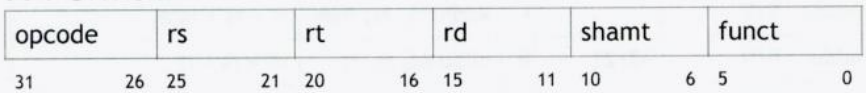
Source: Technische Universität Wien, Computer Organization and Design Course

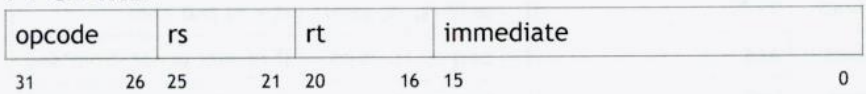
	Instr.	Opcode (dec/hex)	Funct (dec/hex)	F	Example	Meaning
Add	add	0/0	32/20	R	add rd, rs, rt	rd = rs + rt
Add Imm.	addi	8/8	-	I	addi rt, rs, imm	rt = rs + imm
Add Imm. Unsigned	addiu	9/9	-	I	addiu rt, rs, imm	rt = rs + imm
Add Unsigned	addu	0/0	33/21	R	addu rd, rs, rt	rd = rs + rt
And	and	0/0	36/24	R	and rd, rs, rt	rd = rs and rt
And Imm.	andi	12/C	-	I	addi rt, rs, imm	rt = rs and imm
Branch on Equal	beq	4/4	-	I	beq rs, rt, imm	if rs==rt: pc=pc+imm*4+4
Branch on Not Equal	bne	5/5	-	I	bne rs, rt, imm	if rs!=rt: pc=pc+imm*4+4
Jump	j	2/2	-	J	j addr	pc = addr * 4
Jump and Link	jal	3/3	-	J	jal addr	$ra=pc+4; pc=addr*4
Load Byte	lb	32/20	-	I	lb rt, imm(rs)	rt = Mem[rs + imm]
Load Byte Unsigned	lbu	36/24	-	I	lbu rt, imm(rs)	rt = Mem[rs + imm]
Load Halfword	lh	33/21	-	I	lh rt, imm(rs)	rt = Mem[rs + imm]
Load Halfword Unsigned	lhu	37/25	-	I	lhu rt, imm(rs)	rt = Mem[rs + imm]
Load Word	lw	35/23	-	I	lw rt, imm(rs)	rt = Mem[rs + imm]
Multiply	mult	0/0	24/18	R	mult rd, rs, rt	rd = rs * rt
Multiply Unsigned	multu	0/0	25/19	R	multu rd, rs, rt	rd = rs * rt
Nor	nor	0/0	39/27	R	nor rd, rs, rt	rd = rs nor rt
Or	or	0/0	37/25	R	or rd, rs, rt	rd = rs or rt
Or Imm.	ori	13/D	-	I	ori rt, rs, imm	rt = rs or imm
Store Byte	sb	40/28	-	I	sb rt, imm(rs)	Mem[rs + imm] = rt
Store Halfword	sh	41/29	-	I	sh rt, imm(rs)	Mem[rs + imm] = rt
Shift Left Logical	sll	0/0	0/0	R	sll rd, rt, shamt	rd = rt << shamt
Set Less Than	slt	0/0	42/2A	R	slt rd, rs, rt	if rs<rt: rd=1 else rd=0
Set Less Than Imm.	slti	10/A	-	I	slti rt, rs, imm	if rs<imm: rt=1 else rt=0
Set Less Than Imm. Unsigned	sltiu	11/B	-	I	sltiu rt, rs, imm	if rs<imm: rt=1 else rt=0
Set Less Than Unsigned	sltu	0/0	43/2B	R	sltu rd, rs, rt	if rs<rt: rd=1 else rd=0
Shift Right Logical	srl	0/0	2/2	R	srl rd, rt, shamt	rd = rt >> shamt
Subtract	sub	0/0	34/22	R	sub rd, rs, rt	rd = rs - rt
Subtract Unsigned	subu	0/0	35/23	R	subu rd, rs, rt	rd = rs - rt
Store Word	sw	43/2B	-	I	sw rt, imm(rs)	Mem[rs + imm] = rt
Xor	xor	0/0	38/26	R	xor rd, rs, rt	rd = rs xor rt
Xor Imm.	xori	14/E	-	I	xori rt, rs, imm	rt = rs xor imm

Source: Technische Universität Wien, Computer Organization and Design Course

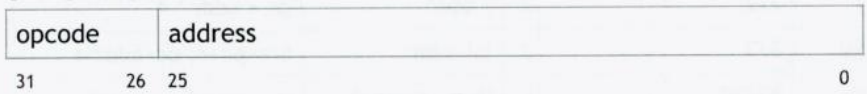
R-Format:



I-Format:



J-Format:



MIPS Registers

Name	Reg. No.	Usage	Preserve on call?
$zero	0	Const. 0	n. a.
$at	1	Reserved for assembler	n. a.
$v0 - $v1	2 - 3	Return values	No
$a0 - $a3	4 - 7	Arguments	Yes
$t0 - $t7	8 - 15	Temporary values	No
$s0 - $s7	16 - 23	Saved values	Yes
$t8 - $t9	24 - 25	Temporary values	No
$k0 - $k1	26 - 27	Reserved for OS kernel	
$gp	28	Global pointer	Yes
$sp	29	Stack pointer	Yes
$fp	30	Frame pointer	Yes
$ra	31	Return address	Yes