



UNIDADE CURRICULAR: Arquitetura de Computadores

CÓDIGO: 21010

DOCENTE: José Coelho

A preencher pelo estudante

NOME: Yrma Marina Vianez Martins

N.º DE ESTUDANTE: 2300212

CURSO: Engenharia Informática

Autoavaliação: 3.75

CrITÉrios	Alínea A	Alínea B	Alínea C	Alínea D
Funcionalidade (50%)	0.5	0.5	0.5	0.5
Simplicidade e Modularidade (10%)	0.2			
Eficiência (10%)	0.2			
Apresentação (10%)	0.35			
Relatório (20%)	1			

Auto-avaliação de acordo com o avaliador: +0.1 na nota do e-fólio

CrITÉrios de correção:

- Funcionalidade: penalizações de funcionamento incorreto divulgadas no lançamento de notas
- Simplicidade e Modularidade: utilização correta de funções e código simples
- Eficiência: o número de ciclos de relógio e instruções for inferior a um valor de referência
- Apresentação: indentação correta; comentários
- Relatório: opções tomadas bem especificadas e justificadas

TRABALHO / RESOLUÇÃO:

Alínea A

O objetivo da sub-rotina é extrair, o código de uma carta (em binário, mas representado em decimal) a partir de 1 input em decimal.

```
;;;;;;;;;;;;; subrotina A ;;;;;;;;;;;;;;

; Extrair código do operador (bits mais significativos)

alineaA:    MOV R2, R1          ; Move R1 para R2 para não alterar o input;
            SHR R2, 4          ; Desloca 4 bits para a direita

            ; Extrair código do número (bits do meio)

            MOV R3, R1          ; Move R1 para R3 para não alterar o input
            SHR R3, 2          ; Desloca 2 bits para a direita
            AND R3, 3          ; Zera os bits mais a esquerda deixando apenas os 2 bits + significativos

            ; Extrair código da letra (bits menos significativos)

            MOV R4, R1          ; Move R1 para R4 para não alterar o input
            AND R4, 3          ; Zera os bits mais a esquerda deixando apenas os 2 bits + significativos
            RET                ; Retornar da sub-rotina

;;;;;;;;;;;;;
```

O código começa por dar um valor a R1, que será a carta que precisamos decodificar, esse valor irá ficar sempre em R1, para não alterar esse valor este é deslocado com o comando MOV para outros registradores (R2, R3 e R4) onde manipula os respetivos bits necessários.

Para extrair os bits mais significativos (que representa o operador) bastou usar o comando SHR 4 no valor de R1 movido para R2. Como o input tem 6 dígitos em binário, este comando vai deslocar os 2bits mais significativos 4 bits para a posição dos 2bits menos significativos. R2 fica então com o código correto para o operador.

Para extrair os 2 bits do meio para R3, que representam o código do número, usei a mesma logica, primeiro desloquei 2bits com o operador SHR 2, fico, portanto, com 4 bits, e depois usei o operador AND 3 (0011b), que vai realizar operação logicas AND bit a bit, portanto tudo o que estiver nos bits correspondentes aos bits de valor 0 do 0011bvai zerar também. Retornando apenas os 2 bits menos significativos, que correspondem aos bits do meio do valor do input que corresponde por sua vez ao código do número da carta, que fica alojado no R3.

Para o código da letra preciso dos bits menos significativos, portanto basta usar o AND 3(000011b), vai zerar todos os 4 bits exceto os 2 últimos bits. Portanto o R4 fica com 2 bits menos significativos do input. Por fim com o comando RET, retorna á posição seguinte de onde foi chamada a sub-rotina.

Alínea B

O objetivo desta sub-rotina é verificar a validade das cartas fornecidas pelo programa de testes e contar a frequência de cada elemento (operador, número e letra) em uma lista de cartas.

```
;;;;;;;;;;;;; sub rotina B;;;;;;;;;;;;;

AlineaB:      MOV R6, R1
proximacarta: MOV R1, M[R6]          ; Passa para R1 a carta que esta na posicao de memoria da lista mais a R7 que é a conta cartas
              CMP R1, 64             ; Compara com 64 para verificar se é carta válida
              BR.N Descod            ; Se R1-64 < 0 a carta é valida,
              RET                    ; Retorna se R1-64 > 0

Descod:       CALL alineaA           ; Usa a subrotina alineaA para descodificar a carta

Operador:     CMP R2, R5              ; Compara R2(codigo do operador) com R5 (iteradora)
              BR.Z memOperador       ; Se for igual, incrementa a frequência do operador
              INC R5                 ; Incrementa o iterador
              BR Operador            ; Volta a 'Operador'

Numero:       CMP R3, R5              ; Compara R3(codigo do numero) com a iteradora
              BR.Z memNumero         ; Se for igual, incrementa a frequência do número
              INC R5                 ; Incrementa o iterador
              BR Numero              ; Volta a 'Numero'

Letra:        CMP R4, R5              ; Compara R4(codigo da Letra) com a iteradora
              BR.Z memLetra          ; Se for igual, incrementa a frequência da letra
              INC R5                 ; Incrementa o iterador
              BR Letra               ; Volta a 'Letra'

memOperador:  INC M[R2+FreqOperador]  ; Incrementa a frequência do operador
              MOV R5, R0              ; Reinicia o iterador
              BR Numero               ; Salta para a próxima comparação

memNumero:    INC M[R3+FreqNumero]    ; Incrementa a frequência da letra
              MOV R5, R0              ; Reinicia o iterador
              BR Letra                ; Incrementa o contador de cartas

memLetra:     INC M[R4+FreqLetra]     ; Incrementa a frequência da letra
              MOV R5, R0              ; Reinicia o iterador
              INC R6
              BR proximacarta

;;;;;;;;;;;;;
```

Começo por mover para R1 a carta que está na posição de memória da Lista1 que é fornecida pelo programa de testes e somo a essa posição o R7, que vai funcionar como um registo iterador para a lista de cartas. Como este registo começa a 0 a primeira carta é chamada para R1, de seguida compara a carta com 64 para verificar se a carta é válida, se for válida vai descodificar a carta caso contrário, irá retornar ao local onde a sub-rotina foi chamada.

Após descodificar compara R2 (código do operador) com R5 que funciona como um único iterador para percorrer os códigos de todos elementos quando forem iguais, incrementa a frequência do operador e reinicia o iterador. Para isso a fórmula 'FreqOperador + R2' como em R2 irá estar sempre um valor entre 0 e 3 vai corresponder precisamente ao local correspondente a cada código de cada elemento. O mesmo processo é repetido para os códigos de número (R3) e letra (R4), incrementando suas respectivas frequências. Após comparar os três elementos da carta, incrementa-se o iterador R7 para avançar para a próxima carta na lista.

O ciclo persiste até encontrar uma carta não válida, momento em que a sub-rotina retorna ao ponto de chamada.

Esta sub-rotina B poderia provavelmente ser mais otimizada com o uso da pilha, mas apenas compreendi o funcionamento da pilha na realização da alínea D e depois não tive tempo para rever esta sub-rotina.

Alínea C

Esta sub-rotina tem como objetivo verificar a ocorrência de elementos repetidos (operador/número/letra) entre duas cartas fornecidas pelo programa de testes.

```
;;;;;;;;;;;;; subrotina C ;;;;;;;;;;;;;;

alineaC:      MOV R7, R2          ; Guarda o valor da segunda carta
              CALL alineaA       ; Descodifica a primeira carta
              MOV R1, R7
              MOV R5, R2          ; Mover o valor do operador da 1ªcarta
              MOV R6, R3          ; Mover o valor do número da 1ªcarta
              MOV R7, R4          ; Mover o valor do letra da 1ªcarta
              CALL alineaA       ; Descodifica a segunda carta
              CMP R5, R2          ; Compara os operadores
              BR.Z igual
              CMP R6, R3          ; Compara os números
              BR.Z igual
              CMP R7, R4          ; Compara as letras
              BR.Z igual
              MOV R3, R0          ; Coloca o valor 0 em R3

              RET

igual:        MOV R3, 1           ;coloca o valor de 1 em R3
              RET
```

Começo por guardar o input da segunda carta (R2) em R7, de modo a preservar a segunda carta para executar corretamente sub-rotina da alínea A. Em seguida, chama a sub-rotina da alínea A, responsável por decodificar a primeira carta, retornando com os valores R2, R3 e R4, com o código da carta 1.

Apos a descodificação da primeira carta desloca para R1 a segunda carta e de seguida transfiro os valores de R2, R3 e R4 para R5, R6 e R7, respetivamente, liberando assim os registos necessários para a decodificação da segunda carta que vou descodificar de seguida fazendo CALL á sub-rotina alínea A novamente. Neste ponto tenho então os códigos correspondentes dos elementos nas posições R2, R5 (operador) R3, R6 (número) R4 e R7 (letra). Agora, procedemos à comparação elemento a elemento. Se forem iguais, a execução é desviada para a etiqueta Igual, onde definimos R3 como 1 indicando a igualdade de 1 elemento nas duas cartas de teste, antes de retornar ao ponto de chamada da sub-rotina. Caso contrário, a comparação continua até que todos os elementos sejam verificados. Se não for encontrado nenhum correspondente, R3 é definido como 0, indicando a ausência de igualdade, e a execução retorna ao ponto de chamada da sub-rotina.

Alínea D

Esta sub-rotina tem como objetivo contar quantos pares de cartas têm um tipo igual numa lista de código de cartas.

```
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;subrotina D;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

alineaD:      PUSH R1                ; Move para stack 1ª a lista
              MOV R2, M[R1]          ; R2 fica com a 1ª carta da lista
              MOV M[8081h], R1        ; Guarda a posicao donde começa a lista

              ;;; Etiqueta para escolher a próxima carta na lista ;;

compara:      INC R1                 ; Avança para a próxima carta a comparar
              MOV R1, M[R1]          ; R1 fica com a próxima carta da lista
              CMP R1, 64              ; Verificar se é carta válida
              BR.NN proxima          ; Se R1-64 < 0, a carta é válida
              CALL alineaC           ; Descodifica e compara as cartas e retorna saída em R3
              CMP R3, 1              ; Verifica se houve um tipo igual
              BR.NZ loop              ; Se sim, salta para o contador
              INC M[8080h]            ; Incrementa ocontador de tipos iguais
loop:         POP R1                 ; Restaura a posição de memória da primeira carta da lista (8028)
              INC R1                 ; Aponta para a próxima carta na lista
              PUSH R1                ; Guarda esse valor na pilha
              MOV R2, M[8081h]        ; R2 fica com a posição de memoria primeira carta da lista
              MOV R2, M[R2]           ; R2 fica com a próxima carta da lista
              BR compara              ; Volta para o início do loop

              ;;;; para avançar para a próxima carta a comparar com a lista

proxima:      INC M[8081h]            ; Atualiza a lista para a próxima carta
              MOV R2, M[8081h]        ; R2 fica com a próxima carta da lista
              POP R0                  ; descarta 1 registo da pilha
              PUSH R2                ; actualiza o início da pilha
              MOV R1, R2              ; Atualiza R1 a posicao da proxima carta
              MOV R2, M[R2]           ; R2 fica com a próxima carta da lista
              CMP R2, 64              ; Verificar se é uma carta válida
              BR.NN fimdalista        ; Se carta invalida não há mais cartas para comparar
              BR compara              ; Se carta valida volta para comparar

fimdalista:   MOV R1, M[8080h]        ; R1 fica com a quantidade total de tipos iguais
              MOV M[8080h], R0        ; Reinicia o contador de tipos iguais
              POP R0                  ; Descarta o valor guardado na pilha
              RET                     ; Retorna da sub-rotina

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
```

Resumidamente o que fiz foi comparar cada carta da lista com todas as cartas seguintes da lista usando a sub-rotina da alínea c, e contar cada vez que essa sub-rotina devolvia o valor 1 na saída. Utilizei a pilha para rastrear a próxima carta a ser comparada. Devido à falta de registos livres, usei duas posições de memória (8080 e 8081). A posição 8080 foi usada como contador de cartas com tipo repetido, e a 8081 guardou a posição da primeira carta na comparação (R2). Fiz uma tabela para me auxiliar a verificar os resultados que se encontra nos Anexos.

Detalhadamente começo por guardar a primeira posição da lista na pilha e na memória (8081h), e preparo a primeira carta que vai comparar com a lista (sempre em R2), e a posição da primeira carta a comparar e isto só ocorre para primeira carta.

Após isso passa para a comparação com o resto das cartas, primeiro incrementa a posição de memória da carta seguinte e posiciona a carta a qual vai ser comparada

(sempre R1), e faz uma verificação se a carta é válida, caso não seja significa que já comparou com todas as cartas da lista, caso seja válida chama a sub-rotina da alínea C para verificar se tem algum tipo igual, se R3 retornar com valor 1 irá incrementar na posição de memória 8080 que é 1 contador de cartas repetidas. Após isso é preciso atualizar o valor da carta seguinte a comparar, para isso faço pop do valor atual, aumento e guardo o valor novamente na pilha, e coloco em R2 a carta que vai comparar ao resto da lista que esta sempre guardada em 8081h aí volta a repetir o ciclo até encontrar uma carta que seja inválida.

Quando é inválida significa que já comparou com a lista toda, aí vai atualizar o valor de R2 para isso incremento a posição de memória guardada em 8081 que é sempre a posição de memória onde está o meu R2, substituo o valor na pilha por esse valor e preparo a próxima carta a comparar. Isto vai se repetir até que R2 tenha valor inválido.

Aí movo para R2 o valor do meu contador de tipos repetidos, faço 1 pop para descartar o valor da pilha e retorna ao programa de testes.

ANEXOS

Resultado dos testes do programa

Zona de memória solicitada:

clock: 433,737

Instructions: 44,537

A	8000	:	0001	0001	0003	0000	0000	0000	0000	0000
	8008	:	0003	0002	0001	0000	0000	0000	0000	0000
B	8010	:	0007	0002	0005	0001	0006	0001	0001	0007
	8018	:	0005	0002	0003	0005	0000	0000	0001	0001	... C...
D	8020	:	0045	0001	00c1	0140	0000	0000	0000	0000

Testes alínea a

Entrada		Saída		
R1	R2	R3	R4	
23	1	1	3	8000 : 0001 0001 0003
57	3	2	1	8008 : 0003 0002 0001
1	0	0	1	8000 : 0000 0000 0001
13	0	3	1	8008 : 0000 0003 0001
30	1	3	2	8000 : 0001 0003 0002
40	2	2	0	8008 : 0002 0002 0000

Testes alínea b

Entrada		Saída												
	R1	FreqOperador				FreqNumero				FreqLetra				
		+	-	*	/	0	1	2	3	A	B	C	D	
Lista 1:	12; 48; 35; 16; 3; 14; 45; 11 47; 34; 15; 2; 13; 44; 20; 100	7	2	5	1	6	1	1	7	5	2	3	5	8010 : 0007 0002 0005 0001 0006 0001 0001 0007 8018 : 0005 0002 0003 0005 0000 0000 0001 0001
Lista 2:	13,40,31,100	1	1	1	0	0	0	1	2	1	1	0	1	8010 : 0001 0001 0001 0000 0000 0000 0001 0002 8018 : 0001 0001 0000 0001 0000 0000 0001 0001
Lista 3:	4140,63,16,58,54,39,23, 3, 53 2,16,39,6,62,39,17,216,17,0 24,57,43,51,31,100	6	8	6	7	9	9	6	3	5	7	6	9	8010 : 0006 0008 0006 0007 0009 0009 0006 0003 8018 : 0005 0007 0006 0009 0000 0000 0001 0001
Lista 4:	4156,52,17,12,22,4,8,3,6,57,60 9,37,55,27,62,4,26,20,42,52, 11 0,28,6,38,18,10,16,36,61,36,100	11	8	6	8	5	13	10	5	14	6	9	4	8010 : 000b 0008 0006 0008 0005 000d 000a 0005 8018 : 000e 0006 0009 0004 0000 0000 0001 0001

Testes Alínea C:

Entrada		Saída
R1	R2	R3
13	40	0
40	31	0
13	31	1
55	22	1

8018 : 0000 0000 0000 0000 0000 0000 0001 0001

Entrada		Saída
R1	R2	R3
0	63	0
11	19	1
23	32	0
15	51	1

8018 : 0000 0000 0000 0000 0000 0001 0000 0001

Testes alínea D:

Entrada: R1		Saída R1	
		decimal	hexa
Lista 1:	12; 48; 35; 16; 3; 14; 45; 11 47; 34; 15; 2; 13; 44; 20; 100	69	45
Lista 2:	13,40,31,100	1	1
Lista 3:	41,40,63,16,58,54,39,23, 3, 53 2,16,39,6,62,39,1,17,21,6,17,0 24,57,43,51,31,100	193	c1
Lista 4:	41,56,52,17,12,22,4,8,3,6,57,60 9,37,55,27,62,4,26,20,42,52, 11 0,28,6,38,18,10,16,36,61,36,100	320	140

8020 : 0045 0001 00c1 0140

Tabelas auxiliares

Tabela de conversão de todas as cartas						
número input		código				
decimal	hexa	R2	R3	R4	Cartas	Binário
0	0	0	0	0	+ 1 A	0 0 0 0
1	1	0	0	1	+ 1 B	0 0 0 1
2	2	0	0	2	+ 1 C	0 0 0 1
3	3	0	0	3	+ 1 D	0 0 0 1
4	4	0	1	0	+ 2 A	0 0 0 1
5	5	0	1	1	+ 2 B	0 0 0 1
6	6	0	1	2	+ 2 C	0 0 0 1
7	7	0	1	3	+ 2 D	0 0 0 1
8	8	0	2	0	+ 3 A	0 0 1 0
9	9	0	2	1	+ 3 B	0 0 1 0
10	a	0	2	2	+ 3 C	0 0 1 0
11	b	0	2	3	+ 3 D	0 0 1 0
12	c	0	3	0	+ 4 A	0 0 1 1
13	d	0	3	1	+ 4 B	0 0 1 1
14	e	0	3	2	+ 4 C	0 0 1 1
15	f	0	3	3	+ 4 D	0 0 1 1
16	10	1	0	0	- 1 A	0 1 0 0
17	11	1	0	1	- 1 B	0 1 0 0
18	12	1	0	2	- 1 C	0 1 0 0
19	13	1	0	3	- 1 D	0 1 0 0
20	14	1	1	0	- 2 A	0 1 0 1
21	15	1	1	1	- 2 B	0 1 0 1
22	16	1	1	2	- 2 C	0 1 0 1
23	17	1	1	3	- 2 D	0 1 0 1
24	18	1	2	0	- 3 A	0 1 1 0
25	19	1	2	1	- 3 B	0 1 1 0
26	1a	1	2	2	- 3 C	0 1 1 0
27	1b	1	2	3	- 3 D	0 1 1 0
28	1c	1	3	0	- 4 A	0 1 1 1
29	1d	1	3	1	- 4 B	0 1 1 1
30	1e	1	3	2	- 4 C	0 1 1 1
31	1f	1	3	3	- 4 D	0 1 1 1
32	20	2	0	0	* 1 A	1 0 0 0
33	21	2	0	1	* 1 B	1 0 0 0
34	22	2	0	2	* 1 C	1 0 0 0
35	23	2	0	3	* 1 D	1 0 0 0
36	24	2	1	0	* 2 A	1 0 0 1
37	25	2	1	1	* 2 B	1 0 0 1
38	26	2	1	2	* 2 C	1 0 0 1
39	27	2	1	3	* 2 D	1 0 0 1
40	28	2	2	0	* 3 A	1 0 1 0
41	29	2	2	1	* 3 B	1 0 1 0
42	2a	2	2	2	* 3 C	1 0 1 0
43	2b	2	2	3	* 3 D	1 0 1 0
44	2c	2	3	0	* 4 A	1 0 1 1
45	2d	2	3	1	* 4 B	1 0 1 1
46	2e	2	3	2	* 4 C	1 0 1 1
47	2f	2	3	3	* 4 D	1 0 1 1
48	30	3	0	0	/ 1 A	1 1 0 0
49	31	3	0	1	/ 1 B	1 1 0 0
50	32	3	0	2	/ 1 C	1 1 0 0
51	33	3	0	3	/ 1 D	1 1 0 0
52	34	3	1	0	/ 2 A	1 1 0 1
53	35	3	1	1	/ 2 B	1 1 0 1
54	36	3	1	2	/ 2 C	1 1 0 1
55	37	3	1	3	/ 2 D	1 1 0 1
56	38	3	2	0	/ 3 A	1 1 1 0
57	39	3	2	1	/ 3 B	1 1 1 0
58	3a	3	2	2	/ 3 C	1 1 1 0
59	3b	3	2	3	/ 3 D	1 1 1 0
60	3c	3	3	0	/ 4 A	1 1 1 1
61	3d	3	3	1	/ 4 B	1 1 1 1
62	3e	3	3	2	/ 4 C	1 1 1 1
63	3f	3	3	3	/ 4 D	1 1 1 1

número a comparar		compara com decimal hexa		Tabela auxiliar alinea D - LISTA1												Total
		12	c	-	-	-	-	-	-	-	-	-	-	-	-	-
12	48	30	100	3	32	231	23	233	33	2	31	230	110	12	7	7
48	35	23	100	3	202	2	230	110	-	-	-	-	-	-	-	9
35	16	10	3	231	23	233	202	33	2	230	-	-	-	-	-	5
16	3	3	202	2	230	110	-	-	-	-	-	-	-	-	-	7
3	14	e	23	233	202	33	2	31	-	-	-	-	-	-	-	8
14	45	2d	23	233	202	33	2	31	230	-	-	-	-	-	-	5
45	11	b	202	33	31	230	-	-	-	-	-	-	-	-	-	4
11	47	2f	33	2	31	-	-	-	-	-	-	-	-	-	-	4
47	34	22	33	31	230	-	-	-	-	-	-	-	-	-	-	2
34	15	f	230	-	-	-	-	-	-	-	-	-	-	-	-	3
15	2	2	31	230	-	-	-	-	-	-	-	-	-	-	-	1
2	13	d	-	-	-	-	-	-	-	-	-	-	-	-	-	1
13	44	2c	-	-	-	-	-	-	-	-	-	-	-	-	-	1
20	20	14	-	-	-	-	-	-	-	-	-	-	-	-	-	0
	100	64	-	-	-	-	-	-	-	-	-	-	-	-	-	69
																45h