MIPS® DSP ASE Instruction Set Quick Reference

RD, RS, RT	—	DESTINATION (RD) AND SOURCE (RS, RT) REGISTERS
AC	—	ACCUMULATOR REGISTER (AC0 – AC3)
C, CC	—	CARRY (BIT 13) AND CONDITION CODE FLAGS (BITS 24-27) IN DSPCONTROL REGISTER
POS, SIZE	—	POSITION AND SIZE (SCOUNT) FIELDS IN DSPCONTROL REGISTER
±	—	SIGNED OPERAND/OPERATION OR SIGN EXTENSION
∅	—	UNSIGNED OPERAND/OPERATION OR ZERO EXTENSION
×	—	INTEGER MULTIPLICATION
⦿ / •	—	FRACTIONAL MULTIPLICATION (IMPLIED SHIFT LEFT BY 1 BIT) WITH / WITHOUT ROUNDING
® / [ ]	—	ROUNDING AND SATURATION OPERATIONS
L / R	—	LEFT / RIGHT 16-BIT PART OF A RESULT OR A REGISTER
LL, LR, RL, RR	—	THE FOUR BYTES IN A 32-BIT REGISTER, FROM LEFT (MSB) TO RIGHT (LSB)
||	—	BOUNDARY BETWEEN TWO OR FOUR SIMD ELEMENTS IN A REGISTER
::	—	CONCATENATION OF BIT FIELDS
R2	—	DSP ASE REVISION 2 INSTRUCTION

ARITHMETIC OPERATIONS: 8-BIT DATA			
ABSQ_S.QB[R2]	RD, RS	$RD_{XY} = [\lvert RS_{XY}^{\pm}\rvert]$	XY ∈ { LL, LR, RL, RR }
ADDU.QB	RD, RS, RT	$RD_{XY} = RS_{XY}^{\varnothing} + RT_{XY}^{\varnothing}$	XY ∈ { LL, LR, RL, RR }
ADDU_S.QB	RD, RS, RT	$RD_{XY} = [RS_{XY}^{\varnothing} + RT_{XY}^{\varnothing}]$	XY ∈ { LL, LR, RL, RR }
ADDUH.QB[R2]	RD, RS, RT	$RD_{XY} = (RS_{XY}^{\varnothing} + RT_{XY}^{\varnothing}) >> 1$	XY ∈ { LL, LR, RL, RR }
ADDUH_R.QB[R2]	RD, RS, RT	$RD_{XY} = (RS_{XY}^{\varnothing} + RT_{XY}^{\varnothing} + 1^{\varnothing}) >> 1$	XY ∈ { LL, LR, RL, RR }
RADDU.W.QB	RD, RS	$RD = RS_{LL}^{\varnothing} + RS_{LR}^{\varnothing} + RS_{RL}^{\varnothing} + RS_{RR}^{\varnothing}$	
REPL.QB	RD, CONST8	RD = CONST8 || CONST8 || CONST8 || CONST8	
REPLV.QB	RD, RS	$RD = RS_{7:0} \,\|\|\, RS_{7:0} \,\|\|\, RS_{7:0} \,\|\|\, RS_{7:0}$	
SUBU.QB	RD, RS, RT	$RD_{XY} = RS_{XY}^{\varnothing} - RT_{XY}^{\varnothing}$	XY ∈ { LL, LR, RL, RR }
SUBU_S.QB	RD, RS, RT	$RD_{XY} = [RS_{XY}^{\varnothing} - RT_{XY}^{\varnothing}]$	XY ∈ { LL, LR, RL, RR }
SUBUH.QB[R2]	RD, RS, RT	$RD_{XY} = (RS_{XY}^{\varnothing} - RT_{XY}^{\varnothing}) >> 1$	XY ∈ { LL, LR, RL, RR }
SUBUH_R.QB[R2]	RD, RS, RT	$RD_{XY} = (RS_{XY}^{\varnothing} - RT_{XY}^{\varnothing} + 1^{\varnothing}) >> 1$	XY ∈ { LL, LR, RL, RR }

SHIFT OPERATIONS: 8-BIT DATA			
SHLL.QB	RD, RS, SHIFT3	$RD_{XY} = RS_{XY} << SHIFT3$	XY ∈ { LL, LR, RL, RR }
SHLLV.QB	RD, RS, RT	$RD_{XY} = RS_{XY} << RT_{2:0}$	XY ∈ { LL, LR, RL, RR }
SHRA.QB[R2]	RD, RS, SHIFT3	$RD_{XY} = RS_{XY}^{\pm} >> SHIFT3$	XY ∈ { LL, LR, RL, RR }
SHRA_R.QB[R2]	RD, RS, SHIFT3	$RD_{XY} = ®(RS_{XY}^{\pm} >> SHIFT3)$	XY ∈ { LL, LR, RL, RR }
SHRAV.QB[R2]	RD, RS, RT	$RD_{XY} = RS_{XY}^{\pm} >> RT_{2:0}$	XY ∈ { LL, LR, RL, RR }
SHRAV_R.QB[R2]	RD, RS, RT	$RD_{XY} = ®(RS_{XY}^{\pm} >> RT_{2:0})$	XY ∈ { LL, LR, RL, RR }
SHRL.QB	RD, RS, SHIFT3	$RD_{XY} = RS_{XY}^{\varnothing} >> SHIFT3$	XY ∈ { LL, LR, RL, RR }
SHRLV.QB	RD, RS, RT	$RD_{XY} = RS_{XY}^{\varnothing} >> RT_{2:0}$	XY ∈ { LL, LR, RL, RR }

UNSIGNED INTEGER MULTIPLY OPERATIONS: 8-BIT/16-BIT DATA » GPR			
MULEU_S.PH.QBL	RD, RS, RT	$RD = [RS_{LL}^{\varnothing} \times RT_{L}^{\varnothing}]_R \,\|\|\, [RS_{LR}^{\varnothing} \times RT_{R}^{\varnothing}]_R$	■■□□ ■■
MULEU_S.PH.QBR	RD, RS, RT	$RD = [RS_{RL}^{\varnothing} \times RT_{L}^{\varnothing}]_R \,\|\|\, [RS_{RR}^{\varnothing} \times RT_{R}^{\varnothing}]_R$	□□■■ ■■

UNSIGNED INTEGER MULTIPLY OPERATIONS: 8-BIT DATA » ACCUMULATOR			
DPAU.H.QBL	AC, RS, RT	$AC \mathrel{+}= (RS_{LL}^{\varnothing} \times RT_{LL}^{\varnothing}) + (RS_{LR}^{\varnothing} \times RT_{LR}^{\varnothing})$	■■□□ ■■□□
DPAU.H.QBR	AC, RS, RT	$AC \mathrel{+}= (RS_{RL}^{\varnothing} \times RT_{RL}^{\varnothing}) + (RS_{RR}^{\varnothing} \times RT_{RR}^{\varnothing})$	□□■■ □□■■
DPSU.H.QBL	AC, RS, RT	$AC \mathrel{-}= (RS_{LL}^{\varnothing} \times RT_{LL}^{\varnothing}) + (RS_{LR}^{\varnothing} \times RT_{LR}^{\varnothing})$	■■□□ ■■□□
DPSU.H.QBR	AC, RS, RT	$AC \mathrel{-}= (RS_{RL}^{\varnothing} \times RT_{RL}^{\varnothing}) + (RS_{RR}^{\varnothing} \times RT_{RR}^{\varnothing})$	□□■■ □□■■

PRECISION EXPANSION (DATA UNPACKING) OPERATIONS: 8-BIT DATA			
PRECEQU.PH.QBL	RD, RS	$RD = (RS_{LL} << 7) \,\|\|\, (RS_{LR} << 7)$	①■⑦①■⑦ ⇦ ■■□□
PRECEQU.PH.QBLA	RD, RS	$RD = (RS_{LL} << 7) \,\|\|\, (RS_{RL} << 7)$	①■⑦①■⑦ ⇦ ■□■□
PRECEQU.PH.QBR	RD, RS	$RD = (RS_{RL} << 7) \,\|\|\, (RS_{RR} << 7)$	①■⑦①■⑦ ⇦ □□■■
PRECEQU.PH.QBRA	RD, RS	$RD = (RS_{LR} << 7) \,\|\|\, (RS_{RR} << 7)$	①■⑦①■⑦ ⇦ □■□■
PRECEU.PH.QBL	RD, RS	$RD = 0^8 \,\|\|\, RS_{LL} \,\|\|\, 0^8 \,\|\|\, RS_{LR}$	○■○■ ⇦ ■■□□
PRECEU.PH.QBLA	RD, RS	$RD = 0^8 \,\|\|\, RS_{LL} \,\|\|\, 0^8 \,\|\|\, RS_{RL}$	○■○■ ⇦ ■□■□
PRECEU.PH.QBR	RD, RS	$RD = 0^8 \,\|\|\, RS_{RL} \,\|\|\, 0^8 \,\|\|\, RS_{RR}$	○■○■ ⇦ □□■■
PRECEU.PH.QBRA	RD, RS	$RD = 0^8 \,\|\|\, RS_{LR} \,\|\|\, 0^8 \,\|\|\, RS_{RR}$	○■○■ ⇦ □■□■

COMPARE AND PICK OPERATIONS: 8-BIT DATA			
CMPGDU.EQ.QB[R2]	RD, RS, RT	$RD_{XY} = CC_{XY} = (RS_{XY}^{\varnothing} = RT_{XY}^{\varnothing})\ ?\ 1 : 0$	XY ∈ { LL, LR, RL, RR }
CMPGDU.LT.QB[R2]	RD, RS, RT	$RD_{XY} = CC_{XY} = (RS_{XY}^{\varnothing} < RT_{XY}^{\varnothing})\ ?\ 1 : 0$	XY ∈ { LL, LR, RL, RR }
CMPGDU.LE.QB[R2]	RD, RS, RT	$RD_{XY} = CC_{XY} = (RS_{XY}^{\varnothing} \le RT_{XY}^{\varnothing})\ ?\ 1 : 0$	XY ∈ { LL, LR, RL, RR }
CMPGU.EQ.QB	RD, RS, RT	$RD_{XY} = (RS_{XY}^{\varnothing} = RT_{XY}^{\varnothing})\ ?\ 1 : 0$	XY ∈ { LL, LR, RL, RR }
CMPGU.LT.QB	RD, RS, RT	$RD_{XY} = (RS_{XY}^{\varnothing} < RT_{XY}^{\varnothing})\ ?\ 1 : 0$	XY ∈ { LL, LR, RL, RR }
CMPGU.LE.QB	RD, RS, RT	$RD_{XY} = (RS_{XY}^{\varnothing} \le RT_{XY}^{\varnothing})\ ?\ 1 : 0$	XY ∈ { LL, LR, RL, RR }
CMPU.EQ.QB	RD, RS, RT	$CC_{XY} = (RS_{XY}^{\varnothing} = RT_{XY}^{\varnothing})\ ?\ 1 : 0$	XY ∈ { LL, LR, RL, RR }
CMPU.LT.QB	RD, RS, RT	$CC_{XY} = (RS_{XY}^{\varnothing} < RT_{XY}^{\varnothing})\ ?\ 1 : 0$	XY ∈ { LL, LR, RL, RR }
CMPU.LE.QB	RD, RS, RT	$CC_{XY} = (RS_{XY}^{\varnothing} \le RT_{XY}^{\varnothing})\ ?\ 1 : 0$	XY ∈ { LL, LR, RL, RR }
PICK.QB	RD, RS, RT	$RD_{XY} = CC_{XY}\ ?\ RS_{XY} : RT_{XY}$	XY ∈ { LL, LR, RL, RR }

COMPARE AND PICK OPERATIONS: 16-BIT DATA		
CMP.EQ.PH	RD, RS, RT	$CC = ((RS_{L}^{\pm} = RT_{L}^{\pm})\ ?\ 1 : 0) \,\|\|\, ((RS_{R}^{\pm} = RT_{R}^{\pm})\ ?\ 1 : 0)$
CMP.LT.PH	RD, RS, RT	$CC = ((RS_{L}^{\pm} < RT_{L}^{\pm})\ ?\ 1 : 0) \,\|\|\, ((RS_{R}^{\pm} < RT_{R}^{\pm})\ ?\ 1 : 0)$
CMP.LE.PH	RD, RS, RT	$CC = ((RS_{L}^{\pm} \le RT_{L}^{\pm})\ ?\ 1 : 0) \,\|\|\, ((RS_{R}^{\pm} \le RT_{R}^{\pm})\ ?\ 1 : 0)$
PICK.PH	RD, RS, RT	$RD = (CC_{L}\ ?\ RS_{L} : RT_{L}) \,\|\|\, (CC_{R}\ ?\ RS_{R} : RT_{R})$

Arithmetic Operations: 16-bit Data		
ABSQ_S.PH	$RD, RS	RD = [|RS_L^±|] || [|RS_R^±|]
ADDQ.PH	$RD, RS, RT	RD = (RS_L^± + RT_L^±) || (RS_R^± + RT_R^±)
ADDQ_S.PH	$RD, RS, RT	RD = [RS_L^± + RT_L^±] || [RS_R^± + RT_R^±]
ADDQH.PH[R2]	$RD, RS, RT	RD = ((RS_L^± + RT_L^±) >> 1) || ((RS_R^± + RT_R^±) >> 1)
ADDQH_R.PH[R2]	$RD, RS, RT	RD = ®((RS_L^± + RT_L^±) >> 1) || ®((RS_R^± + RT_R^±) >> 1)
ADDU.PH[R2]	$RD, RS, RT	RD = (RS_L^∅ + RT_L^∅) || (RS_R^∅ + RT_R^∅)
ADDU_S.PH[R2]	$RD, RS, RT	RD = [RS_L^∅ + RT_L^∅] || [RS_R^∅ + RT_R^∅]
REPL.PH	$RD, const10	RD = const10^± || const10^±
REPLV.PH	$RD, RS	RD = RS_15:0 || RS_15:0
SUBQ.PH	$RD, RS, RT	RD = (RS_L^± − RT_L^±) || (RS_R^± − RT_R^±)
SUBQ_S.PH	$RD, RS, RT	RD = [RS_L^± − RT_L^±] || [RS_R^± − RT_R^±]
SUBQH.PH[R2]	$RD, RS, RT	RD = ((RS_L^± − RT_L^±) >> 1) || ((RS_R^± − RT_R^±) >> 1)
SUBQH_R.PH[R2]	$RD, RS, RT	RD = ®((RS_L^± − RT_L^±) >> 1) || ®((RS_R^± − RT_R^±) >> 1)
SUBU.PH[R2]	$RD, RS, RT	RD = (RS_L^∅ − RT_L^∅) || (RS_R^∅ − RT_R^∅)
SUBU_S.PH[R2]	$RD, RS, RT	RD = [RS_L^∅ − RT_L^∅] || [RS_R^∅ − RT_R^∅]

Shift Operations: 16-bit Data		
SHLL.PH	$RD, RS, shift4	RD = (RS_L << shift4) || (RS_R << shift4)
SHLL_S.PH	$RD, RS, shift4	RD = [RS_L << shift4] || [RS_R << shift4]
SHLLV.PH	$RD, RS, RT	RD = (RS_L << RT_3:0) || (RS_R << RT_3:0)
SHLLV_S.PH	$RD, RS, RT	RD = [RS_L << RT_3:0] || [RS_R << RT_3:0]
SHRA.PH	$RD, RS, shift4	RD = (RS_L^± >> shift4) || (RS_R^± >> shift4)
SHRA_R.PH	$RD, RS, shift4	RD = ®(RS_L^± >> shift4) || ®(RS_R^± >> shift4)
SHRAV.PH	$RD, RS, RT	RD = (RS_L^± >> RT_3:0) || (RS_R^± >> RT_3:0)
SHRAV_R.PH	$RD, RS, RT	RD = ®(RS_L^± >> RT_3:0) || ®(RS_R^± >> RT_3:0)
SHRL.PH[R2]	$RD, RS, shift4	RD = (RS_L^∅ >> shift4) || (RS_R^∅ >> shift4)
SHRLV.PH[R2]	$RD, RS, RT	RD = (RS_L^∅ >> RT_3:0) || (RS_R^∅ >> RT_3:0)

Precision Reduction (Data Packing) Operations: 16-bit Data			
PRECR.QB.PH[R2]	$RD, RS, RT	RD = RS_LR || RS_RR || RT_LR || RT_RR	□■□■ □■□■
PRECRQ.QB.PH	$RD, RS, RT	RD = RS_LL || RS_RL || RT_LL || RT_RL	■□■□ ■□■□
PRECRQU_S.QB.PH	$RD, RS, RT	RD = [RS_L^∅]_14:7 || [RS_R^∅]_14:7 || [RT_L^∅]_14:7 || [RT_R^∅]_14:7	

Precision Expansion (Data Unpacking) Operations: 16-bit Data			
PRECEQ.W.PHL	$RD, RS	RD = RS_L << 16	■⬭ ⇦ ■□
PRECEQ.W.PHR	$RD, RS	RD = RS_R << 16	■⬭ ⇦ □■

Integer Multiply Operations: 16-bit Data » Accumulator		
DPA.W.PH[R2]	$AC, RS, RT	AC += (RS_L^± × RT_L^±) + (RS_R^± × RT_R^±)
DPAX.W.PH[R2]	$AC, RS, RT	AC += (RS_L^± × RT_R^±) + (RS_R^± × RT_L^±)
DPS.W.PH[R2]	$AC, RS, RT	AC −= (RS_L^± × RT_L^±) + (RS_R^± × RT_R^±)
DPSX.W.PH[R2]	$AC, RS, RT	AC −= (RS_L^± × RT_R^±) + (RS_R^± × RT_L^±)
MULSA.W.PH[R2]	$AC, RS, RT	AC += (RS_L^± × RT_L^±) − (RS_R^± × RT_R^±)

Integer Multiply Operations: 16-bit Data » GPR		
MUL.PH[R2]	$RD, RS, RT	RD = (RS_L^± × RT_L^±)_R || (RS_R^± × RT_R^±)_R
MUL_S.PH[R2]	$RD, RS, RT	RD = [RS_L^± × RT_L^±]_R || [RS_R^± × RT_R^±]_R

Fractional Multiply Operations: 16-bit Data » GPR			
MULQ_RS.PH	$RD, RS, RT	RD = [RS_L^± ⦿ RT_L^±]_L || [RS_R^± ⦿ RT_R^±]_L	■■ ■■
MULQ_S.PH[R2]	$RD, RS, RT	RD = [RS_L^± • RT_L^±]_L || [RS_R^± • RT_R^±]_L	■■ ■■
MULEQ_S.W.PHL	$RD, RS, RT	RD = [RS_L^± • RT_L^±]	■□ ■□
MULEQ_S.W.PHR	$RD, RS, RT	RD = [RS_R^± • RT_R^±]	□■ □■

Fractional Multiply Operations: 16-bit Data » Accumulator			
DPAQ_S.W.PH	$AC, RS, RT	AC += [RS_L^± • RT_L^±] + [RS_R^± • RT_R^±]	■■ ■■
DPAQX_S.W.PH[R2]	$AC, RS, RT	AC += [RS_L^± • RT_R^±] + [RS_R^± • RT_L^±]	■■ ■■
DPAQX_SA.W.PH[R2]	$AC, RS, RT	AC = [AC + [RS_L^± • RT_R^±] + [RS_R^± • RT_L^±]]	■■ ■■
DPSQ_S.W.PH	$AC, RS, RT	AC −= [RS_L^± • RT_L^±] + [RS_R^± • RT_R^±]	■■ ■■
DPSQX_S.W.PH[R2]	$AC, RS, RT	AC −= [RS_L^± • RT_R^±] + [RS_R^± • RT_L^±]	■■ ■■
DPSQX_SA.W.PH[R2]	$AC, RS, RT	AC = [AC − [RS_L^± • RT_R^±] + [RS_R^± • RT_L^±]]	■■ ■■
MULSAQ_S.W.PH	$AC, RS, RT	AC += [RS_L^± • RT_L^±] − [RS_R^± • RT_R^±]	■■ ■■
MAQ_S.W.PHL	$AC, RS, RT	AC += [RS_L^± • RT_L^±]	■□ ■□
MAQ_S.W.PHR	$AC, RS, RT	AC += [RS_R^± • RT_R^±]	□■ □■
MAQ_SA.W.PHL	$AC, RS, RT	AC = [AC^± + [RS_L^± • RT_L^±]]	■□ ■□
MAQ_SA.W.PHR	$AC, RS, RT	AC = [AC^± + [RS_R^± • RT_R^±]]	□■ □■

Accumulator Extract Operations: 16-bit Data		
EXTR_S.H	$RD, AC, shift5	RD = [AC >> shift5]_R
EXTRV_S.H	$RD, AC, RT	RD = [AC >> RT_4:0]_R

DSPControl Register Access Operations		
RDDSP	$RD, mask10	RD = ReadDSPControl(mask10_5:0)
WRDSP	$RS, mask10	WriteDSPControl(mask10_5:0, RS)

Arithmetic Operations: 32-bit Data		
ABSQ_S.W	R_D, R_S	$R_D = [\lvert R_S \rvert]$
ADDQ_S.W	R_D, R_S, R_T	$R_D = [R_S^{\pm} + R_T^{\pm}]$
ADDQH.W[R2]	R_D, R_S, R_T	$R_D = (R_S^{\pm} + R_T^{\pm}) >> 1$
ADDQH_R.W[R2]	R_D, R_S, R_T	$R_D = ®((R_S^{\pm} + R_T^{\pm}) >> 1)$
ADDSC	R_D, R_S, R_T	$C::R_D = R_S^{\varnothing} + R_T^{\varnothing}$
ADDWC	R_D, R_S, R_T	$R_D = R_S^{\varnothing} + R_T^{\varnothing} + C$
SUBQ_S.W	R_D, R_S, R_T	$R_D = [R_S - R_T]$
SUBQH.W[R2]	R_D, R_S, R_T	$R_D = (R_S^{\pm} - R_T^{\pm}) >> 1$
SUBQH_R.W[R2]	R_D, R_S, R_T	$R_D = ®((R_S^{\pm} - R_T^{\pm}) >> 1)$

Shift Operations: 32-bit Data		
SHLL_S.W	R_D, R_S, SHIFT5	$R_D = [R_S << \text{SHIFT5}]$
SHLLV_S.W	R_D, R_S, R_T	$R_D = [R_S << R_{T4:0}]$
SHRA_R.W	R_D, R_S, SHIFT5	$R_D = ®(R_S^{\pm} >> \text{SHIFT5})$
SHRAV_R.W	R_D, R_S, R_T	$R_D = ®(R_S^{\pm} >> R_{T4:0})$

Fractional Multiply Operations: 32-bit Data » GPR		
MULQ_RS.W[R2]	R_D, R_S, R_T	$R_D = [R_S^{\pm} \circledcirc R_T^{\pm}]_L$
MULQ_S.W[R2]	R_D, R_S, R_T	$R_D = [R_S^{\pm} \bullet R_T^{\pm}]_L$

Fractional Multiply Operations: 32-bit Data » Accumulator		
DPAQ_SA.L.W	A_C, R_S, R_T	$A_C = [A_C^{\pm} + [R_S^{\pm} \bullet R_T^{\pm}]]$
DPSQ_SA.L.W	A_C, R_S, R_T	$A_C = [A_C^{\pm} - [R_S^{\pm} \bullet R_T^{\pm}]]$

Integer Multiply Operations: 32-bit Data » Accumulator		
MADD[R2]	A_C, R_S, R_T	$A_C \mathrel{+}= R_S^{\pm} \times R_T^{\pm}$
MADDU[R2]	A_C, R_S, R_T	$A_C \mathrel{+}= R_S^{\varnothing} \times R_T^{\varnothing}$
MSUB[R2]	A_C, R_S, R_T	$A_C \mathrel{-}= R_S^{\pm} \times R_T^{\pm}$
MSUBU[R2]	A_C, R_S, R_T	$A_C \mathrel{-}= R_S^{\varnothing} \times R_T^{\varnothing}$
MULT[R2]	A_C, R_S, R_T	$A_C = R_S^{\pm} \times R_T^{\pm}$
MULTU[R2]	A_C, R_S, R_T	$A_C = R_S^{\varnothing} \times R_T^{\varnothing}$

Precision Reduction (Data Packing) Operations: 32-bit Data		
PRECRQ.PH.W	R_D, R_S, R_T	$R_D = R_{S_L} \parallel R_{T_L}$
PRECRQ_RS.PH.W	R_D, R_S, R_T	$R_D = ®[R_S^{\pm}]_L \parallel ®[R_T^{\pm}]_L$
PRECR_SRA.PH.W[R2]	R_D, R_S, SHIFT5	$R_D = (R_D^{\pm} >> \text{SHIFT5})_R \parallel (R_S^{\pm} >> \text{SHIFT5})_R$
PRECR_SRA_R.PH.W[R2]	R_D, R_S, SHIFT5	$R_D = ®(R_D^{\pm} >> \text{SHIFT5})_R \parallel ®(R_S^{\pm} >> \text{SHIFT5})_R$

Accumulator Extract Operations: 32-bit Data		
EXTR.W	R_D, A_C, SHIFT5	$R_D = (A_C >> \text{SHIFT5})_R$
EXTR_R.W	R_D, A_C, SHIFT5	$R_D = ®(A_C >> \text{SHIFT5})_R$
EXTR_RS.W	R_D, A_C, SHIFT5	$R_D = ®[A_C >> \text{SHIFT5}]_R$
EXTRV.W	R_D, A_C, R_T	$R_D = (A_C >> R_{T4:0})_R$
EXTRV_R.W	R_D, A_C, R_T	$R_D = ®(A_C >> R_{T4:0})_R$
EXTRV_RS.W	R_D, A_C, R_T	$R_D = ®[A_C >> R_{T4:0}]_R$

Bit-Field Extract Operations		
BPOSGE32	OFF18	IF POS ≥ 32, PC += OFF18[±]
EXTP	R_D, A_C, SIZE5	$R_D = A_{C\text{POS:POS–SIZE5}}$
EXTPDP	R_D, A_C, SIZE5	$R_D = A_{C\text{POS:POS–SIZE5}}$, POS −= (SIZE5 + 1)
EXTPV	R_D, A_C, R_S	$R_D = A_{C\text{POS:POS–RS4:0}}$
EXTPDPV	R_D, A_C, R_S	$R_D = A_{C\text{POS:POS–RS4:0}}$, POS −= ($R_{S4:0}$ + 1)
MTHLIP	R_S, A_C	$A_{C\text{HI}} = A_{C\text{LO}}$, $A_{C\text{LO}} = R_S$, POS += 32

Load Data Operations		
BITREV	R_D, R_S	$R_D = R_{S0:15}$
LBUX	R_D, $R_S(R_T)$	$R_D = \text{MEM8}(R_S + R_T)^{\varnothing}$
LHX	R_D, $R_S(R_T)$	$R_D = \text{MEM16}(R_S + R_T)^{\pm}$
LWX	R_D, $R_S(R_T)$	$R_D = \text{MEM32}(R_S + R_T)$
MODSUB	R_D, R_S, R_T	$R_D = (R_S \neq 0)\ ?\ (R_S - R_{T7:0}) : R_{T23:8}$

Accumulator Operations		
MFHI	R_D, A_C	$R_D = A_{C\text{HI}}$
MFLO	R_D, A_C	$R_D = A_{C\text{LO}}$
MTHI	R_S, A_C	$A_{C\text{HI}} = R_S$
MTLO	R_S, A_C	$A_{C\text{LO}} = R_S$
SHILO	A_C, SHIFT6	$A_C = (\text{SHIFT6}^{\pm} \geq 0)\ ?\ (A_C >> \text{SHIFT6}^{\pm}) : (A_C << -\text{SHIFT6}^{\pm})$
SHILOV	A_C, R_S	$A_C = (R_{S5:0}^{\pm} \geq 0)\ ?\ (A_C >> R_{S5:0}^{\pm}) : (A_C << -R_{S5:0}^{\pm})$

Bit-Field Operations		
APPEND[R2]	R_D, R_S, SHIFT5	$R_D = (R_D << \text{SHIFT5}) :: R_{S\text{SHIFT5–1:0}}$
BALIGN[R2]	R_D, R_S, BPOS2	$R_D = (R_D << (8 \times \text{BPOS2})) :: (R_S >> (32 - 8 \times \text{BPOS2}))$
INSV	R_D, R_S	$R_{D\text{POS+SIZE–1:POS}} = R_{S\text{SIZE–1:0}}$
PACKRL.PH	R_D, R_S, R_T	$R_D = R_{S_R} \parallel R_{T_L}$
PREPEND[R2]	R_D, R_S, SHIFT5	$R_D = R_{S\text{SHIFT5–1:0}} :: (R_D >> \text{SHIFT5})$