

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

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	opco	de	r	S		t	r	d	sha	mt	fur	nct
-	6 bit	S	5 b	oits	5	oits	5	oits	5 b	its	6 k	oits
and \$t1, \$zero, \$t3	0н		0	н	E	Вн	ç	н	0	н	24	4 н
	0000	00	000	000	01	011	01	001	000	000	100	100
or \$t0, \$t1, \$t2	0н		9	н	P	Н	8	Н	0	Н	2	5 н
	0000	00	010	001	01	010	01	000	000	000	100	101
xor \$t1, \$t2, \$t3	0н		A	λΗ	E	Вн	ç	Н	0	Н	20	Э н
	0000	00	010)10	01	011	01	001	000	000	100	110
nor \$t0, \$t0, \$t3	0н		8	н	E	Вн	8	Вн	0	Н	27	7н
	0000	00	010)11	01	011	01	001	000	000	100	111

Operações Lógicas (andi, ori, xori)

31 26 25 21 20 16 15 0 Tipo I opcode Deslocam. ou oper. imediato rs rt 6 bits 5 bits 5 bits 16 bits andi \$t1, \$t0, 0xA0A0 **А0А0**н Сн 8н 9н 001100 01000 01001 1010000010100000 ori \$t0, \$zero, 0xFFFF Dн 0н 8н FFFF_H 001101 00000 01000 11111111111111111 xori \$t1, \$t1, 0x538 Ен 9н 9н 538н 001110 01001 01001 0000010100111000

Operações Lógicas (sll, slr)

Tipo R

31	26	25	21	20	16	15	11	10	6	5	0
орс	ode	rs	5	rt		ro	t	sha	amt	fu	nct
6 b	oits	5 b	its	5 bi	ts	5 b	its	5 k	oits	6	bits

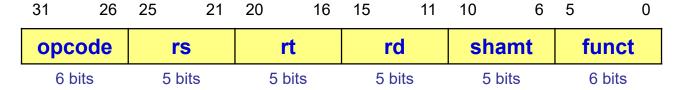
sll \$t0, \$t1, 12

srl \$t5, \$t5, 28

0 н	0 н	9н	8н	Сн	0 н
000000	00000	01001	01000	01100	000000
0н	0н	Dн	Dн	1Сн	2н
000000	00000	01101	01101	11100	000010

Operações Aritméticas (sra)

Tipo R



sra \$t1, \$t0, 4

0н	0н	8н	9н	4н	3н
000000	00000	01000	01001	00100	000011

Operações Aritméticas (add, sub, addu, subu)

	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	0	0	•			00
Sub φιz, φιυ, φι i	0н	8н	9н	Ан	0н	22н
Sub φιΖ, φιο, φι ι	000000	8 н 01000	9 н 01001	А н 01010	00000	100010
addu \$t2, \$t0, \$t1						
	000000	01000	01001	01010	00000	100010
	000000 0 н	01000 8 н	01001 9 н	01010 А н	00000 О н	100010 21 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
			,			
mult \$t0, \$t1	0н	8н	9н	0н	0н	18н
	000000	01000	01001	00000	00000	011000
multu \$t0, \$t1	0 н	8н	9н	0н	0н	19н
multu \$t0, \$t1	0 н 000000	8 н 01000	9 н 01001	О н 00000	О н 00000	19 н 011001
multu \$t0, \$t1 mfhi \$t2						
,	000000	01000	01001	00000	00000	011001
,	000000 О н	01000 Он	01001 Он	00000 А н	00000 Он	011001 10 н

Operações Aritméticas (div, divu, mfhi, mflo)

· •	31 2	26	25	21	20	16	15	11	10	6	5	0
Tipo R	opcode	9	r	S		t	re	d	sh	amt	fur	ct
_	6 bits		5 b	its	5	oits	5 b	its	5	bits	6 b	its
J: #40 #44	_				_							
div \$t0, \$t1	0н		8	н	5)н	0	Н	(Эн	1 <i>A</i>	І Н
	000000)	010	000	01	001	000	000	00	000	0110	010
divu \$t0, \$t1	0н		8	н	9	н	0	Н	(Эн	1E	Вн
	000000)	010	000	01	001	000	000	00	000	011	011
mfhi \$t2	0н		0	н	0	Н	Α	н	0)н	10	Н
	000000)	000	00	000	000	010	10	00	000	0100	000
mflo \$t3	0н		0	н	0	Н	В	Н	C)н	12	Н

Operações Aritméticas (addi, addiu)

Tipo I



addi \$t2, \$t0, 0x12

8н	8н	Ан	12н
001000	01000	01010	000000000010010

addiu \$t2, \$t0, 0x12

2	9н	8н	A H	12н
	001001	01000	01010	000000000010010

Acesso à Memória (Iw, sw)

Tipo I



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

23н	9н	Ан	60н
100011	01001	01010	000000001100000

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

2Вн	9н	A H	60н
101011	01000	01010	000000001100000

Acesso à Memória (Ih, sh)

Tipo I

31	26	25	21	20	16	15 0
opcode		rs		rt		Deslocam. ou oper. imediato
6 k	oits	5 b	its	5 k	oits	16 bits

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

21н	9н	Ан	62н
100001	01001	01010	000000001100010

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

25 н	9н	Ан	62н
100101	01000	01010	000000001100010

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

29н	9н	Ан	62н
101001	01000	01010	000000001100010

Acesso à Memória (lb, sb)

Tipo I

	31	20	23	۷ ۱	20	10	15
	opco	ode	r	છુ	r	t	Deslocam. ou oper. imediato
•	6 bi	ts	5	bits	5 k	oits	16 bits

15

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

20 н	9н	Ан	63н
100000	01001	01010	000000001100011

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

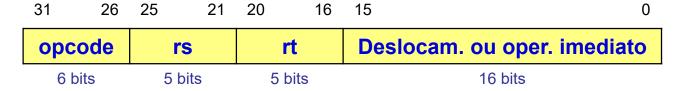
24 н	9н	Ан	63н
100100	01000	01010	000000001100011

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

31	26	25	21	20	16	15 0
opco	ode	r	S	r	t	Deslocam. ou oper. imediato
6 bi	its	5 b	oits	5 k	oits	16 bits

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

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

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

5 н	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		ocode rs		rt	rt rd		shamt		funct		
6 b	oits	5 b	its	5 bi	ts	5 b	its	5 k	oits	6	bits

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

0 н	8н	9н	10н	10н Он	
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



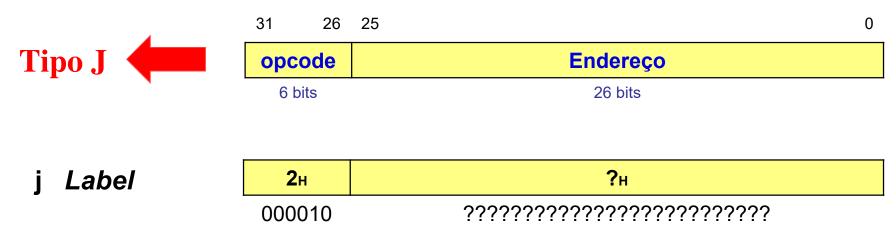
slti \$t0, \$t1, 0x63

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		r	S	rt		r	d	sha	amt	fu	nct
6 bits		5 k	oits	5 bi	its	5 b	oits	5 I	oits	6	bits

jr \$t0

0н	8н	0 н	0н	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
                           # escreve -3 em 0x10010004
      .word -3
      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