

MIT 6.004 ISA Reference Card: Instructions

| Instruction | Syntax | Description | Execution |
|-------------|----------------------------|----------------------------------|--|
| LUI | lui rd, immU | Load Upper Immediate | reg[rd] <= immU << 12 |
| JAL | jal rd, immJ | Jump and Link | reg[rd] <= pc + 4 pc <= pc + immJ |
| JALR | jalr rd, rs1, immJ | Jump and Link Register | reg[rd] <= pc + 4 pc <= {(reg[rs1] + immJ)[31:1], 1'b0} |
| BEQ | beq rs1, rs2, immB | Branch if = | pc <= (reg[rs1] == reg[rs2]) ? pc + immB : pc + 4 |
| BNE | bne rs1, rs2, immB | Branch if ≠ | pc <= (reg[rs1] != reg[rs2]) ? pc + immB : pc + 4 |
| BLT | blt rs1, rs2, immB | Branch if < (Signed) | pc <= (reg[rs1] < _s reg[rs2]) ? pc + immB : pc + 4 |
| BGE | bge rs1, rs2, immB | Branch if ≥ (Signed) | pc <= (reg[rs1] >= _s reg[rs2]) ? pc + immB : pc + 4 |
| BLTU | bltu rs1, rs2, immB | Branch if < (Unsigned) | pc <= (reg[rs1] < _u reg[rs2]) ? pc + immB : pc + 4 |
| BGEU | bgeu rs1, rs2, immB | Branch if ≥ (Unsigned) | pc <= (reg[rs1] >= _u reg[rs2]) ? pc + immB : pc + 4 |
| LW | lw rd, immI(rs1) | Load Word | reg[rd] <= mem[reg[rs1] + immI] |
| SW | sw rs2, immS(rs1) | Store Word | mem[reg[rs1] + immS] <= reg[rs2] |
| ADDI | addi rd, rs1, immI | Add Immediate | reg[rd] <= reg[rs1] + immI |
| SLTI | slti rd, rs1, immI | Compare < Immediate (Signed) | reg[rd] <= (reg[rs1] < _s immI) ? 1 : 0 |
| SLTIU | sltiu rd, rs1, immI | Compare < Immediate (Unsigned) | reg[rd] <= (reg[rs1] < _u immI) ? 1 : 0 |
| XORI | xori rd, rs1, immI | Xor Immediate | reg[rd] <= reg[rs1] ^ immI |
| ORI | ori rd, rs1, immI | Or Immediate | reg[rd] <= reg[rs1] immI |
| ANDI | andi rd, rs1, immI | And Immediate | reg[rd] <= reg[rs1] & immI |
| SLLI | slli rd, rs1, immI | Shift Left Logical Immediate | reg[rd] <= reg[rs1] << immI |
| SRLI | srl rd, rs1, immI | Shift Right Logical Immediate | reg[rd] <= reg[rs1] >> _u immI |
| SRAI | srai rd, rs1, immI | Shift Right Arithmetic Immediate | reg[rd] <= reg[rs1] >> _s immI |
| ADD | add rd, rs1, rs2 | Add | reg[rd] <= reg[rs1] + reg[rs2] |
| SUB | sub rd, rs1, rs2 | Subtract | reg[rd] <= reg[rs1] - reg[rs2] |
| SLL | sll rd, rs1, rs2 | Shift Left Logical | reg[rd] <= reg[rs1] << reg[rs2] |
| SLT | slt rd, rs1, rs2 | Compare < (Signed) | reg[rd] <= (reg[rs1] < _s reg[rs2]) ? 1 : 0 |
| SLTU | sltu rd, rs1, rs2 | Compare < (Unsigned) | reg[rd] <= (reg[rs1] < _u reg[rs2]) ? 1 : 0 |
| XOR | xor rd, rs1, rs2 | Xor | reg[rd] <= reg[rs1] ^ reg[rs2] |
| SRL | srl rd, rs1, rs2 | Shift Right Logical | reg[rd] <= reg[rs1] >> _u reg[rs2] |
| SRA | sra rd, rs1, rs2 | Shift Right Arithmetic | reg[rd] <= reg[rs1] >> _s reg[rs2] |
| OR | or rd, rs1, rs2 | Or | reg[rd] <= reg[rs1] reg[rs2] |
| AND | and rd, rs1, rs2 | And | reg[rd] <= reg[rs1] & reg[rs2] |

MIT 6.004 ISA Reference Card: Pseudoinstructions

| Pseudoinstruction | Description | Execution |
|-----------------------------|--|--|
| li rd, constant | Load Immediate | reg[rd] <= constant |
| mv rd, rs1 | Move | reg[rd] <= reg[rs1] + 0 |
| not rd, rs1 | Logical Not | reg[rd] <= reg[rs1] ^ -1 |
| neg rd, rs1 | Arithmetic Negation | reg[rd] <= 0 - reg[rs1] |
| j label | Jump | pc <= label |
| jal label | Jump and Link (with ra) | reg[ra] <= pc + 4 pc <= label |
| jr rs | Jump Register | pc <= reg[rs1] & ~1 |
| jalr rs | Jump and Link Register (with ra) | reg[ra] <= pc + 4 pc <= reg[rs1] & ~1 |
| ret | Return from Subroutine | pc <= reg[ra] |
| bgt rs1, rs2, label | Branch > (Signed) | pc <= (reg[rs1] > _s reg[rs2]) ? label : pc + 4 |
| ble rs1, rs2, label | Branch ≤ (Signed) | pc <= (reg[rs1] <= _s reg[rs2]) ? label : pc + 4 |
| bgtu rs1, rs2, label | Branch > (Unsigned) | pc <= (reg[rs1] > _s reg[rs2]) ? label : pc + 4 |
| bleu rs1, rs2, label | Branch ≤ (Unsigned) | pc <= (reg[rs1] <= _s reg[rs2]) ? label : pc + 4 |
| beqz rs1, label | Branch = 0 | pc <= (reg[rs1] == 0) ? label : pc + 4 |
| bnez rs1, label | Branch ≠ 0 | pc <= (reg[rs1] != 0) ? label : pc + 4 |
| bltz rs1, label | Branch < 0 (Signed) | pc <= (reg[rs1] < _s 0) ? label : pc + 4 |
| bgez rs1, label | Branch ≥ 0 (Signed) | pc <= (reg[rs1] >= _s 0) ? label : pc + 4 |
| bgtz rs1, label | Branch > 0 (Signed) | pc <= (reg[rs1] > _s 0) ? label : pc + 4 |
| blez rs1, label | Branch ≤ 0 (Signed) | pc <= (reg[rs1] <= _s 0) ? label : pc + 4 |

MIT 6.004 ISA Reference Card: Calling Convention

| Registers | Symbolic names | Description | Saver |
|----------------|----------------|--------------------------------------|--------|
| x0 | zero | Hardwired zero | — |
| x1 | ra | Return address | Caller |
| x2 | sp | Stack pointer | Callee |
| x3 | gp | Global pointer | — |
| x4 | tp | Thread pointer | — |
| x5-x7 | t0-t2 | Temporary registers | Caller |
| x8-x9 | s0-s1 | Saved registers | Callee |
| x10-x11 | a0-a1 | Function arguments and return values | Caller |
| x12-x17 | a2-a7 | Function arguments | Caller |
| x18-x27 | s2-s11 | Saved registers | Callee |
| x28-x31 | t3-t6 | Temporary registers | Caller |

MIT 6.004 ISA Reference Card: Instruction Encodings

| | | | | | | | | | | | | |
|-----------------------|----|-----|----|-----|----|--------|----|-------------|---|--------|---|--------|
| 31 | 25 | 24 | 20 | 19 | 15 | 14 | 12 | 11 | 7 | 6 | 0 | |
| funct7 | | rs2 | | rs1 | | funct3 | | rd | | opcode | | R-type |
| imm[11:0] | | | | rs1 | | funct3 | | rd | | opcode | | I-type |
| imm[11:5] | | rs2 | | rs1 | | funct3 | | imm[4:0] | | opcode | | S-type |
| imm[12:10:5] | | rs2 | | rs1 | | funct3 | | imm[4:1 11] | | opcode | | B-type |
| imm[31:12] | | | | | | | | rd | | opcode | | U-type |
| imm[20:10:1 11:19:12] | | | | | | | | rd | | opcode | | J-type |

RV32I Base Instruction Set (MIT 6.004 subset)

| | | | | | | |
|-----------------------|-------|-----|-----|-------------|---------|-------|
| imm[31:12] | | | | rd | 0110111 | LUI |
| imm[20 10:1 11 19:12] | | | | rd | 1101111 | JAL |
| imm[11:0] | | rs1 | 000 | rd | 1100111 | JALR |
| imm[12 10:5] | rs2 | rs1 | 000 | imm[4:1 11] | 1100011 | BEQ |
| imm[12 10:5] | rs2 | rs1 | 001 | imm[4:1 11] | 1100011 | BNE |
| imm[12 10:5] | rs2 | rs1 | 100 | imm[4:1 11] | 1100011 | BLT |
| imm[12 10:5] | rs2 | rs1 | 101 | imm[4:1 11] | 1100011 | BGE |
| imm[12 10:5] | rs2 | rs1 | 110 | imm[4:1 11] | 1100011 | BLTU |
| imm[12 10:5] | rs2 | rs1 | 111 | imm[4:1 11] | 1100011 | BGEU |
| imm[11:0] | | rs1 | 010 | rd | 0000011 | LW |
| imm[11:5] | rs2 | rs1 | 010 | imm[4:0] | 0100011 | SW |
| imm[11:0] | | rs1 | 000 | rd | 0010011 | ADDI |
| imm[11:0] | | rs1 | 010 | rd | 0010011 | SLTI |
| imm[11:0] | | rs1 | 011 | rd | 0010011 | SLTIU |
| imm[11:0] | | rs1 | 100 | rd | 0010011 | XORI |
| imm[11:0] | | rs1 | 110 | rd | 0010011 | ORI |
| imm[11:0] | | rs1 | 111 | rd | 0010011 | ANDI |
| 0000000 | shamt | rs1 | 001 | rd | 0010011 | SLLI |
| 0000000 | shamt | rs1 | 101 | rd | 0010011 | SRLI |
| 0100000 | shamt | rs1 | 101 | rd | 0010011 | SRAI |
| 0000000 | rs2 | rs1 | 000 | rd | 0110011 | ADD |
| 0100000 | rs2 | rs1 | 000 | rd | 0110011 | SUB |
| 0000000 | rs2 | rs1 | 001 | rd | 0110011 | SLL |
| 0000000 | rs2 | rs1 | 010 | rd | 0110011 | SLT |
| 0000000 | rs2 | rs1 | 011 | rd | 0110011 | SLTU |
| 0000000 | rs2 | rs1 | 100 | rd | 0110011 | XOR |
| 0000000 | rs2 | rs1 | 101 | rd | 0110011 | SRL |
| 0100000 | rs2 | rs1 | 101 | rd | 0110011 | SRA |
| 0000000 | rs2 | rs1 | 110 | rd | 0110011 | OR |
| 0000000 | rs2 | rs1 | 111 | rd | 0110011 | AND |