Instruções da Máquina Nativa				Instruções da Máquina Virtual				DETI-UA - ACI
Transferê	ncia Memória-Registo ( <i>Load</i> )	-		Transferência Memória-Registo (Load)		Salto Relativo (Branch)		
lb	Rdst,addr	add	Rdst,Rsrc1,Rsrc2	1.d	FPdst,addr	b	Label	
lbu	Rdst,addr	addi	Rdst,Rsrc,Imm	1.s	FPdst,addr	beqz	Rsrc, Labe	1
lw	Rdst,addr	addiu	Rdst,Rsrc,Imm			bge	Rsrc,Src,L	abel
lwcz	CReg,addr	addu	Rdst,Rsrc1,Rsrc2	Transfe	rência Registo-Memória (Store)	bgeu	Rsrc,Src,L	abel
		div	Rsrc1,Rsrc2	s.d	FPsrc,addr	bgt	Rsrc,Src,L	abel
Transferê	ncia Registo-Memória (Store)	divu	Rsrc1,Rsrc2	s.s	FPsrc,addr	bgtu	Rsrc,Src,L	abel
sb	Rsrc,addr	mult	Rsrc1,Rsrc2			ble	Rsrc,Src,L	abel
sw	Rsrc,addr	multu	Rsrc1,Rsrc2	Transfe	rência Registo-Registo (Move)	bleu	Rsrc,Src,L	abel
SWCZ	Creg,addr	sub	Rdst,Rsrc1,Rsrc2	move	Rdst,Rsrc	blt	Rsrc,Src,L	abel
		subu	Rdst,Rsrc1,Rsrc2			bltu	Rsrc,Src,L	abel
Transferê	ncia Registo-Registo (Move)	Cálculo c/	Inteiros: Op. Lógicas Bitwise	Manipu	lação de Const. (Load Imm/sym)	bnez	Rsrc,Label	
mfhi	Rdst	and	Rdst,Rsrc1,Rsrc2	la	Rdst,sym			
mflo	Rdst	andi	Rdst,Rsrc,Imm	li	Rdst,IMM			
mthi	Rsrc	nor	Rdst,Rsrc1,Rsrc2	1.d	FPdst,sym			
mtlo	Rsrc	or	Rdst,Rsrc1,Rsrc2	1.s	FPdst,sym			
${\tt mfc} z$	Rdst,Creg	ori	Rdst,Rsrc,Imm					
mtcz	Rsrc,Creg	xor	Rdst,Rsrc1,Rsrc2	Cálculo	c/ Inteiros: Op. Aritméticas			
mov.d	FPdst,FPsrc	xori	Rdst,Rsrc,Imm	abs	Rdst,Rsrc			
mov.s	FPdst,FPsrc	Cálculo c/	Inteiros: Operações de Shift	div	Rdst,Rsrc,Src			
		sll	Rdst,Rsrc1,Imm5	divu	Rdst,Rsrc,Src			
Manipula	ção de Const. (Load Immediate)	sllv	Rdst,Rsrc1,Rsrc2	mul	Rdst,Rsrc,Src			
lui	Rdst,Imm	sra	Rdst,Rsrc1,Imm5	mulu	Rdst,Rsrc,Src		Tabela I: R	egistos do MIPS e convenção de uso
		srav	Rdst,Rsrc1,Rsrc2	mulo	Rdst,Rsrc,Src	Nome Lo		Uso Convencionado
Instruções	s de Comparação	srl	Rdst,Rsrc1,Imm5	mulou	Rdst,Rsrc,Src	\$zero	\$0	Constante 0
slt	Rdst,Rsrc1,Rsrc2	srlv	Rdst,Rsrc1,Rsrc2	neg	Rdst,Rsrc	\$at	\$1	Reservado pelo assemblador
sltu	Rdst,Rsrc1,Rsrc2	Cálculo er	n Vírgula Flutuante	negu	Rdst,Rsrc	\$v0\$v1	\$2\$3	Cálculo de expressões e valor de retorno das funções.
slti	Rdst,Rsrc,Imm	abs.p	FPdst, FPsrc	rem	Rdst,Rsrc,Src	\$a0\$a3	\$4\$7	Primeiros 4 parâmetros das funções
sltiu	Rdst,Rsrc,Imm	add.p	FPdst,FPsrc1,FPsrc2	remu	Rdst,Rsrc,Src	\$t0\$t7	\$8\$15	Geral (não são preservados pelas funções)
		c.eq.p	FPsrc1,FPsrc2	Cálculo	c/ Inteiros: Op. Lógicas Bitwise	\$s0\$s7	\$16\$23	Geral (não podem ser alterados pelas funções)
Salto Rela	tivo (Branch) e Absoluto (Jump)	c.le.p	FPsrc1,FPsrc2	not	Rdst,Rsrc	\$t8\$t9	\$24\$25	Geral (não são preservados pelas funções)
bczf	Label	c.lt.p	FPsrc1,FPsrc2			\$k0\$k1	\$26\$27	Reservado pelo kernel do S.O.
bczt	Label		FPdst,FPsrc	Cálculo	c/ Inteiros: Operações de <i>Rotate</i>	\$gp	\$28	Ponteiro para área global (Global Pointer)
beq	Rsrc1,Rsrc2,Label	cvt.d.w	FPdst,FPsrc	rol	Rdst,Rsrc,Src	\$sp	\$29	Stack Pointer
bgez	Rsrc,Label	cvt.s.d	FPdst,FPsrc	ror	Rdst,Rsrc,Src	\$fp	\$30	Frame Pointer
bgezal	Rsrc,Label		FPdst,FPsrc		, ,	\$ra	\$31	Endereço de retorno das funções ( <i>Return Address</i> )
bgtz	Rsrc,Label		FPdst,FPsrc	Instrucô	es de Comparação	4	[44.5	
blez	Rsrc,Label		FPdst,FPsrc	seq	Rdst,Rsrc,Src	r	Tabela II: Regis	tos da FPU do MIPS e convenção de uso
bltz	Rsrc,Label	div.p	FPdst,FPsrc1,FPsrc2	sge	Rdst,Rsrc,Src	Nome L		Uso Convencionado
bltzal	Rsrc,Label	mul.p	FPdst,FPsrc1,FPsrc2	sgeu	Rdst,Rsrc,Src		\$f2(\$f3)	Cálculo de expressões e valor de retorno das funções
bne	Rsrc1,Rsrc2,Label	neg.p	FPdst,FPsrc	sgt	Rdst,Rsrc,Src		\$f10(\$f11)	Geral (não são preservados pelas funções)
j	Label	sub.p	FPdst, FPsrc1, FPsrc2	sgtu	Rdst,Rsrc,Src			Passagem de parâmetros para funções.
jal	Label	-	ção de Excepções e <i>Traps</i>	sle	Rdst,Rsrc,Src			Geral (não são preservados pelas funções)
jalr	Rsrc	break	n	sleu	Rdst,Rsrc,Src			Geral (não podem ser alterados pelas funções)
jr	Rsrc	nop		sne	Rdst,Rsrc,Src	Ψ120(Ψ12	-, φιο σ(φιο 1)	, I
	-	eret			-,, <del></del>			Rev 2016 - MBC, JLA, AO, LAU, ACF
1		1		1		1		• • • • •

syscall

## DETI-UA - ACI

Tabela III:	Notação		
Imm	Valor imediato (constante) de 16 bits	addr	Endereço na forma $Imm(Rsrc) = (Rsrc) + Imm$
IMM	Valor imediato de 32 bits	B <sub>k</sub> (Rsrc)	Byte índice k de Rsrc
Rsrc(1,2)	Registo fonte (1 ou 2)	FPdst	Registo destino do coprocessador aritmético
(Rsrc)	Conteudo de <b>Rsrc</b>	FPsrc(1,2)	Registo fonte do coprocessador aritmético (1 ou 2)
Rdst	Registo destino	Cz	Coprocessador nº z
CReg	Registo do Coprocessador C <sub>z</sub>	Src	Rsrc ou IMM
sym	Endereço do símbolo (label) sym	Imm5	Valor imediato (constante) de 5 bits

Tabela V - Directivas do Assembler						
Directivas	Descrição					
Para controlo dos Segmentos						
.data [address]	Coloca os próximos items no segmento de dados do utilizador (opcionalmente a partir de address).					
.text [address]	Coloca os próximos items no segmento de código do utilizador (opcionalmente a partir de address).					
.kdata [address]	Coloca os próximos items no segmento de dados do kernel (opcionalmente a partir de address).					
.ktext [address]	Coloca os próximos items no segmento de código do kernel (opcionalmente a partir de address).					
Para criação de constantes e variáveis em memória:						
.ascii str	Armazena uma string em memória sem lhe acrescentar o terminador '\0'.					
.eqv label, valor	Substitui todas as ocorrências de label no programa por valor.					
.asciiz str	Armazena uma string em memória acrescentando-lhe o terminador \0'.					
.byte $b_1$ ,, $b_n$	Armazena as grandezas de 8 bits b <sub>1</sub> ,, b <sub>n</sub> em sucessivos bytes de memória.					
.half $h_1$ ,, $h_n$	Armazena as grandezas de 16 bits h <sub>1</sub> ,, h <sub>n</sub> em sucessivas meias palavras de memória.					
.word $w_1$ ,, $w_n$	Armazena as grandezas de 32 bits $w_1$ ,, $w_n$ em sucessivas palavras de memória.					
.float $f_1$ ,, $f_n$	Armazena f <sub>1</sub> ,, f <sub>n</sub> em vírgula flutuante, precisão simples (32 bits) no seg. de dados.					
.double $d_1$ ,, $d_n$	Armazena d <sub>1</sub> ,, d <sub>n</sub> em vírgula flutuante, precisão dupla (64 bits) no seg. de dados.					
.space n	Reserva n bytes no segmento de dados, sem inicializar					
Para controlo do alinhamento:						
.align n	Alinha o próximo item num endereço múltiplo de 2 <sup>n</sup> .					
Para referências externas:						
.globl sym	Declara que o símbolo sym é global e pode ser referenciado em outros ficheiros.					
.extern sym size	Declara que o item associado a sym ocupa size bytes e é um símbolo global.					

Tabela IV: System Calls do MARS								
Protótipo equivalent em C		Parâmetros de entrada	Retorno					
<pre>void print_int10(int value)</pre>	1	a0 = int						
<pre>void print_float(float value)</pre>	2	f12 = float						
<pre>void print_double(double value)</pre>	3	\$f12 = double						
<pre>void print_string(char *str)</pre>	4	\$a0 = string						
<pre>int read_int(void)</pre>	5		\$v0					
float read_float(void)	6		\$f0					
double read_double(void)	7		\$f0					
<pre>void read_string(char *buf, int len)</pre>	8	a0 = buf, $a1 = length$						
<pre>void *sbrk(int amount)</pre>	9	a0 = amount	\$v0					
void exit(void)	10							
<pre>void print_char(char value)</pre>	11	a0 = character						
char read_char(void)	12		\$v0					
<pre>void print_int16(unsigned int value)</pre>	34	\$a0						
<pre>void print_int2(unsigned int value)</pre>	35	\$a0						
<pre>void print_intu10(unsigned int value)</pre>	36	\$a0						