Guia de instrução rápida

CONJUNTO DE INSTRUÇÕES BÁSICO /

					OPCODE/
NOME	MNEMÔ- NICO	FORMATO	OPERAÇÃO (EM VERILOG)		FUNCT (HEXA)
Add	add	R	R[rd] = R[rs] + R[rt]	(1)	0/20 _{hexa}
Add Immediate	addi	1	R[rt] = R[rs] + ImSinExt	(1)(2)	8 _{hexa}
Add Imm. Unsigned	addiu	1	R[rt] = R[rs] + ImSinExt	(2)	9 _{hexa}
Add Unsigned	addu	R	R[rd] = R[rs] + R[rt]		0/21 _{hexa}
And	and	R	R[rd] = R[rs] & R[rt]		0/24 _{hexa}
And Immediate	andi	1	R[rt] = R[rs] & ImZeroExt	(3)	c _{hexa}
Branch On Equal	beq	1	if(R[rs]==R[rt]) PC=PC+4+ EndBranch	(4)	4 _{hexa}
Branch On Not Equal	bne	1	If(R[rs]!=R[rt]) PC=PC+4+EndBranch	(4)	5 _{hexa}
Jump	j	J	PC=EndJump	(5)	2 _{hexa}
Jump And Link	jal	J	R[31]=PC+4;PC=EndJump	(5)	3 _{hexa}
Jump Register	jr	R	PC=R[rs]		0/08 _{hexa}
Load Byte Unsigned	lbu		$ R[rt]=\{24'b0,M[R[rs]\\+lmSinExt](7:0)\} $	(2)	0/24 _{hexa}
Load Halfword Unsigned	lhu	Ê	$R[rt]=\{16'b0,M[R[rs]+ ImSinExt](15:0)\}$	(2)	0/25 _{hexa}
Load Upper Imm.	lui	Ĺ	$R[rt] = \{imm, 16'b0\}$		f _{hexa}
Load Word	lw	Ī	R[rt] = M[R[rs] + ImSinExt]	(2)	0/23 _{hexa}
Nor	nor	R	$R[rd] = \sim (R[rs] R[rt])$		0/27 _{hexa}
Or o	or	R	R[rd] = R[rs] R[rt]		0/25 _{hexa}
Or Immediate	ori	1	$R[rt] = R[rs] \mid ImZeroExt$	(3)	d _{hexa}
Set Less Than	sit	R	R[rd] = (R[rs] < R[rt]) ? 1 : 0		0/2a _{hexa}
Set Less Than Imm.	slti	I)	R[rt] = (R[rs] < ImSinExt) ? 1 :0	(2)	a _{hexa}
Set Less Than Imm. Unsigned	sltiu	Ü	R[rt] = (R[rs] < ImSinExt) ? 1 : 0	(2)(6)	b _{hexa}
Set Less Than Unsigned	sltu	R	R[rd] = (R[rs] < R[rt]) ? 1 : 0	(6)	0/2b _{hexa}
Shift Left Logical	sll	R	R[rd] = R[rs] << shamt		0/00 _{hexa}
Shift Right Logical	srl	R	R[rd] = R[rs] >> shamt		0/02 _{hexa}
Store Byte	sb	1	M[R[rs]+ImSinExt](7:0) = R[rt](7:0))(2)	28 _{hexa}
Store Halfword	sh	I	$\begin{split} &\text{M[R[rs]+ ImSinExt](15:0)} = \\ &\text{R[rt](15:0)} \end{split}$	(2)	29 _{hexa}
Store Word	sw	Ĭ.	M[R[rs]+ImSinExt] = R[rt]	(2)	2b _{hexa}
Subtract	sub	R	R[rd] = R[rs] - R[rt]	(1)	0/22 _{hexa}
Subtract Unsigned	subu	R	R[rd] = R[rs] - R[rt]		0/23 _{hexa}

- (1) Pode causar exceção de overflow

- (2) ImmSinExt = { 16(imediato[15]}, imediato }
 (3) ImmZeroExt = { 16(ib'0}, imediato }
 (4) EndBranch = { 14(imediato [15]), imediato, 2'b0 }
 (5) JumpAddr = { PC[31:28], address, 2'b0 }
- (6) Operandos considerados números sem sinal (vs. compl. 2)

FORMATOS DE INSTRUÇÃO BÁSICOS

opcode	rs	rt	Rd		shamt		funct	
31	26 25	21 20	16 15	11	10	6	5	0
opcode	rs	rt	imediat	:0				
31	26 25	21 20	16 15					0
opcode	Endereç	0						
31	26 25							0

CONJUNTO DE INSTRUÇÕES BÁSICAS ARITMÉTICAS

0

NOME	MNEMÔ- NICO	FORMATO	OPERAÇÃO	OPCODE/FMT/ FT/FUNCT (HEXA)
Branch On FP True	belt	FI	if(FPcond)PC=PC+4+EndBranch (4)	11/8/1/-
Branch On FP False	bclf	FI	if(!FPcond)PC=PC+4+EndBranch (4)	11/8/0/-
Divide	div	R	Lo=R[rs]/R[rt]; Hi=R[rs]%R[rt]	0/-/-/1a
Divide Unsigned	divu	R	Lo=R[rs]/R[rt]; Hi=R[rs]%R[rt] (6)	0/-/-/1b
FP Add Single	add.s	FR	F[fd] = F[fs] + F[ft]	11/10/-/0
FP Add Double	Add.d	FR	${F[fd],F[fd+1]} = {F[fs],F[fs+1]} + {F[ft],F[ft+1]}$	11/11/-/0
FP Compare Single	C.x.s*	FR	FPcond = (F[fs] op F[ft]) ? 1:0	11/10/-/y
FP Compare Double	C.x.d*	FR	FPcond = ({F[fs],F[fs+1]} op {F[ft],F[ft+1]})? 1:0	11/11/-/y
	* (x é e	q, lt ou le)	(op é ==, < ou <=) (y é 32, 3c ou 3e)	
FP Divide Single	div.s	FR	F[fd] = F[fs]/F[ft]	11/10/-/3
FP Divide Double	div.d	FR	${F[fd],F[fd+1]} = {F[fs],F[fs+1]}/$ ${F[ft],F[ft+1]}$	11/11/-/3
FP Multiply Single	mul.s	FR	F[fd] = F[fs] * F[ft]	11/10/-/2
FP Multiply Double	mul.d	FR	${F[fd],F[fd+1]} = {F[fs],F[fs+1]}*$ ${F[ft],F[ft+1]}$	11/11/-/2
FP Subtract Single	sub.s	FR	F[fd]=F[fs]-F[ft]	11/10/-/1
FP Subtract Double	sub.d	FR	${F[fd],F[fd+1]} = {F[fs],F[fs+1]}-{F[ft],F[ft+1]}$	11/11/-/1
Load FP Single	lwc1	I)	F[rt]=M[R[rs]+ ImSinExt] (2)	31/-/-/-
Load FP Double	ldc1	E	F[rt]=M[R[rs]+ ImSinExt]; F[rt+I]=M[R[rs]+ ImSinExt+4] (2)	35/-/-/-
Move From Hi	mfhi	R	R[rd] = Hi	0/-/-/10
Move From Lo	mflo	R	R[rd] = Lo	0/-/-/12
Move From Control	mfc0	R	R[rd] = CR[rs]	16/0/-/0
Multiply	mult	R	${Hi,Lo} = R[rs] * R[rt]$	0/-/-/18
Multiply Unsigned	multu	R	${Hi,Lo} = R[rs] * R[rt] (6)$	0/-/-/19
Store FP Single	swc1	1	M[R[rs]+ImSinExt] = F[rt] (2)	39/-/-/-
Store FP Double	sdc1	1 .	M[R[rs]+ ImSinExt] = F[rt]; (2) M[R[rs]+ ImSinExt+4] = F[rt+1]	3d/-/-/-

FORMATOS DAS INSTRUÇÕES DE PONTO FLUTUANTE

R	opcode	fmt	ft	fs	fd	funct	
	31	26 25	21 20	16 15	11 10	6 5	0
ı	opcode	fmt	ft	imediato			
	31	26 25	21 20	16 15			0

CONJUNTO DE PSEUDO-INSTRUÇÕES

NOME	MNEMÔNICO	OPERAÇÃO
Branch Less Than	bit	if(R[rs] <r[rt]) pc="Label</td"></r[rt])>
Branch Greater Than	bgt	if(R[rs]>R[rt]) PC = Label
Branch Less Than or Equal	Ble	if(R[rs]<=R[rt]) PC = Label
Branch Greater Than or Equal	Bge	if(R[rs]>=R[rt]) PC = Label
Load Immediate	Li	R[rd] = imediato
Move	Move	R[rd] = R[rs]

NOME DO REGISTRADOR, NÚMERO, USO E CONVENÇÃO DE CHAMADA

NOME	NÚMERO	USO	PRESERVADO ENTRE CHAMADAS?
\$zero	0	O valor constante O	N.A.
\$at	1	Temporário do montador	Não
\$v0-\$v1	2-3	Valores para resultados de função e avaliação de expressão	Não
\$a0-\$a3	4-7	Argumentos	Não
St0-\$t7	8-15	Temporários	Não
\$s0-\$s7	16-23	Temporários salvos	Sim
\$t8-\$t9	24-25	Temporários	Não
\$k0-\$k1	26-27	Reservado para kernel do SO	Não
\$gp	28	Ponteiro global	Sim
\$sp	29	Stack Pointer	Sim
\$fp	30	Frame Pointer	Sim
\$ra	31	Endereço de retorno	Sim

OPCODES, CONVERSÃO DE BASE, SÍMBOLOS ASCII

OPCODE MIPS (31:26)	(1) MIPS FUNCT (5:0)	(2) MIPS FUNCT (5:0)	BINÁRIO	DECIMAL	HEXA	ASCII	DECIMAL	HEXA	ASCII
(1)	sll	add.f	00 0000	0	0	NUL	64	40	@
		sub.f	00 0001	1	1	SOH	65	41	Α
j	srl	mul.f	00 0010	2	2	STX	66	42	В
jal	sra	div.f	00 0011	3	3	ETX	67	43	С
beq	sllv	sqrt.f	00 0100	4	4	EOT	68	44	D
bne		abs.f	00 0101	5	5	ENQ	69	45	E
blez	srlv	mov.f	00 0110	6	6	ACK	70	46	F
bgtz	srav	neg.f	00 0111	7	7	BEL	71	47	G
addi	jr		00 1000	8	8	BS	72	48	Н
addiu	jalr		00 1001	9	9	HT	73	49	1
slti	movz		00 1010	10	а	LF	74	4a	J
sltiu	movn		00 1011	11	b	VT	75	4b	K
andi	syscali	round.w.f	00 1100	12	С	FF	76	4c	L
ori	break	trunc.w.f	00 1101	13	d	CR	77	4d	M
xori		ceil.w.f	00 1110	14	е	SO	78	4e	N
lui	sync	floor.w.f	00 1111	1.5	f	SI	79	4f	0
200	mfhi		01 0000	16	10	DLE	80	50	Р
(2)	mthi		01 0001	17	11	DC1	81	51	Q
	mflo	movz.f	01 0010	18	12	DC2	82	52	R
	mtlo	movn.f	01 0011	19	13	DC3	83	53	S
			01 0100	20	14	DC4	84	54	T
			01 0101	21	15	NAK	85	55	U
			01 0110	22	16	SYN	86	56	V
	- 20		01 0111	23	17	ETB	87	57	W
	mult		01 1000	24	18	CAN	88	58	X
	multu		01 1001	25	19	EM	89	59	Υ
	div		01 1010	26	1a	SUB	90	5a	Z
	divu		01 1011	27	1b	ESC	91	5b	
			01 1100	28	1c	FS	92	5c	/
			01 1101	29	1d	GS	93	5d	1
			01 1110	30	1e	RS	94	5e	٨
			01 1111	31	1f	US	95	5f	
lb	add	cvt.s.f	10 0000	32	20	Espaço	96	60	
lh	addu	cvt.d.f	10 0001	33	21	!	97	61	а
lwl	sub		10 0010	34	22		98	62	b
lw	subu		10 0011	35	23	#	99	63	С
lbu	and	cvt.w.f	10 0100	36	24	\$	100	64	d
lhu	or		10 0101	37	25	%	101	65	е
lwr	xor		10 0110	38	26	&	102	66	f
	nor		10 0111	39	27		103	67	g
sb			10 1000	40	28	(104	68	h
sh	CIF		10 1001		29)	105	69	i
swl	SIt		10 1010		2a	*	106	6a	j
SW	Sltu		10 1011	43	2b	+	107	6b	k
			10 1100		2c	•	108	6c	1
21112			10 1101		2d	-	109	6d	m
swr			10 1110		2e	,	110	6e	n
cache	tan	- 54	10 1111	0836	2f	/ -	111	6f	0
1	tge	c.f.f	11 0000		30	0	112	70	P
wc1	tgeu	c.un.f	11 0001		31	1	113	71	q
wc2	tlt	c.eq.f	11 0010		32	2	114	72	r
oref	tltu	c.ueq.f	11 0011	51	33	3	115	73	S
	teq	colt.f		52	34	4	116	74	t
dc1		cult.f	11 0101		35	5	117	75 70	u
dc2	tne	cole.f	11 0110		36	6	118	76	V
		c.ule.f	11 0111	55	37	7	119	77	w
3C		c.sf.f		56	38	8	120	78	X
swc1		c.ngle.f	11 1001	57	39	9	121	79 70	У
swc2		c.seq.f	11 1010		3a		122	7a	Z
		c.ngl.f		59	3b	;	123	7b	{
		en a company and a company	11 1100		3c	<	124	7c	1
sdc1		c.nge.f	11 1101		3d	=	125	7d	}
sdc2		c.le.f	11 1110		3e	>	126	7e	~
		c.ngt.f	11 1111	63	3f	?	127	7f	DEL

⁽¹⁾ opcode(31:26)==0

PADRÃO IEEE 754 DE PONTO FLUTUANTE

(-1)^S x (1 + Fração) x 2^(Expoente-Bias) onde Bias Precisão simples = 127, Bias Precisão dupla = 1023.

Símbolos IEEE 754

Exponente	Fração	Objeto
0	0	±0
0	≠0	± Denorm
La MAX – 1	qualquer coisa	± Núm. Pt. Flut.
MAX	0	±∞
MAX	≠0	NaN

S.P. MAX = 255, D.P. MAX = 2047

Formatos IEEE de precisão simples e dupla:

S	Expoente	Fração
31 3	0 23 22	
S	Expoente	Fração
63 6	2 52.51	

ALOCAÇÃO DE MEMÓRIA FRAME DE PILHA Endereços \$sp ---> 7fff fffc_{hexa} de memória Argumento 6 maiores Argumento 5 \$fp -Registradores salvos Pilha Dados dinâmicos \$gp - 1000 8000_{hexa} Cresce Dados estáticos Variáveis 1000 0000_{hexa} locais \$sp-Texto pc → 0040 0000_{hexa} Endereços de memória Reservado menores O_{hexa}

ALINHAMENTO DE DADOS

Word dupla								
Word				Word				
Half word	Half	word		Half wo	ord	Half w	ord	
Byte Byte	Byte	Byte		Byte E	Byte	Byte	Byte	
0	1	2	3	4	5	6	7	

Valor de três bits menos significativos do endereço de byte (Big Endian)

REGISTRADORES DE CONTROLE DE EXCEÇÃO: CAUSA E STATUS

B D		Máscara de interrupção		Interrup pende			L.
31	15	8	6	4	2		
		Código de exceção		U M		E L	I E
	15	8		4		1	0

BD = Branch Delay, UM = User Mode, EL = Exception Level, IE = Interrupt Enable

CÓDIGOS DE EXCEÇÃO

NÚMERO NOME		CAUSA DA EXCEÇÃO	NÚMERO	NOME	CAUSA DA EXCEÇÃO
0	Int	Interface (hardware)	9	Вр	Exceção de breakpoint
4	AdEL	Address Error Exception (load ou busca de instrução)	10	RI	Exceção de instrução reservada
5	AdES	Address Error Exception (store)	11	CpU	Co-processador não implementado
6	IBE	Erro de barramento na busca de instrução	12	Ov	Exceção de overflow aritmético
7	DBE	Erro de barramento no Load ou Store	13	Tr	Trap
8	Sys	Exceção de Syscall	15	FPE	Exceção de ponto flutuante

PREFIXOS DE TAMANHO (10^X para Disco, Comunicação; 2^X para Memória)

TAMANHO	PREFIXO	TAMANHO	PREFIXO	TAMANHO	PREFIXO	TAMANHO	PREFIXO
10 ³ ,2 ¹⁰	Kilo-	10 ¹⁵ ,2 ⁵⁰	Peta-	10-3	mili-	10 ⁻¹⁵	femto-
10 ⁶ ,2 ²⁰	Mega-	10 ¹⁸ ,2 ⁶⁰	Exa-	10.6	micro-	10 ⁻¹⁸	atto-
10 ⁹ ,2 ³⁰	Giga-	10 ²¹ ,2 ⁷⁰	Zetta-	10.9	nano-	10 ⁻²¹	zepto-
10 ¹² ,2 ⁴⁰	Tera-	10 ²⁴ ,2 ⁸⁰	Yotta-	10-12	pico-	10 ⁻²⁴	yocto-

O símbolo para cada prefixo é apenas sua primeira letra, exceto que é usado para micro.

⁽²⁾ opcode(31:26) = 17_{dec} (11_{hexa}); if fmt(25:21)== 16_{dec} (10_{hexa}) f= s (single); if fmt(25:21)== 17_{dec} (11_{hexa}) f = d (double)