Sub-conjunto de instruções do MIPS

B Ridest, Imm16(Rarc) 1 0x20 load byte from memory lbu Ridest, Imm16(Rarc) 1 0x24 load word from memory sb Rarc2, Imm16(Rarc1) 1 0x24 load word from memory sb Rarc2, Imm16(Rarc1) 1 0x26 store word to memory sw Rsrc2, Imm16(Rarc1) 1 0x26 store word to memory sw Rsrc2, Imm16(Rarc1) 1 0x26 store word to memory sw Rsrc2, Imm16(Rarc1) 1 0x26 store word to memory sw Rsrc2, Imm16(Rarc1) 1 0x26 store word to memory sw Roset, Imm16 1 0x0 0x11 move from in memory	Grupo		Sintaxe	Tipo	Op	Func	Comentário
w Rdest, Imm16(Rsrc)	-	lb	Rdest, Imm16(Rsrc)	1	0x20	-	load byte from memory
Store byte to memory Store byte to memory Store word to move from it Store word to St		lw		1	0x23	-	load word from memory
Swape Resc2 Imm16 Ricest R 00 0x10 0x90 move from him Ricest R 00 0x10 move from him Ricest R 00 0x11 move to him Ricest R 00 0x13 move to him Ricest R 0x10 0x13 move to him Ricest R 0x10 0x13 move to him Ricest Rices		lbu		- 1		-	
Transferência de de de de de de de d		sb		1		-	
Mest Rest R				I		-	
informação				I		-	
mthi	II						
mtlo	informação						
II		_					
la				K	00	UXIS	
move move add Rdest, Rsrc1, Rsrc2 R 00 0x20 addition (with overflow) addition with overflow) addition with overflow with overflow addition with overflow addition with overflow addition with overflow addition with overflow with overflow addition			,				
add Rdest, Rsrc1, Rsrc2 R 00 0x20 addition (with overflow) addu Rdest, Rsrc1, Imm16 I 08 0x21 addition (with overflow) addition imm, (with ov.) addition (without overflow) addition imm, (with ov.) addition (without overflow) addition imm, (with ov.) addition (without overflow) addition (without overflow) addition imm, (with ov.) addition (without overflow) addition (without overflow							
addi		_		R	00	0x20	
addu Rdest, Rsrc1, Rsrc2 R 00 0x21 addition (without overflow)							
Addiu Rdest, Rsrc1, Imm16				R	00	0x21	
Sub Rdest, Rsrc1, Rsrc2 R 00 0x22 Subtract (with overflow)		addiu		1	09	-	
Operações aritméticas		sub	Rdest, Rsrc1, Rsrc2		00	0x22	subtract (with overflow)
aritméticas		subu					
div	11	mult					
divu	aritméticas						
abs Rdest, Rsrc mul Rdest, Rsrc1, Rsrc2 multiply multiple							
multiply Rdest, Rsrc1, Rsrc2 multiply multiply divide remainder multiply mu				R	00	0x1b	
div Rdest, Rsrc1, Rsrc2							
rem Rdest, Rsrc1, Rsrc2 R 00 0x24 AND mmediate							multiply
and Rdest, Rsrc1, Rsrc2 and Rdest, Rsrc1, Imm16		_					
Andi		_		R	00	0x24	
Or Rdest, Rsrc1, Imm16						-	
Operações lógicas lógicas ori Rdest, Rsrc1, Rsrc2 Rdest, Rsrc1, Imm16 I loxôbe - OR immediate OR immediate e de nor Rdest, Rsrc1, Rsrc2 et de comparação Rdest, Rsrc1, Rsrc2 R OU OX23 Set if less than slit Rdest, Rsrc1, Rsrc2 R OU OX24 Set if less than immediate sltu Rdest, Rsrc1, Imm16 I OX0b - Set if less than immediate sltu Rdest, Rsrc1, Imm16 I OX0b - Set if less than unsigned set if less than unsigned set if less than unsigned imm. Operações de deslocamento de bits sll Rdest, Rsrc1, Shamt5 R OU OV2b Shift left logical shift right logic			Rdest Rsrc1 Rsrc2			0x25	
Operações Iógicas Xor Xori Rdest, Rsrc1, Rsrc2 Rdest, Rsrc1, Imm16 R 00 0x26 VOR XOR XOR immediate comparação sit Rdest, Rsrc1, Rsrc2 R 00 0x2a Set if less than set if less than immediate siti Rdest, Rsrc1, Imm16 I 0x0a 0x2b Set if less than immediate set if less than immediate sitiu Rdest, Rsrc1, Imm16 I 0x0b 0x0b 0x0b - set if less than immediate sitiu Rdest, Rsrc1, Imm16 I 0x0b 0x0b - set if less than immediate sitiu Rdest, Rsrc1, Shamt5 R 00 0x2b 0x0b set if less than immediate sitiu Rdest, Rsrc1, Shamt5 R 00 0x0b 0x0b - set if less than immediate sitiu Rdest, Rsrc1, Shamt5 R 00 0x0b 0x0b - set if less than immediate sitiu Rdest, Rsrc1, Shamt5 R 00 0x0b 0x0b - set if less than immediate set if less than immediate set if less than immediate set if less than immediate set if less than immediate			Rdest, Rsrc1, Imm16				
Togicas e de	Operações	xor		R	00	0x26	
Sit Rdest, Rsrc1, Rsrc2 R 00 0x2a set if less than set if less than set if less than set if less than immediate set if less than immediate set if less than immediate set if less than unsigned imm. not Rdest, Rsrc1, Imm16 I 0x0b - set if less than unsigned imm. not Rdest, Rsrc1, Shamt5 R 00 00 Shift left logical shift right arithmetic rotate left rotate left rotate right J address28 J 00 03 shift right arithmetic rotate right J address28 J 02 J I J J J J J J J J J J J J J J J J J		xori	Rdest, Rsrc1, Imm16	1	0x0e	-	XOR immediate
Stit Rdest, Rsrc1, Imm16 I 0x0a - set if less than immediate set if less than unsigned set if less	e de	nor	Rdest, Rsrc1, Rsrc2	R	00	0x27	
Situ Rdest, Rsrc1, Rsrc2 R 00 0x2b set if less than unsigned shift sh	comparação					0x2a	
Sitiu Rdest, Rsrc1, Imm16							
Not Rdest, Rsrc							
SII				'	duxu	-	
Operações de deslocamento de bits srl Rdest, Rsrc1, Shamt5 R 00 00 03 shift right logical shift right arithmetic rol Rdest, Rsrc1, Rsrc2 ror Rdest, Rsrc1, Rsrc2 ror Rdest, Rsrc1, Rsrc2 rotate left rotate right j address28 jr Rsrc Nerc2 beq Rsrc1, Rsrc2, address18 bne Rsrc1, Rsrc2, address18 lone Rsrc1, Rsrc2, address18 lone Rsrc1, Rsrc2, address18 lone Rsrc1, Rsrc2, address18 lone Rsrc, address18 lone Rsrc, address18 lone Nor equal (relative addr) brown on the equal expression of the equal expre					00	00	
Sra Rdest, Rsrc1, Shamt5 R 00 03 Shift right arithmetic	Operaçãos do						
Tolate left	1						
ror Rdest, Rsrc1, Rsrc2 rotate right j address28 J 02 - jump (absolute addr) jr Rsrc R 00 08 jump register beq Rsrc1, Rsrc2, address18 J 04 - branch on equal (relative addr) bne Rsrc1, Rsrc2, address18 J 05 - branch on not equal (relative addr) branch on equal credit equal credit equal credit equal credit equal credit equal equal credit equal				- ' '			
j address28	do bito						
jr Rsrc		i		J	02	-	Ÿ
beq		<u>j́r</u>				80	
bne		beq		I		-	branch on equal (relative addr)
Description		bne	Rsrc1, Rsrc2, address18	I		-	branch on not equal (relative addr)
blez						01*	br. on greater than equal zero ("")
bltz	de salto			!			
jal address28 jalr Rsrc b address18/32 b <cnd> Rsrc1, Rsrc2, address18/32 b<cnd> Rsrc1, Rsrc2, address18/32</cnd></cnd></cnd></cnd></cnd></cnd></cnd></cnd></cnd></cnd></cnd></cnd></cnd></cnd></cnd></cnd></cnd></cnd></cnd></cnd></cnd></cnd></cnd></cnd></cnd></cnd></cnd></cnd></cnd></cnd></cnd></cnd></cnd></cnd></cnd></cnd></cnd></cnd></cnd></cnd></cnd></cnd></cnd></cnd></cnd></cnd></cnd></cnd></cnd></cnd></cnd></cnd></cnd></cnd></cnd></cnd></cnd></cnd></cnd>				!			
jalr Rsrc R 00 09 jump and link register b address18/32 branch inconditional (relative addr) b b br. on <cnd> = [gt, ge, lt, le] ("") b cnd>u Rsrc1, Rsrc2, address18/32 br. on <cnd> = [gt, ge, lt, le] uns. ("") br. on <cnd> = [gt, ge, lt, le] uns. ("") creturn from exception system call</cnd></cnd></cnd>							, ,
b address18/32 b <cnd> b ranch inconditional (relative addr) br. on <cnd> = [gt, ge, lt, le] ("") br. on <cnd> = [gt, ge, lt, le] ("") br. on <cnd> = [gt, ge, lt, le] uns. ("") br. on <cnd> = [gt, ge, lt, le] uns. ("") br. on <cnd> = [gt, ge, lt, le] uns. ("") br. on <cnd> = [gt, ge, lt, le] uns. ("") br. on <cnd> = [gt, ge, lt, le] uns. ("") br. on <cnd> = [gt, ge, lt, le] uns. ("") br. on <cnd> = [gt, ge, lt, le] uns. ("") br. on <cnd> = [gt, ge, lt, le] uns. ("") br. on <cnd> = [gt, ge, lt, le] uns. ("") br. on <cnd> = [gt, ge, lt, le] uns. ("") br. on <cnd> = [gt, ge, lt, le] uns. ("") br. on <cnd> = [gt, ge, lt, le] uns. ("") br. on <cnd> = [gt, ge, lt, le] uns. ("") br. on <cnd> = [gt, ge, lt, le] uns. ("") br. on <cnd> = [gt, ge, lt, le] uns. ("") br. on <cnd> = [gt, ge, lt, le] uns. ("") br. on <cnd> = [gt, ge, lt, le] uns. ("") br. on <cnd> = [gt, ge, lt, le] uns. ("") br. on <cnd> = [gt, ge, lt, le] uns. ("") br. on <cnd> = [gt, ge, lt, le] uns. ("") br. on <cnd> = [gt, ge, lt, le] uns. ("") br. on <cnd> = [gt, ge, lt, le] uns. ("") br. on <cnd> = [gt, ge, lt, le] uns. ("") br. on <cnd> = [gt, ge, lt, le] uns. ("") br. on <cnd> = [gt, ge, lt, le] uns. ("") br. on <cnd> = [gt, ge, lt, le] uns. ("") br. on <cnd> = [gt, ge, lt, le] uns. ("") br. on <cnd> = [gt, ge, lt, le] uns. ("") br. on <cnd> = [gt, ge, lt, le] uns. ("") br. on <cnd> = [gt, ge, lt, le] uns. ("") br. on <cnd> = [gt, ge, lt, le] uns. ("") br. on <cnd> = [gt, ge, lt, le] uns. ("") br. on <cnd> = [gt, ge, lt, le] uns. ("") br. on <cnd> = [gt, ge, lt, le] uns. ("") br. on <cnd> = [gt, ge, lt, le] uns. ("") br. on <cnd> = [gt, ge, lt, le] uns. ("") br. on <cnd> = [gt, ge, lt, le] uns. ("") br. on <cnd> = [gt, ge, lt, le] uns. ("") br. on <cnd> = [gt, ge, lt, le] uns. ("") br. on <cnd> = [gt, ge, lt, le] uns. ("") br. on <cnd> = [gt, ge, lt, le] uns. ("") br. on <cnd> = [gt, ge, lt, le] uns. ("") br. on <cnd> = [gt, ge, lt, le] uns. ("") br. on <cnd> = [gt, ge, lt, le] uns. ("") br. on <cnd> = [gt, ge, lt, le] uns. ("") br. on <cnd> = [gt, ge</cnd></cnd></cnd></cnd></cnd></cnd></cnd></cnd></cnd></cnd></cnd></cnd></cnd></cnd></cnd></cnd></cnd></cnd></cnd></cnd></cnd></cnd></cnd></cnd></cnd></cnd></cnd></cnd></cnd></cnd></cnd></cnd></cnd></cnd></cnd></cnd></cnd></cnd></cnd></cnd></cnd></cnd></cnd></cnd></cnd></cnd></cnd></cnd></cnd>							
$b < cnd > Rsrc1, Rsrc2, address18/32 \\ b < cnd > u Rsrc1, Rsrc2, address18/32 \\ rfe \\ Excepção $				11	- 50	09	
b <cnd>u Rsrc1, Rsrc2, address18/32 br. on <cnd> = [gt, ge, lt, le] uns. ("") rfe R 0x10 0x20 return from exception syscall R 00 0x0c system call</cnd></cnd>		-					
rfe R 0x10 0x20 return from exception Excepção syscall R 00 0x0c system call							
Excepção syscall R 00 0x0c system call				R	0x10	0x20	
	Excepção						
produce 1 to over broak	- 21-3	break	code20	R	00	0x0d	break

* especificado no campo rt

Formatos das instruções e exemplos

	nstrução	Tipo	Op/6 bits	Rs/5 bits	Rt/5 bits	Rd/5 bits	Sa/5 bits	Func/6 bits
add	\$a0, \$t0, \$s0	R	0	8 (\$t0)	0x10 (\$s0)	4 (\$a0)	0	0x20
addi	\$8, \$9, 0x100	1	8	9	8	0x100		
sw	\$6, -4 (\$7)	- 1	0x2b	7	6	0xfffc (-4)		
beq	\$4, \$5, 0x60	1	4	4	5	0x18 (0x60/4)		
j	0x80000	J	2	0x20000 (0x80000/4)				

Convenção dos registos do MIPS

Nome	Número	Utilização	Preservado na chamada?
\$zero	0	Constante 0	n.a.
\$v0-\$v1	2-3	Valores para resultados e avaliação de expressões	Não
\$a0-\$a3	4-7	Argumentos	Sim
\$t0-\$t7	8-15	Temporários	Não
\$s0-\$s7	16-23	Seguros	Sim
\$t8-\$t9	24-25	Mais temporários	Não
\$gp	28	Apontador global	Sim
\$sp	29	Apontador para pilha	Sim
\$fp	30	Apontador para a frame	Sim
\$ra	31	Endereço de retorno	Sim

O registo 1 (\$at) está reservado para o assemler, os registos 26-27 (\$k0-\$k1) estão reservados para o sistema operativo