a seguinte instrução:

```
FIM:  JMP FIM          ;Fim do programa
```

Não são aceites entregas fora da plataforma Moodle.

Avaliação

Cotação:

A cotação encontra-se junto de cada uma das alíneas, entre [].

Critérios de Correcção:

Funcionalidade: 50%

Simplicidade e Modularidade: 10%

Eficiência (serão contabilizados o número de instruções e ciclos de relógio): 10%

Apresentação do código (indentação e comentários): 20%

Relatório (Legibilidade e Justificação dos Resultados e das Opções): 10%

Descontos:

Trabalhos entregues que não estejam em conformidade com as regras de entrega do e-fólio B: até 10%

Código sem comentários, ou apenas com comentários a refletir o significado da instrução (exemplo MOV R1,R2 ;mover o conteúdo de R2 para R1) : até 50%

Deteção de fraude (total ou parcial): 100%

Trabalhos entregues após a data limite (o recurso de entrega estará aberto até ao início da manhã do dia seguinte, no entanto entregas após a data/hora indicada, têm desconto na nota) : 10%