3.0 INSTRUCTION SET

The PIC32MX3XX/4XX family instruction set complies with the MIPS32 Release 2 instruction set architecture. PIC32MX does not support the following features:

- CoreExtend instructions
- Coprocessor 1 instructions
- Coprocessor 2 instructions

Table 3-1 provides a summary of the instructions that are implemented by the PIC32MX3XX/4XX family core.

Note: Refer to *"MIPS32® Architecture for Programmers Volume II: The MIPS32® Instruction Set"* at www.mips.com for more information.

TABLE 3-1: PIC32MX3XX/4XX INSTRUCTION SET

Instruction	Description	Function
ADD	Integer Add	Rd = Rs + Rt
ADDI	Integer Add Immediate	Rt = Rs + Immed
ADDIU	Unsigned Integer Add Immediate	Rt = Rs $+_U$ Immed
ADDIUPC	Unsigned Integer Add Immediate to PC (MIPS16e™ only)	Rt = PC $+_u$ Immed
ADDU	Unsigned Integer Add	Rd = Rs $+_U$ Rt
AND	Logical AND	Rd = Rs & Rt
ANDI	Logical AND Immediate	Rt = Rs & (0_{16} || Immed)
B	Unconditional Branch (Assembler idiom for: BEQ r0, r0, offset)	PC += (int)offset
BAL	Branch and Link (Assembler idiom for: BGEZAL r0, offset)	GPR[31> = PC + 8 PC += (int)offset
BEQ	Branch On Equal	if Rs == Rt PC += (int)offset

BEQL	Branch On Equal Likely	if Rs == Rt
		  PC += (int)offset
		else
		  Ignore Next Instruction

BGEZ	Branch on Greater Than or Equal To Zero	if !Rs[31> PC += (int)offset
BGEZAL	Branch on Greater Than or Equal To Zero And Link	GPR[31> = PC + 8 if !Rs[31> PC += (int)offset

BGEZALL	Branch on Greater Than or Equal To Zero And Link Likely	GPR[31> = PC + 8
		if !Rs[31>
		  PC += (int)offset
		else
		  Ignore Next Instruction

BGEZL	Branch on Greater Than or Equal To Zero Likely	if !Rs[31>
		  PC += (int)offset
		else
		  Ignore Next Instruction

BGTZ	Branch on Greater Than Zero	if !Rs[31> && Rs != 0 PC += (int)offset

BGTZL	Branch on Greater Than Zero Likely	if !Rs[31> && Rs != 0
		  PC += (int)offset
		else
		  Ignore Next Instruction

TABLE 3-1: PIC32MX3XX/4XX INSTRUCTION SET (CONTINUED)

Instruction	Description	Function
BLEZ	Branch on Less Than or Equal to Zero	if Rs[31> || Rs == 0 PC += (int)offset

BLEZL	Branch on Less Than or Equal to Zero Likely	if Rs[31> || Rs == 0
		  PC += (int)offset
		else
		  Ignore Next Instruction

BLTZ	Branch on Less Than Zero	if Rs[31> PC += (int)offset
BLTZAL	Branch on Less Than Zero And Link	GPR[31> = PC + 8 if Rs[31> PC += (int)offset

BLTZALL	Branch on Less Than Zero And Link Likely	GPR[31> = PC + 8
		if Rs[31>
		  PC += (int)offset
		else
		  Ignore Next Instruction

BLTZL	Branch on Less Than Zero Likely	if Rs[31>
		  PC += (int)offset
		else
		  Ignore Next Instruction

BNE	Branch on Not Equal	if Rs != Rt PC += (int)offset

BNEL	Branch on Not Equal Likely	if Rs != Rt
		  PC += (int)offset
		else
		  Ignore Next Instruction

BREAK	Breakpoint	Break Exception
CLO	Count Leading Ones	Rd = NumLeadingOnes(Rs)
CLZ	Count Leading Zeroes	Rd = NumLeadingZeroes(Rs)
COP0	Coprocessor 0 Operation	See Software User's Manual
DERET	Return from Debug Exception	PC = DEPC Exit Debug Mode
DI	Atomically Disable Interrupts	Rt = Status; $Status_{IE}$ = 0
DIV	Divide	LO = (int)Rs / (int)Rt HI = (int)Rs % (int)Rt
DIVU	Unsigned Divide	LO = (uns)Rs / (uns)Rt HI = (uns)Rs % (uns)Rt
EHB	Execution Hazard Barrier	Stop instruction execution until execution hazards are cleared
EI	Atomically Enable Interrupts	Rt = Status; $Status_{IE}$ = 1

ERET	Return from Exception	if SR[2>
		  PC = ErrorEPC
		else
		  PC = EPC
		  SR[1> = 0
		SR[2> = 0
		LL = 0

EXT	Extract Bit Field	Rt = ExtractField(Rs, pos, size)

TABLE 3-1: PIC32MX3XX/4XX INSTRUCTION SET (CONTINUED)

Instruction	Description	Function
INS	Insert Bit Field	Rt = InsertField(Rs, Rt, pos, size)
J	Unconditional Jump	PC = PC[31:28> || offset<<2
JAL	Jump and Link	GPR[31> = PC + 8 PC = PC[31:28> || offset<<2
JALR	Jump and Link Register	Rd = PC + 8 PC = Rs
JALR.HB	Jump and Link Register with Hazard Barrier	Like JALR, but also clears execution and instruction hazards
JALRC	Jump and Link Register Compact – do not execute instruction in jump delay slot (MIPS16e™ only)	Rd = PC + 2 PC = Rs
JR	Jump Register	PC = Rs
JR.HB	Jump Register with Hazard Barrier	Like JR, but also clears execution and instruction hazards
JRC	Jump Register Compact – do not execute instruction in jump delay slot (MIPS16e only)	PC = Rs
LB	Load Byte	Rt = (byte)Mem[Rs+offset>
LBU	Unsigned Load Byte	Rt = (ubyte))Mem[Rs+offset>
LH	Load Halfword	Rt = (half)Mem[Rs+offset>
LHU	Unsigned Load Halfword	Rt = (uhalf)Mem[Rs+offset>
LL	Load Linked Word	Rt = Mem[Rs+offset> LL = 1 LLAdr = Rs + offset
LUI	Load Upper Immediate	Rt = immediate << 16
LW	Load Word	Rt = Mem[Rs+offset>
LWPC	Load Word, PC relative	Rt = Mem[PC+offset>
LWL	Load Word Left	See Architecture Reference Manual
LWR	Load Word Right	See Architecture Reference Manual
MADD	Multiply-Add	HI | LO += (int)Rs * (int)Rt
MADDU	Multiply-Add Unsigned	HI | LO += (uns)Rs * (uns)Rt
MFC0	Move From Coprocessor 0	Rt = CPR[0, Rd, sel>
MFHI	Move From HI	Rd = HI
MFLO	Move From LO	Rd = LO
MOVN	Move Conditional on Not Zero	if Rt ¼ 0 then Rd = Rs
MOVZ	Move Conditional on Zero	if Rt = 0 then Rd = Rs
MSUB	Multiply-Subtract	HI | LO -= (int)Rs * (int)Rt
MSUBU	Multiply-Subtract Unsigned	HI | LO -= (uns)Rs * (uns)Rt
MTC0	Move To Coprocessor 0	CPR[0, n, Sel> = Rt
MTHI	Move To HI	HI = Rs
MTLO	Move To LO	LO = Rs
MUL	Multiply with register write	HI | LO =Unpredictable Rd = ((int)Rs * (int)Rt)$_{31..0}$
MULT	Integer Multiply	HI | LO = (int)Rs * (int)Rd
MULTU	Unsigned Multiply	HI | LO = (uns)Rs * (uns)Rd

TABLE 3-1: PIC32MX3XX/4XX INSTRUCTION SET (CONTINUED)

Instruction	Description	Function
NOP	No Operation (Assembler idiom for: SLL r0, r0, r0)	
NOR	Logical NOR	Rd = ~(Rs | Rt)
OR	Logical OR	Rd = Rs | Rt
ORI	Logical OR Immediate	Rt = Rs | Immed
RDHWR	Read Hardware Register	Allows unprivileged access to registers enabled by HWREna register
RDPGPR	Read GPR from Previous Shadow Set	Rt = SGPR[SRSCtl$_{PSS}$, Rd>
RESTORE	Restore registers and deallocate stack frame (MIPS16e™ only)	See Architecture Reference Manual
ROTR	Rotate Word Right	Rd = Rt$_{sa-1..0}$ || Rt$_{31..sa}$
ROTRV	Rotate Word Right Variable	Rd = Rt$_{Rs-1..0}$ || Rt$_{31..Rs}$
SAVE	Save registers and allocate stack frame (MIPS16e only)	See Architecture Reference Manual
SB	Store Byte	(byte)Mem[Rs+offset> = Rt
SC	Store Conditional Word	if LL = 1 mem[Rs+offset> = Rt Rt = LL
SDBBP	Software Debug Break Point	Trap to SW Debug Handler
SEB	Sign-Extend Byte	Rd = (byte)Rs
SEH	Sign-Extend Half	Rd = (half)Rs
SH	Store Half	(half)Mem[Rs+offset> = Rt
SLL	Shift Left Logical	Rd = Rt << sa
SLLV	Shift Left Logical Variable	Rd = Rt << Rs[4:0>

SLT	Set on Less Than	if (int)Rs < (int)Rt
		  Rd = 1
		else
		  Rd = 0

SLTI	Set on Less Than Immediate	if (int)Rs < (int)Immed
		  Rt = 1
		else
		  Rt = 0

SLTIU	Set on Less Than Immediate Unsigned	if (uns)Rs < (uns)Immed
		  Rt = 1
		else
		  Rt = 0

SLTU	Set on Less Than Unsigned	if (uns)Rs < (uns)Immed
		  Rd = 1
		else
		  Rd = 0

SRA	Shift Right Arithmetic	Rd = (int)Rt >> sa
SRAV	Shift Right Arithmetic Variable	Rd = (int)Rt >> Rs[4:0>
SRL	Shift Right Logical	Rd = (uns)Rt >> sa
SRLV	Shift Right Logical Variable	Rd = (uns)Rt >> Rs[4:0>
SSNOP	Superscalar Inhibit No Operation	NOP
SUB	Integer Subtract	Rt = (int)Rs - (int)Rd
SUBU	Unsigned Subtract	Rt = (uns)Rs - (uns)Rd
SW	Store Word	Mem[Rs+offset> = Rt
SWL	Store Word Left	See Architecture Reference Manual

TABLE 3-1: PIC32MX3XX/4XX INSTRUCTION SET (CONTINUED)

Instruction	Description	Function
SWR	Store Word Right	See Architecture Reference Manual
SYNC	Synchronize	See Software User's Manual
SYSCALL	System Call	SystemCallException
TEQ	Trap if Equal	if Rs == Rt TrapException
TEQI	Trap if Equal Immediate	if Rs == (int)Immed TrapException
TGE	Trap if Greater Than or Equal	if (int)Rs >= (int)Rt TrapException
TGEI	Trap if Greater Than or Equal Immediate	if (int)Rs >= (int)Immed TrapException
TGEIU	Trap if Greater Than or Equal Immediate Unsigned	if (uns)Rs >= (uns)Immed TrapException
TGEU	Trap if Greater Than or Equal Unsigned	if (uns)Rs >= (uns)Rt TrapException
TLT	Trap if Less Than	if (int)Rs < (int)Rt TrapException
TLTI	Trap if Less Than Immediate	if (int)Rs < (int)Immed TrapException
TLTIU	Trap if Less Than Immediate Unsigned	if (uns)Rs < (uns)Immed TrapException
TLTU	Trap if Less Than Unsigned	if (uns)Rs < (uns)Rt TrapException
TNE	Trap if Not Equal	if Rs != Rt TrapException
TNEI	Trap if Not Equal Immediate	if Rs != (int)Immed TrapException
WAIT	Wait for Interrupts	Stall until interrupt occurs
WRPGPR	Write to GPR in Previous Shadow Set	$SGPR[SRSCtl_{PSS}, Rd> = Rt$
WSBH	Word Swap Bytes Within Halfwords	$Rd = Rt_{23..16} \,||\, Rt_{31..24} \,||\, Rt_{7..0} \,||\, Rt_{15..8}$
XOR	Exclusive OR	Rd = Rs ^ Rt
XORI	Exclusive OR Immediate	Rt = Rs ^ (uns)Immed
ZEB	Zero-extend byte (MIPS16e™ only)	Rt = (ubyte) Rs
ZEH	Zero-extend half (MIPS16e only)	Rt = (uhalf) Rs

NOTES:

