

#### **Universidade Federal de Pelotas**

Centro de Desenvolvimento Tecnológico Bacharelado em Ciência da Computação Engenharia de Computação

# Arquitetura e Organização de Computadores I

**Prática** 

Aula 6

Revisão

Prof. Guilherme Corrêa gcorrea@inf.ufpel.edu.br

Prof. Bruno Zatt

#### MIPS: Registradores

Registrador	Nome	Uso (convenção)
\$0	\$zero	Zero
\$1	\$at	Assembler Temporary
\$2, \$3	\$v0, \$v1	Valor de retorno de subrotina
\$4 – \$7	\$a0 – \$a3	Argumentos de subrotina
\$8 - \$15	\$t0 - \$t7	Temporários (locais à função)
\$16 – \$23	\$s0 - \$s7	Salvos (não alterados na função)
\$24, \$25	\$t8, \$t9	Temporários
\$26, \$27	\$k0, \$k1	Kernel (reservado para SO)
\$28	\$gp	Global Pointer
\$29	\$sp	Stack Pointer
\$30	\$fp	Frame Pointer
\$31	\$ra	Endereço de Retorno

#### Operações Lógicas (and, or, xor, nor)

	31 26	25 21	20 16	15 11	10 6	5 0
Tipo R	opcode	rs	rt	rd	shamt	funct
	6 bits	5 bits	5 bits	5 bits	5 bits	6 bits
and \$t1, \$zero, \$t3	<b>0</b> н	0н	Вн	9н	0н	24н
	000000	00000	01011	01001	00000	100100
or \$t0, \$t1, \$t2	0н	0	۸	0	0н	25н
OI DIO. DII. DIZ	UH	1 9H	I AH	OH	UH	l Zijh i
οι φιο, φιι, φιΖ		<b>9</b> н	<b>A</b> H	<b>8</b> н		
Οι φιο, φιι, φιΖ	000000	01001	01010	<b>о</b> н 01000	00000	100101
xor \$t1, \$t2, \$t3						
, , ,	000000	01001	01010	01000	00000	100101
, , ,	000000 <b>0</b> н	01001 <b>А</b> н	01010 <b>В</b> н	01000 <b>9</b> н	00000 Он	100101 <b>26</b> н

#### Operações Lógicas (andi, ori, xori)

	31 26	25 21	20 16	15 0
Tipo I	opcode	rs	rt	Deslocam. ou oper. imediato
	6 bits	5 bits	5 bits	16 bits
andi \$t1, \$t0, 0xA0A0	Сн	8н	9н	А0А0н
	001100	01000	01001	1010000010100000
ori \$t0, \$zero, 0xFFFF	<b>D</b> н	0н	8н	FFFFH
	001101	00000	01000	111111111111111
xori \$t1, \$t1, 0x538	Ен	9н	9н	<b>538</b> н
	001110	01001	01001	0000010100111000

#### Operações Lógicas (sll, slr)

Tipo R

31	26	25	21	20	16	15	11	10	6	5	0
орс	ode	r	S	r	t	ro	b	sha	amt	fu	ınct
6 b	its	5 b	its	5 k	oits	5 b	its	5 b	oits	6	bits

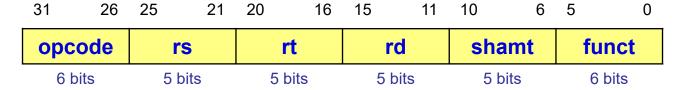
SII	\$tU,	Φť1,	12

srl \$t5, \$t5, 28

0н	<b>0</b> н	9н	8н	Сн	0н
000000	00000	01001	01000	01100	000000
0н	0н	<b>D</b> н	<b>D</b> н	1Сн	2н
000000	00000	01101	01101	11100	000010

#### Operações Aritméticas (sra)

Tipo R



sra \$t1, \$t0, 4

<b>0</b> н	0н	8н	9н	4н	3н
000000	00000	01000	01001	00100	000011

#### Operações Aritméticas (add, sub, addu, subu)

, .			<del>-</del>			
	31 26	25 21	20 16	15 11	10 6	5 0
Tipo R	opcode	rs	rt	rd	shamt	funct
_	6 bits	5 bits	5 bits	5 bits	5 bits	6 bits
		,				,
add \$t2, \$t0, \$t1	0н	8н	9н	Ан	0н	20н
	000000	01000	01001	01010	00000	100000
sub \$t2, \$t0, \$t1	0н	8н	9н	Ан	0н	22н
sub \$t2, \$t0, \$t1	<b>О</b> н 000000	<b>8</b> н 01000	<b>9</b> н 01001	<b>А</b> н 01010	<b>О</b> н 00000	<b>22</b> н 100010
sub \$t2, \$t0, \$t1 addu \$t2, \$t0, \$t1						
	000000	01000	01001	01010	00000	100010
	000000 <b>О</b> н	01000 <b>8</b> н	01001 <b>9</b> н	01010 <b>А</b> н	00000 <b>О</b> н	100010 <b>21</b> н

#### Operações Aritméticas (mult, multu, mfhi, mflo)

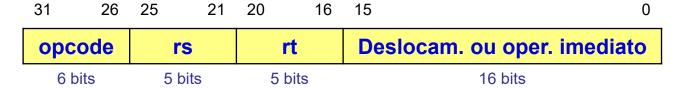
, ,			•	•	•	•
	31 26	25 21	20 16	15 11	10 6	5 0
Tipo R	opcode	rs	rt	rd	shamt	funct
_	6 bits	5 bits	5 bits	5 bits	5 bits	6 bits
					<del>,</del>	,
mult \$t0, \$t1	0н	8н	9н	0н	0н	18н
	000000	01000	01001	00000	00000	011000
multu \$t0, \$t1	<b>0</b> н	8н	9н	<b>0</b> н	0н	19н
multu \$t0, \$t1	<b>0</b> н 000000	<b>8</b> н 01000	<b>9</b> н 01001	<b>О</b> н 00000	<b>О</b> н 00000	<b>19</b> н 011001
multu \$t0, \$t1 mfhi \$t2						
	000000	01000	01001	00000	00000	011001
	000000 <b>О</b> н	01000 Он	01001 Он	00000 <b>А</b> н	00000 Он	011001 <b>10</b> н

#### Operações Aritméticas (div, divu, mfhi, mflo)

-	31 26	25 21	20 16	15 11	10 6	5 0
Tipo R	opcode	rs	rt	rd	shamt	funct
•	6 bits	5 bits	5 bits	5 bits	5 bits	6 bits
		T				
div \$t0, \$t1	0н	8н	9н	0н	0н	1Ан
	000000	01000	01001	00000	00000	011010
di ¢10 ¢11	•					45
divu \$t0, \$t1	0н	8н	9н	0н	0н	1Вн
αινα φιυ, φι ι	000000	<b>8</b> н 01000	<b>9</b> н 01001	00000	00000	<b>1В</b> н 011011
mfhi \$t2						
·	000000	01000	01001	00000	00000	011011
·	000000 <b>О</b> н	01000 Он	01001 Он	00000 <b>А</b> н	00000 Он	011011 <b>10</b> н

#### Operações Aritméticas (addi, addiu)

Tipo I



addi \$t2, \$t0, 0x12

8н	8н	Ан	12н
001000	01000	01010	000000000010010

addiu \$t2, \$t0, 0x12

9н	8н	Ан	12н
001001	01000	01010	000000000010010

21

20

#### Acesso à Memória (lw, sw)

25

31

Tipo I

opcode	rs	rt	Deslocam. ou oper. imediato
6 bits	5 bits	5 bits	16 bits

15

Iw \$t2, 0x60(\$t1)

23н	9н	Ан	60н
100011	01001	01010	000000001100000

sw \$t2, 0x60(\$t1)

2Вн	9н	Ан	60н
101011	01000	01010	000000001100000

0

20

#### Acesso à Memória (Ih, sh)

31

Tipo I

<b>0</b> 1	20	20	۷ ۱	20	10	10
opco	ode	r	S	rt		Deslocam. ou oper. imediato
6 bi	ts	5 k	oits	5 bit	S	16 bits

15

Ih \$t2, 0x62(\$t1)

<b>21</b> н	9н	Ан	62н
100001	01001	01010	000000001100010

Ihu \$t2, 0x62(\$t1)

<b>25</b> н	9н	Ан	62н
100101	01000	01010	000000001100010

sh \$t2, 0x62(\$t1)

29н	9н	Ан	62н
101001	01000	01010	000000001100010

Λ

#### Acesso à Memória (lb, sb)

Tipo I

J1	20	23	۷ ۱	20	10	15
opco	ode	r	Ś	ri	t	Deslocam. ou oper. imediato
6 bi	ts	5 l	bits	5 b	its	16 bits

15

lb \$t2, 0x63(\$t1)

<b>20</b> н	9н	Ан	63н
100000	01001	01010	000000001100011

Ibu \$t2, 0x62(\$t1)

<b>24</b> н	9н	Ан	63н
100100	01000	01010	000000001100011

sb \$t2, 0x63(\$t1)

28н	9н	Ан	63н
101000	01000	01010	000000001100011

#### Acesso à Memória (lui)

Tipo I



Iui \$t0, 0x1001

Fн	0н	8н	1001н
001111	00000	01000	000100000000001

#### Desvios Condicionais (beq, bne)

Tipo I

<u>ي</u> ا	20	25	21	20	10	15 0	
opcode		rs		rt		Deslocam. ou oper. imediato	
6 bits		5 k	oits	5 bi	ts	16 bits	

beq \$t0, \$t1, *Label* 

4н	8н	9н	?н
000100	01000	01001	??????????????

bne \$t0, \$t1, *Label* 

5н	8н	9н	?н
000101	01000	01001	??????????????

#### Desvios Condicionais (slt, sltu)

Tipo R

	31	26	25	21	20	16	15	11	10	6	5	0
	opcode		rs		rt		rd		shamt		fu	nct
6 bits		5 b	oits	5 b	its	5 b	its	5 k	oits	6	bits	

slt \$t0, \$t1, \$t2

<b>0</b> н	8н	9н	10н	0н	2Ан	
000000	01000	01001	01010	00000	101010	

sltu \$t0, \$t1, \$t2

0н	8н	9н	10н	0н	2Вн
000000	01000	01001	01010	00000	101011

#### Desvios Condicionais (slti, sltiu)

Tipo I



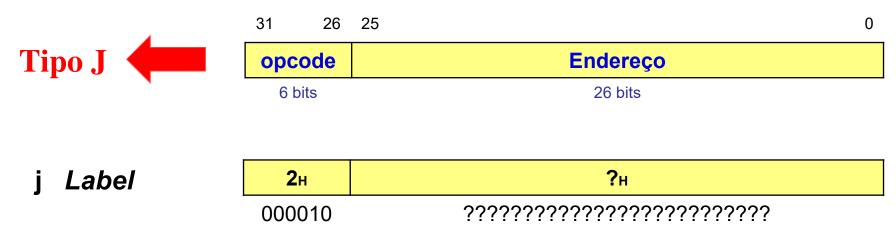
slti \$t0, \$t1, 0x63

Ан	8н	9н	63н
001010	01000	01001	000000001100011

sltiu \$t0, \$t1, 0x63

Вн	8н	9н	63н
001011	01000	01001	000000001100010

#### Desvios Incondicionais (j)



#### Desvios Incondicionais (jr)

Tipo R

	31	26	25	21	20	16	15	11	10	6	5	0
	opcode		rs		rt		rd		shamt		fu	nct
6 bits		5 k	oits	5 b	oits	5 k	oits	5 I	oits	6	bits	

jr \$t0

<b>0</b> н	8н	0н	<b>0</b> н	0н	08н
000000	01000	00000	00000	00000	001000

## Definir seção de dados no MARS

- Utilizamos a diretiva .data
- O início da área de dados é a posição **0x10010000**
- Atenção: .data NÃO É uma instrução do MIPS; é apenas uma diretiva do montador!

```
      .data
      # inicia a seção de dados

      .word 1
      # escreve 1 em 0x10010000

      .word -3
      # escreve -3 em 0x10010004

      .word 15
      # escreve 15 em 0x10010008
```

## Definir seção de dados no MARS

- A partir de agora, utilizaremos:
  - .data antes da seção de dados
  - .text antes da seção de instruções

```
.data
                           # inicia a seção de dados
                           # escreve 1 em 0x10010000
      word 1
      .word -3
                           # escreve -3 em 0x10010004
      word 15
                           # escreve 15 em 0x10010008
.text
                           # inicia a seção de instruções
      lui $t0, 0x1001
                           # carrega o reg $t0 com 0x10010000
      lw $t1, 8($t0)
                           # carrega $t1 com a word em 0x10010008
      add $s2, $t1, $t1
                           # soma $t1 com $t1 e coloca o resultado em $s2
      sw $s2, 12($t0)
                           # escreve a word $s2 em 0x1001000C
```

## Definir seção de dados no MARS

• Outras diretivas para .data

```
.byte reserva um byte
```

.half reserva uma halfword

.space x reserva x bytes