Instructions:

| | ICCIONS: | | • |
|--|---|--|--|
| <u>Inst.</u> | | Usage | Meaning - |
| add | add | add rd, rs1, rs2 | rd = rs1 + rs2 |
| sub | subtract | sub rd, rs1, rs2 | rd = rs1 - rs2 |
| and | and | and rd, rs1, rs2 | rd = rs1 & rs2 |
| | | | rd = rs1 rs2 |
| or | or | | |
| xor | xor | xor rd, rs1, rs2 | $rd = rs1 ^ rs2$ |
| slt | set less than | slt rd, rs1, rs2 | rd = rs1 < rs2 |
| sltu | set less than (U) | sltu rd, rs1, rs2 | rd = rs1 < rs2 |
| sll | shift left logic. | sll rd, rs1, rs2 | rd = rs1 << rs2[4:0] |
| srl | shift right logic. | srl rd, rs1, rs2 | rd = rs1 >> rs2[4:0] |
| | | | |
| <u>sra</u> | shift right arith. | sra rd, rs1, rs2 | <u>rd = rs1 >> rs2[4:0]</u> - |
| addi | add immediate | addi rd, rs, const | rd = rs + const |
| andi | and immediate | andi rd, rs, const | rd = rs & const |
| ori | or immediate | ori rd, rs, const | rd = rs const |
| xori | xor immediate | xori rd, rs, const | rd = rs ^ const |
| slti | set less than imm. | slti rd, rs, const | rd = rs < const |
| | set less than imm. (U) | sltiu rd, rs, const | rd = rs < const |
| | | | |
| slli | shift left logic. imm. | slli rd, rs, const | rd = rs << const[4:0] |
| srli | shift right logic. imm. | srli rd, rs, const | rd = rs >> const[4:0] |
| srai | shift right arith. imm. | srai rd, rs, const | rd = rs >> const[4:0] - |
| beg | branch if == | beg rs1, rs2, offset | if (rs1 == rs2) pc += offset |
| bne | branch if != | bne rs1, rs2, offset | if (rs1 != rs2) pc += offset |
| | | · | |
| | branch if < (U) | bltu rs1, rs2, offset | if (rs1 < rs2) pc += offset |
| blt | branch if < | blt rs1, rs2, offset | if (rs1 < rs2) pc += offset |
| bgeu | branch if $>=$ (U) | bgeu rs1, rs2, offset | if (rs1 >= rs2) pc += offset |
| bge | branch if >= | bge rs1, rs2, offset | if $(rs1 >= rs2)$ pc += offset - |
| lui | load upper imm. | lui rd, const | rd = const << 12 |
| | | auipc rd, const | rd = pc + const << 12 |
| | add upper imm. to pc | | 10 = pc + const << 12 - |
| jal | jump and link | jal rd, offset | rd = pc + 4; pc += offset |
| jalr | jump and link reg. | jalr rd, offset(rs) | rd = pc + 4; pc += rs + offset |
| lw | load word | <pre>lw rd, offset(rs)</pre> | rd = mem[rs + offset][31:0] |
| lh | load half | lh rd, offset(rs) | rd = mem[rs + offset] [15:0] |
| | | , (, | |
| lhu | load half (II) | lhu rd offcat (ra) | |
| lhu lh | load half (U) | lhu rd, offset(rs) | rd = mem[rs + offset][15:0] |
| lb | load byte | lb rd, offset(rs) | rd = mem[rs + offset][15:0] rd = mem[rs + offset][7:0] |
| lb lbu | load byte load byte (U) | <pre>lb rd, offset(rs) lbu rd, offset(rs)</pre> | <pre>rd = mem[rs + offset] [15:0] rd = mem[rs + offset] [7:0] rd = mem[rs + offset] [7:0] -</pre> |
| lb | load byte | lb rd, offset(rs) | <pre>rd = mem[rs + offset] [15:0] rd = mem[rs + offset] [7:0] rd = mem[rs + offset] [7:0] mem[rs1 + offset] [31:0] = rs2</pre> |
| lb lbu | load byte load byte (U) | <pre>lb rd, offset(rs) lbu rd, offset(rs)</pre> | <pre>rd = mem[rs + offset] [15:0] rd = mem[rs + offset] [7:0] rd = mem[rs + offset] [7:0] mem[rs1 + offset] [31:0] = rs2</pre> |
| lb lbu sw sh | load byte (U) store word store half | <pre>lb rd, offset(rs) lbu rd, offset(rs) sw rs2, offset(rs1) sh rs2, offset(rs1)</pre> | <pre>rd = mem[rs + offset] [15:0] rd = mem[rs + offset] [7:0] rd = mem[rs + offset] [7:0] - mem[rs1 + offset] [31:0] = rs2 mem[rs1 + offset] [15:0] = rs2[15:0]</pre> |
| lb <u>lbu</u> sw | load byte (U) store word | <pre>lb rd, offset(rs) lbu rd, offset(rs) sw rs2, offset(rs1)</pre> | <pre>rd = mem[rs + offset] [15:0] rd = mem[rs + offset] [7:0] rd = mem[rs + offset] [7:0] mem[rs1 + offset] [31:0] = rs2</pre> |
| lb lbu sw sh sb | load byte load byte (U) store word store half store byte | <pre>lb rd, offset(rs) lbu rd, offset(rs) sw rs2, offset(rs1) sh rs2, offset(rs1)</pre> | <pre>rd = mem[rs + offset] [15:0] rd = mem[rs + offset] [7:0] rd = mem[rs + offset] [7:0] - mem[rs1 + offset] [31:0] = rs2 mem[rs1 + offset] [15:0] = rs2[15:0]</pre> |
| lb lbu sw sh sb | load byte load byte (U) store word store half store byte Instructions: | lb rd, offset(rs) lbu rd, offset(rs) sw rs2, offset(rs1) sh rs2, offset(rs1) sb rs2, offset(rs1) | <pre>rd = mem[rs + offset] [15:0] rd = mem[rs + offset] [7:0] rd = mem[rs + offset] [7:0] - mem[rs1 + offset] [31:0] = rs2 mem[rs1 + offset] [15:0] = rs2[15:0] mem[rs1 + offset] [7:0] = rs2[7:0] -</pre> |
| lb lbu sw sh sb | load byte load byte (U) store word store half store byte Instructions: | <pre>lb rd, offset(rs) lbu rd, offset(rs) sw rs2, offset(rs1) sh rs2, offset(rs1)</pre> | <pre>rd = mem[rs + offset] [15:0] rd = mem[rs + offset] [7:0] rd = mem[rs + offset] [7:0] - mem[rs1 + offset] [31:0] = rs2 mem[rs1 + offset] [15:0] = rs2[15:0]</pre> |
| lb lbu sw sh sb Pseude Inst. | load byte load byte (U) store word store half store byte Instructions: Name | lb rd, offset(rs) lbu rd, offset(rs) sw rs2, offset(rs1) sh rs2, offset(rs1) sb rs2, offset(rs1) | <pre>rd = mem[rs + offset] [15:0] rd = mem[rs + offset] [7:0] rd = mem[rs + offset] [7:0] - mem[rs1 + offset] [31:0] = rs2 mem[rs1 + offset] [15:0] = rs2[15:0] mem[rs1 + offset] [7:0] = rs2[7:0] -</pre> <pre>Meaning or Implementation -</pre> |
| lb lbu sw sh sb Pseudo Inst. nop | load byte load byte (U) store word store half store byte Instructions: Name no operation | lb rd, offset(rs) lbu rd, offset(rs) sw rs2, offset(rs1) sh rs2, offset(rs1) sb rs2, offset(rs1) Usage nop | <pre>rd = mem[rs + offset] [15:0] rd = mem[rs + offset] [7:0] rd = mem[rs + offset] [7:0] - mem[rs1 + offset] [31:0] = rs2 mem[rs1 + offset] [15:0] = rs2[15:0] mem[rs1 + offset] [7:0] = rs2[7:0] - Meaning or Implementation - addi x0, x0, 0 -</pre> |
| lb lbu sw sh sb Pseudo Inst. nop | load byte load byte (U) store word store half store byte -Instructions: Name no operation load immediate | lb rd, offset(rs) lbu rd, offset(rs) sw rs2, offset(rs1) sh rs2, offset(rs1) sb rs2, offset(rs1) Usage nop li rd, constant | rd = mem[rs + offset] [15:0] rd = mem[rs + offset] [7:0] rd = mem[rs + offset] [7:0] - mem[rs1 + offset] [31:0] = rs2 mem[rs1 + offset] [15:0] = rs2[15:0] mem[rs1 + offset] [7:0] = rs2[7:0] - Meaning or Implementation addi x0, x0, 0 rd = constant |
| lb lbu sw sh sb Pseudd Inst. nop li | load byte load byte (U) store word store half store byte D-Instructions: Name no operation load immediate move | lb rd, offset(rs) lbu rd, offset(rs) sw rs2, offset(rs1) sh rs2, offset(rs1) sb rs2, offset(rs1) Usage nop li rd, constant mv rd, rs | rd = mem[rs + offset] [15:0] rd = mem[rs + offset] [7:0] rd = mem[rs + offset] [7:0] - mem[rs1 + offset] [31:0] = rs2 mem[rs1 + offset] [15:0] = rs2[15:0] mem[rs1 + offset] [7:0] = rs2[7:0] - - Meaning or Implementation - addi x0, x0, 0 - rd = constant - addi rd, rs, 0 |
| lb lbu sw sh sb Pseudo Inst. nop li mv not | load byte load byte (U) store word store half store byte D-Instructions: Name no operation load immediate move not | lb rd, offset(rs) lbu rd, offset(rs) sw rs2, offset(rs1) sh rs2, offset(rs1) sb rs2, offset(rs1) Usage nop li rd, constant mv rd, rs not rd, rs | rd = mem[rs + offset] [15:0] rd = mem[rs + offset] [7:0] rd = mem[rs + offset] [7:0] mem[rs1 + offset] [31:0] = rs2 mem[rs1 + offset] [15:0] = rs2[15:0] mem[rs1 + offset] [7:0] = rs2[7:0] - Meaning or Implementation addi x0, x0, 0 rd = constant addi rd, rs, 0 xori rd, rs, -1 |
| lb lbu sw sh sb Pseudd Inst. nop li | load byte load byte (U) store word store half store byte D-Instructions: Name no operation load immediate move not negate | lb rd, offset(rs) lbu rd, offset(rs) sw rs2, offset(rs1) sh rs2, offset(rs1) sb rs2, offset(rs1) Usage nop li rd, constant mv rd, rs not rd, rs neg rd, rs | rd = mem[rs + offset] [15:0] rd = mem[rs + offset] [7:0] rd = mem[rs + offset] [7:0] |
| lb lbu sw sh sb Pseudo Inst. nop li mv not | load byte load byte (U) store word store half store byte D-Instructions: Name no operation load immediate move not negate set equal to zero | lb rd, offset(rs) lbu rd, offset(rs) sw rs2, offset(rs1) sh rs2, offset(rs1) sb rs2, offset(rs1) Usage nop li rd, constant mv rd, rs not rd, rs | rd = mem[rs + offset] [15:0] rd = mem[rs + offset] [7:0] rd = mem[rs + offset] [7:0] mem[rs1 + offset] [31:0] = rs2 mem[rs1 + offset] [15:0] = rs2[15:0] mem[rs1 + offset] [7:0] = rs2[7:0] - Meaning or Implementation addi x0, x0, 0 rd = constant addi rd, rs, 0 xori rd, rs, -1 sub rd, x0, rs sltui rd, rs, 1 |
| lb lbu sw sh sb Pseudo Inst. nop li mv not neg | load byte load byte (U) store word store half store byte D-Instructions: Name no operation load immediate move not negate set equal to zero | lb rd, offset(rs) lbu rd, offset(rs) sw rs2, offset(rs1) sh rs2, offset(rs1) sb rs2, offset(rs1) Usage nop li rd, constant mv rd, rs not rd, rs neg rd, rs seqz rd, rs | rd = mem[rs + offset] [15:0] rd = mem[rs + offset] [7:0] rd = mem[rs + offset] [7:0] mem[rs1 + offset] [31:0] = rs2 mem[rs1 + offset] [15:0] = rs2[15:0] mem[rs1 + offset] [7:0] = rs2[7:0] - Meaning or Implementation addi x0, x0, 0 rd = constant addi rd, rs, 0 xori rd, rs, -1 sub rd, x0, rs sltui rd, rs, 1 |
| lb lbu sw sh sb Pseudo Inst. nop li mv not neg seqz snez | load byte load byte (U) store word store half store byte D-Instructions: Name no operation load immediate move not negate set equal to zero set not equal to zero | lb rd, offset(rs) lbu rd, offset(rs) sw rs2, offset(rs1) sh rs2, offset(rs1) sb rs2, offset(rs1) Usage nop li rd, constant mv rd, rs not rd, rs neg rd, rs seqz rd, rs snez rd, rs | rd = mem[rs + offset] [15:0] rd = mem[rs + offset] [7:0] rd = mem[rs + offset] [7:0] mem[rs1 + offset] [31:0] = rs2 mem[rs1 + offset] [15:0] = rs2[15:0] mem[rs1 + offset] [7:0] = rs2[7:0] - Meaning or Implementation addi x0, x0, 0 rd = constant addi rd, rs, 0 xori rd, rs, -1 sub rd, x0, rs sltui rd, rs, 1 sltu rd, x0, rs |
| lb lbu sw sh sb Pseudo Inst. nop li mv not neg seqz snez sltz | load byte load byte (U) store word store half store byte D-Instructions: Name no operation load immediate move not negate set equal to zero set not equal to zero set less than zero | lb rd, offset(rs) lbu rd, offset(rs) sw rs2, offset(rs1) sh rs2, offset(rs1) sb rs2, offset(rs1) Usage nop li rd, constant mv rd, rs not rd, rs neg rd, rs seqz rd, rs snez rd, rs sltz rd, rs | rd = mem[rs + offset] [15:0] rd = mem[rs + offset] [7:0] rd = mem[rs + offset] [7:0] - mem[rs1 + offset] [31:0] = rs2 mem[rs1 + offset] [15:0] = rs2[15:0] mem[rs1 + offset] [7:0] = rs2[7:0] - - Meaning or Implementation - addi x0, x0, 0 - rd = constant - addi rd, rs, 0 xori rd, rs, -1 sub rd, x0, rs sltui rd, rs, 1 sltu rd, x0, rs slt rd, rs, x0 |
| lb lbu sw sh sb Pseude Inst. nop li mv not neg seqz snez sltz sqtz | load byte load byte (U) store word store half store byte D-Instructions: Name no operation load immediate move not negate set equal to zero set not equal to zero set less than zero set greater than zero | lb rd, offset(rs) lbu rd, offset(rs) sw rs2, offset(rs1) sh rs2, offset(rs1) sb rs2, offset(rs1) Usage nop li rd, constant mv rd, rs not rd, rs neg rd, rs seqz rd, rs snez rd, rs sltz rd, rs sqtz rd, rs | rd = mem[rs + offset] [15:0] rd = mem[rs + offset] [7:0] rd = mem[rs + offset] [7:0] - mem[rs1 + offset] [31:0] = rs2 mem[rs1 + offset] [15:0] = rs2[15:0] mem[rs1 + offset] [7:0] = rs2[7:0] |
| Pseudo Inst. nop li mv not neg seqz snez sltz sqtz beqz | load byte load byte (U) store word store half store byte D-Instructions: Name no operation load immediate move not negate set equal to zero set not equal to zero set less than zero set greater than zero branch if == 0 | lb rd, offset(rs) lbu rd, offset(rs) sw rs2, offset(rs1) sh rs2, offset(rs1) sb rs2, offset(rs1) Usage nop li rd, constant mv rd, rs not rd, rs neg rd, rs seqz rd, rs snez rd, rs sltz rd, rs setz rd, rs | rd = mem[rs + offset] [15:0] rd = mem[rs + offset] [7:0] rd = mem[rs + offset] [7:0] - mem[rs1 + offset] [31:0] = rs2 mem[rs1 + offset] [15:0] = rs2[15:0] mem[rs1 + offset] [7:0] = rs2[7:0] - |
| Pseudo Inst. nop li mv not neg seqz snez sltz sqtz beqz bnez | load byte load byte (U) store word store half store byte D-Instructions: Name no operation load immediate move not negate set equal to zero set not equal to zero set less than zero set greater than zero branch if == 0 branch if != 0 | lb rd, offset(rs) lbu rd, offset(rs) sw rs2, offset(rs1) sh rs2, offset(rs1) sb rs2, offset(rs1) Usage nop li rd, constant mv rd, rs not rd, rs neg rd, rs seqz rd, rs snez rd, rs sltz rd, rs sqtz rd, rs beqz rs, offset bnez rs, offset | rd = mem[rs + offset] [15:0] rd = mem[rs + offset] [7:0] rd = mem[rs + offset] [7:0] |
| Pseudo Inst. nop li mv not neg seqz snez sltz sqtz beqz | load byte load byte (U) store word store half store byte D-Instructions: Name no operation load immediate move not negate set equal to zero set not equal to zero set less than zero set greater than zero branch if == 0 branch if != 0 | lb rd, offset(rs) lbu rd, offset(rs) sw rs2, offset(rs1) sh rs2, offset(rs1) sb rs2, offset(rs1) Usage nop li rd, constant mv rd, rs not rd, rs neg rd, rs seqz rd, rs snez rd, rs sltz rd, rs setz rd, rs | rd = mem[rs + offset] [15:0] rd = mem[rs + offset] [7:0] rd = mem[rs + offset] [7:0] - mem[rs1 + offset] [31:0] = rs2 mem[rs1 + offset] [15:0] = rs2[15:0] mem[rs1 + offset] [7:0] = rs2[7:0] - |
| Pseudo Inst. nop li mv not neg seqz snez sltz sqtz beqz bnez blez | load byte load byte (U) store word store half store byte D-Instructions: Name no operation load immediate move not negate set equal to zero set not equal to zero set less than zero set greater than zero branch if == 0 branch if != 0 branch if <= 0 | lb rd, offset(rs) lbu rd, offset(rs) sw rs2, offset(rs1) sh rs2, offset(rs1) sb rs2, offset(rs1) Usage nop li rd, constant mv rd, rs not rd, rs neg rd, rs seqz rd, rs snez rd, rs sltz rd, rs setz rd, rs beqz rs, offset bnez rs, offset blez rs, offset | rd = mem[rs + offset] [15:0] rd = mem[rs + offset] [7:0] rd = mem[rs + offset] [7:0] - mem[rs1 + offset] [31:0] = rs2 mem[rs1 + offset] [15:0] = rs2[15:0] mem[rs1 + offset] [7:0] = rs2[7:0] - |
| Pseudo Inst. nop li mv not neg seqz snez sltz sgtz beqz blez bgez | load byte load byte (U) store word store half store byte D-Instructions: Name no operation load immediate move not negate set equal to zero set not equal to zero set less than zero set greater than zero branch if == 0 branch if != 0 branch if <= 0 branch if >= 0 | lb rd, offset(rs) lbu rd, offset(rs) sw rs2, offset(rs1) sh rs2, offset(rs1) sb rs2, offset(rs1) Usage nop li rd, constant mv rd, rs not rd, rs neg rd, rs seqz rd, rs snez rd, rs sltz rd, rs beqz rs, offset bnez rs, offset blez rs, offset bgez rs, offset bgez rs, offset | rd = mem[rs + offset] [15:0] rd = mem[rs + offset] [7:0] rd = mem[rs + offset] [7:0] - mem[rs1 + offset] [31:0] = rs2 mem[rs1 + offset] [15:0] = rs2[15:0] mem[rs1 + offset] [7:0] = rs2[7:0] - |
| Pseudo Inst. nop li mv not neg seqz sltz sgtz beqz bnez blez bgez bltz | load byte load byte (U) store word store half store byte D-Instructions: Name no operation load immediate move not negate set equal to zero set not equal to zero set less than zero set greater than zero branch if == 0 branch if <= 0 branch if <= 0 branch if <= 0 branch if <= 0 | lb rd, offset(rs) lbu rd, offset(rs) sw rs2, offset(rs1) sh rs2, offset(rs1) sb rs2, offset(rs1) Usage nop li rd, constant mv rd, rs not rd, rs neg rd, rs seqz rd, rs snez rd, rs sltz rd, rs sgtz rd, rs beqz rs, offset bnez rs, offset blez rs, offset blez rs, offset bltz rs, offset bltz rs, offset bltz rs, offset | rd = mem[rs + offset] [15:0] rd = mem[rs + offset] [7:0] rd = mem[rs + offset] [7:0] - mem[rs1 + offset] [31:0] = rs2 mem[rs1 + offset] [15:0] = rs2[15:0] mem[rs1 + offset] [7:0] = rs2[7:0] - |
| Pseudo Inst. nop li mv not neg seqz sltz sqtz beqz bnez blez bgez bltz bgtz | load byte load byte (U) store word store half store byte D-Instructions: Name no operation load immediate move not negate set equal to zero set not equal to zero set less than zero set greater than zero branch if == 0 branch if <= 0 branch if <= 0 branch if <= 0 branch if < 0 branch if < 0 branch if > 0 | lb rd, offset(rs) lbu rd, offset(rs) sw rs2, offset(rs1) sh rs2, offset(rs1) sb rs2, offset(rs1) Usage nop li rd, constant mv rd, rs not rd, rs neg rd, rs seqz rd, rs snez rd, rs sltz rd, rs sgtz rd, rs beqz rs, offset bnez rs, offset blez rs, offset bltz rs, offset bltz rs, offset bdtz rs, offset bdtz rs, offset bdtz rs, offset bdtz rs, offset | <pre>rd = mem[rs + offset] [15:0] rd = mem[rs + offset] [7:0] rd = mem[rs + offset] [7:0]</pre> |
| Pseudo Inst. nop li mv not neg seqz sltz sqtz bnez blez bgez bltz bgtz | load byte load byte (U) store word store half store byte D-Instructions: Name no operation load immediate move not negate set equal to zero set not equal to zero set less than zero branch if == 0 branch if != 0 branch if <= 0 branch if <= 0 branch if <= 0 branch if <= 0 branch if < 0 branch if > 0 | lb rd, offset(rs) lbu rd, offset(rs) sw rs2, offset(rs1) sh rs2, offset(rs1) sb rs2, offset(rs1) Usage nop li rd, constant mv rd, rs not rd, rs neg rd, rs seqz rd, rs shz rd, rs shz rd, rs sptz rd, rs beqz rs, offset bnez rs, offset blez rs, offset bdtz rs, offset bgtz rs, offset bgt rs1, rs2, offset | <pre>rd = mem[rs + offset] [15:0] rd = mem[rs + offset] [7:0] rd = mem[rs + offset] [7:0]</pre> |
| Pseudo Inst. nop li mv not neg seqz snez sltz sqtz bnez blez bgez bltz bgtz | load byte load byte (U) store word store half store byte D-Instructions: Name no operation load immediate move not negate set equal to zero set not equal to zero set less than zero branch if == 0 branch if != 0 branch if != 0 branch if <= 0 branch if <= 0 branch if <= 0 branch if < 0 branch if > 0 | lb rd, offset(rs) lbu rd, offset(rs) sw rs2, offset(rs1) sh rs2, offset(rs1) sb rs2, offset(rs1) Usage nop li rd, constant mv rd, rs not rd, rs neg rd, rs seqz rd, rs snez rd, rs sltz rd, rs sptz rd, rs beqz rs, offset bnez rs, offset blez rs, offset bltz rs, offset bgtz rs, offset bgt rs1, rs2, offset bgtu rs1, rs2, offset | <pre>rd = mem[rs + offset] [15:0] rd = mem[rs + offset] [7:0] rd = mem[rs + offset] [7:0]</pre> |
| Pseudo Inst. nop li mv not neg seqz sltz sqtz bnez blez bgez bltz bgtz | load byte load byte (U) store word store half store byte D-Instructions: Name no operation load immediate move not negate set equal to zero set not equal to zero set less than zero branch if == 0 branch if == 0 branch if <= 0 branch if <= 0 branch if <= 0 branch if <= 0 branch if >= 0 branch if > 0 | lb rd, offset(rs) lbu rd, offset(rs) sw rs2, offset(rs1) sh rs2, offset(rs1) sb rs2, offset(rs1) Usage nop li rd, constant mv rd, rs not rd, rs neg rd, rs seqz rd, rs setz rd, rs stz rd, rs stz rd, rs sptz rd, rs beqz rs, offset bnez rs, offset blez rs, offset blez rs, offset bgtz rs, offset bgtz rs, offset bgt rs1, rs2, offset ble rs1, rs2, offset ble rs1, rs2, offset | <pre>rd = mem[rs + offset] [15:0] rd = mem[rs + offset] [7:0] rd = mem[rs + offset] [7:0]</pre> |
| Pseudo Inst. nop li mv not neg seqz snez sltz sqtz bnez blez bgez bltz bgtz | load byte load byte (U) store word store half store byte D-Instructions: Name no operation load immediate move not negate set equal to zero set not equal to zero set less than zero branch if == 0 branch if == 0 branch if <= 0 branch if <= 0 branch if <= 0 branch if <= 0 branch if >= 0 branch if > 0 | lb rd, offset(rs) lbu rd, offset(rs) sw rs2, offset(rs1) sh rs2, offset(rs1) sb rs2, offset(rs1) Usage nop li rd, constant mv rd, rs not rd, rs neg rd, rs seqz rd, rs setz rd, rs stz rd, rs stz rd, rs sptz rd, rs beqz rs, offset bnez rs, offset blez rs, offset blez rs, offset bgtz rs, offset bgtz rs, offset bgt rs1, rs2, offset ble rs1, rs2, offset ble rs1, rs2, offset | <pre>rd = mem[rs + offset] [15:0] rd = mem[rs + offset] [7:0] rd = mem[rs + offset] [7:0] mem[rs1 + offset] [31:0] = rs2 mem[rs1 + offset] [15:0] = rs2[15:0] mem[rs1 + offset] [7:0] = rs2[7:0] - Meaning or Implementation addi x0, x0, 0 rd = constant addi rd, rs, 0 xori rd, rs, -1 sub rd, x0, rs sltui rd, rs, 1 sltu rd, rs, 1 sltu rd, x0, rs slt rd, rs, x0 slt rd, x0, rs beq rs, x0, offset bne rs, x0, offset bge x0, rs, offset blt rs, x0, offset blt rs, x0, offset blt rs2, rs1, offset bge rs2, rs1, offset bge rs2, rs1, offset bge rs2, rs1, offset</pre> |
| Pseudo Inst. nop li mv not neg seqz snez sltz sqtz bnez blez bgez bltz bgtz bgtu ble bleu | load byte load byte (U) store word store half store byte D-Instructions: Name no operation load immediate move not negate set equal to zero set not equal to zero set less than zero set qreater than zero branch if == 0 branch if <= 0 branch if <= 0 branch if <= 0 branch if > 0 branch if <= 0 branch if > 0 branch if > 0 branch if > 0 branch if > 0 branch if <= 0 branch if > 0 | lb rd, offset(rs) lbu rd, offset(rs) sw rs2, offset(rs1) sh rs2, offset(rs1) sb rs2, offset(rs1) Usage nop li rd, constant mv rd, rs not rd, rs neg rd, rs segz rd, rs sltz rd, rs sytz rd, rs beqz rs, offset bnez rs, offset bnez rs, offset blez rs, offset bgz rs, offset bgt rs1, rs2, offset bgt rs1, rs2, offset ble rs1, rs2, offset | rd = mem[rs + offset] [15:0] rd = mem[rs + offset] [7:0] rd = mem[rs + offset] [7:0] mem[rs1 + offset] [31:0] = rs2 mem[rs1 + offset] [15:0] = rs2[15:0] mem[rs1 + offset] [7:0] = rs2[7:0] - Meaning or Implementation addi x0, x0, 0 rd = constant addi rd, rs, 0 xori rd, rs, -1 sub rd, x0, rs sltui rd, rs, 1 sltu rd, x0, rs sltui rd, rs, x0 slt rd, x0, rs slt rd, rs, x0 slt rd, x0, rs beq rs, x0, offset bne rs, x0, offset bne rs, x0, offset blt rs, x0, offset blt x0, rs, offset blt x0, rs, offset blt rs2, rs1, offset bge rs2, rs1, offset bgeu rs2, rs1, offset bgeu rs2, rs1, offset bgeu rs2, rs1, offset |
| Pseudo Inst. nop li mv not neg seqz sltz sqtz beqz blez blez bgt bgt bgt bgt bgt ble ble ble ble u j | load byte load byte (U) store word store half store byte D-Instructions: Name no operation load immediate move not negate set equal to zero set not equal to zero set less than zero set greater than zero branch if == 0 branch if <= 0 branch if <= 0 branch if <= 0 branch if > (U) branch if <= (U) jump | lb rd, offset(rs) lbu rd, offset(rs) sw rs2, offset(rs1) sh rs2, offset(rs1) sb rs2, offset(rs1) Usage nop li rd, constant mv rd, rs not rd, rs neg rd, rs segz rd, rs sltz rd, rs sltz rd, rs stz rd, rs beqz rs, offset bnez rs, offset blez rs, offset blez rs, offset bgtz rs, offset bgtz rs, offset bgt rs1, rs2, offset ble rs1, rs2, offset bleu rs1, rs2, offset bleu rs1, rs2, offset bleu rs1, rs2, offset | rd = mem[rs + offset] [15:0] rd = mem[rs + offset] [7:0] rd = mem[rs + offset] [7:0] |
| Pseudo Inst. nop li mv not neg seqz sltz sqtz beqz blez blez bgt bgt bgt uble bleu j jr | load byte load byte (U) store word store half store byte D-Instructions: Name no operation load immediate move not negate set equal to zero set not equal to zero set less than zero set greater than zero branch if == 0 branch if <= 0 branch if <= 0 branch if <= 0 branch if > 0 branch if <= 0 | lb rd, offset(rs) lbu rd, offset(rs) sw rs2, offset(rs1) sh rs2, offset(rs1) sb rs2, offset(rs1) Usage nop li rd, constant mv rd, rs not rd, rs neg rd, rs segz rd, rs sltz rd, rs sqtz rd, rs sqtz rd, rs beqz rs, offset bnez rs, offset blez rs, offset bgtz rs, offset bgtz rs, offset bgtz rs, offset bgt rs1, rs2, offset bleu rs1, rs2, offset bleu rs1, rs2, offset j offset j offset j offset j offset j offset | rd = mem[rs + offset] [15:0] rd = mem[rs + offset] [7:0] rd = mem[rs + offset] [7:0] mem[rs1 + offset] [31:0] = rs2 mem[rs1 + offset] [15:0] = rs2[15:0] mem[rs1 + offset] [7:0] = rs2[7:0] - Meaning or Implementation addi x0, x0, 0 rd = constant addi rd, rs, 0 xori rd, rs, -1 sub rd, x0, rs sltui rd, rs, 1 sltui rd, rs, 1 sltui rd, rs, x0 slt rd, x0, rs slt rd, x0, rs slt rd, x0, rs slt rd, x0, rs beq rs, x0, offset bne rs, x0, offset bge x0, rs, offset blt rs, x0, offset blt rs, x0, offset blt rs2, rs1, offset bge rs2, rs1, offset |
| Pseudo Inst. nop li mv not neg seqz snez sltz sqtz beqz blez bgez bltz bgtbgtu ble bleu jr call | load byte load byte (U) store word store half store byte D-Instructions: Name no operation load immediate move not negate set equal to zero set not equal to zero set less than zero set greater than zero branch if == 0 branch if <= 0 branch if <= 0 branch if < 0 branch if > 0 branch if > 0 branch if > (U) branch if <= 0 branch if <= (U) jump jump register call subroutine | lb rd, offset(rs) lbu rd, offset(rs) sw rs2, offset(rs1) sh rs2, offset(rs1) sb rs2, offset(rs1) Usage nop li rd, constant mv rd, rs not rd, rs neg rd, rs seqz rd, rs snez rd, rs sltz rd, rs sqtz rd, rs beqz rs, offset bnez rs, offset blez rs, offset blez rs, offset bgtz rs, offset bgtz rs, offset bgt rs1, rs2, offset bgtu rs1, rs2, offset bleu rs1, rs2, offset j offset j offset j offset j offset j offset | rd = mem[rs + offset] [15:0] rd = mem[rs + offset] [7:0] rd = mem[rs + offset] [7:0] mem[rs1 + offset] [31:0] = rs2 mem[rs1 + offset] [15:0] = rs2[15:0] mem[rs1 + offset] [7:0] = rs2[7:0] - Meaning or Implementation addi x0, x0, 0 rd = constant addi rd, rs, 0 xori rd, rs, -1 sub rd, x0, rs sltui rd, rs, 1 sltui rd, rs, 1 sltui rd, rs, x0 slt rd, x0, rs slt rd, x0, rs slt rd, x0, rs slt rd, x0, offset bne rs, x0, offset bne rs, x0, offset bge x0, rs, offset blt rs, x0, offset blt rs, x0, offset blt rs2, rs1, offset bge rs2, rs1, offset bgeu rs2, rs1, offset jal x0, offset jal x0, offset |
| Pseudo Inst. nop li mv not neg seqz sltz sqtz beqz blez blez bgt bgt bgt uble bleu j jr | load byte load byte (U) store word store half store byte D-Instructions: Name no operation load immediate move not negate set equal to zero set not equal to zero set less than zero set greater than zero branch if == 0 branch if <= 0 branch if <= 0 branch if <= 0 branch if > 0 branch if <= 0 | lb rd, offset(rs) lbu rd, offset(rs) sw rs2, offset(rs1) sh rs2, offset(rs1) sb rs2, offset(rs1) Usage nop li rd, constant mv rd, rs not rd, rs neg rd, rs segz rd, rs sltz rd, rs sqtz rd, rs sqtz rd, rs beqz rs, offset bnez rs, offset blez rs, offset bgtz rs, offset bgtz rs, offset bgtz rs, offset bgt rs1, rs2, offset bleu rs1, rs2, offset bleu rs1, rs2, offset j offset j offset j offset j offset j offset | rd = mem[rs + offset] [15:0] rd = mem[rs + offset] [7:0] rd = mem[rs + offset] [7:0] mem[rs1 + offset] [31:0] = rs2 mem[rs1 + offset] [15:0] = rs2[15:0] mem[rs1 + offset] [7:0] = rs2[7:0] - Meaning or Implementation addi x0, x0, 0 rd = constant addi rd, rs, 0 xori rd, rs, -1 sub rd, x0, rs sltui rd, rs, 1 sltui rd, rs, 1 sltui rd, rs, x0 slt rd, x0, rs slt rd, x0, rs slt rd, x0, rs slt rd, x0, rs beq rs, x0, offset bne rs, x0, offset bge x0, rs, offset blt rs, x0, offset blt rs, x0, offset blt rs2, rs1, offset bge rs2, rs1, offset |

| Register | | D | D | 37 | D |
|----------|-------|----------------|------------|--------|---------------------|
| Reg. | Name | Purpose | Reg. | Name | Purpose |
| x0 | zero | zero | x8 | s0/fp | saved/frame pointer |
| x1 | ra | return address | x 9 | s1 | saved |
| x2 | sp | stack pointer | x10-x11 | a0-a1 | args/return values |
| x3 | ab | global pointer | x12-x17 | a2-a7 | args |
| x4 | tp | thread pointer | x17-x27 | s2-s11 | saved |
| x5-x7 | tŌ-t2 | temporary | x28-x31 | t3-t6 | temporary |