RISC-V Instruction Set Summary

	31	: 25	24:20	19:15	14:12	11:7	6:0	
	fun	ct7	rs2	rs1	funct3	rd	op	R-Type
	imm₁	imm _{11:0}		rs1	funct3	rd	op	I-Type
	imm _{11:5} imm _{12,10:5}		rs2	rs1	funct3	imm _{4:0}	op	S-Type
Ī			rs2	rs1	funct3	imm _{4:1,11}	ор	B-Type
	imm ₃	imm _{31:12}				rd	op	U-Type
	imm ₂	0,10:1,11,19	9:12		rd	op	J-Type	
Ī	fs3	funct2	fs2	fs1	funct3	fd	ор	R4-Type
	5 bits	2 bits	5 bits	5 bits	3 bits	5 bits	7 bits	•

Figure B.1 RISC-V 32-bit instruction formats

imm: signed immediate in imm_{11:0}
uimm: 5-bit unsigned immediate in i

uimm: 5-bit unsigned immediate in imm_{4:0}
 upimm: 20 upper bits of a 32-bit immediate, in imm_{31:12}
 Address: memory address: rs1 + SignExt(imm_{11:0})

• [Address]: data at memory location Address

 $\label{eq:bounds} \begin{array}{ll} \bullet \mbox{ BTA:} & \mbox{branch target address: } PC + \mbox{SignExt}(\{\mbox{imm}_{12:1}, 1'b0\}) \\ \bullet \mbox{ JTA:} & \mbox{jump target address: } PC + \mbox{SignExt}(\{\mbox{imm}_{20:1}, 1'b0\}) \\ \end{array}$

label: text indicating instruction address
 SignExt: value sign-extended to 32 bits
 ZeroExt: value zero-extended to 32 bits
 csr: control and status register

Table B.1 RV32I: RISC-V integer instructions

op	funct3	funct7	Type	Instruction		Description	Operation
0000011 (3)	000	_	I	lb rd,	imm(rs1)	load byte	rd = SignExt([Address] _{7:0})
0000011 (3)	001	_	I	lh rd,	imm(rs1)	load half	rd = SignExt([Address] _{15:0})
0000011 (3)	010	_	I	lw rd,	imm(rs1)	load word	rd = [Address] _{31:0}
0000011 (3)	100	_	I	lbu rd,	imm(rs1)	load byte unsigned	rd = ZeroExt([Address] _{7:0})
0000011 (3)	101	_	I	lhu rd,	imm(rs1)	load half unsigned	rd = ZeroExt([Address] _{15:0})
0010011 (19)	000	_	I	addi rd,	rs1, imm	add immediate	rd = rs1 + SignExt(imm)
0010011 (19)	001	0000000*	I	slli rd,	rs1, uimm	shift left logical immediate	rd = rs1 << uimm
0010011 (19)	010	_	I	slti rd,	rs1, imm	set less than immediate	rd = (rs1 < SignExt(imm))
0010011 (19)	011	_	I	sltiu rd,	rs1, imm	set less than imm. unsigned	rd = (rs1 < SignExt(imm))
0010011 (19)	100	_	I	xori rd,	rs1, imm	xor immediate	rd = rs1 ^ SignExt(imm)
0010011 (19)	101	0000000*	I	srli rd,	rs1, uimm	shift right logical immediate	rd = rs1 >> uimm
0010011 (19)	101	0100000*	I	srai rd,	rs1, uimm	shift right arithmetic imm.	rd = rs1 >>> uimm
0010011 (19)	110	_	I	ori rd,	rs1, imm	or immediate	rd = rs1 SignExt(imm)
0010011 (19)	111	_	I	andi rd,	rs1, imm	and immediate	rd = rs1 & SignExt(imm)
0010111 (23)	_	-	U	auipc rd,	upimm	add upper immediate to PC	rd = {upimm, 12'b0} + PC
0100011 (35)	000	_	S	sb rs2,	imm(rs1)	store byte	$[Address]_{7:0} = rs2_{7:0}$
0100011 (35)	001	_	S	sh rs2,	imm(rs1)	store half	[Address] _{15:0} = rs2 _{15:0}
0100011 (35)	010	-	S	sw rs2,	imm(rs1)	store word	[Address] _{31:0} = rs2
0110011 (51)	000	0000000	R	add rd,	rs1, rs2	add	rd = rs1 + rs2
0110011 (51)	000	0100000	R	sub rd,	rs1, rs2	sub	rd = rs1 - rs2
0110011 (51)	001	0000000	R	sll rd,	rs1, rs2	shift left logical	rd = rs1 << rs2 _{4:0}
0110011 (51)	010	0000000	R	slt rd,	rs1, rs2	set less than	rd = (rs1 < rs2)
0110011 (51)	011	0000000	R	sltu rd,	rs1, rs2	set less than unsigned	rd = (rs1 < rs2)
0110011 (51)	100	0000000	R	xor rd,	rs1, rs2	xor	rd = rs1 ^ rs2
0110011 (51)	101	0000000	R	srl rd,	rs1, rs2	shift right logical	rd = rs1 >> rs2 _{4:0}
0110011 (51)	101	0100000	R	sra rd,	rs1, rs2	shift right arithmetic	rd = rs1 >>> rs2 _{4:0}
0110011 (51)	110	0000000	R	or rd,	rs1, rs2	or	rd = rs1 rs2
0110011 (51)	111	0000000	R	and rd,	rs1, rs2	and	rd = rs1 & rs2
0110111 (55)	-	_	U	lui rd,	upimm	load upper immediate	rd = {upimm, 12'b0}
1100011 (99)	000	_	В	beq rs1,	rs2, label	branch if =	if (rs1 == rs2) PC = BTA
1100011 (99)	001	_	В		rs2, label		if (rs1 ≠ rs2) PC = BTA
1100011 (99)	100	_	В	blt rs1,	rs2, label	branch if <	if (rs1 < rs2) PC = BTA
1100011 (99)	101	_	В		rs2, label	Drumen n =	if (rs1 ≥ rs2) PC = BTA
1100011 (99)	110	-	В			branch if < unsigned	if (rs1 < rs2) PC = BTA
1100011 (99)	111	_	В	bgeu rs1,	rs2, label	branch if ≥ unsigned	if (rs1 ≥ rs2) PC = BTA
1100111 (103)	000	_	I	jalr rd,	rs1, imm	jump and link register	PC = rs1 + SignExt(imm), rd = PC + 4
1101111 (111)	-	_	J	jal rd,	label	jump and link	PC = JTA, $rd = PC + 4$