|  |  |  |  |
| --- | --- | --- | --- |
| 23 16 | 15 11 | 10 6 | 5 0 |
| Immediate8 | Ra5 | Rt5 | Opcode6 |

|  |  |  |
| --- | --- | --- |
| 39 11 | 10 6 | 5 0 |
| Immediate29 | Rt5 | Opcode6 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 23 21 | 20 16 | 15 11 | 10 6 | 5 0 |
| Func3 | Rb5 | Ra5 | Rt5 | Opcode6 |

### 

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| 0x | 0 8  BRK | 1 16  ADDI | 2 24  {R2A} | 3 24  {R2L} | 4 24  ADDI | 5 24  CMPI | 6 24  MULI | 7 24  DIVI |
|  | 8 24  ANDI | 9 24  ORI | 10 24  EORI | 11 24  {R2S} | 12 24  SUBFI | 13 24  SLTI | 14 24  MULUI | 15 24  DIVUI |
| 1x | 16 8  RET  RETA  ERET | 17 16  RETD | 18 24  CALL  BRA  JMP | 19 32  LCALL  LBRA  LJMP | 20 40  XLCALL  XLBRA  XLJMP | 21 | 22 | 23 |
|  | 24 24  BEQ | 25 24  BNE | 26 24  BLT | 27 24  BLE | 28 24  BLTU | 29 24  BLEU | 30 32  LBcc | 31 40  XLBcc |
| 2x | 32 24  LDB | 33 24  LDBU | 34 24  LDW | 35 24  LDWU | 36 24  LDT | 37 24 | 38 24  LDA | 39 24  CACHE |
|  | 40 24  LDBX | 41 24  LDBUX | 42 24  LDWX | 43 24  LDWUX | 44 24  LDTX | 45 24 | 46 24  LDAX | 47 24  CACHEX |
| 3x | 48 24  STB | 49 24  STW | 50 24  STT | 51 | 52 24  STBX | 53 24  STWX | 54 24  STTX | 55 |
|  | 56 16  PFX8 | 57 24  PFX16 | 58 32  PFX24 | 59 40  PFX32 | 60 72  PFX64 | 61 | 62 | 63 8  NOP |

### {R2A} Arithmetic Operations

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| 2 | 0  ADD | 1  CMP | 2  MUL | 3  DIV | 4  SUB | 5 | 6  MULU | 7  DIVU |

### {R2L} Logic Operations

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| 3 | 0  AND | 1  OR | 2  EOR | 3  ANDN | 4  NAND | 5  NOR | 6  ENOR | 7  ORN |

### {R2S} Logic Operations

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| 11 | 0  SEQ | 1  SNE | 2  SLT | 3  SLE | 4  SLTU | 5  SLEU | 6 | 7 |

### ADD – Addition

**Description:**

Add two source operands and place the result in the target register. Operands are treated as signed twos complement values.

**Instruction Format:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 23 21 | 20 16 | 15 11 | 10 6 | 5 0 |
| 03 | Rb5 | Ra5 | Rt5 | 26 |

### ADDI – Add Immediate

**Description:**

Add two source operands and place the result in the target register. The first operand is in register Ra the second is an immediate specified in the instruction. Operands are treated as signed twos complement values.

**Instruction Format:**

|  |  |  |  |
| --- | --- | --- | --- |
| 23 16 | 15 11 | 10 6 | 5 0 |
| Immediate8 | Ra5 | Rt5 | 46 |

|  |  |  |
| --- | --- | --- |
| 15 11 | 10 6 | 5 0 |
| Imm5 | Rt5 | 16 |

### AND – Bitwise ‘And’

**Description:**

Bitwise ‘and’ two source operands and place the result in the target register.

**Instruction Format:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 23 21 | 20 16 | 15 11 | 10 6 | 5 0 |
| 03 | Rb5 | Ra5 | Rt5 | 36 |

### BEQ – Branch if Equal

**Description:**

If Ra and Rb are equal then branch to the target address. The target address is formed as the sum of the program counter and an immediate constant in the instruction.

**Instruction Format:**

|  |  |  |  |
| --- | --- | --- | --- |
| 23 16 | 15 11 | 10 6 | 5 0 |
| Disp8 | Rb5 | Ra5 | 246 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 31 19 | 18 16 | 15 11 | 10 6 | 5 0 |
| Displacement13 | 03 | Rb5 | Ra5 | 306 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 39 19 | 18 16 | 15 11 | 10 6 | 5 0 |
| Displacement21 | 03 | Rb5 | Ra5 | 316 |

### BRA – Branch

**Description:**

Branch to a target address. The target address is the sum of the program counter and a constant in the instruction.

**Instruction Format:**

|  |  |  |
| --- | --- | --- |
| 23 8 | 7 6 | 5 0 |
| Displacement16 | 22 | 186 |

|  |  |  |
| --- | --- | --- |
| 31 8 | 7 6 | 5 0 |
| Displacement24 | 22 | 196 |

|  |  |  |
| --- | --- | --- |
| 39 8 | 7 6 | 5 0 |
| Displacement32 | 22 | 206 |

### CALL – Call Subroutine

**Description:**

Call a subroutine by storing the program counter in a link register then branching to a target address. The target address is the sum of the program counter and a constant in the instruction.

**Instruction Format:**

|  |  |  |
| --- | --- | --- |
| 23 8 | 7 6 | 5 0 |
| Displacement16 | Lr2 | 186 |

|  |  |  |
| --- | --- | --- |
| 31 8 | 7 6 | 5 0 |
| Displacement24 | Lr2 | 196 |

|  |  |  |
| --- | --- | --- |
| 39 8 | 7 6 | 5 0 |
| Displacement32 | Lr2 | 206 |

### JMP – Jump to Address

**Description:**

Jump to a target address. The target address is a constant in the instruction.

**Instruction Format:**

|  |  |  |
| --- | --- | --- |
| 23 8 | 7 6 | 5 0 |
| Displacement16 | 32 | 186 |

|  |  |  |
| --- | --- | --- |
| 31 8 | 7 6 | 5 0 |
| Displacement24 | 32 | 196 |

|  |  |  |
| --- | --- | --- |
| 39 8 | 7 6 | 5 0 |
| Displacement32 | 32 | 206 |

### OR – Bitwise ‘Or’

**Description:**

Bitwise ‘or’ two source operands and place the result in the target register.

**Instruction Format:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 23 21 | 20 16 | 15 11 | 10 6 | 5 0 |
| 13 | Rb5 | Ra5 | Rt5 | 36 |

### LDB – Load Byte

**Description:**

Load a byte from the memory addressed as the sum of register Ra and a displacement constant encoded in the instruction. The loaded byte is sign extended to the width of the machine.

**Instruction Format:**

|  |  |  |  |
| --- | --- | --- | --- |
| 23 16 | 15 11 | 10 6 | 5 0 |
| Immediate8 | Ra5 | Rt5 | 326 |

### LDBX – Load Byte, Indexed

**Description:**

Load a byte from the memory addressed as the sum of register Ra and scaled register Rb. The loaded byte is sign extended to the width of the machine.

**Instruction Format:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 23 | 22 21 | 20 16 | 15 11 | 10 6 | 5 0 |
| ~ | Sc2 | Rb5 | Ra5 | Rt5 | 406 |

### PFX8 – Eight-bit Postfix Immediate

**Description:**

Replace either the A operand or the B operand with an eight-bit immediate constant.

**Instruction Format:**

|  |  |  |
| --- | --- | --- |
| 16 8 | 7 6 | 5 0 |
| Immed8 | Op2 | 566 |

|  |  |
| --- | --- |
| Op2 | Operand Replaced |
| 0 | Replace operand Ra |
| 1 | Replace operand Rb |
| 2 | Replace operand Rc |
| 3 | reserved |

### PFX16 – Sixteen-bit Postfix Immediate

**Description:**

Replace either the A operand or the B operand with a sixteen-bit immediate constant.

**Instruction Format:**

|  |  |  |
| --- | --- | --- |
| 23 8 | 7 6 | 5 0 |
| Immed16 | Op2 | 576 |

### PFX24 – 24-bit Postfix Immediate

**Description:**

Replace either the A operand or the B operand with a 24-bit immediate constant.

**Instruction Format:**

|  |  |  |
| --- | --- | --- |
| 31 8 | 7 6 | 5 0 |
| Immed24 | Op2 | 586 |

### PFX32 – 32-bit Postfix Immediate

**Description:**

Replace either the A operand or the B operand with a 32-bit immediate constant.

**Instruction Format:**

|  |  |  |
| --- | --- | --- |
| 39 8 | 76 | 5 0 |
| Immed32 | Op2 | 596 |

### RET - Return

**Description:**

Load the PC from LR0.

**Instruction Format:**

|  |  |  |
| --- | --- | --- |
| 7 6 | 5 0 | Mnemonic |
| 0 | 166 | RET |

### RETA – Return using Alternate Link Register

**Description:**

Load the PC from LR1.

**Instruction Format:**

|  |  |  |
| --- | --- | --- |
| 7 6 | 5 0 | Mnemonic |
| 1 | 166 | RETA |

### RETD – Return and Deallocate

**Description:**

Load the PC from LR0 and add a constant to the SP.

**Instruction Format:**

|  |  |
| --- | --- |
| 15 6 | 5 0 |
| Imm10 | 176 |

### ERET – Return from Exception

**Description:**

Load the PC from the internal exception stack or perform a two-up level return using the internal exception stack.

**Instruction Format:**

|  |  |  |
| --- | --- | --- |
| 7 6 | 5 0 | Mnemonic |
| 2 | 166 | ERET |
| 3 | 166 | ERET2UP |