

# MIPS Reference Data



①

②

## CORE INSTRUCTION SET

NAME, MNEMONIC	FOR-MAT	OPERATION (in Verilog)	OPCODE / FUNCT (Hex)
Add	add R R[rd] = R[rs] + R[rt]	(1) 0/20hex	
Add Immediate	addi I R[rt] = R[rs] + SignExtImm	(1,2) 8hex	
Add Imm. Unsigned	addiu I R[rt] = R[rs] + SignExtImm	(2) 9hex	
Add Unsigned	addu R R[rd] = R[rs] + R[rt]	0/21hex	
And	and R R[rd] = R[rs] & R[rt]	0/24hex	
And Immediate	andi I R[rt] = R[rs] & ZeroExtImm	(3) 9hex	
Branch On Equal	beq I iR[rs]=R[rt]	4hex	
	PC=PC+4+BranchAddr		
Branch On Not Equal	bne I iR[rs]=R[rt]	5hex	
	PC=PC+4+BranchAddr		
Jump	j J PC=JumpAddr	(4) 2hex	
Jump And Link	jal J R[31]=PC+8; PC=JumpAddr	(5) 3hex	
Jump Register	jr R PC=R[rs]	0/08hex	
Load Byte Unsigned	lbu I R[rt] = (2*160; M[R[rs]] + SignExtImm)[7:0]	(2) 24hex	
Load Halfword Unsigned	lhu I R[rt] = (16*160; M[R[rs]] + SignExtImm)[15:0]	(2) 25hex	
Load Limited	ll I R[rt] = M[R[rs]; SignExtImm]	(2,7) 30hex	
Load Upper Imm.	lui I R[rt] = imm, 16*160	flex	
Load Word	lw I R[rt] = M[R[rs]; SignExtImm]	(2) 23hex	
Nor	nor R R[rd] = ~ (R[rs]   R[rt])	0/27hex	
Or	or R R[rd] = R[rs]   R[rt]	0/25hex	
Or Immediate	ori I R[rt] = R[rs]   ZeroExtImm	(3) 6hex	
Set Less Than	slt I R[rt] = (R[rs] < R[rt]) ? 1 : 0	0/28hex	
Set Less Than Imm.	slti I R[rt] = (R[rs] < R[rt]) ? 1 : 0	9hex	
Set Less Than Imm. Unsigned	sltiu I R[rt] = (R[rs] < SignExtImm) ? 1 : 0	(2,6) 6hex	
Set Less Than Unsig. sltu	R R[rd] = (R[rs] < R[rt]) ? 1 : 0	(6) 0/2bhex	
Shift Left Logical	sll I R[rd] = R[rt] << shamt	0/00hex	
Shift Right Logical	srl I R[rd] = R[rt] >> shamt	0/02hex	
Store Byte	sb I M[R[rs]; SignExtImm][7:0] = R[rt][7:0]	(2) 28hex	
Store Conditional	sc I M[R[rs]; SignExtImm] = R[rt]; R[rt] = (atomic) ? 1 : 0	(2,7) 38hex	
Store Halfword	sh I M[R[rs]; SignExtImm][15:0] = R[rt][15:0]	(2) 29hex	
Store Word	sw I M[R[rs]; SignExtImm] = R[rt]	(2) 2bhex	
Subtract	sub R R[rd] = R[rs] - R[rt]	(1) 0/22hex	
Subtract Unsigned	subu R R[rd] = R[rs] - R[rt]	0/23hex	

## BASIC INSTRUCTION FORMATS

R	opcode	rs	rt	rd	shamt	funct
31	26-25	21-20	16-15	11-10	6-5	0
I	opcode	rs	rt		immediate	
J	opcode	26-25	21-20	16-15	address	

②

①

## ARITHMETIC CORE INSTRUCTION SET

NAME, MNEMONIC	FOR-MAT	OPERATION	OPCODE / FUNCT (Hex)
Branch On FP True	bf I iF[FpCond]PC=PC+4+BranchAddr	(4) 11/81/-	
Branch On FP False	bfi I iF[FpCond]PC=PC+4+BranchAddr	(4) 11/80/-	
Divide	div R Lo=R[rs]/R[rt]; Hi=R[rs]%R[rt]	0/-/-/1a	
Divide Unsigned	divu R Lo=R[rs]/R[rt]; Hi=R[rs]%R[rt]	(6) 0/-/-/1b	
FP Add Single	FPAdd Fld = F[rs] + F[rt]	11/10/-/0	
FP Add Double	FPAdd Fld = F[rs] + F[rt]	11/11/-/0	
FP Compare Single	FPCompd = (F[rs] op F[rt]) ? 1 : 0	11/10/-/1	
FP Compare Double	FPCompd = (F[rs] op F[rt]) ? 1 : 0	11/11/-/1	
FP Divide Single	FPDivd = (F[rs] / F[rt])	11/10/-/3	
FP Divide Double	FPDivd = (F[rs] / F[rt])	11/11/-/3	
FP Multiply Single	FPMuld = F[rs] * F[rt]	11/10/-/2	
FP Multiply Double	FPMuld = F[rs] * F[rt]	11/11/-/2	
FP Subtract Single	FPSubd = F[rs] - F[rt]	11/10/-/1	
FP Subtract Double	FPSubd = F[rs] - F[rt]	11/11/-/1	
Lead FP Single	lwc1 I F[rt] = M[R[rs]; SignExtImm]	(2) 31/-/-/4-	
Lead FP Double	lwc2 I F[rt] = M[R[rs]; SignExtImm]	(2) 35/-/-/4-	
Move From Hi	mfhi R R[rd] = Hi	0/-/-/10	
Move From Lo	mflo R R[rd] = Lo	0/-/-/12	
Move From Control	mtc0 R R[rd] = CR[rs]	10/00/-/0	
Multiply	mult R {Hi,Lo} = R[rs] * R[rt]	0/-/-/18	
Multiply Unsigned	multu R {Hi,Lo} = R[rs] * R[rt]	(6) 0/-/-/19	
Shift Right Arith.	sra R R[rd] = R[rt] >>> shamt	0/-/-/3	
Shift Right Logical	srl R R[rd] = R[rt] >>> shamt	(2) 39/-/-/4-	
Store FP Single	swc1 I M[R[rs]; SignExtImm] = F[rt]	(2) 39/-/-/4-	
Store FP Double	swc2 I M[R[rs]; SignExtImm] = F[rt]	(2) 3d/-/-/4-	

## FLOATING-POINT INSTRUCTION FORMATS

FR	opcode	fm1	ft	fs	fd	funct
31	26-25	21-20	16-15	11-10	6-5	0
FI	opcode	fm1	ft	fs	fd	funct
31	26-25	21-20	16-15	11-10	6-5	0

## PSEUDOINSTRUCTION SET

NAME	MNEMONIC	OPERATION
Branch Less Than	blt	iR[rs] < R[rt] PC = Label
Branch Greater Than	bgt	iR[rs] > R[rt] PC = Label
Branch Less Than or Equal	btle	iR[rs] <= R[rt] PC = Label
Branch Greater Than or Equal	bgtle	iR[rs] >= R[rt] PC = Label
Load Immediate	li	R[rd] = immediate
Move	move	R[rd] = R[rs]

## REGISTER NAME, NUMBER, USE, CALL CONVENTION

NAME	NUMBER	USE	PRESERVED/CROSS A CALL?
\$zero	0	The Constant Value 0	N.A.
\$at	1	Assembler Temporary	No
\$v0-\$v1	2-3	Values for Function Results and Expression Evaluation	No
\$a0-\$a3	4-7	Arguments	No
\$t0-\$t7	8-15	Temporaries	No
\$s0-\$s7	16-23	Saved Temporaries	Yes
\$k0-\$k1	24-25	Temporaries	No
\$k2-\$k3	26-27	Reserved for OS Kernel	No
\$gp	28	Global Pointer	Yes
\$sp	29	Stack Pointer	Yes
\$fp	30	Frame Pointer	Yes
\$ra	31	Return Address	Yes