RISC-V Reference Card V0.2

Base format: Inst rd, rs1, rs2 (rd: destination register, rs1, rs2 source registers).

Other formats: Inst rs1, rs2
Inst rs1, immediate

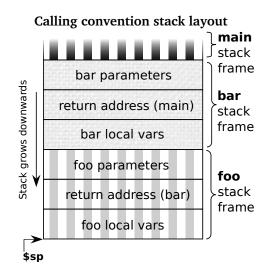
### **RV32I Base Integer Instructions**

<b>T</b> .	7.7	D 1 11 (C)
Inst	Name	Description (C)
add	ADD	rd = rs1 + rs2
sub	SUB	rd = rs1 - rs2
xor	XOR	$rd = rs1 ^rs2$
or	OR	rd = rs1   rs2
and	AND	rd = rs1 & rs2
sll	Shift Left Logical	$rd = rs1 \ll rs2$
srl	Shift Right Logical	$rd = rs1 \gg rs2$
sra	Shift Right Arith*	$rd = rs1 \gg rs2$
slt	Set Less Than	rd = (rs1 < rs2)?1:0
sltu	Set Less Than (U)	rd = (rs1 < rs2)?1:0
addi	ADD Immediate	rd = rs1 + imm
xori	XOR Immediate	rd = rs1 ^ imm
ori	OR Immediate	rd = rs1   imm
andi	AND Immediate	rd = rs1 & imm

### Registers

regioters			
Register	ABI Name	Description	Saver
x0	zero	Zero constant	_
x1	ra	Return address	Caller
x2	sp	Stack pointer	_
x3	gp	Global pointer	_
x4	tp	Thread pointer	Callee
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-x31	t3-t6	Temporaries	Caller

andi   A	AND Immediate	rd = r	s1 & imm	
slli	Shift Left Logical I	mm	rd = rs1 << im	m[0:4]
srli	Shift Right Logical Imm		rd = rs1 >> imm[0:4]	
srai	Shift Right Arith Imm		$rd = rs1 \gg imm[0:4]$	
slti	Set Less Than Imm		rd = (rs1 < irr	nm)?1:0
sltiu	Set Less Than Imm (U)		rd = (rs1 < irr	nm)?1:0
lb	Load Byte		rd = M[rs1+ir]	nm][0:7]
lh	Load Half		rd = M[rs1+ir]	nm][0:15]
lw	Load Word		rd = M[rs1+ir]	nm][0:31]
1bu	Load Byte (U)		rd = M[rs1+ir]	nm][0:7]
lhu	Load Half (U)		rd = M[rs1+ir]	nm][0:15]
sb	Store Byte		M[rs1+imm][	0:7] = rs2[0:7]
sh	Store Half		M[rs1+imm][0:15] = rs2[0:15]	
SW	Store Word		M[rs1+imm][0:31] = rs2[0:31]	
beq	Branch ==		if(rs1 == rs2)	PC += imm
bne	Branch !=		if(rs1 != rs2) PC += imm	
blt	Branch <		if(rs1 < rs2) PC += imm	
bge	Branch ≤		if(rs1 >= rs2) PC += imm	
bltu	Branch < (U)		if(rs1 < rs2)	PC += imm
bgeu	Branch $\geq$ (U)		if(rs1 >= rs2)	PC += imm
jal	Jump And Link		rd = PC+4; PC += imm	
jalr	Jump And Link Reg	g	rd = PC+4; PC	C = rs1 + imm
lui	Load Upper Imm		rd = imm << 1	2
auipc	Add Upper Imm to	PC	rd = PC + (im)	m << 12)
ecall	Environment Call		Transfer control to OS	
ebreak	Environment Break	ζ.	Transfer contro	ol to debugger



## **RV32M Multiply Extension**

KV32M Multiply Extension				
Inst	Name	Description (C)		
mul	MUL	rd = (rs1 * rs2)[31:0]		
mulh	MUL High	rd = (rs1 * rs2)[63:32]		
mulsu	MUL High (S) (U)	rd = (rs1 * rs2)[63:32]		
mulu	MUL High (U)	rd = (rs1 * rs2)[63:32]		
div	DIV	rd = rs1 / rs2		
divu	DIV (U)	rd = rs1 / rs2		
rem	Remainder	rd = rs1 % rs2		
remu	Remainder (U)	rd = rs1 % rs2		
	Inst mul mulh mulsu mulu div divu rem	Inst Name  mul MUL  mulh MUL High  mulsu MUL High (S) (U)  mulu MUL High (U)  div DIV  divu DIV (U)  rem Remainder		

C compiler datatype sizes

C compiler datatype sizes			
C type	Description	Bytes in RV32	
char	Character value/byte	1	
short	Short integer	2	
int	Integer	4	
long	Long integer	4	
long long	Long long integer	8	
void*	Pointer	4	
float	Single-precision float	4	
double	Double-precision float	8	
long double	Extended-precision float	16	

RISC-V Reference Card V0.2

## **Pseudo Instructions**

auipc rd, symbol[31:12] addi rd, rd, symbol[11:0] s{b h w d} rd, symbol stipe rd, symbol[11:0] s{b h w d} rd, symbol rt, symbol[31:12] l{b h w d} rd, symbol auipc rt, symbol[11:0](rd) auipc rt, symbol[11:0](rd) auipc rt, symbol[11:0](rd) auipc rt, symbol[11:0](rt) store global  No operation Load immediate Myriad sequences wv rd, rs addi rd, rs, 0 Copy register not rd, rs xori rd, rs, -1 One's complement neg rd, rs subr rd, x0, rs Two's complement neg rd, rs subw rd, x0, rs Two's complement sext. w rd, rs addiw rd, rs, 0 Sign extend word seqz rd, rs sltu rd, x0, rs Set if ≠ zero snez rd, rs sltu rd, x0, rs Set if ≠ zero snez rd, rs slt rd, rs, x0 Set if < zero sgtz rd, rs slt rd, x0, rs Set if > zero beqz rs, offset beq rs, x0, offset Branch if ≡ zero blez rs, offset bge x0, rs, offset Branch if ≡ zero blez rs, offset bge rs, x0, offset Branch if ≥ zero blez rs, offset bge rs, x0, offset Branch if ≥ zero blez rs, offset blt rs, x0 offset Branch if ≥ zero bltz rs, offset blt rs, x0, offset Branch if ≥ zero bltz rs, offset blt rs, x0, offset Branch if ≥ zero bgtz rs, offset blt rs, x0, offset Branch if ≥ zero bgtz rs, offset blt rs, x0, offset Branch if ≥ zero bgtz rs, offset blt rs, x0, offset Branch if ≥ zero bgtz rs, offset blt rs, x0, offset Branch if ≥ zero bgtz rs, offset blt rs, x0, offset Branch if ≥ zero bgtz rs, offset blt rs, x0, offset Branch if ≥ zero bgt rs, rt, offset blt rr, rs, offset Branch if ≤ zero bgt rs, rt, offset blt rr, rs, offset Branch if ≤ zero bgt rs, rt, offset blt rr, rs, offset Branch if ≤ zero bgt rs, rt, offset blt rr, rs, offset Branch if ≤ zero bgt rs, rt, offset blt rs, x0, offset Branch if ≤ zero bgt rs, rt, offset blt rs, x0, offset Branch if ≤ zero bgt rs, rt, offset blt rs, x0, offset Branch if ≤ zero bgt rs, rt, offset blt rs, x0, offset Branch if ≤ zero bgt rs, rt, offset blt rs, x0, offset Branch if ≤ zero bgt rs, rt, offset blt rs, x0, offset Branch if ≤ zero bgt rs, rt, offset blt rs, x0, offset Branch if ≤ zero bgt rs, rt, offset blt rs, x0, offset Branch if ≤ zero b	Pseudoinstruction	Base Instruction(s)	Meaning
	la rd, symbol		Load address
s(b n w d) rd, symbol, rt  nop     addi x0, x0, 0     No operation li rd, immediate     Myriad sequences     wr rd, rs     addi rd, rs, 0     not rd, rs     not rd, rs     subr rd, x0, rs     regw rd, rs     subr rd, x0, rs     sext.w rd, rs     sext.f rd, rs     sext.w rd, rs	l{b h w d} rd, symbol	· · · · · · · · · · · · · · · · · · ·	Load global
li rd, immediate $mv rd$ , rs addi rd, rs, 0 Copy register $mv rd$ , rs addi rd, rs, 0 Copy register $mv rd$ , rs $m$	s{b h w d} rd, symbol, rt	· · · · · · · · · · · · · · · · · · ·	Store global
mv rd, rs not rd, rs not rd, rs neg rd, rs sub rd, x0, rs sub rd, x0, rs rtwo's complement regw rd, rs subw rd, x0, rs sext.w rd, rs sext.w rd, rs set if = zero set if = zero set if >	nop	addi x0, x0, 0	No operation
not rd, rs	li rd, immediate	Myriad sequences	Load immediate
neg rd, rs negw rd, rs subw rd, x0, rs sext.w rd, rs addiw rd, rs, 0 sign extend word seqz rd, rs subt rd, rs, 1 set if = zero sltz rd, rs sltu rd, x0, rs set if ≠ zero sltz rd, rs slt rd, rs, x0 set if < zero set if	mv rd, rs	addi rd, rs, 0	Copy register
negw rd, rs subw rd, x0, rs Two's complement word sext.w rd, rs addiw rd, rs, 0 Sign extend word seqz rd, rs sltu rd, x0, rs Set if = zero snez rd, rs sltu rd, x0, rs Set if $\neq$ zero sltz rd, rs slt rd, rs, x0 Set if $<$ zero sgtz rd, rs slt rd, x0, rs Set if $<$ zero sgtz rd, rs slt rd, x0, rs Set if $<$ zero sgtz rd, rs slt rd, x0, rs Set if $<$ zero specification of the property of the prope	not rd, rs	xori rd, rs, −1	One's complement
sext.w rd, rs seqz rd, rs seqz rd, rs sltiu rd, rs, 1 Set if = zero snez rd, rs sltu rd, x0, rs Set if ≠ zero sltz rd, rs slt rd, rs, x0 Set if < zero sgtz rd, rs slt rd, x0, rs Set if > zero sgtz rd, rs slt rd, x0, rs Set if > zero sgtz rd, rs slt rd, x0, rs Set if > zero sgtz rd, rs slt rd, x0, rs Set if > zero sgtz rd, rs slt rd, x0, rs Set if > zero sgtz rd, rs slt rd, x0, rs Set if > zero sgtz rd, rs slt rd, x0, rs Set if > zero Set if > zero sgtz rd, rs slt rd, x0, rs Set if > zero sgtz rd, rs slt rd, x0, rs Set if > zero Set if > zero sgtz rd, rs slt rd, x0, rs Set if > zero set if > zero sgtz rd, rs slt rd, x0, rs Set if > zero s	neg rd, rs	sub rd, x0, rs	Two's complement
seqz rd, rs sltiu rd, rs, 1 Set if = zero snez rd, rs sltu rd, x0, rs Set if $\neq$ zero sltz rd, rs slt rd, rs, x0 Set if $\neq$ zero sgtz rd, rs slt rd, x0, rs Set if $\neq$ zero sgtz rd, rs slt rd, x0, rs Set if $\neq$ zero beqz rs, offset beq rs, x0, offset Branch if = zero blez rs, offset bge x0, rs, offset Branch if $\neq$ zero blez rs, offset bge x0, rs, offset Branch if $\neq$ zero blez rs, offset bge rs, x0, offset Branch if $\neq$ zero bltz rs, offset blt rs, x0, offset Branch if $\neq$ zero bltz rs, offset blt x0, rs, offset Branch if $\neq$ zero bgtz rs, offset blt x0, rs, offset Branch if $\neq$ zero bgtz rs, offset blt x0, rs, offset Branch if $\neq$ zero bgt rs, rt, offset blt rt, rs, offset Branch if $\neq$ ble rs, rt, offset bge rt, rs, offset Branch if $\neq$ blu rt, rs, offset Branch if $\neq$ blu rt, rs, offset Branch if $\neq$ blu rs, rt, offset bgeu rt, rs, offset Branch if $\neq$ unsigned bleu rs, rt, offset bgeu rt, rs, offset Branch if $\neq$ unsigned j offset jal x0, offset Jump jal offset jal x1, offset Jump and link jr rs jalr x0, rs, 0 Jump and link register ret jalr x0, x1, 0 Return from subroutine call offset jal x1, offset[31:12] jalr x1, x1, offset[11:0] Call far-away subroutine tail offset jalr x0, x6, offset[11:0]	negw rd, rs	subw rd, x0, rs	Two's complement word
snez rd, rs sltu rd, x0, rs Set if $\neq$ zero sltz rd, rs slt rd, rs, x0 Set if $<$ zero sgtz rd, rs slt rd, x0, rs Set if $<$ zero sgtz rd, rs slt rd, x0, rs Set if $<$ zero beqz rs, offset beq rs, x0, offset Branch if $=$ zero blez rs, offset bge x0, rs, offset Branch if $\leq$ zero bgez rs, offset bge rs, x0, offset Branch if $\leq$ zero bgez rs, offset blt rs, x0, offset Branch if $\leq$ zero bltz rs, offset blt rs, x0, offset Branch if $\leq$ zero bgtz rs, offset blt x0, rs, offset Branch if $>$ zero bgt rs, rt, offset blt rt, rs, offset Branch if $>$ blt rt, rs, offset Branch if $\leq$ blt rs, rt, offset bge rt, rs, offset Branch if $>$ blt rt, rs, offset Branch if $>$ unsigned bleu rs, rt, offset bgeu rt, rs, offset Branch if $>$ unsigned bleu rs, rt, offset bgeu rt, rs, offset Branch if $\leq$ unsigned $=$ j offset jal x0, offset Jump jal offset jal x1, offset Jump and link jr rs jalr x0, rs, 0 Jump and link register ret jalr x0, x1, 0 Return from subroutine call offset auipc x1, offset[31:12] jalr x1, x1, offset[11:0] Tail call far-away subroutine	sext.w rd, rs	addiw rd, rs, 0	Sign extend word
sltz rd, rs slt rd, rs, x0 Set if < zero sgtz rd, rs slt rd, x0, rs Set if > zero beqz rs, offset beq rs, x0, offset Branch if = zero bnez rs, offset bne rs, x0, offset Branch if $\neq$ zero blez rs, offset bge x0, rs, offset Branch if $\leq$ zero bgez rs, offset bge rs, x0, offset Branch if $\leq$ zero bgez rs, offset blt rs, x0, offset Branch if $\leq$ zero bltz rs, offset blt rs, x0, offset Branch if $\leq$ zero bgtz rs, offset blt x0, rs, offset Branch if $\geq$ zero bgt rs, rt, offset blt rt, rs, offset Branch if $\geq$ begt rs, rt, offset bge rt, rs, offset Branch if $\leq$ bgtu rs, rt, offset bge rt, rs, offset Branch if $\leq$ bgtu rs, rt, offset bltu rt, rs, offset Branch if $\leq$ unsigned bleu rs, rt, offset bgeu rt, rs, offset Branch if $\leq$ unsigned $\leq$ j offset jal x0, offset Jump jal offset jal x1, offset Jump and link jr rs jalr x0, rs, 0 Jump and link register ret jalr x0, x1, 0 Return from subroutine call offset auipc x1, offset[31:12] jalr x1, x1, offset[11:0] Tail call far-away subroutine	seqz rd, rs	sltiu rd, rs, 1	Set if $=$ zero
sgtz rd, rsslt rd, x0, rsSet if > zerobeqz rs, offsetbeq rs, x0, offsetBranch if = zerobnez rs, offsetbne rs, x0, offsetBranch if $\neq$ zeroblez rs, offsetbge x0, rs, offsetBranch if $\leq$ zerobgez rs, offsetbge rs, x0, offsetBranch if $\geq$ zerobltz rs, offsetblt rs, x0, offsetBranch if $<$ zerobgtz rs, offsetblt x0, rs, offsetBranch if $>$ zerobgt rs, rt, offsetblt rt, rs, offsetBranch if $>$ ble rs, rt, offsetble rs, rt, offsetbge rt, rs, offsetBranch if $>$ unsignedbleu rs, rt, offsetbltu rt, rs, offsetBranch if $>$ unsignedbleu rs, rt, offsetbgeu rt, rs, offsetBranch if $>$ unsignedj offsetjal x0, offsetJumpjal offsetjal x1, offsetJump and linkjr rsjalr x0, rs, 0Jump and link registerretjalr x0, x1, 0Return from subroutinecall offsetauipc x1, offset[31:12] jalr x1, x1, offset[11:0]Call far-away subroutinetail offsetauipc x6, offset[31:12] jalr x0, x6, offset[11:0]Tail call far-away subroutine	snez rd, rs	sltu rd, x0, rs	Set if $\neq$ zero
beqz rs, offset beq rs, x0, offset Branch if = zero bnez rs, offset bne rs, x0, offset Branch if $\neq$ zero blez rs, offset bge x0, rs, offset Branch if $\leq$ zero bgez rs, offset bge rs, x0, offset Branch if $\leq$ zero bgtz rs, offset blt rs, x0, offset Branch if $\leq$ zero bgtz rs, offset blt x0, rs, offset Branch if $\leq$ zero bgt rs, rt, offset blt x0, rs, offset Branch if $>$ zero bgt rs, rt, offset blt rt, rs, offset Branch if $>$ ble rs, rt, offset bge rt, rs, offset Branch if $\leq$ bgtu rs, rt, offset bge rt, rs, offset Branch if $\leq$ bgtu rs, rt, offset bltu rt, rs, offset Branch if $\leq$ bgtu rs, rt, offset bgeu rt, rs, offset Branch if $\leq$ , unsigned bleu rs, rt, offset bgeu rt, rs, offset Branch if $\leq$ , unsigned j offset jal x0, offset Jump jal offset jal x1, offset Jump and link jr rs jalr x0, rs, 0 Jump register jalr rs jalr x0, rs, 0 Jump and link register ret jalr x0, x1, 0 Return from subroutine call offset auipc x1, offset[31:12] jalr x1, x1, offset[11:0] Tail call far-away subroutine tail offset jalr x0, x6, offset[11:0]	sltz rd, rs	slt rd, rs, x0	Set if < zero
bnez rs, offset bne rs, x0, offset Branch if $\neq$ zero blez rs, offset bge x0, rs, offset Branch if $\leq$ zero bgez rs, offset bge rs, x0, offset Branch if $\leq$ zero bltz rs, offset blt rs, x0, offset Branch if $\leq$ zero bgtz rs, offset blt x0, rs, offset Branch if $>$ zero bgt rs, rt, offset blt rt, rs, offset Branch if $>$ blt rt, rs, offset Branch if $>$ ble rs, rt, offset bge rt, rs, offset Branch if $>$ bge rt, rs, offset Branch if $>$ blu rs, rt, offset blu rt, rs, offset Branch if $>$ unsigned bleu rs, rt, offset bgeu rt, rs, offset Branch if $\leq$ , unsigned bleu rs, rt, offset bgeu rt, rs, offset Branch if $\leq$ , unsigned j offset jal x0, offset Jump jal offset jal x1, offset Jump and link jr rs jalr x0, rs, 0 Jump register jalr rs ret jalr x0, x1, 0 Return from subroutine call offset jalr x1, x1, offset[31:12] jalr x1, x1, offset[31:12] Call far-away subroutine tail offset jalr x0, x6, offset[31:12] Tail call far-away subroutine	sgtz rd, rs	slt rd, x0, rs	Set if > zero
blez rs, offset bge x0, rs, offset Branch if $\leq$ zero bgez rs, offset bge rs, x0, offset Branch if $\geq$ zero bltz rs, offset blt rs, x0, offset Branch if $<$ zero bgtz rs, offset blt x0, rs, offset Branch if $>$ zero bgt rs, rt, offset blt rt, rs, offset Branch if $>$ blt rt, rs, offset Branch if $>$ blt rt, rs, offset Branch if $\leq$ bgtu rs, rt, offset bge rt, rs, offset Branch if $\leq$ bgtu rs, rt, offset bltu rt, rs, offset Branch if $\leq$ nusigned bleu rs, rt, offset bgeu rt, rs, offset Branch if $\leq$ , unsigned bleu rs, rt, offset bgeu rt, rs, offset Branch if $\leq$ , unsigned bleu rs, rt, offset bgeu rt, rs, offset Branch if $\leq$ , unsigned bleu rs, rt, offset bgeu rt, rs, offset Branch if $\leq$ , unsigned bleu rs, rt, offset bgeu rt, rs, offset Branch if $\leq$ nusigned bleu rs, rt, offset bgeu rt, rs, offset Branch if $\leq$ nusigned bleu rs, rt, offset Branch if $\leq$ nusigned bleu rs, rt, offset bgeu rt, rs, offset Branch if $\leq$ nusigned bleu rs, rt, offset Branch if $\leq$ nusigned bleu rs, rt, offset Branch if $\leq$ nusigned bleu rs, rt, offset Branch if $\leq$ nusigned Branch if $\geq$ nusigned Branch if $\leq$	beqz rs, offset	beq rs, x0, offset	Branch if = zero
bgez rs, offset bge rs, x0, offset Branch if $\geq$ zero bltz rs, offset blt rs, x0, offset Branch if $<$ zero bgtz rs, offset blt x0, rs, offset Branch if $>$ zero bgt rs, rt, offset blt rt, rs, offset Branch if $>$ blt rt, rs, offset Branch if $>$ blt rt, rs, offset Branch if $<$ bge rt, rs, offset Branch if $<$ bgtu rs, rt, offset bltu rt, rs, offset Branch if $>$ unsigned bleu rs, rt, offset bgeu rt, rs, offset Branch if $<$ , unsigned bleu rs, rt, offset bgeu rt, rs, offset Branch if $<$ , unsigned $<$ j offset jal x0, offset Jump jal offset jal x1, offset Jump and link jr rs jalr x0, rs, 0 Jump register jalr rs jalr x1, rs, 0 Jump and link register ret jalr x0, x1, 0 Return from subroutine call offset auipc x1, offset[31:12] jalr x1, x1, offset[11:0] Tail call far-away subroutine tail offset jalr x0, x6, offset[11:0]	bnez rs, offset	bne rs, x0, offset	Branch if $\neq$ zero
bltz rs, offset blt rs, x0, offset Branch if < zero bgtz rs, offset blt x0, rs, offset Branch if > zero  bgt rs, rt, offset blt rt, rs, offset Branch if > ble rs, rt, offset bge rt, rs, offset Branch if ≤ bgtu rs, rt, offset bltu rt, rs, offset Branch if ≤ bltu rt, rs, offset Branch if >, unsigned bleu rs, rt, offset bgeu rt, rs, offset Branch if ≤, unsigned  j offset jal x0, offset Jump jal offset jal x1, offset Jump and link jr rs jalr x0, rs, 0 Jump register jalr rs jalr x1, rs, 0 Jump and link register ret jalr x0, x1, 0 Return from subroutine  call offset jalr x1, x1, offset[31:12] jalr x1, x1, offset[11:0] Tail call far-away subroutine  tail offset jalr x0, x6, offset[11:0]	blez rs, offset	bge x0, rs, offset	Branch if $\leq$ zero
bgtz rs, offset blt x0, rs, offset Branch if > zero  bgt rs, rt, offset blt rt, rs, offset Branch if > ble rs, rt, offset bge rt, rs, offset Branch if ≤ bgtu rs, rt, offset bltu rt, rs, offset Branch if >, unsigned bleu rs, rt, offset bgeu rt, rs, offset Branch if ≤, unsigned  j offset jal x0, offset Jump jal offset jal x1, offset Jump and link jr rs jalr x0, rs, 0 Jump register jalr rs jalr x1, rs, 0 Jump and link register ret jalr x0, x1, 0 Return from subroutine  call offset jalr x1, x1, offset[31:12] jalr x1, x1, offset[31:12] jalr x0, x6, offset[31:12] Tail call far-away subroutine		bge rs, x0, offset	Branch if $\geq$ zero
bgt rs, rt, offset blt rt, rs, offset Branch if > ble rs, rt, offset bge rt, rs, offset Branch if ≤ bgtu rs, rt, offset bltu rt, rs, offset Branch if >, unsigned bleu rs, rt, offset bgeu rt, rs, offset Branch if >, unsigned bleu rs, rt, offset bgeu rt, rs, offset Branch if ≤, unsigned  j offset jal x0, offset Jump jal offset jal x1, offset Jump and link jr rs jalr x0, rs, 0 Jump register jalr rs jalr x1, rs, 0 Jump and link register ret jalr x0, x1, 0 Return from subroutine  call offset auipc x1, offset[31:12] jalr x1, x1, offset[11:0] Call far-away subroutine  tail offset jalr x0, x6, offset[11:0] Tail call far-away subroutine	bltz rs, offset	blt rs, x0, offset	Branch if < zero
ble rs, rt, offset bge rt, rs, offset Branch if $\leq$ bgtu rs, rt, offset blu rt, rs, offset Branch if $>$ , unsigned bleu rs, rt, offset bgeu rt, rs, offset Branch if $\leq$ , unsigned j offset jal x0, offset Jump jal offset jal x1, offset Jump and link jr rs jalr x0, rs, 0 Jump register jalr rs jalr x1, rs, 0 Jump and link register ret jalr x0, x1, 0 Return from subroutine call offset jalr x1, x1, offset[31:12] jalr x1, x1, offset[31:12] jalr x0, x6, offset[31:12] Tail call far-away subroutine	bgtz rs, offset	blt x0, rs, offset	Branch if > zero
bgtu rs, rt, offset bltu rt, rs, offset Branch if >, unsigned bleu rs, rt, offset bgeu rt, rs, offset Branch if ≤, unsigned j offset jal x0, offset Jump jal offset jal x1, offset Jump and link jr rs jalr x0, rs, 0 Jump register jalr rs jalr x1, rs, 0 Jump and link register ret jalr x0, x1, 0 Return from subroutine auipc x1, offset[31:12] jalr x1, x1, offset[11:0] Call far-away subroutine tail offset jalr x0, x6, offset[31:12] Tail call far-away subroutine	bgt rs, rt, offset	blt rt, rs, offset	Branch if >
bleu rs, rt, offset bgeu rt, rs, offset Branch if $\leq$ , unsigned j offset jal x0, offset Jump jal offset jal x1, offset Jump and link jr rs jalr x0, rs, 0 Jump register jalr rs jalr x1, rs, 0 Jump and link register ret jalr x0, x1, 0 Return from subroutine call offset auipc x1, offset[31:12] jalr x1, x1, offset[11:0] Call far-away subroutine tail offset jalr x0, x6, offset[11:0] Tail call far-away subroutine	ble rs, rt, offset	bge rt, rs, offset	Branch if $\leq$
j offset jal x0, offset Jump  jal offset jal x1, offset Jump and link  jr rs jalr x0, rs, 0 Jump register  jalr rs jalr x1, rs, 0 Jump and link register  ret jalr x0, x1, 0 Return from subroutine  call offset auipc x1, offset[31:12]  jalr x1, x1, offset[31:12]  tail offset jalr x0, x6, offset[31:12]  Tail call far-away subroutine	bgtu rs, rt, offset	bltu rt, rs, offset	
jal offsetjal x1, offsetJump and linkjr rsjalr x0, rs, 0Jump registerjalr rsjalr x1, rs, 0Jump and link registerretjalr x0, x1, 0Return from subroutinecall offsetauipc x1, offset[31:12] jalr x1, x1, offset[11:0]Call far-away subroutinetail offsetauipc x6, offset[31:12] jalr x0, x6, offset[11:0]Tail call far-away subroutine	bleu rs, rt, offset	bgeu rt, rs, offset	Branch if $\leq$ , unsigned
jr rsjalr x0, rs, 0Jump registerjalr rsjalr x1, rs, 0Jump and link registerretjalr x0, x1, 0Return from subroutinecall offsetauipc x1, offset[31:12] jalr x1, x1, offset[11:0]Call far-away subroutinetail offsetauipc x6, offset[31:12] jalr x0, x6, offset[11:0]Tail call far-away subroutine	j offset	jal x0, offset	Jump
jalr rsjalr x1, rs, 0Jump and link registerretjalr x0, x1, 0Return from subroutinecall offsetauipc x1, offset[31:12] jalr x1, x1, offset[11:0]Call far-away subroutinetail offsetauipc x6, offset[31:12] jalr x0, x6, offset[11:0]Tail call far-away subroutine	jal offset	jal x1, offset	Jump and link
ret jalr x0, x1, 0 Return from subroutine  call offset auipc x1, offset[31:12] jalr x1, x1, offset[11:0]  tail offset auipc x6, offset[31:12] jalr x0, x6, offset[11:0]  Tail call far-away subroutine	jr rs	jalr x0, rs, 0	
call offset auipc x1, offset[31:12] auipc x1, x1, offset[11:0] Call far-away subroutine tail offset auipc x6, offset[31:12] auipc x6, offset[31:12] jalr x0, x6, offset[11:0] Tail call far-away subroutine	jalr rs	jalr x1, rs, 0	Jump and link register
tail offset jalr x1, x1, offset[11:0]  tail offset jalr x0, x6, offset[11:0]  Tail call far-away subroutine	ret	jalr x0, x1, 0	Return from subroutine
tail offset auipc x6, offset[31:12] auipc x6, offset[11:0] Tail call far-away subroutine	call offset		Call far-away subroutine
fence fence iorw, iorw Fence on all memory and I/O	tail offset	auipc x6, offset[31:12]	Tail call far-away subroutine
	fence	fence iorw, iorw	Fence on all memory and I/O

# Control and status registers (CSRs)

Example usage	Description	←Modify CSR registers		
	Atomic Read/Clear CSR: read from the		· ·	
csrrc t0, fcsr, t1	CSR into t0 and clear bits of the CSR			
	according to t1			CSR register description $\downarrow$
	Atomic Read/Clear CSR Immediate:	Number	Name	Description
csrrci t0, fcsr, 10	read from the CSR into t0 and clear bits	User trap setup		
	of the CSR according to a constant	0x000	ustatus	User status register.
	Atomic Read/Set CSR: read from the	0x004	uie	User interrupt-enable register
csrrs t0, fcsr, t1	CSR into t0 and logical or t1 into the	0x005	utvec	User trap handler base address
	CSR	User trap handling		
	Atomic Read/Set CSR Immediate: read	0x040	uscratch	Scratch reg. for user handlers
csrrsi t0, fcsr, 10	from the CSR into t0 and logical or a	0x041	uepc	User exception program counter
	constant into the CSR	0x042	ucause	User trap cause
	A. I. D. LOWY. COD. 16 .1	0x043	utval	User bad address/instruction
csrrw t0, fcsr, t1	Atomic Read/Write CSR: read from the	0x044	uip	User interrupt pending
, ,	CSR into t0 and write t1 into the CSR			
	Atomic Read/Write CSR Immediate:	1		
csrrwi t0, fcsr, 10	read from the CSR into t0 and write a			
, ,	constant into the CSR			