

Base Integer Instructions: RV32I, RV64I, and RV128I							RV Privileged Instructions																		
Category	Name	Fmt	RV32I Base			+RV{64,128}			Category	Name	RV mnemonic														
Loads	Load Byte	I	LB	rd,rs1,imm		L{D Q} rd,rs1,imm			CSR Access	Atomic R/W	CSRRW	rd,csr,rs1													
	Load Halfword	I	LH	rd,rs1,imm						Atomic Read & Set Bit	CSRRS	rd,csr,rs1													
	Load Word	I	LW	rd,rs1,imm						Atomic Read & Clear Bit	CSRRC	rd,csr,rs1													
	Load Byte Unsigned	I	LBU	rd,rs1,imm						Atomic R/W Imm	CSRRWI	rd,csr,imm													
	Load Half Unsigned	I	LHU	rd,rs1,imm						Atomic Read & Set Bit Imm	CSRSSI	rd,csr,imm													
Stores	Store Byte	S	SB	rs1,rs2,imm		S{D Q} rs1,rs2,imm			Change Level	Env. Call	ECALL														
	Store Halfword	S	SH	rs1,rs2,imm						Environment Breakpoint	EBREAK														
	Store Word	S	SW	rs1,rs2,imm						Environment Return	ERET														
Shifts	Shift Left	R	SLL	rd,rs1,rs2		SLL{W D} rd,rs1,rs2			Trap Redirect to Supervisor	MRTS															
	Shift Left Immediate	I	SLLI	rd,rs1,shamt		SLLI{W D} rd,rs1,shamt				Redirect Trap to Hypervisor	MRTH														
	Shift Right	R	SRL	rd,rs1,rs2		SRL{W D} rd,rs1,rs2				Hypervisor Trap to Supervisor	HRTS														
	Shift Right Immediate	I	SRLI	rd,rs1,shamt		SRLI{W D} rd,rs1,shamt				Interrupt	WFI														
	Shift Right Arithmetic	R	SRA	rd,rs1,rs2		SRA{W D} rd,rs1,rs2				MMU	Supervisor FENCE	SFENCE.VM rs1													
	Shift Right Arith Imm	I	SRAI	rd,rs1,shamt		SRAI{W D} rd,rs1,shamt																			
Arithmetic	ADD	R	ADD	rd,rs1,rs2		ADD{W D} rd,rs1,rs2			Optional Compressed (16-bit) Instruction Extension: RVC																
	ADD Immediate	I	ADDI	rd,rs1,imm		ADDI{W D} rd,rs1,imm																			
	SUBtract	R	SUB	rd,rs1,rs2		SUB{W D} rd,rs1,rs2																			
	Load Upper Imm	U	LUI	rd,imm		C.LW rd',rs1',imm																			
	Add Upper Imm to PC	U	AUIPC	rd,imm						C.LWSP	rd,imm	LW rd',sp,imm*4													
Logical	XOR	R	XOR	rd,rs1,rs2		C.LD rd',rs1',imm				C.LDSP	rd,imm	LD rd',sp,imm*8													
	XOR Immediate	I	XORI	rd,rs1,imm						C.LQ	rd',rs1',imm	LQ rd',sp,imm*16													
	OR	R	OR	rd,rs1,rs2		C.LQSP				C.LQSP	rd,imm	LQ rd',rs1',imm*16													
	OR Immediate	I	ORI	rd,rs1,imm																					
	AND	R	AND	rd,rs1,rs2		C.SW rs1',rs2',imm																			
Compare	AND Immediate	I	ANDI	rd,rs1,imm						C.SWSP	rs2,imm	SW rs2,sp,imm*4													
	Set <	R	SLT	rd,rs1,rs2		C.SD rs1',rs2',imm				C.SDSP	rs2,imm	SD rs2,sp,imm*8													
	Set < Immediate	I	SLTI	rd,rs1,imm						C.SQ	rs1',rs2',imm	SQ rs1',sp,imm*16													
	Set < Unsigned	R	SLTU	rd,rs1,rs2		C.SQSP rs2,imm				C.SQSP	rs2,imm	SQ rs2,sp,imm*16													
	Set < Imm Unsigned	I	SLTIU	rd,rs1,imm		C.BEQZ rs1',imm																			
Branches	Branch =	SB	BEQ	rs1,rs2,imm		C.BNEZ rs1',imm																			
	Branch ≠	SB	BNE	rs1,rs2,imm																					
	Branch <	SB	BLT	rs1,rs2,imm		C.BGE rs1',imm																			
	Branch ≥	SB	BGE	rs1,rs2,imm																					
	Branch < Unsigned	SB	BLTU	rs1,rs2,imm		C.BLTU rs1',imm																			
Jump & Link	J&L	UJ	JAL	rd,imm																					
	Jump & Link Register	UJ	JALR	rd,rs1,imm		C.J imm																			
Synch	Synch thread	I	FENCE																						
	Synch Instr & Data	I	FENCE.I			C.LUI rd,imm																			
System	System CALL	I	SCALL																						
	System BREAK	I	SBREAK			C.MV rd,rs1																			
Counters	ReaD CYCLE	I	RDCYCLE	rd																					
	ReaD CYCLE upper Half	I	RDCYCLES	rd		C.RDTIME rd																			
	ReaD TIME	I	RDTIME	rd																					
	ReaD TIME upper Half	I	RDTIMES	rd		C.RDINSTRET rd																			
	ReaD INSTR RETired	I	RDINSTRET	rd																					
	ReaD INSTR upper Half	I	RDINSTRETH	rd		C.RDINSTRETH rd																			
32-bit Instruction Formats							16-bit (RVC) Instruction Formats																		
R		31	30	25 24	21 20	19	15 14	12 11	8	7	6	0	CR	15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0											
I		funct7		rs2		rs1		funct3		rd		opcode		funct4		rd/rs1		rs2		op					
S		imm[11:0]		rs1		funct3		rd		opcode		funct3		imm		rs2		op							
SB		imm[11:5]		rs2		rs1		funct3		imm[4:0]		opcode		funct3		imm		rd'		op					
U		imm[12]		imm[10:5]		rs2		rs1		funct3		imm[4:1]		imm[11]		opcode		funct3		imm		rd'		op	
UJ		imm[31:12]		imm[10:1]		imm[11]		imm[19:12]		rd		opcode		funct3		offset		rs1'		offset		op			
CJ		imm[20]		imm[11]		imm[11]		imm[19:12]		rd		opcode		funct3		jump target		op		funct3		jump target		op	

RISC-V Integer Base (RV32I/64I/128I), privileged, and optional compressed extension (RVC). Registers x1-x31 and the pc are 32 bits wide in RV32I, 64 in RV64I, and 128 in RV128I (x0=0). RV64I/128I add 10 instructions for the wider formats. The RVI base of <50 classic integer RISC instructions is required. Every 16-bit RVC instruction matches an existing 32-bit RVI instruction. See risc.org.

Optional Multiply-Divide Instruction Extension: RVM					
Category	Name	Fmt	RV32M (Multiply-Divide)	+RV{64,128}	
Multiply	MULTiply	R	MUL rd,rs1,rs2	MUL{W D} rd,rs1,rs2	
	MULTiply upper Half	R	MULH rd,rs1,rs2		
	MULTiply Half Sign/Uns	R	MULHSU rd,rs1,rs2		
	MULTiply upper Half Uns	R	MULHU rd,rs1,rs2		
Divide	DIVide	R	DIV rd,rs1,rs2	DIV{W D} rd,rs1,rs2	
	DIVide Unsigned	R	DIVU rd,rs1,rs2		
Remainder	REMAinder	R	REM rd,rs1,rs2	REM{W D} rd,rs1,rs2	
	REMAinder Unsigned	R	REMU rd,rs1,rs2	REMU{W D} rd,rs1,rs2	

Optional Atomic Instruction Extension: RVA					
Category	Name	Fmt	RV32A (Atomic)	+RV{64,128}	
Load	Load Reserved	R	LR.W rd,rs1	LR.{D Q} rd,rs1	
Store	Store Conditional	R	SC.W rd,rs1,rs2	SC.{D Q} rd,rs1,rs2	
Swap	SWAP	R	AMOSWAP.W rd,rs1,rs2	AMOSWAP.{D Q} rd,rs1,rs2	
Add	ADD	R	AMOADD.W rd,rs1,rs2	AMOADD.{D Q} rd,rs1,rs2	
Logical	XOR	R	AMOXOR.W rd,rs1,rs2	AMOXOR.{D Q} rd,rs1,rs2	
	AND	R	AMOAND.W rd,rs1,rs2	AMOAND.{D Q} rd,rs1,rs2	
	OR	R	AMOOR.W rd,rs1,rs2	AMOOR.{D Q} rd,rs1,rs2	
Min/Max	MINimum	R	AMOMIN.W rd,rs1,rs2	AMOMIN.{D Q} rd,rs1,rs2	
	MAXimum	R	AMOMAX.W rd,rs1,rs2	AMOMAX.{D Q} rd,rs1,rs2	
	MINimum Unsigned	R	AMOMINU.W rd,rs1,rs2	AMOMINU.{D Q} rd,rs1,rs2	
	MAXimum Unsigned	R	AMOMAXU.W rd,rs1,rs2	AMOMAXU.{D Q} rd,rs1,rs2	

Three Optional Floating-Point Instruction Extensions: RVF, RVD, & RVQ					
Category	Name	Fmt	RV32{F D Q} (HP/SP,DP,QP Fl Pt)	+RV{64,128}	
Move	Move from Integer	R	FMV.{H S}.X rd,rs1	FMV.{D Q}.X rd,rs1	
	Move to Integer	R	FMV.X.{H S} rd,rs1	FMV.X.{D Q} rd,rs1	
Convert	Convert from Int	R	FCVT.{H S D Q}.W rd,rs1	FCVT.{H S D Q}.{L T} rd,rs1	
	Convert from Int Unsigned	R	FCVT.{H S D Q}.WU rd,rs1	FCVT.{H S D Q}.{L T}U rd,rs1	
	Convert to Int	R	FCVT.W.{H S D Q} rd,rs1	FCVT.{L T}.{H S D Q} rd,rs1	
	Convert to Int Unsigned	R	FCVT.WU.{H S D Q} rd,rs1	FCVT.{L T}U.{H S D Q} rd,rs1	

RISC-V Calling Convention					
Register	ABI Name	Saver	Description		
x0	zero	---	Hard-wired zero		
x1	ra	Caller	Return address		
x2	sp	Callee	Stack pointer		
x3	gp	---	Global pointer		
x4	tp	---	Thread pointer		
x5-x7	t0-2	Caller	Temporaries		
x8	s0/fp	Callee	Saved register/frame pointer		
x9	s1	Callee	Saved register		
x10-x11	a0-1	Caller	Function arguments/return values		
x12-x17	a2-7	Caller	Function arguments		
x18-27	s2-11	Callee	Saved registers		
x28-31	t3-t6	Caller	Temporaries		
f0-7	ft0-7	Caller	FP temporaries		
f8-9	fs0-1	Callee	FP saved registers		
f10-11	fa0-1	Caller	FP arguments/return values		
f12-17	fa2-7	Caller	FP arguments		
f18-27	fs2-11	Callee	FP saved registers		
f28-31	ft8-11	Caller	FP temporaries		

RISC-V calling convention and five optional extensions: 10 multiply-divide instructions (RVM); 11 optional atomic instructions (RVA); and 25 floating-point instructions each for single-, double-, and quadruple-precision (RVF, RVD, RVQ). The latter add registers f0-f31, whose width matches the widest precision, and a floating-point control and status register fcsr. Each larger address adds some instructions: 4 for RVM, 11 for RVA, and 6 each for RVF/D/Q. Using regex notation, {} means set, so L{D|Q} is both LD and LQ. See riscv.org. (8/21/15 revision)