

Instruções da Máquina Nativa		Instruções da Máquina Virtual			
Transferência Memória-Registo (<i>Load</i>)		Cálculo c/ Inteiros: Operações Aritméticas		Salto Relativo (<i>Branch</i>)	
lb	Rdst, addr	add	Rdst, Rsrc1, Rsrc2	b	Label
lbu	Rdst, addr	addi	Rdst, Rsrc, Imm	beqz	Rsrc, Label
lw	Rdst, addr	addiu	Rdst, Rsrc, Imm	bge	Rsrc, Src, Label
		addu	Rdst, Rsrc1, Rsrc2	bgeu	Rsrc, Src, Label
		sub	Rdst, Rsrc1, Rsrc2	bgt	Rsrc, Src, Label
		subu	Rdst, Rsrc1, Rsrc2	bgtu	Rsrc, Src, Label
				ble	Rsrc, Src, Label
				bleu	Rsrc, Src, Label
				blt	Rsrc, Src, Label
				bltu	Rsrc, Src, Label
				bnez	Rsrc, Label
Transferência Registo-Memória (<i>Store</i>)		Transferência Registo-Registo (<i>Move</i>)			
sb	Rsrc, addr	move	Rdst, Rsrc		
sw	Rsrc, addr				
		Manipulação de Const. (<i>Load Imm/sym</i>)			
		la	Rdst, sym		
		li	Rdst, IMM		
		Cálculo c/ Inteiros: Op. Aritméticas			
		abs	Rdst, Rsrc		
		div	Rdst, Rsrc, Src		
		divu	Rdst, Rsrc, Src		
		mul	Rdst, Rsrc, Src		
		mulo	Rdst, Rsrc, Src		
		mulou	Rdst, Rsrc, Src		
		rem	Rdst, Rsrc, Src		
		remu	Rdst, Rsrc, Src		
		neg	Rdst, Rsrc		
		negu	Rdst, Rsrc		
		Cálculo c/ Inteiros: Op. Lógicas Bitwise			
		not	Rdst, Rsrc		
		Cálculo c/ Inteiros: Operações de Rotate			
		rol	Rdst, Rsrc, Src		
		ror	Rdst, Rsrc, Src		

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_k(Rsrc)	Byte índice k de Rsrc
Rsrc (1, 2)	Registo fonte (1 ou 2)	FPdst	Registo destino do coprocessador aritmético
(Rsrc)	Conteúdo de Rsrc	FPsrc (1, 2)	Registo fonte do coprocessador aritmético (1 ou 2)
Rdst	Registo destino	C_z	Coprocessador nº z
CReg	Registo do Coprocessador C_z	Src	Rsrc ou IMM
sym	Endereço do símbolo (label) sym	Imm5	Valor imediato (constante) de 5 bits

Tabela IV: <i>System Calls</i> da placa DETPIC32			
Protótipo equivalent em C	\$v0	Parâmetros de entrada	Retorno
char inkey(void)	1		\$v0
char getChar(void)	2		\$v0
void putChar(char ch)	3	\$a0 = character	
unsigned int readInt(unsigned int base)	4	\$a0 = base	\$v0
int readInt10(void)	5		\$v0
void printInt(unsigned int val, unsigned int base)	6	\$a0 = val, \$a1 = base	
void printInt10(int val)	7	\$a0	
void printStr(char *str)	8	\$a0 = str	
void readStr(char *buffer, unsigned int nc)	9	\$a0 = buffer, \$a1 = nc	
void exit(int code)	10	\$a0 = exit code	
unsigned int readCoreTimer(void)	11		\$v0
void resetCoreTimer(void)	12		

printInt(), "base": **16 lsbits** – [2.. 16], **16 msbits** – número de caracteres com que o resultado é apresentado (o valor por omissão é 0, i.e. sem formatação)

Tabela V - Directivas do Assembler	
Directivas	Descrição
Para controlo dos Segmentos	
.data [address]	Coloca os próximos itens no segmento de dados do utilizador (opcionalmente a partir de <i>address</i>).
.text [address]	Coloca os próximos itens no segmento de código do utilizador (opcionalmente a partir de <i>address</i>).
.kdata [address]	Coloca os próximos itens no segmento de dados do <i>kernel</i> (opcionalmente a partir de <i>address</i>).
.ktext [address]	Coloca os próximos itens no segmento de código do <i>kernel</i> (opcionalmente a partir de <i>address</i>).
Para criação de constantes e variáveis em memória:	
.ascii str	Armazena uma <i>string</i> em memória sem lhe acrescentar o terminador '\0'.
.asciiz str	Armazena uma <i>string</i> em memória acrescentando-lhe o terminador '\0'.
.byte b ₁ , ..., b _n	Armazena as grandezas de 8 bits b ₁ , ..., b _n em sucessivos bytes 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 ₁ , ..., f _n em vírgula flutuante, precisão simples (32 bits) no seg. de dados.
.double d ₁ , ..., d _n	Armazena d ₁ , ..., d _n em vírgula flutuante, precisão dupla (64 bits) no seg. de dados.
.space n	Reserva <i>n</i> bytes no segmento de dados, sem inicializar
.equ name, val	Atribui ao símbolo "name" o valor "val"
Para controlo do alinhamento:	
.align n	Alinha o próximo item num endereço múltiplo de 2 ⁿ .
Para referências externas:	
.globl sym	Declara que o símbolo <i>sym</i> é global e pode ser referenciado em outros ficheiros.
.extern sym size	Declara que o item associado a <i>sym</i> ocupa <i>size</i> bytes e é um símbolo global.
.include <filename>	Inclui o ficheiro especificado no campo "filename"

Tabela VI: Registos do CP0 do MIPS		
Nome Lógico	Nome Real	Conteúdo
\$BadVAddr	\$8	Endereço de memória inválido que causou a exceção
\$Status	\$12	Interrupt mask & Enable bits
\$Cause	\$13	Tipo de exceção e interrupt bits
\$EPC	\$14	Endereço da instrução que causou a exceção
Tabela VII: Valores dos bits [5..2] do registo Cause		
Valor	Nome	Significado
0	INT	External Interrupt
4	ADDRL	Add error exception (load or store)
5	ADDRS	Add error exception (fetch)
6	IBUS	Bus error on instruction fetch
7	DBUS	Bus error on data load or store
8	SYSCALL	Syscall exception
9	BKPT	Break point exception
10	RI	Reserved instruction exception
12	OVF	Overflow exception