



2ª Lista de Exercícios

Observação:

Os exercícios desta lista não serão cobrados. Porém, é altamente recomendável que os alunos tentem resolvê-los, a fim de se prepararem minimamente para a 1ª prova semestral.

Exercício 1

- a) Quais instruções do processador MIPS acessam a memória de dados? Quais seriam as implicações no projeto do processador MIPS, caso houvesse instruções aritméticas cujos operandos (de entrada) tivessem que ser lidos da memória de dados?
- b) Considere que o registrador *\$pc* (PC) do processador MIPS armazene o valor *0x000C0000* em uma instrução *beq*. Após a execução da instrução *beq* qual o maior endereço que pode ser atingido? Qual o menor endereço?
- c) Considere ainda que o registrador *\$pc* (PC) armazene o valor *0x000C0000*. Caso seja utilizada uma instrução *jump* para realizar o desvio, qual o maior endereço que pode ser atingido? Qual o menor endereço?
- d) Proponha um trecho de código MIPS que realize $e = (a+b) - (c-d)$. Considere que *a*, *b*, *c* e *d* estão nos registradores *\$s0*, *\$s1*, *\$s2* e *\$s3* e que a variável *e* corresponde ao registrador *\$s4*.
- e) Proponha um trecho de código MIPS que realize $A[20] = a + A[10]$. Considere que o endereço base do array *A* esteja no registrador *\$s1* e a variável *a* esteja no registrador *\$s0*.
- f) As instruções **add**, **sub**, **and** e **or** são instruções do tipo R que buscam seus operandos em registradores. Caso um dos operandos seja uma constante, o programador pode, alternativamente, utilizar certas instruções do tipo I para realizar somas, subtrações, ands ou or. Consulte o manual do MIPS (no livro) e apresente essas instruções juntamente com os seus códigos de operação.

Exercício 2

Para resolver a presente questão, considere as instruções do processador MIPS descritas a seguir.

Mnemônico	Instrução	Linguagem de Montagem	Significado
addi	Adição imediata	addi \$s1, \$s2, const	$\$s1 \leftarrow \$s2 + \text{const}$
add	Adição	add \$s1, \$s2, \$s3	$\$s1 \leftarrow \$s2 + \$s3$
sub	Subtração	sub \$s1, \$s2, \$s3	$\$s1 \leftarrow \$s2 - \$s3$
slt	Set on less than	slt \$s1, \$s2, \$s3	if($\$s2 < \$s3$) $\$s1 \leftarrow 1$; else $\$s1 \leftarrow 0$
lw	Load word	lw \$s1, desl(\$s2)	$\$s1 \leftarrow \text{Mem}[\$s2 + \text{desl}]$
sw	Store word	sw \$s1, desl(\$s2)	$\text{Mem}[\$s2 + \text{desl}] \leftarrow \$s1$
beq	Branch on equal	beq \$s1, \$s2, L	if($\$s1 == \$s2$) desvia para L
j	Jump	j L	desvia para a linha cujo label é "L"

Complete o código em linguagem de montagem gerado pelo compilador do processador MIPS para o trecho de código em linguagem C dado a seguir, onde *X* é um vetor de *n* elementos que foi alocado estaticamente na memória e seu endereço-base está armazenado no registrador *\$s0*. Considere também que neste trecho de código as variáveis *i*, *total* e *n* estejam nos registradores *\$s1*, *\$s2* e *\$s3*, respectivamente, e que o registrador *\$zero* contenha a constante zero (i.e., *\$zero*=0).

```
i=0;
total=0;
while( i < n ) {
    total=total+X[i];
    i=i+1;
}
```


A Tab. 1 mostra como as instruções *add*, *sub*, *lw*, *sw*, *beq* e *jump* estão definidas no MIPS.

Tab. 1 – Descrição das instruções *add*, *sub*, *lw*, *sw*, *beq* e *jump* do MIPS.

Mnemônico	Instrução	Linguagem de Montagem	Significado
<i>add</i>	Adição	<i>add \$s1, \$s2, \$s3</i>	$\$s1 := \$s2 + \$s3$
<i>sub</i>	Subtração	<i>add \$s1, \$s2, \$s3</i>	$\$s1 := \$s2 - \$s3$
<i>lw</i>	Load word	<i>lw \$s1, deslocam(\$s2)</i>	$\$s1 \leftarrow \text{Mem}[\$s2 + \text{deslocam}]$
<i>sw</i>	Load word	<i>sw \$s1, deslocam (\$s2)</i>	$\text{Mem}[\$s2 + \text{deslocam}] \leftarrow \$s1$
<i>beq</i>	branch on equal	<i>beq \$s1, \$s2, deslocam</i>	$\text{if}(\$s1 == \$s2) \text{ then } PC \leftarrow PC + 4 + 4 \times \text{deslocam}$
<i>j</i>	jump	<i>j, endereço</i>	$PC \leftarrow \text{endereço}$

Nesta definição *\$s1*, *\$s2* e *\$s3* são registradores e *deslocam* é uma constante inteira de 32 bits, obtida a partir da extensão de sinal dos 16 bits menos significativos da instrução (formato "I").

- b) Preencha a Tab. 2 com os valores adequados dos sinais de controle que configuram o bloco operativo (*datapath*) do MIPS monociclo para a execução de cada uma das instruções da Tab. 1. Lembre-se que, conforme discutido em aula, os sinais que controlam leitura e escrita de elementos de memória (memórias e registradores) não podem valer "X" (*don't care*).

Tab. 2 – Sinais de controle para o bloco operativo do MIPS monociclo executar *add*, *sub*, *lw*, *sw*, *beq* e *jump* (a completar).

	DVI	DvC	ULAFonte	ULAOp	LerMem	EscMem	MemParaReg	RegDst	EscReg
<i>add</i>				10					
<i>sub</i>				10					
<i>lw</i>				00					
<i>sw</i>				00					
<i>beq</i>				01					
<i>j</i>				XX					

A Tab. 3 mostra valores para as características temporais dos componentes do MIPS monociclo da Fig. 1.

Para resolver os itens que seguem, considere o diagrama de blocos do MIPS monociclo mostrado na Fig. 1 e os atrasos de seus componentes apresentados na Tab. 2. Considere também a seguinte nomenclatura: dado um sinal "nome_do_sinal", $TE(\text{nome_do_sinal})$ designa seu tempo de estabilização (nesta questão, em picossegundos – ps).

- Calcule $TE(I)$, $TE(S1)$, e $TE(S2)$. Estes TEs dependem da instrução a ser executada? Explique.
- Calcule os tempos de estabilização (TEs) dos sinais de controle e do sinal *cULA* do MIPS da Fig. 1.
- Calcule $TE(A)$ e $TE(B)$. Quais instruções fazem uso do valor fornecido pelo sinal *B*? Estes TEs dependem da instrução a ser executada? Explique.
- Considerando a instrução *add*, calcule $TE(M5)$. Quais instruções não possuem o mesmo valor para $ME(5)$? Explique.
- Considerando ainda a instrução *add*, calcule $TE(M3)$.
- Qual é o atraso crítico para a instrução *add*? Escreva o caminho crítico para esta instrução.
- Considerando a instrução *lw*, calcule $TE(M3)$.
- Qual é o atraso crítico para a instrução *lw*? Escreva o caminho crítico desta instrução.
- Para determinar o atraso crítico do MIPS monociclo (considerando as características temporais da Tab. 2), é preciso realizar a análise timing para as demais instruções vistas em aula? (Sim ou não?) Explique.
- Qual é atraso crítico do MIPS monociclo desta questão? Justifique e ou mostre o cálculo.

- m) Calcule a frequência máxima de operação para o processador MIPS monociclo considerando as características temporais da Tab. 2. Indique seu valor em GHz (gigahertz) e mostre o cálculo.

Tab. 3 – Características temporais dos componentes do MIPS, versão monociclo. Obs: $1 \text{ ps} = 1 \times 10^{-12} \text{ s}$

Componente	Característica	Símbolo	Valor
Memória de instruções	tempo para leitura	$td_{L\text{MEM}}$	300 ps
Memória de dados	tempo para leitura	$td_{L\text{MEM}}$	300 ps
Memória de dados	tempo para escrita (setup)	$td_{E\text{MEM}}$	300 ps
Banco de Registradores	tempo para leitura	$td_{L\text{REG}}$	70 ps
Banco de Registradores	tempo para escrita (setup)	$td_{E\text{REG}}$	70 ps
ULA	atraso para qualquer operação	td_{ULA}	60 ps
Qualquer somador	atraso	td_{soma}	20 ps
Qualquer mux 2:1	atraso	td_{mux}	2 ps
PC	tempo de setup	tsu	5 ps
PC	tempo de carga	tco	5 ps
PC	tempo de hold	th	Desprezível (0 ps)
Deslocador, extensão de sinal, porta AND	atraso	-	Desprezível (0 ps)
Controle	atraso	td_{controle}	75 ps
Controle da ULA	atraso	td_{ctrlULA}	5 ps

Exercício 4

Considere a instrução `addi`, a qual é definida da seguinte forma:

Mnemônico	Instrução	Linguagem de Montagem	Significado
<code>addi</code>	Adição imediata	<code>addi \$s1, \$s2, deslocam</code>	$\$s1 := \$s2 + \text{deslocam}$

Considere que esta instrução utiliza o **formato I** (o mesmo utilizado nas instruções `lw` e `sw`), possuindo porém um código de operação ("opcode") próprio, diferente daquele utilizado nas instruções aritméticas e lógicas que operam somente com registradores (`add`, `sub`, `and`, `or`, `slt`). O valor `deslocam` é uma constante inteira de 32 bits, obtida a partir da extensão de sinal dos 16 bits menos significativos da instrução.

- Quais alterações são necessárias no bloco operativo (*datapath*) do MIPS monociclo para que `addi` possa ser executada?
- Quais alterações são necessárias no bloco de controle do MIPS monociclo para que `addi` possa ser executada?
- Complete na tabela a seguir os valores dos sinais de controle que devem ser aplicados ao bloco operativo (*datapath*) do MIPS monociclo para que `addi` seja executada corretamente.

	DVI	DvC	ULAFonte	ULAOp	LerMem	EscMem	MemParaReg	RegDst	EscReg
<code>addi</code>				00					