Table 3 : RV32I RISC-V Integer instructions

op	Func3	Func7	Type	Mnemonic	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		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	0000000	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 immediate	rd = rs1 >> uimm
0010011(19)	110		I	ori rd, rs1, uimm	OR immediate	rd = rs1 | SignExt(imm)
0010011(19)	111		I	andi rd, rs1, uimm	AND immediate	rd = rs1 & SignExt(imm)
0010111(23)			U	auipc rd, rs1, uimm	ADD upper immediate to PC	rd = (upimm, 12’b0) + PC
0100011(35)			S	sb rs2,imm(rs1)	Store byte	[Address]$_{7:0}$ = rs2$_{7:0}$
0100011(35)			S	sh rs2,imm(rs1)	Store half	[Address]$_{15:0}$ = rs2$_{15:0}$
0100011(35)			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		B	beq rs1,rs2, label	Branch if equal =	if (rs1 == rs2) PC = BTA
1100011(99)	001		B	bne rs1,rs2, label	Branch if not equal ≠	if (rs1 != rs2) PC = BTA
1100011(99)	010		B	blt rs1,rs2, label	Branch if lower than <	if (rs1 < rs2) PC = BTA
1100011(99)	011		B	bge rs1,rs2, label	Branch if greater / equal ≥	if (rs1 ≥ rs2) PC = BTA
1100011(99)	100		B	bltu rs1,rs2, label	Branch if lower than unsigned <	if (rs1 < rs2) PC = BTA
1100011(99)	101		B	bgeu rs1,rs2, label	Branch if greater / equal unsign. ≥	if (rs1 ≥ rs2) PC = BTA
1100111(103)	000		I	jalr rd, rs1, label	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