RISC-V Reference

Base Integer Instructions

Inst	Name	FMT	Opcode	funct3	funct7	Description	Note
add	ADD	R	0110011	000	000 0000	R[rd] = R[rs1] + R[rs2]	
sub	SUB	R	0110011	000	010 0000	R[rd] = R[rs1] - R[rs2]	
xor	XOR	R	0110011	100	000 0000	R[rd] = R[rs1] ^ R[rs2]	
or	OR	R	0110011	110	000 0000	R[rd] = R[rs1] | R[rs2]	
and	AND	R	0110011	111	000 0000	R[rd] = R[rs1] & R[rs2]	
sll	Shift Left Logical	R	0110011	001	000 0000	R[rd] = R[rs1] << R[rs2]	
srl	Shift Right Logical	R	0110011	101	000 0000	R[rd] = R[rs1] >> R[rs2]	
sra	Shift Right Arith*	R	0110011	101	010 0000	R[rd] = R[rs1] >> R[rs2]	sign-extends
slt	Set Less Than	R	0110011	010	000 0000	R[rd] = (rs1 < rs2)?1:0	
addi	ADD Immediate	I	0010011	000		R[rd] = R[rs1] + SE(imm)	
xori	XOR Immediate	I	0010011	100		R[rd] = R[rs1] ^ SE(imm)	
ori	OR Immediate	I	0010011	110		R[rd] = R[rs1] | SE(imm)	
andi	AND Immediate	I	0010011	111		R[rd] = R[rs1] & SE(imm)	
slli	Shift Left Logical Imm	I	0010011	001	imm[11:5] =0x00	R[rd] = R[rs1] << imm[4:0]	
srli	Shift Right Logical Imm	I	0010011	101	imm[11:5] =0x00	R[rd] = R[rs1] >> imm[4:0]	
srai	Shift Right Arith Imm	I	0010011	101	imm[11:5] =0x20	R[rd] = R[rs1] >> imm[4:0]	sign-extends
lw	Load Word	I	0000011	010		R[rd] = M[R[rs1]+SE(imm)]	
sw	Store Word	S	0100011	010		M[R[rs1]+SE(imm)] = R[rs2]	
beq	Branch ==	SB	1100011	000		if(rs1 == rs2) PC += SE(imm) << 1	
bne	Branch !=	SB	1100011	001		if(rs1 != rs2) PC += SE(imm) << 1	
blt	Branch <	SB	1100011	100		if(rs1 < rs2) PC += SE(imm) <<1	
bge	Branch >=	SB	1100011	101		if(rs1 >= rs2) PC += SE(imm) <<1	
jal	Jump And Link	UJ	1101111			R[rd] = PC+4; PC += SE(imm) <<1	
jalr	Jump And Link Reg	I	1100111	000		R[rd] = PC+4; PC = R[rs1]+ SE(imm)	
lui	Load Upper Imm	U	0110111			R[rd] = SE(imm) << 12	
auipc	Add Upper Imm to PC	U	0010111			R[rd] = PC + (SE(imm) << 12)	
csrrw	CSR read & write	I	1110011	001		R[rd] = CSRs[csr]; CSRs[csr] = R[rs1]	
csrrs	CSR read & set	I	1110011	010		R[rd] = CSRs[csr]; CSRs[csr] = CSRs[csr] | R[rs1]	
csrrc	CSR read & clear	I	1110011	011		R[rd] = CSRs[csr]; CSRs[csr] = CSRs[csr] & ∼R[rs1]	
ecall	Environment Call	I	1110011	000	imm=0x0	Transfer control to OS	
ebreak	Environment Break	I	1110011	000	imm=0x1	Transfer control to debugger	

R = Register file access CSR = Coprocessor Register
SE = Sign extend (e.g., scause, sepc)

Core Instruction Formats

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[11|9:4]	rs2	rs1	funct3	imm[3:0|10]	opcode	SB-type
imm[19:0]				rd	opcode	U-type
imm[19|9:0|10|18:11]				rd	opcode	UJ-type

Opcodes, Base conversion

Binary	Hex	Opcode
000 0000	00	
000 0001	01	
000 0010	02	
000 0011	03	lw
000 0100	04	
000 0101	05	
000 0110	06	
000 0111	07	
000 1000	08	
000 1001	09	
000 1010	0A	
000 1011	0B	
000 1100	0C	
000 1101	0D	
000 1110	0E	
000 1111	0F	
001 0000	10	
001 0001	11	
001 0010	12	
001 0011	13	I-type
001 0100	14	
001 0101	15	
001 0110	16	
001 0111	17	auipc
001 1000	18	
001 1001	19	
001 1010	1A	
001 1011	1B	
001 1100	1C	
001 1101	1D	
001 1110	1E	
001 1111	1F	

Binary	Hex	Opcode
010 0000	20	
010 0001	21	
010 0010	22	
010 0011	23	sw
010 0100	24	
010 0101	25	
010 0110	26	
010 0111	27	
010 1000	28	
010 1001	29	
010 1010	2A	
010 1011	2B	
010 1100	2C	
010 1101	2D	
010 1110	2E	
010 1111	2F	
011 0000	30	
011 0001	31	
011 0010	32	
011 0011	33	R-type
011 0100	34	
011 0101	35	
011 0110	36	
011 0111	37	lui
011 1000	38	
011 1001	39	
011 1010	3A	
011 1011	3B	
011 1100	3C	
011 1101	3D	
011 1110	3E	
011 1111	3F	

Binary	Hex	Opcode
100 0000	40	
100 0001	41	
100 0010	42	
100 0011	43	
100 0100	44	
100 0101	45	
100 0110	46	
100 0111	47	
100 1000	48	
100 1001	49	
100 1010	4A	
100 1011	4B	
100 1100	4C	
100 1101	4D	
100 1110	4E	
100 1111	4F	
101 0000	50	
101 0001	51	
101 0010	52	
101 0011	53	
101 0100	54	
101 0101	55	
101 0110	56	
101 0111	57	
101 1000	58	
101 1001	59	
101 1010	5A	
101 1011	5B	
101 1100	5C	
101 1101	5D	
101 1110	5E	
101 1111	5F	

Binary	Hex	Opcode
110 0000	60	
110 0001	61	
110 0010	62	
110 0011	63	SB-type
110 0100	64	
110 0101	65	
110 0110	66	
110 0111	67	jalr
110 1000	68	
110 1001	69	
110 1010	6A	
110 1011	6B	
110 1100	6C	
110 1101	6D	
110 1110	6E	
110 1111	6F	jal
111 0000	70	
111 0001	71	
111 0010	72	
111 0011	73	exceptions
111 0100	74	
111 0101	75	
111 0110	76	
111 0111	77	
111 1000	78	
111 1001	79	
111 1010	7A	
111 1011	7B	
111 1100	7C	
111 1101	7D	
111 1110	7E	

Registers

Register	Name	Description	Saver
x0	zero	Zero constant	—
x1	ra	Return address	Caller
x2	sp	Stack pointer	Callee
x3	gp	Global pointer	—
x4	tp	Thread pointer	—
x5-x7	t0-t2	Temporaries	Caller
x8	s0 / fp	Saved / frame pointer	Callee
x9	s1	Saved register	Callee
x10-x11	a0-a1	Fn args/return values	Caller
x12-x17	a2-a7	Fn args	Caller
x18-x27	s2-s11	Saved registers	Callee
x28-x30	t3-t5	Temporaries	Caller
x31	at	Assembler Temporary	Caller

Memory Allocation