Machine coding of selected MIPS instructions

R type fields (bits):	opc 6	rs 5	rt 5	rd 5	sa 5	fn 6	mnem $ rd,$rs,$rt mnem $rs,$rt or mnem $rd	
Instruction	Decimal Coding in Memory						Coding in Assembly Language	
sll	0	0	2	1	3	0	sll $1,$2, 3	shift-logical-left
jr	0	31	0	0	0	8	jr $31 (5)	jump-to-register
mflo	0	0	0	1	0	18	mflo $1	move-from-LO
mult	0	2	3	0	0	24	mult $2,$3	signed-multiply
multu	0	2	3	0	0	25	multu $2,$3	unsigned-multiply
add	0	2	3	1	0	32	add $1,$2,$3	add
addu	0	2	3	1	0	33	addu $1,$2,$3	unsigned-add
sub	0	2	3	1	0	34	sub $1,$2,$3	subtract
subu	0	2	3	1	0	35	subu $1,$2,$3	unsigned-subtract
and	0	2	3	1	0	36	and $1,$2,$3	bitwise-and
or	0	2	3	1	0	37	or $1,$2,$3	bitwise-or
slt	0	2	3	1	0	42	slt $1,$2,$3	set-if-less-than
sltu	0	2	3	1	0	43	sltu $1,$2,$3	unsigned-set-if-less-than

J- type fields (bits):	**opc** 6	**immJ (word-address)** 26	mnem byte-address	
Instruction	Decimal Coding in Memory		Coding in Assembly Language	
j	2	10000 (4)	j 40000	jump
jal	3	10000 (4)	jal 40000	jump-and-link

I- type field-width (bit):	opc 6	rs 5	rt 5	imm 16	mnem $rd,$rs,imm (1) (2) or mnem $rt, displacement($rs) (2)	
Instruction	Decimal Coding in Memory				Coding in Assembly Language	
beq	4	1	2	100 (3)	beq $1,$2,400	branch-if-equal
bne	5	1	2	100 (3)	bne $1,$2,400	branch-if-not-equal
addi	8	2	1	100	addi $1,$2,100 (1)	add immediate
slti	10	2	1	100	slti $1,$2,100	set-less-than-immed.
lui	15	0	1	100	lui $1,100	load-upper-immediate
lw	35	2	1	100	lw $1,100($2) (2)	load-word
sw	43	2	1	100	sw $1,100($2) (2)	store-word

(1) **immediate-value (16-bit signed);** (2) **base/displacement addressing (16-bit, byte-address)**	(3) **PC-relative (16-bit word-address)** (4) **26-bit word-address** (5) **register-addressing.**

MIPS Assembly convention for general-purpose register usage:

$zero	$0 , contains zero	$at (reserved)	$1 , temporary for pseudo-codes
$v0 ... $v1	$2 ... $3, return values from expression and procedure evaluation	$a0 ... $a3	$4 ... $7, arguments for procedure
$t0 ... $t7	$8 ... $15 , temporaries	$s0 ... $s7	$16 ... $23, saved variables
$t8 ... $t9	$24 ... $25, more temporaries	$gp	$28, global pointer
$sp	$29, stack pointer	$fp	$30, frame pointer
$ra	$31, return address		