

RISC-V Instruction Set Summary

32-bit Base Integer ISA (RV32I)

Arithmetic Operations

Mnemonic	Instruction	Type	Description
ADD rd, rs1, rs2	Add	R	$rd \leftarrow rs1 + rs2$
SUB rd, rs1, rs2	Subtract	R	$rd \leftarrow rs1 - rs2$
ADDI rd, rs1, imm12	Add immediate	I	$rd \leftarrow rs1 + imm12$
SLT rd, rs1, rs2	Set less than	R	$rd \leftarrow rs1 < rs2 ? 1 : 0$
SLTI rd, rs1, imm12	Set less than immediate	I	$rd \leftarrow rs1 < imm12 ? 1 : 0$
SLTU rd, rs1, rs2	Set less than unsigned	R	$rd \leftarrow rs1 < rs2 ? 1 : 0$
SLTIU rd, rs1, imm12	Set less than immediate unsigned	I	$rd \leftarrow rs1 < imm12 ? 1 : 0$
LUI rd, imm20	Load upper immediate	U	$rd \leftarrow imm20 \ll 12$
AUIPC rd, imm20	Add upper immediate to PC	U	$rd \leftarrow pc + imm20 \ll 12$

Logical Operations

Mnemonic	Instruction	Type	Description
AND rd, rs1, rs2	AND	R	$rd \leftarrow rs1 \& rs2$
OR rd, rs1, rs2	OR	R	$rd \leftarrow rs1 \mid rs2$
XOR rd, rs1, rs2	XOR	R	$rd \leftarrow rs1 \wedge rs2$
ANDI rd, rs1, imm12	AND immediate	I	$rd \leftarrow rs1 \& imm12$
ORI rd, rs1, imm12	OR immediate	I	$rd \leftarrow rs1 \mid imm12$
XORI rd, rs1, imm12	XOR immediate	I	$rd \leftarrow rs1 \wedge imm12$
SLL rd, rs1, rs2	Shift left logical	R	$rd \leftarrow rs1 \ll rs2$
SRL rd, rs1, rs2	Shift right logical	R	$rd \leftarrow rs1 \gg rs2$
SRA rd, rs1, rs2	Shift right arithmetic	R	$rd \leftarrow rs1 \gg rs2$
SLLI rd, rs1, shamt	Shift left logical immediate	I	$rd \leftarrow rs1 \ll shamt$
SRLI rd, rs1, shamt	Shift right logical imm.	I	$rd \leftarrow rs1 \gg shamt$
SRAI rd, rs1, shamt	Shift right arithmetic immediate	I	$rd \leftarrow rs1 \gg shamt$

32-bit Instruction Formats

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	funct							rs2					rs1				funct			rd				opcode								
I	immediate[11:0]												rs1				funct			rd				opcode								
S	imm[11:5]							rs2					rs1				funct			imm[4:0]				opcode								
B	imm[12][10:5]							rs2					rs1				funct			imm[4:1][11]				opcode								
U	immediate[31:12]																				rd				opcode							
J	imm[20][10:1][11]												imm[19:12]							rd				opcode								

Load/Store Operations

Mnemonic	Instruction	Type	Description
LW rd, imm12(rs1)	Load word	I	$rd \leftarrow mem[rs1 + imm12]$
LH rd, imm12(rs1)	Load halfword	I	$rd \leftarrow mem[rs1 + imm12]$
LB rd, imm12(rs1)	Load byte	I	$rd \leftarrow mem[rs1 + imm12]$
LHU rd, imm12(rs1)	Load halfword unsigned	I	$rd \leftarrow mem[rs1 + imm12]$
LBU rd, imm12(rs1)	Load byte unsigned	I	$rd \leftarrow mem[rs1 + imm12]$
SW rs2, imm12(rs1)	Store word	S	$rs2(31:0) \rightarrow mem[rs1 + imm12]$
SH rs2, imm12(rs1)	Store halfword	S	$rs2(15:0) \rightarrow mem[rs1 + imm12]$
SB rs2, imm12(rs1)	Store byte	S	$rs2(7:0) \rightarrow mem[rs1 + imm12]$

Branching

Mnemonic	Instruction	Type	Description
BEQ rs1, rs2, imm12	Branch equal	B	if $rs1 = rs2$ $pc \leftarrow pc + imm12$
BNE rs1, rs2, imm12	Branch not equal	B	if $rs1 \neq rs2$ $pc \leftarrow pc + imm12$
BGE rs1, rs2, imm12	Branch greater than or equal	B	if $rs1 \geq rs2$ $pc \leftarrow pc + imm12$
BGEU rs1, rs2, imm12	Branch greater than or equal unsigned	B	if $rs1 \geq rs2$ $pc \leftarrow pc + imm12$
BLT rs1, rs2, imm12	Branch less than	B	if $rs1 < rs2$ $pc \leftarrow pc + imm12$
BLTU rs1, rs2, imm12	Branch less than unsigned	B	if $rs1 < rs2$ $pc \leftarrow pc + imm12$
JAL rd, imm20	Jump and link	J	$rd \leftarrow pc + 4$ $pc \leftarrow pc + imm20$
JALR rd, imm12(rs1)	Jump and link register	I	$rd \leftarrow pc + 4$ $pc \leftarrow rs1 + imm12$

Pseudoinstructions

Mnemonic	Instruction	Base instruction(s)
LI rd, imm12	Load immediate (near)	ADDI rd, zero, imm12
LI rd, imm	Load immediate (far)	LUI rd, imm[31:12] ADDI rd, rd, imm[11:0]
LA rd, sym	Load address (far)	AUIPC rd, (sym-pc)[31:12] ADDI rd, rd, sym(sym-pc)[11:0]
MV rd, rs	Copy register	ADDI rd, rs, 0
NOT rd, rs	One's complement	XORI rd, rs, -1
NEG rd, rs	Two's complement	SUB rd, zero, rs
BGT rs1, rs2, offset	Branch if rs1 > rs2	BLT rs2, rs1, offset
BLE rs1, rs2, offset	Branch if rs1 ≤ rs2	BGE rs2, rs1, offset
BGTU rs1, rs2, offset	Branch if rs1 > rs2 (unsigned)	BLTU rs2, rs1, offset
BLEU rs1, rs2, offset	Branch if rs1 ≤ rs2 (unsigned)	BGEU rs2, rs1, offset
BEQZ rs1, offset	Branch if rs1 = 0	BEQ rs1, zero, offset
BNEZ rs1, offset	Branch if rs1 ≠ 0	BNE rs1, zero, offset
BGEZ rs1, offset	Branch if rs1 ≥ 0	BGE rs1, zero, offset
BLEZ rs1, offset	Branch if rs1 ≤ 0	BGE zero, rs1, offset
BGTZ rs1, offset	Branch if rs1 > 0	BLT zero, rs1, offset
J offset	Unconditional jump	JAL zero, offset
CALL offset	Call subroutine (far)	AUIPC ra, offset[31:12] JALR ra, offset[11:0](ra)
RET	Return from subroutine	JALR zero, 0(ra)
NOP	No operation	ADDI zero, zero, 0

Register File

Register Names

x0	x1	x2	x3	x4	x5	x6	x7
x8	x9	x10	x11	x12	x13	x14	x15
x16	x17	x18	x19	x20	x21	x22	x23
x24	x25	x26	x27	x28	x29	x30	x31



Register Aliases

zero	ra	sp	gp	tp	t0	t1	t2
s0/fp	s1	a0	a1	a2	a3	a4	a5
a6	a7	s2	s3	s4	s5	s6	s7
s8	s9	s10	s11	t3	t4	t5	t6

ra: Return address

sp: Stack pointer

gp: Global pointer

tp: Thread pointer

t0–t6: Temporary registers

s0–s11: Callee-saved registers

a0–a7: Argument registers

a0–a1: Return value(s)