2.21 Exercícios[1]

[1] Contribuição de John Oliver, da Cal Poly, San Luis Obispo, com colaborações de Nicole Kaiyan (Universidade de Adelaide) e Milos Prvulovic (Georgia Tech)

O Apêndice B descreve o simulador do MIPS, que é útil para estes exercícios. Embora o simulador aceite pseudoinstruções, tente não usá-las em qualquer exercício que pedir para produzir código do MIPS. Seu objetivo deverá ser aprender o conjunto de instruções MIPS real, e se você tiver de contar instruções, sua contagem deverá refletir as instruções reais executadas, e não as pseudoinstruções.
Existem alguns casos em que as pseudoinstruções precisam ser usadas (por exemplo, a instrução la quando um valor real não é conhecido durante a codificação em assembly).
Em muitos casos, elas são muito convenientes e resultam em código mais legível (por exemplo, as instruções li e move. Se você decidir usar pseudoinstruções por esses motivos, por favor, acrescente uma sentença ou duas à sua solução, indicando quais pseudoinstruções usou e por quê.

Exercício 2.1
Os problemas a seguir lidam com a tradução de C para MIPS. Suponha que as variáveis f, g, h e i sejam dadas e possam ser consideradas inteiros de 32 bits, conforme declarado em um programa C.

a. f=g-h;
b. f=g+(h-5);

2.1.1 [5] <2.2> Para essas instruções C, qual é o código assembly do MIPS correspondente? Use um número mínimo de instruções assembly do MIPS.

```
sub f, g, h

addi t0, h, -5
add f, g, t0
```

2.1.2 [5] <2.2> Para essas instruções C, quantas instruções assembly do MIPS são necessárias a fim de executar a instrução C?

Para a primeira, é necessária 1 instrução assembly do MIPS, e para a segunda, 2.

2.1.3 [5] <2.2> Se as variáveis f, g, h e i possuem o valor de 1,

2, 3 e 4, respectivamente, qual é o valor final de f?

a) f = -1
b) f = 0

Os problemas a seguir lidam com a tradução de MIPS para C. Suponha que as variáveis g, h, i e j sejam dadas e possam ser consideradas inteiros de 32 bits, conforme declarado em um programa C.

```
a.   addi f, f, 4
b.   add f, g, h
     add f, i, f
```

2.1.4 [5] <2.2> Para essas instruções MIPS, qual é a instrução C correspondente?

a) f += 4;

b) f = g + h;
f += i;

2.1.5 [5] <2.2> Se as variáveis f, g, h e i têm valores 1, 2, 3 e 4, respectivamente, qual é o valor final de f?

a) f = 5
b) f = 9

Exercício 2.3
Os problemas a seguir lidam com a tradução de C para MIPS. Considere que as variáveis f e g sejam dadas e possam ser consideradas inteiros de 32 bits, conforme declarado em um programa C.

```
a. f = —g — f;
b. f = g + (—f — 5);
```

2.3.1 [5] <2.2> Para essas instruções C, qual é o código assembly do MIPS correspondente? Use um número mínimo de instruções assembly do MIPS.

a) add f, -g, -f

b) addi t0, -f, -5
add f, g, t0

2.3.2 [5] <2.2> Para as instruções C anteriores, quantas instruções assembly do MIPS são necessárias a fim de executar a instrução C?

Para a primeira, foi necessária 1 instrução assembly do MIPS, para a segunda, 2.

2.3.3 [5] <2.2> Se as variáveis f, g, h, i e j têm valores 1, 2, 3, 4 e 5, respectivamente, qual é o valor final de f?

a) f = -3
b) f = -4

Os problemas a seguir lidam com a tradução de MIPS para C. Suponha que as variáveis g, h, i e j sejam dadas e possam ser consideradas inteiros de 32 bits, conforme declarado em um programa C.

```
a. addi   f, f, — 4
b. add    i, g, h
   add    f, i, f
```

2.3.4 [5] <2.2> Para essas instruções MIPS, qual é a instrução C correspondente?

a) f = f – 4;

b) i = g + h;
f = i + f;

2.3.5 [5] <2.2> Se as variáveis f, g, h e i têm valores 1, 2, 3 e 4, respectivamente, qual é o valor final de f?

a) f = -3
b) f = 6

Exercício 2.5

Nos problemas a seguir estaremos investigando as operações da memória no contexto de um processador MIPS. A tabela a seguir mostra os valores de um array armazenado na memória. Considere que o endereço de base do array está armazenado no registrador $s6 e faça o offset considerando o endereço de base do array.

a.	**Endereço**	**Dados**
	20	4
	24	5
	28	3
	32	2
	34	1
b.	**Endereço**	**Dados**
	24	2
	38	4
	32	3
	36	6
	40	1

2.5.1 [10] <2.2, 2.3> Para os locais de memória na tabela anterior, escreva o código C de modo a classificar os dados do mais baixo ao mais alto, colocando o menor valor no menor local de memória mostrado na figura. Suponha que os dados mostrados representem a variável C chamada Array, que é um array do tipo int. Suponha que essa máquina em particular seja uma máquina endereçável por byte e uma word consista em 4 bytes.

a)

```
temp = c[0];
c[0] = c[4];
c[4] = temp;

temp = c[1];
c[1] = c[4];
c[4] = temp;

temp = c[3];
c[3] = c[1];
c[1] = temp;
```

b)

```
temp = c[0];
c[0] = c[4];
c[4] = temp;

temp = c[1];
c[1] = c[4];
c[4] = temp;

temp = c[3];
c[3] = c[4];
c[4] = temp;
```

a.	**Endereço**	**Dados**	organizado
[0]	**20**	**4**	1
[1]	**24**	**5**	2
[2]	**28**	**3**	3
[3]	**32**	**2**	4
[4]	**36**	**1**	5
b.	**Endereço**	**Dados**	
	24	**2**	1
	28	**4**	2
	32	**3**	3
	36	**6**	4
	40	**1**	6

2.5.2 [10] <2.2, 2.3> Para os locais de memória na tabela anterior, escreva o código MIPS que classifique os dados do mais baixo ao mais alto, colocando o menor valor no menor local de memória. Use um número mínimo de instruções MIPS. Suponha que o endereço de base de Array esteja armazenado no registrador $s6.

a)

```
lw      $t0, 20($s6)
lw      $t1, 36($s6)
sw      $t1, 20($s6)
sw      $t0, 36($s6)

lw      $t0, 24($s6)
lw      $t1, 36($s6)
sw      $t1, 24($s6)
sw      $t0, 36($s6)

lw      $t0, 32($s6)
lw      $t1, 24($s6)
sw      $t1, 32($s6)
sw      $t0, 24($s6)
```

b)

```
lw      $t0, 24($s6)
lw      $t1, 40($s6)
sw      $t1, 24($s6)
sw      $t0, 40($s6)

lw      $t0, 28($s6)
lw      $t1, 40($s6)
sw      $t1, 28($s6)
sw      $t0, 40($s6)

lw      $t0, 36($s6)
lw      $t1, 40($s6)
sw      $t1, 36($s6)
sw      $t0, 40($s6)
```

a.	Endereço	Dados		
	20	4		
	24	5		
	28	3		
	32	2		
	36	1		
b.	Endereço	Dados		
	24	2		
	28	4		
	32	3		
	36	6		
	40	1		

2.5.3 [5] <2.2, 2.3> A fim de classificar o array anterior, quantas instruções são necessárias para o código MIPS? Se você não tiver permissão para usar o campo imediato nas instruções lw e sw, de quantas instruções MIPS você precisa?

Foram necessárias 12 instruções para cada código MIPS.

a.	Endereço	Dados			
	20	4			
	24	5			
	28	3			
	32	2			

	36	**1**			
b.	**Endereço**	**Dados**			
	24	**2**			
	28	**4**			
	32	**3**			
	36	**6**			
	40	**1**			

Exercício 2.10

Nos problemas a seguir, a tabela de dados contém bits que representam o opcode de uma instrução. Você deverá traduzir as entradas para o código assembly e determinar que formato da instrução MIPS os bits representam.

a. $0000\ 0010\ 0001\ 0000\ 1000\ 0000\ 0010\ 0000_{dois}$
b. $0000\ 0001\ 0100\ 1011\ 0100\ 1000\ 0010\ 0010_{dois}$

Utilizar MIPS Reference Data Card.pdf					
R Op (6 bits)	Rs (5 bits)	Rt (5 bits)	Rd (5 bits)	Shamt (5 bits)	Funct (6 bits)
I Op	Rs	Rt	Endereço (16 bits)		
J Op	Endereço (26 bits)				

a. 0000 0010 0001 0000 1000 0000 0010 0000$_{dois}$

add $s0, $s0, $s0

opcode	rs	rt	rd	shamt	funct
0 0 0 0 0 0	1 0 0 0 0	1 0 0 0 0	1 0 0 0 0	0 0 0 0 0	1 0 0 0 0 0

b. 0000 0001 0100 1011 0100 1000 0010 0010$_{dois}$

sub $t1, $t2, $t3

opcode	rs	rt	rd	shamt	funct
0 0 0 0 0 0	0 1 0 1 0	0 1 0 1 1	0 1 0 0 1	0 0 0 0 0	1 0 0 0 1 0

2.10.1 [5] <2.5> Para essas entradas binárias, que instrução elas representam?

No a, add, e no b, sub.

2.10.2 [5] <2.5> Que tipo de instrução (tipo I, tipo R) as mesmas entradas binárias representam?

Tipo R.

2.10.3 [5] <2.4, 2.5> Se as entradas binárias anteriores fossem bits de dados, que número elas representariam em hexadecimal?

add -> 20hex 0x101010020
sub -> 22hex 0xab9022

Nos problemas a seguir, a tabela de dados contém instruções MIPS. Você deverá traduzir as entradas para os bits do opcode e determinar qual é o formato da instrução MIPS.

a. addi $t0 ,$t0 ,0
b. sw $t1, 32($t2)

R Op (6 bits)	Rs (5 bits)	Rt (5 bits)	Rd (5 bits)	Shamt (5 bits)	Funct (6 bits)
I Op	Rs	Rt	Endereço (16 bits)		
J Op	Endereço (26 bits)				

a. addi $t0 ,$t0 ,0

op						rs					rt					immediate															
0	0	1	0	0	0	0	1	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

b. sw $t1, 32($t2)

op						rs					rt					endereço															
1	0	1	0	1	1	0	1	0	1	1	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0

2.10.4 [5] <2.4, 2.5> Mostre a representação hexadecimal dessas instruções.

a) 8880
b) 2bba20

2.10.5 [5] <2.5> Que tipo (tipo I, tipo R) essas instruções representam?

Tipo I

2.10.6 [5] <2.5> Qual é a representação hexadecimal dos campos opcode, Rs e Rt nessa instrução? Para as instruções de tipo R, qual é a representação hexadecimal dos campos Rd e funct? Para as instruções de tipo I, qual é a representação hexadecimal do campo imediato?

Cada instrução tem uma tradução diferente, como visto no exercicio 2.10.4. rd de acordo com o registrador em questão, e funct de acordo com a função (add, sub, etc). Basta traduzir a constante em para hexadecimal (exemplo: 26 torna-se 1a).

Exercício 2.16

Para estes problemas, a tabela mantém diversos valores binários para o registrador $t0.
Dado o valor de $t0, você deverá avaliar o resultado de diferentes desvios.

a. $t0 = 0010 0100 1001 0010 0100 1001 0010 0100_{dois}
b. $t0 = 0101 1111 1011 1110 0100 0000 0000 0000_{dois}

2.16.1 [5] <2.7> Suponha que o registrador $t0 contenha um desses valor e $t1 tenha o valor

$t1 = 0011 1111 1111 1000 0000 0000 0000 0000_{dois}

Note o resultado da execução de tais instruções em certos registradores. Qual é o valor de $t2 depois das seguintes instruções?

```
              slt $t2, $t0, $t1
              beq $t2, $ZERO, ELSE
              j DONE
ELSE:     addi $t2, $0, 2
DONE:
```

a) $t2 = 1
b) $t2 = 2

2.16.4 [5] <2.7> Suponha que o registrador $t0 contenha um valor da tabela anterior.
Qual é o valor de $t2 após as instruções a seguir?

```
              slt $t2, $0, $t0
              bne $t2, $ZERO, ELSE
              j DONE
ELSE:     addi $t2, $t2, 2
DONE:
```

$t2 = 3