

### MIPS operands

Name	Example	Comments
32 registers	\$0, \$1, \$2, . . . , \$31, Hi, Lo	Fast locations for data. In MIPS, data must be in registers to perform arithmetic. MIPS register \$0 always equals 0. Register \$1 is reserved for the assembler to handle pseudoinstructions and large constants. Hi and Lo are 32-bit registers containing the results of multiply and divide.
2 <sup>30</sup> memory words	Memory[0], Memory[4], . . . , Memory[4294967292]	Accessed only by data transfer instructions. MIPS uses byte addresses, so sequential words differ by 4. Memory holds data structures, such as arrays, and spilled registers, such as those saved on procedure calls.

### MIPS assembly language

Category	Instruction	Example	Meaning	Comments
Arithmetic	add	add \$1,\$2,\$3	$\$1 = \$2 + \$3$	3 operands; exception possible
	subtract	sub \$1,\$2,\$3	$\$1 = \$2 - \$3$	3 operands; exception possible
	add immediate	addi \$1,\$2,100	$\$1 = \$2 + 100$	+ constant; exception possible
	add unsigned	addu \$1,\$2,\$3	$\$1 = \$2 + \$3$	3 operands; no exceptions
	subtract unsigned	subu \$1,\$2,\$3	$\$1 = \$2 - \$3$	3 operands; no exceptions
	add imm. unsign.	addiu \$1,\$2,100	$\$1 = \$2 + 100$	+ constant; no exceptions
	Move fr. copr. reg.	mfc0 \$1,\$epc	$\$1 = \$epc$	Used to get exception PC
	multiply	mult \$2,\$3	$Hi, Lo = \$2 \times \$3$	64-bit signed product in Hi, Lo
	multiply unsigned	multu \$2,\$3	$Hi, Lo = \$2 \times \$3$	64-bit unsigned product in Hi, Lo
	divide	div \$2,\$3	$Lo = \$2 \div \$3, Hi = \$2 \bmod \$3$	Lo = quotient, Hi = remainder
	divide unsigned	divu \$2,\$3	$Lo = \$2 \div \$3, Hi = \$2 \bmod \$3$	Unsigned quotient and remainder
	Move from Hi	mfhi \$1	$\$1 = Hi$	Used to get copy of Hi
	Move from Lo	mflo \$1	$\$1 = Lo$	Use to get copy of Lo
Logical	and	and \$1,\$2,\$3	$\$1 = \$2 \& \$3$	3 register operands; logical AND
	or	or \$1,\$2,\$3	$\$1 = \$2 \mid \$3$	3 register operands; logical OR
	and immediate	andi \$1,\$2,100	$\$1 = \$2 \& 100$	Logical AND register, constant
	or immediate	ori \$1,\$2,100	$\$1 = \$2 \mid 100$	Logical OR register, constant
	shift left logical	sll \$1,\$2,10	$\$1 = \$2 \ll 10$	Shift left by constant
	shift right logical	srl \$1,\$2,10	$\$1 = \$2 \gg 10$	Shift right by constant
Data transfer	load word	lw \$1,100(\$2)	$\$1 = \text{Memory}[\$2+100]$	Data from memory to register
	store word	sw \$1,100(\$2)	$\text{Memory}[\$2+100] = \$1$	Data from register to memory
	load upper imm.	lui \$1,100	$\$1 = 100 \times 2^{16}$	Loads constant in upper 16 bits
Conditional branch	branch on equal	beq \$1,\$2,100	if $(\$1 = \$2)$ go to $PC+4+100 \times 4$	Equal test; PC relative branch
	branch on not eq.	bne \$1,\$2,100	if $(\$1 \neq \$2)$ go to $PC+4+100 \times 4$	Not equal test; PC relative
	set on less than	slt \$1,\$2,\$3	if $(\$2 < \$3)$ $\$1=1$ ; else $\$1=0$	Compare less than; 2's complement
	set less than imm.	slti \$1,\$2,100	if $(\$2 < 100)$ $\$1=1$ ; else $\$1=0$	Compare < constant; 2's comp.
	set less than uns.	sltu \$1,\$2,\$3	if $(\$2 < \$3)$ $\$1=1$ ; else $\$1=0$	Compare less than; natural number
	set l.t. imm. uns.	sltiu \$1,\$2,100	if $(\$2 < 100)$ $\$1=1$ ; else $\$1=0$	Compare < constant; natural
Unconditional jump	jump	j 10000	go to 10000	Jump to target address
	jump register	jr \$31	go to \$31	For switch, procedure return
	jump and link	jal 10000	$\$31 = PC + 4$ ; go to 10000	For procedure call

Main MIPS assembly language instruction set. The floating-point instructions are shown in Figure 4.44 on page 241. Appendix A gives the full MIPS assembly language instruction set.