Instruction Set Summary

1 ADD ADD Rd, Ra, Rb Rd ← Ra + rb C, ν, n, z A 00010 - 2 ADDI ADDI Rd, Ra, #imm5 Rd ← Ra + rimm5 C, ν, n, z B 00011 - 3 ADDIB ADDI Rd, #imm8 Rd ← Rd + rimm8 C, ν, n, z B 00011 - 4 ADC ADC Rd, Ra, Rb Rd ← Ra + rimm5 + C C, ν, n, z A 00100 - 5 ADCI ADC Rd, Ra, frimm5 Rd ← Ra + rimm5 + C C, ν, n, z A 00100 - 6 NEG NEG Rd, Ra Rb Rd ← Ra - rimm5 + C C, ν, n, z A 10110 - 7 SUB SUB SUB Rd, Ra, Bb Rd ← Ra - rimm5 + C C, ν, n, z A 10110 - 8 SUB SUB SUB Rd, Ra, mm8 Rd ← Ra - rimm5 + C C, ν, n, z A 10110 - 9 SUB RD SUB Rd, #imm8 Rd ← Ra - rimm5 + C C, ν, n, z A 10110 - 11 SUC SUC Rd, Ra, Rb Rd ← Ra - rimm5 + C C, ν, n, z A 10110 - 12 CMP CMP Ra, Rb Rd ← Ra - rimm5 + C C, ν, n, z A 10110 - 13 CMPI CMP Ra, Rb Rd ← Ra - rimm5 + C C, ν, n, z A 10110 - 14 AND AND Rd, Ra, Rb Rd ← Ra - rimm5 + C C, ν, n, z A 10111 - 15 OR OR Rd, Ra, Rb Rd ← Ra - rimm5 + C C, ν, n, z A 10111 - 16 XOR XOR Rd, Ra, Rb Rd ← Ra - Rb C, ν, n, z A 10111 - 17 NOT NOT Rd, Ra Rb Rd ← Ra OR Rb R, z A Rb Rd ← Ra OR Rb Rd ← Ra		Mnemonic	Syntax	Semantics	Flags	Encoding	Opcode	Cond.
ADDIB ADDIB Rd, #imm8 Rd ← Rd + imm8 C, V,n,z A O0010 - ADC Rd, Ra, Rb ADCI ADC Rd, Ra, Bm S Rd ← Ra + Rb + C C, V,n,z A O0100 - NEG NEG NEG Rd, Ra, Bmm5 Rd ← Ra + imm5 + C C, V,n,z A O0101 - SUB SUBI SUB Rd, Ra, Rb Rd ← Ra - imm5 + C C, V,n,z A O0101 - SUB SUBI SUB Rd, Ra, Bmm5 Rd ← Ra - imm5 C, V,n,z A O11010 - SUC SUC Rd, Ra, Rb Rd ← Rd - imm8 C, V,n,z B O1101 - SUC SUC Rd, Ra, Rb Rd ← Ra - imm5 C, V,n,z A O1100 - SUC SUC Rd, Ra, Rb Rd ← Ra - imm5 - C C, V,n,z A O1100 - CMP Ra, Rb Rd ← Ra - imm5 - C C, V,n,z A O1100 - CMP Ra, Rb Rd ← Ra - imm5 - C C, V,n,z A O1100 - CMP Ra, Rb Rd ← Ra - imm5 - C C, V,n,z A O1101 - CMP CMP Ra, #imm5 Rd ← Ra - imm5 - C C, V,n,z A O1101 - CMP CMP Ra, #imm5 Rd ← Ra - Rb C, V,n,z A O1111 - CMP CMP Ra, Rb Rd ← Ra - Rb C, V,n,z A O1111 - CMP CMP Ra, Rb Rd ← Ra - Rb C, V,n,z A O1111 - CMP CMP Ra, Rb Rd ← Ra - Rb C, V,n,z A O1111 - CMP CMP Ra, Rb Rd ← Ra - Rb C, V,n,z A O1111 - CMP CMP Ra, Rb Rd ← Ra - Rb C, V,n,z A O1111 - CMP CMP Ra, Rb Rd ← Ra - Rb C, V,n,z A O1111 - CMP CMP Ra, Rb Rd ← Ra - Rb C, V,n,z A O1111 - CMP CMP Ra, Rb Rd ← Ra - Rb C, V,n,z A O1111 - CMP CMP Ra, Rb Rd ← Ra NDR Rb N,z A O1111 - CMP CMP Ra, Rb Rd ← Ra NDR Rb N,z A O1001 - CMP Ra, Rb Rd ← Ra NDR Rb N,z A O1001 - CMP Ra, Rb Rd ← Ra NDR Rb N,z A O1001 - CMP Ra, Rb Rd ← Ra NDR Rb N,z A O1001 - CMP Ra, Rb Rd ← Ra NDR Rb N,z A O1011 - CMP Ra, Rb Rd ← Ra NDR Rb N,z A O1011 - CMP Ra, Rb Rd ← Ra NDR Rb N,z A O1011 - CMP Ra, Rb Rd ← Ra NDR Rb N,z A O1011 - CMP Ra, Rb Rd ← Ra NDR Rb N,z A O1011 - CMP Ra, Rb Rd ← Ra NDR Rb N,z A O1011 - CMP Ra, Rb Rd ← Ra NDR Rb N,z A O1011 - CMP Ra, Rb Rd ← Ra NDR Rb N,z A O1011 - CMP Ra, Rb Rd ← Ra NDR Rb N,z A O1011 - CMP Ra, Rb Rd ← Ra NDR Rb N,z A O1011 - CMP Ra, Rb Rd ← Ra NDR Rb N,z A O1011 - CMP Ra, Rb Rd ← Ra NDR Rb N,z A O1011 - CMP Ra, Rb Rd ← Ra NDR Rb N,z A O1011 - CMP Ra, Rb Rd ← Ra NDR Rb N,z A O1011 - CMP Ra, Rb Rd ← Ra NDR Rb N,z A O1011 - CMP Ra, Rb Rd ← Ra NDR Rb N,z A O1011 - CMP Ra, Rb Rd ← Ra NDR Rb N,z A O1011 - CMP Ra, Rb Rd ← Ra NDR Rb N,z A O1011 - CMP Ra, R	1	ADD	ADD Rd, Ra, Rb	Rd ← Ra + Rb	c,v,n,z	Α	00010	-
ADC ADC Rd, Ra, Rb Rd ← Ra + Rb + c C, V, n, z A 00100 - C ADC Rd, Ra, #imm5 Rd ← Ra + imm5 + c C, V, n, z A 11010 - C RD RG Rd, Ra Rd ← Ra + imm5 + c C, V, n, z A 11010 - C RD Rd Rd ← Ra - Rb C, V, n, z A 11010 - C RD Rd Rd ← Ra - Rb C, V, n, z A 11010 - C RD Rd Rd ← Ra - Rb C, V, n, z A 11010 - C RD Rd Rd ← Ra - Rb C, V, n, z A 11010 - C RD Rd Rd ← Ra - Rb C, V, n, z A 10110 - C RD Rd Rd ← Ra - Rb C, V, n, z A 10110 - C RD Rd Rd ← Ra - Rb C, V, n, z A 10110 - C RD Rd Rd ← Ra - Rb - c RD Rd Rd ← Ra - Rb - c RD Rd Rd ← Ra - Rb - c RD Rd Rd ← Ra - Rb - c RD Rd Rd ← Ra - Rb - c RD Rd Rd ← Ra - Rb - c RD Rd Rd ← Ra - Rb - c RD Rd Rd ← Ra - Rb - c RD Rd Rd Rd - Ra - Rb - c RD Rd Rd Rd - Ra - Rb - c RD Rd Rd Rd - Ra - Rb - c RD Rd Rd Rd - Ra - Rb - c RD Rd Rd Rd - Ra - Rb - c RD Rd Rd Rd - Ra - Rb - c RD Rd Rd Rd - Ra - Rb - c RD Rd Rd Rd - Ra - Rb - c RD Rd Rd Rd - Ra - Rb - c RD Rd Rd Rd - Ra - Rb - c RD Rd Rd Rd - Ra - Rb - c RD Rd Rd Rd - Ra - Rb - c RD Rd Rd Rd - Ra - Rb - c RD Rd Rd Rd - Ra - Rb - c RD Rd Rd Rd - Ra - Rb - c RD Rd Rd Rd - Ra - Rb - c RD Rd Rd - Ra - Rb - c RD Rd Rd - Ra - Rb - c RD Rd Rd - Ra - Rb - c RD Rd Rd - Ra - Rb - c RD Rd Rd - Ra - Rb - c RD Rd Rd - Ra - Rb - c Rd Rd - Ra - Rb - c Rd Rd - Ra - Rb - c Rd Rd - Rd - Rd Rd - Rd - Rd - Rd -	2	ADDI	ADDI Rd, Ra, #imm5	Rd ← Ra + imm5	c,v,n,z	Α	00110	-
S ADCI NEG ADCI Rd, Ra, #imm5 Rd ← Ra + imm5 + c Rd ← 0 - Ra C, ν, ν, ν, λ A 00101 - 6 NEG NEG Rd, Ra Nb Rd ← 0 - Ra C, ν, ν, λ A 01010 - 7 SUB SUBI Rd, Ra, #b Rd ← Ra - Rb C, ν, ν, λ A 01010 - 8 SUBI SUBI Rd, #imm5 Rd ← Ra - imm5 C, ν, ν, λ A 01110 - 9 SUBIB SUBIB Rd, #imm5 Rd ← Ra - imm5 C, ν, ν, λ A 01110 - 10 SUC SUC Rd, Ra, Rb Rd ← Ra - imm5 - C C, ν, ν, λ A 01101 - 11 SUCI SUCI Rd, Ra, #imm5 Rd ← Ra - imm5 - C C, ν, ν, λ A 01101 - 12 CMP CMP Ra, Rb Rd ← Ra - imm5 - C C, ν, ν, λ A 01111 - 12 CMP CMP Ra, Rb Rd ← Ra - Rb C, ν, ν, λ A 01111 - 14 AND AND Rd, Ra, Rb Rd ← Ra AND Rb N, λ A 100011 - 15 OR OR Rd, Ra, Rb Rd ← Ra NAD Rb N, λ A 10011 - 16 XOR XOR Rd, Ra, Rb Rd ← Ra NAD Rb N, λ A 10111 <td>3</td> <td>ADDIB</td> <td>ADDIB Rd, #imm8</td> <td>$Rd \leftarrow Rd + imm8$</td> <td>c,v,n,z</td> <td>В</td> <td>00011</td> <td>-</td>	3	ADDIB	ADDIB Rd, #imm8	$Rd \leftarrow Rd + imm8$	c,v,n,z	В	00011	-
6 NEG NEG Rd, Ra Rd ← 0 - Ra C, V, n, 2 A 11010 - 7 SUB SUB Rd, Ra, Rb Rd ← Ra - Rb C, V, n, 2 A 01110 - 9 SUBIB SUBIB Rd, #imm5 Rd ← Ra - Rb C, V, n, 2 A 01110 - 10 SUC SUC Rd, Ra, #b Rd ← Ra - Rb - C C, V, n, 2 A 01101 - 11 SUCI SUCI Rd, Ra, #b Rd ← Ra - Rb - C C, V, n, 2 A 01101 - 12 CMP CMP Ra, Rb Rd ← Ra - Rb - C C, V, n, 2 A 01101 - 12 CMP CMP Ra, Rb Rd ← Ra - Rb C, V, n, 2 A 01101 - 12 CMP CMP Ra, Rb Rd ← Ra - Rb C, V, n, 2 A 01101 - 13 CMPI CMP Ra, Rb Rd ← Ra ND Rb n, 2 A 10001 - 14 AND RD, Ra, Rb Rd ← Ra ND Rb n, 2 A 10001 <td>4</td> <td>ADC</td> <td>ADC Rd, Ra, Rb</td> <td>$Rd \leftarrow Ra + Rb + c$</td> <td>c,v,n,z</td> <td>Α</td> <td>00100</td> <td>-</td>	4	ADC	ADC Rd, Ra, Rb	$Rd \leftarrow Ra + Rb + c$	c,v,n,z	Α	00100	-
7 SUB SUB Rd, Ra, Rb Rd ← Ra - Rb c,v,n,z A 01010 - 8 SUBI SUBI Rd, Ra, simms Rd ← Ra - imms c,v,n,z A 01110 - 10 SUC SUC Rd, Ra, Rb Rd ← Ra - imms c,v,n,z A 01100 - 11 SUC SUC Rd, Ra, simms Rd ← Ra - imms - c c,v,n,z A 01101 - 12 CMP CMP Ra, Rb Rd ← Ra - imms - c c,v,n,z A 00111 - 13 CMPI CMP Ra, Rb Rd ← Ra - Rb c,v,n,z A 00111 - 13 CMPI CMP Ra, Rb Rd ← Ra - Rb c,v,n,z A 00111 - 14 AND AND Rd, Ra, Rb Rd ← Ra - Rb c,v,n,z A 00111 - 15 OR CMPI Ra, Rb Rd ← Ra NAD Rb n,z A 10001 - 16 XOR XOR Rd, Ra, Rb Rd ← Ra OR Rb n,z A <td< td=""><td>5</td><td>ADCI</td><td>ADCI Rd, Ra, #imm5</td><td>$Rd \leftarrow Ra + imm5 + c$</td><td>c,v,n,z</td><td>Α</td><td>00101</td><td>-</td></td<>	5	ADCI	ADCI Rd, Ra, #imm5	$Rd \leftarrow Ra + imm5 + c$	c,v,n,z	Α	00101	-
8 SUBI SUBIR Rd, Ra, #imm5 Rd ← Ra - imm5 C,v,n,z A 01110 - 9 SUBIB Rd, #imm8 Rd ← Rd - imm8 C,v,n,z B 01011 - 10 SUC SUC Rd, Ra, #imm5 Rd ← Ra - Rb - c C,v,n,z A 01100 - 11 SUCI SUCI Rd, Ra, Rb Rd ← Ra - imm5 - c C,v,n,z A 001101 - 12 CMP CMP Ra, Rb Ra - imm5 - c C,v,n,z A 00111 - 12 CMPI CMP Ra, Rb Rd ← Ra - Rb C,v,n,z A 00111 - 14 AND AND Rd, Ra, Rb Rd ← Ra AND Rb n,z A 10000 - 15 OR OR Rd, Ra, Rb Rd ← Ra OR Rb n,z A 10001 - 16 XOR XOR Rd, Ra, Rb Rd ← Ra NAND Rb n,z A 10010 - 17 NOT NOR Rd, Ra, Rb Rd ← Ra NOR Rb n,z A 10111	6	NEG	NEG Rd, Ra	Rd ← 0 - Ra	c,v,n,z	Α	11010	-
9 SUBIB SUC BIB Rd, #imm8 Rd ← Rd - imm8 C,v,n,z B 01011 - 10 SUC SUC Rd, Ra, Rb Rd ← Ra - Rb - c C,v,n,z A 01100 - 12 CMP CMP Ra, Rb Rd ← Ra - Rb - c C,v,n,z A 01101 - 12 CMP CMP Ra, Rb Rd ← Ra - Rb C,v,n,z A 00111 - 13 CMPI CMPI Ra, #imm5 Ra - imm5 C,v,n,z A 00111 - 14 AND AND Rd, Ra, Rb Rd ← Ra OR Rb n,z A 10000 - 15 OR OR Rd, Ra, Rb Rd ← Ra NOR Rb n,z A 10001 - 16 XOR XOR Rd, Ra, Rb Rd ← Ra NOR Rb n,z A 10010 - 17 NOT NOT Rd, Ra Rd ← Ra NOR Rb n,z A 10110 - 18 NAND NOR Rd, Ra, #imm4 Rd ← Ra NOR Rb n,z A 10111 - <	7	SUB	SUB Rd, Ra, Rb	Rd ← Ra - Rb	c,v,n,z	Α	01010	-
10 SUC SUCRd, Ra, Rb Rd ← Ra - Imm5 - c C, V, N, Z A 01100 - 11 SUCI Rd, Ra, #imm5 Rd ← Ra - imm5 - c C, V, N, Z A 01101 - 12 CMP CMP Ra, Rb Ra - imm5 C, V, N, Z A 00111 - 13 CMPI CMPI Ra, #imm5 Ra - imm5 C, V, N, Z A 00111 - 14 AND AND Rd, Ra, Rb Rd ← Ra AND Rb N, Z A 10000 - 15 OR OR Rd, Ra, Rb Rd ← Ra AND Rb N, Z A 10001 - 16 XOR XOR Rd, Ra, Rb Rd ← Ra AND Rb N, Z A 10010 - 17 NOT NOT Rd, Ra Rd ← Ra NAND Rb N, Z A 10010 - 18 NAND NAND Rd, Ra, Rb Rd ← Ra NAND Rb N, Z A 10110 - 19 NOR NCR Rd, Ra, #imm4 Rd ← Ra S C imm4 n, Z A 10111 <td>8</td> <td>SUBI</td> <td>SUBI Rd, Ra, #imm5</td> <td>Rd ← Ra - imm5</td> <td>c,v,n,z</td> <td>Α</td> <td>01110</td> <td>-</td>	8	SUBI	SUBI Rd, Ra, #imm5	Rd ← Ra - imm5	c,v,n,z	Α	01110	-
11 SUCI SUCI Rd, Ra, #imm5 Rd ← Ra - imm5 - c C, V, n, 2 A 01101 - 12 CMP CMP Ra, Rb Ra - Rb c, V, n, 2 A 001111 - 13 CMPI CMP Ra, Rb Ra - Rb c, V, n, 2 A 001111 - 14 AND AND Rd, Ra, Rb Rd ← Ra AND Rb n, 2 A 10000 - 15 OR OR Rd, Ra, Rb Rd ← Ra NOR Rb n, 2 A 10001 - 16 XOR XOR Rd, Ra, Rb Rd ← Ra NOR Rb n, 2 A 10010 - 16 XOR XOR Rd, Ra, Rb Rd ← Ra NOR Rb n, 2 A 10010 - 18 NAND NAND Rd, Ra, Rb Rd ← Ra NOR Rb n, 2 A 10111 - 19 NOR NOR Rd, Ra, #imm4 Rd ← Ra <> imm4 n, 2 A 10111 - 20 LSL LSR Rd, Ra, #imm4 Rd ← Ra >> imm4 n, 2 A <	9	SUBIB	SUBIB Rd, #imm8	Rd ← Rd - imm8	c,v,n,z	В	01011	-
12 CMP CMP Ra, Rb Ra - Rb C,v,n,z A 00111 - 13 CMPI CMPI Ra, #imm5 Ra - imm5 c,v,n,z A 01111 - 14 AND AND Rd, Ra, Rb Rd ← Ra AND Rb n,z A 10000 - 15 OR OR Rd, Ra, Rb Rd ← Ra OR Rb n,z A 10000 - 16 XOR XOR Rd, Ra, Rb Rd ← Ra OR Rb n,z A 10011 - 16 XOR XOR Rd, Ra, Rb Rd ← Ra NAND Rb n,z A 10010 - 17 NOT NOR Rd, Ra, Rb Rd ← Ra NAND Rb n,z A 10110 - 18 NAND NOR Rd, Ra, Rb Rd ← Ra S NA RD Rb n,z A 10111 - 19 NOR NOR Rd, Ra, #imm4 Rd ← Ra < mMA	10	SUC	SUC Rd, Ra, Rb	Rd ← Ra - Rb - c	c,v,n,z	Α	01100	-
13 CMPI CMPI Ra, #imm5 Ra - imm5 c,v,n,z A 01111 - 14 AND AND Rd, Ra, Rb Rd ← Ra AND Rb n,z A 10000 - 15 OR OR Rd, Ra, Rb Rd ← Ra OR Rb n,z A 10001 - 16 XOR XOR Rd, Ra, Rb Rd ← Ra OR Rb n,z A 10011 - 17 NOT NOT Rd, Ra Rd ← NOT Ra n,z A 10010 - 18 NAND NAND Rd, Ra, Rb Rd ← Ra NAND Rb n,z A 10110 - 19 NOR NOR Rd, Ra, Rb Rd ← Ra NAOR Rb n,z A 10111 - 19 NOR NOR Rd, Ra, Rb Rd ← Ra SN ROR Rb n,z A 10111 - 20 LSL LSL Rd, Ra, #imm4 Rd ← Ra SN Bma n,z A 11101 - 21 LSR LSR Rd, Ra, #imm4 Rd ← Ra SN Bma n,z A 11101 <	11	SUCI	SUCI Rd, Ra, #imm5	Rd ← Ra - imm5 - c	c,v,n,z	Α	01101	-
14 AND AND Rd, Ra, Rb Rd ← Ra AND Rb n,z A 10000 - 15 OR OR Rd, Ra, Rb Rd ← Ra OR Rb n,z A 10001 - 16 XOR XOR Rd, Ra, Rb Rd ← Ra XOR Rb n,z A 10010 - 17 NOT NOT Rd, Ra Rd ← NOT Ra n,z A 10010 - 18 NAND NAND Rd, Ra, Rb Rd ← Ra NOR Rb n,z A 10010 - 19 NOR NOR Rd, Ra, Rb Rd ← Ra NOR Rb n,z A 10110 - 19 NOR NOR Rd, Ra, Rb Rd ← Ra NOR Rb n,z A 10111 - 20 LSL LSL Rd, Ra, #imm4 Rd ← Ra > imm4 n,z A 11111 - 21 LSR LSR Rd, Ra, #imm4 Rd ← Ra > imm4 n,z A 11101 - 22 ASR ASR Rd, Ra, #imm4 Rd ← Ra > imm4 n,z A 11101 <td< td=""><td>12</td><td>CMP</td><td>CMP Ra, Rb</td><td>Ra - Rb</td><td>c,v,n,z</td><td>Α</td><td>00111</td><td>-</td></td<>	12	CMP	CMP Ra, Rb	Ra - Rb	c,v,n,z	Α	00111	-
15 OR OR Rd, Ra, Rb Rd ← Ra OR Rb n,z A 10001 - 16 XOR XOR Rd, Ra, Rb Rd ← Ra OR Rb n,z A 10011 - 17 NOT NOT Rd, Ra Rd ← NOT Ra n,z A 10010 - 18 NAND NAND Rd, Ra, Rb Rd ← Ra NOR Rb n,z A 10110 - 19 NOR NOR Rd, Ra, Rb Rd ← Ra NOR Rb n,z A 10111 - 19 NOR NOR Rd, Ra, Rb Rd ← Ra NOR Rb n,z A 10111 - 20 LSL LSL Rd, Ra, #imm4 Rd ← Ra >> imm4 n,z A 11110 - 21 LSR LSR Rd, Ra, #imm4 Rd ← Ra >> imm4 n,z A 11110 - 21 LSR LSR Rd, Ra, #imm4 Rd ← Ra >> imm4 n,z A 11100 - 22 ASR ASR Rd, Ra, #imm4 Rd ← Ra >> imm4 n,z A 11110	13	CMPI	CMPI Ra, #imm5	Ra - imm5	c,v,n,z	Α	01111	-
16 XOR XOR Rd, Ra, Rb Rd ∈ Ra XOR Rb n,z A 10011 - 17 NOT NOT Rd, Ra Rd ∈ NOT Ra n,z A 10010 - 18 NAND NAND Rd, Ra, Rb Rd ∈ Ra NAND Rb n,z A 10110 - 19 NOR NOR Rd, Ra, Rb Rd ∈ Ra NOR Rb n,z A 10111 - 20 LSL LSL Rd, Ra, #imm4 Rd ∈ Ra >> imm4 n,z A 11111 - 21 LSR LSR Rd, Ra, #imm4 Rd ∈ Ra >> imm4 n,z A 11110 - 22 ASR ASR Rd, Ra, #imm4 Rd ∈ Ra >> imm4 n,z A 11100 - 23 LDW LDW Rd, [Ra, #imm5] Rd € Mem[Ra + imm5] - C 00000 - 24 STW SDW Rd, [Ra, #imm5] Rd € Mem[Ra + imm5] - C C 01000 - 25 LUI LUI Rd, #imm8 Rd € Rd[I5:8], imm8] -	14	AND	AND Rd, Ra, Rb	Rd ← Ra AND Rb	n,z	Α	10000	-
17 NOT NOT Rd, Ra Rd ← NOT Ra n,z A 10010 - 18 NAND NAND Rd, Ra, Rb Rd ← Ra NAND Rb n,z A 10110 - 19 NOR NOR Rd, Ra, Rb Rd ← Ra NOR Rb n,z A 10111 - 20 LSL LSL Rd, Ra, #imm4 Rd ← Ra <> imm4 n,z A 11111 - 21 LSR LSR Rd, Ra, #imm4 Rd ← Ra >> imm4 n,z A 11110 - 22 ASR ASR Rd, Ra, #imm4 Rd ← Ra >> imm4 n,z A 11100 - 23 LDW LDW Rd, [Ra, #imm5] Rd ← Mem[Ra + imm5] - C 00000 - 24 STW SDW Rd, [Ra, #imm5] Mem[Ra + imm5] ← Rd - C 00000 - 25 LUI LUI Rd, #imm8 Rd ← [Rd[fs.8], imm8] - B 10100 - 26 LLI LLI Rd, #imm8 Rd ← [Rd[fs.8], imm8] - B	15	OR	OR Rd, Ra, Rb	Rd ← Ra OR Rb	n,z	Α	10001	-
18 NAND NAND Rd, Ra, Rb Rd ← Ra NAND Rb n,z A 10110 - 19 NOR NOR Rd, Ra, Rb Rd ← Ra NOR Rb n,z A 10111 - 20 LSL LSL Rd, Ra, #imm4 Rd ← Ra <	16	XOR	XOR Rd, Ra, Rb	$Rd \leftarrow Ra XOR Rb$	n,z	Α	10011	-
19 NOR NOR Rd, Ra, Rb Rd ← Ra NOR Rb n,z A 10111 - 20 LSL LSL Rd, Ra, #imm4 Rd ← Ra < < imm4	17	NOT	NOT Rd, Ra	Rd ← NOT Ra	n,z	Α	10010	-
20 LSL LSL Rd, Ra, #imm4 Rd ← Ra << imm4	18	NAND	NAND Rd, Ra, Rb	Rd ← Ra NAND Rb	n,z	Α	10110	-
$ \begin{array}{cccccccccccccccccccccccccccccccccccc$	19	NOR	NOR Rd, Ra, Rb	Rd ← Ra NOR Rb	n,z	Α	10111	-
$ \begin{array}{cccccccccccccccccccccccccccccccccccc$	20	LSL	LSL Rd, Ra, #imm4	Rd ← Ra << imm4	n,z	Α	11111	-
23 LDW LDW Rd, [Ra, #imm5] Rd ← Mem[Ra + imm5] - C 00000 - 24 STW SDW Rd, [Ra, #imm5] Mem[Ra + imm5] ← Rd - C 01000 - 25 LUI LUI Rd, #imm8 Rd ← {imm8, 0} - B 10100 - 26 LLI LLI Rd, #imm8 Rd ← {Rd[15:8], imm8} - B 10101 - 27 BR BR LABEL PC ← PC + imm8 - D - 000 28 BNE BNE LABEL (z==0)? PC ← PC + imm8 - D - 110 29 BE BE LABEL (z==0)? PC ← PC + imm8 - D - 110 30 BLT BLT LABEL (n& YOR ~n& Y)? PC ← PC + imm8 - D - 100 31 BGE BGE LABEL (n& YOR ~n& Y)? PC ← PC + imm8 - D - 101 32 BWL BWL LABEL LR ← PC + 1; PC ← PC + imm8 - D -	21	LSR	LSR Rd, Ra, #imm4	Rd ← Ra >> imm4	n,z	Α	11101	-
$ \begin{array}{cccccccccccccccccccccccccccccccccccc$	22	ASR	ASR Rd, Ra, #imm4	Rd ← Ra >>> imm4	n,z	Α	11100	-
$ \begin{array}{cccccccccccccccccccccccccccccccccccc$	23	LDW	LDW Rd, [Ra, #imm5]	Rd ← Mem[Ra + imm5]	-	С	00000	-
$ \begin{array}{cccccccccccccccccccccccccccccccccccc$	24	STW	SDW Rd, [Ra, #imm5]	Mem[Ra + imm5] ← Rd	-	С	01000	-
$ \begin{array}{cccccccccccccccccccccccccccccccccccc$	25	LUI	LUI Rd, #imm8	Rd ← {imm8, 0}	-	В	10100	-
28 BNE BNE LABEL $(z==0)$? $PC \leftarrow PC + imm8$ - D - 110 29 BE BE LABEL $(z==1)$? $PC \leftarrow PC + imm8$ - D - 111 30 BLT BLT LABEL $(n\& \sim V \cap R \sim V)$? $PC \leftarrow PC + imm8$ - D - 100 31 BGE BGE LABEL $(n\& V \cap R \sim N\& \sim V)$? $PC \leftarrow PC + imm8$ - D - 101 32 BWL BWL LABEL $LR \leftarrow PC + 1$; $PC \leftarrow PC + imm8$ - D - 101 33 RET RET $PC \leftarrow LR$ - D - 011 34 JMP JMP Ra, #imm5 $PC \leftarrow Ra + imm5$ - D - 001 35 PUSH PUSH Ra PUSH LR $R7 \leftarrow R7 - 1$; $Mem[R7] \leftarrow Ra$ R7 $\leftarrow R7 - 1$; $Mem[R7] \leftarrow RL$ - E - - 36 POP POP Ra POP LR Ra $\leftarrow Mem[R7]$; $R7 \leftarrow R7 + 1$ - E - - 37 RETI RETI PC $\leftarrow Mem[R7]$; $R7 \leftarrow R7 + 1$ - F - 000 38 ENAI	26	LLI	LLI Rd, #imm8	Rd ← {Rd[15:8], imm8}	-	В	10101	-
$\begin{array}{cccccccccccccccccccccccccccccccccccc$	27	BR	BR LABEL	PC ← PC + imm8	-	D	-	000
30 BLT BLT LABEL $(n\&^{\sim}v \text{ OR }^{\sim}n\&^{\circ}v)? \text{ PC} \leftarrow \text{ PC} + \text{imm8}$ - D - 100 31 BGE BGE LABEL $(n\&v \text{ OR }^{\sim}n\&^{\circ}v)? \text{ PC} \leftarrow \text{ PC} + \text{imm8}$ - D - 101 32 BWL BWL LABEL LR $\leftarrow \text{ PC} \leftarrow \text{ H; PC} \leftarrow \text{ PC} + \text{ imm8}$ - D - 011 33 RET RET PC $\leftarrow \text{ LR}$ - D - 010 34 JMP JMP Ra, #imm5 PC $\leftarrow \text{ Ra} + \text{ imm5}$ - D - 001 35 PUSH Ra PUSH Ra R7 $\leftarrow \text{ R7} - 1; \text{ Mem}[R7] \leftarrow \text{ Ra}$ - E - - 001 36 POP POP Ra RA $\leftarrow \text{ Mem}[R7]; R7 \leftarrow R7 + 1$ - E - - - 36 POP RETI PC $\leftarrow \text{ Mem}[R7]; R7 \leftarrow R7 + 1$ - E - - - 37 RETI RETI PC $\leftarrow \text{ Mem}[R7]; R7 \leftarrow R7 + 1$ - F - 000 38 ENAI ENAI IntEnFlag $\leftarrow 0$ - F - 010	28	BNE	BNE LABEL	$(z==0)$? PC \leftarrow PC + imm8	-	D	-	110
31 BGE BGE LABEL $(n\&v OR \sim n\&\sim v)? PC \leftarrow PC + imm8$ - D - 101 32 BWL BWL LABEL $LR \leftarrow PC + 1; PC \leftarrow PC + imm8$ - D - 011 33 RET RET PC \leftarrow LR - D - 010 34 JMP JMP Ra, #imm5 PC \leftarrow Ra + imm5 - D - 001 35 PUSH PUSH Ra R7 \leftarrow R7 - 1; Mem[R7] \leftarrow Ra - E - - - 36 POP POP Ra Ra \leftarrow Mem[R7]; R7 \leftarrow R7 + 1 - E - - - 37 RETI RETI PC \leftarrow Mem[R7]; R7 \leftarrow R7 + 1 - F - 000 38 ENAI ENAI IntEnFlag \leftarrow 1 - F - 001 39 DISI DISI IntEnFlag \leftarrow 0 - F - 010 40 STF STF STF R7 \leftarrow R7 - 1; Mem[R7] \leftarrow Flags - F - 011	29	BE	BE LABEL	$(z==1)$? PC \leftarrow PC + imm8	-	D	-	111
31 BGE BGE LABEL $(n\&v OR \sim n\&\sim v)? PC \leftarrow PC + imm8$ - D - 101 32 BWL BWL LABEL $LR \leftarrow PC + 1; PC \leftarrow PC + imm8$ - D - 011 33 RET RET PC \leftarrow LR - D - 010 34 JMP JMP Ra, #imm5 PC \leftarrow Ra + imm5 - D - 001 35 PUSH PUSH Ra R7 \leftarrow R7 - 1; Mem[R7] \leftarrow Ra - E - - - 36 POP POP Ra Ra \leftarrow Mem[R7]; R7 \leftarrow R7 + 1 - E - - - 37 RETI RETI PC \leftarrow Mem[R7]; R7 \leftarrow R7 + 1 - F - 000 38 ENAI ENAI IntEnFlag \leftarrow 1 - F - 001 39 DISI DISI IntEnFlag \leftarrow 0 - F - 010 40 STF STF STF R7 \leftarrow R7 - 1; Mem[R7] \leftarrow Flags - F - 011	30	BLT	BLT LABEL	$(n\&^v OR ^n\&v)? PC \leftarrow PC + imm8$	-	D	_	100
33 RET RET PC \leftarrow LR - D - 010 34 JMP JMP Ra, #imm5 PC \leftarrow Ra + imm5 - D - 001 35 PUSH PUSH Ra PUSH LR R7 \leftarrow R7 - 1; Mem[R7] \leftarrow Ra RT - E -	31	BGE	BGE LABEL	$(n\&v OR \sim n\&\sim v)? PC \leftarrow PC + imm8$	-	D	-	101
$ \begin{array}{c ccccccccccccccccccccccccccccccccccc$	32	BWL	BWL LABEL	$LR \leftarrow PC + 1$; $PC \leftarrow PC + imm8$	-	D	-	011
35 PUSH PUSH Ra PUSH LR PUSH LR R7 \leftarrow R7 - 1; Mem[R7] \leftarrow Ra R7 \leftarrow R7 - 1; Mem[R7] \leftarrow RL R7 \leftarrow R7 - 1; Mem[R7] \leftarrow RL R8 \leftarrow Mem[R7]; R7 \leftarrow R7 + 1 RETI RETI RETI PC \leftarrow Mem[R7]; R7 \leftarrow R7 + 1 PC \leftarrow Mem[R7] PC \leftarrow Mem[R7] PC \leftarrow Mem[R7] PC \leftarrow Mem[R7] PC \leftarrow PC \leftarrow Mem[R7] PC \leftarrow PC	33	RET	RET	PC ← LR	-	D	-	010
35 PUSH PUSH LR $R7 \leftarrow R7 - 1$; $Mem[R7] \leftarrow RL$ - E - - 36 POP POP Ra POP LR $Ra \leftarrow Mem[R7]$; $R7 \leftarrow R7 + 1$ - E - - 37 RETI RETI RETI PC $\leftarrow Mem[R7]$ PC $\leftarrow Mem[R7]$ PC $\leftarrow Mem[R7]$ PF	34	JMP	JMP Ra, #imm5	PC ← Ra + imm5	-	D	-	001
35 PUSH PUSH LR $R7 \leftarrow R7 - 1$; $Mem[R7] \leftarrow RL$ - E - - 36 POP POP Ra POP LR $Ra \leftarrow Mem[R7]$; $R7 \leftarrow R7 + 1$ - E - - 37 RETI RETI RETI PC $\leftarrow Mem[R7]$ PC $\leftarrow Mem[R7]$ PC $\leftarrow Mem[R7]$ PF	2-	511611	PUSH Ra	R7 ← R7 - 1; Mem[R7] ← Ra		_		
36 POP POP LR $RL \leftarrow Mem[R7]; R7 \leftarrow R7 + 1$ - E - - 37 RETI RETI PC $\leftarrow Mem[R7]$ - F - 000 38 ENAI ENAI IntEnFlag $\leftarrow 1$ - F - 001 39 DISI DISI IntEnFlag $\leftarrow 0$ - F - 010 40 STF STF R7 \leftarrow R7 - 1; Mem[R7] \leftarrow Flags - F - 011	35	PUSH	PUSH LR	$R7 \leftarrow R7 - 1$; Mem[R7] $\leftarrow RL$	-	Ł	-	-
37 RETI RETI $PC \leftarrow Mem[R7]; R7 \leftarrow R7 + 1$ - F - 000 38 ENAI ENAI IntEnFlag $\leftarrow 1$ - F - 001 39 DISI DISI IntEnFlag $\leftarrow 0$ - F - 010 40 STF STF R7 \leftarrow R7 - 1; Mem[R7] \leftarrow Flags - F - 011	2.6	202	POP Ra	$Ra \leftarrow Mem[R7]; R7 \leftarrow R7 + 1$		_		
37 RETI RETI $PC \leftarrow Mem[R7]$ - F - 000 38 ENAI ENAI IntEnFlag \leftarrow 1 - F - 001 39 DISI DISI IntEnFlag \leftarrow 0 - F - 010 40 STF STF R7 \leftarrow R7 - 1; Mem[R7] \leftarrow Flags - F - 011	36	РОР	POP LR	$RL \leftarrow Mem[R7]; R7 \leftarrow R7 + 1$	-	Ł	-	-
38 ENAI ENAI IntEnFlag \leftarrow 1 - F - 001 39 DISI DISI IntEnFlag \leftarrow 0 - F - 010 40 STF STF R7 \leftarrow R7 - 1; Mem[R7] \leftarrow Flags - F - 011	37	RETI	RETI	- - -	-	F	-	000
39 DISI DISI IntEnFlag \leftarrow 0 - F - 010 40 STF STF R7 \leftarrow R7 - 1; Mem[R7] \leftarrow Flags - F - 011					-		_	
40 STF STF R7 \leftarrow R7 \leftarrow R7 \leftarrow 1; Mem[R7] \leftarrow Flags \rightarrow F \rightarrow 011				_	-		_	
				<u> </u>	-		_	
41 LUF LUF Flags \leftarrow Mem $ R/ $; $R/\leftarrow R/+1$ $ C,V,N,Z $ F $ C-V $ 100 $ C-V $	41	LDF	LDF	Flags ← Mem[R7]; R7 ← R7 + 1	c,v,n,z	F	_	100

General Instruction Formatting

	Instruction Type	Sub-Type	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
A1	Data Manipulation	Register	Opcode		Rd		Do		Rb X X			Х						
A2		Immediate		Οŀ	,,,,,	JE			ιτα		Ra		imm4/5					
В	Byte Immediate		Opcode				Rd			imm8								
С	Data Transfer		0	LS	0	0	0		Rd			Ra			ir	nm	5	
D1	Control Transfer	Others	1	1	1	1	0	Cond.				imm8						
D2		Jump	1	1		1	U	Cona.		cona.		Ra			ir	nm	5	
E	Stack Operations		0	U	0	0	1	L	Х	Χ		Ra		0	0	0	0	1
F	Interrupts		1	1	0	0	1	IC	ono	J.	1	1	1	Х	Х	Χ	Х	Х

LS: 0 = Load Data, 1 = Store Data

U: 1 = PUSH, 0 = POP

L: 1 = Use Link, 0 = Don't use Link

Example Coding

Data Manipulation

These operations are performed by the Arithmetic Logic Unit and examples are shown below.

1	ADD R5, R3, R4	R5 ← R3 + R4	13	CMPI R3, #9	R3 - 9
2	ADDI R5, R3, #9	R5 ← R3 + 9	14	AND R5, R3, R4	$R5 \leftarrow R3 \text{ AND } R4$
4	ADC R5, R3, R4	R5 ← R3 + R4 + c	15	OR R5, R3, R4	R5 ← R3 OR R4
5	ADCI R5, R3, #9	$R5 \leftarrow R3 + 9 + c$	16	XOR R5, R3, R4	R5 ← R3 XOR R4
6	NEG R5	R5 ← 0 - R5	17	NOT R5, R3	R5 ← NOT R3
7	SUB R5, R3, R4	R5 ← R3 - R4	18	NAND R5, R3, R4	R5 ← R3 NAND R4
8	SUBI R5, R3, #9	R5 ← R3 - 9	19	NOR R5, R3, R4	R5 ← R3 NOR R4
10	SUC R5, R3, R4	R5 ← R3 - R4 - NOT c	20	LSL R5, R3, #3	R5 ← R3 << 3
11	SUCI R5, R3, #9	$R5 \leftarrow R3 - 9 - NOT c$	21	LSR R5, R3, #3	R5 ← R3 >> 3
12	CMP R3, R4	R3 - R4	22	ASR R5, R3, #3	R5 ← R3 >>> 3

The value 'c' corresponds to the carry bit flag in the ALU from the previous calculation.

CMP, CMPI are comparison instructions for performing a subtraction without saving the result. The updated status flags can then be used for a conditional branch.

Byte Immediate

These instructions ADD/SUB an 8-bit immediate value from the given register, replacing the result back in that register. Alternatively, the same formatting is used for loading the upper/lower byte of a register with an 8-bit immediate value.

3	ADDIB R5, #150	R5 ← R5 + 150
9	SUBIB R5, #150	R5 ← R5 - 150
25	LUI R5, #150	R5[15:8] ← 150
26	LLI R5, #150	R5[7:0] ← 150

Data Transfer

When loading data, the value at the memory location held in Ra, adds an offset held in Ro, and replaces the returned value in register Rd. When storing data, the same functionality is used, only with data transferring in opposite direction.

23 LDW R5, [R3, #imm5]	R5 ← Mem[R3 + imm5]
24 STW R5, [R3, #imm5]	Mem[R3+imm5] ← R5

Control Transfer

This set of instructions adjust the value of the program counter by a relative amount determined by the location of the given label. Conditions are as follows:

BR — Branch Always — Unconditionally branch to the stated location

■ BNE — Branch if != — Conditionally branch if zero status flag (z) equals zero

Team R4 – H. Lovett, A. Reddy, A. Robinson, M. Wearn

BE — Branch if = — Conditionally branch if zero status flag (z) equals one
 BLT — Branch if < — Conditionally branch if negative status flag (n) equals one
 BGE — Branch if ≥ — Conditionally branch if negative status flag (n) equals zero

BWL — Branch with link — Unconditionally branch to stated location, saving PC to link register (LR)

• RET — Return — Unconditionally jump to the value stored in the link register (LR)

JMP – Jump – Unconditionally jump to the location held in register Ra plus an 5-bit offset

Stack Operations

These operations are for popping or pushing either a general purpose register or the link register onto the stack, useful for saving register values before or during a subroutine call. PUSH pre-decrements stack pointer (R7) and POP post-increments stack pointer (R7) for a top-down growing stack. The 'U' bit indicates if a PUSH or POP operation is to be performed. If the 'L' bit is set, the link register value will be used instead of the value in register Ra.

Combined Branching & Stack Example

Below is an example showing how PUSH/POP operations and branches can be used to call a subroutine. ".sub" is a label used in assembly language to refer to a different line of code, it is converted to a relative address by an assembler. Here it is calculated as 3 + 4 = 7, if the destination address was before the calling instruction the relative value would be negative.

	PUSH R1	:Save R1
	PUSH R2	:Save R2
	BWL .sub	:Call subroutine
	POP R2	:Restore R2
	POP R1	:Restore R1
	BR .end	:Branch to end of memory
.sub PUSH LR		:Save Link Register
		:Subroutine does something
POP LR		:Restore Link Register
	JMP :Return to where subroutine was ca	
.end	BR .end	