## Chapter 20

## RISC-V Assembly Programmer's Handbook

This chapter is a placeholder for an assembly programmer's manual.

Table 20.1 lists the assembler mnemonics for the x and f registers and their role in the standard calling convention.

Register	ABI Name	Description	Saver
х0	zero	Hard-wired zero	
x1	ra	Return address	Caller
x2	sp	Stack pointer	Callee
х3	gp	Global pointer	—
x4	tp	Thread pointer	—
x5	t0	Temporary/alternate link register	Caller
x6-7	t1-2	Temporaries	Caller
x8	s0/fp	Saved register/frame pointer	Callee
x9	s1	Saved register	Callee
x10-11	a0-1	Function arguments/return values	Caller
x12-17	a2-7	Function arguments	Caller
x18-27	s2-11	Saved registers	Callee
x28-31	t3-6	Temporaries	Caller
f0-7	ft0-7	FP temporaries	Caller
f8-9	fs0-1	FP saved registers	Callee
f10-11	fa0-1	FP arguments/return values	Caller
f12-17	fa2-7	FP arguments	Caller
f18-27	fs2-11	FP saved registers	Callee
f28-31	ft8-11	FP temporaries	Caller

Table 20.1: Assembler mnemonics for RISC-V integer and floating-point registers.

Tables 20.2 and 20.3 contain a listing of standard RISC-V pseudoinstructions.

auipc rd, symbol [31:12] addir dr, dr, symbol[[11:0]] s{b h w d} rd, symbol auipc rd, symbol[[31:12]] t{b h w d} rd, symbol, rt fl{w d} rd, symbol, rt fs{w d} rd, symbol, rd fs{w d} rd, symbol, rt fs{w d}	Pseudoinstruction	Base Instruction(s)	Meaning
	la rd, symbol		Load address
Solinwid   Td., symbol, rt   Solinwid   Td., symbol [11:0] (rt)   auipc rt, symbol [31:12]   fi   wild   rd., symbol [31:12]   fi   wild   rd., symbol [11:0] (rt)   auipc rt, symbol [31:12]   fo   wild   rd., symbol [11:0] (rt)   auipc rt, symbol [31:12]   fo   wild   rd., symbol [11:0] (rt)   rop   add x0, x0, 0	$l\{b h w d\}$ rd, symbol	<u>-</u>	Load global
fl{vld} rd, symbol, rt fl{vld} rd, symbol[11:0](rt) autpc rt, symbol[11:0](rt) autpc rt, symbol[11:0](rt) fs{vld} rd, symbol[11:0](rt)  nop li rd, immediate mv rd, rs not rd, rs neg rd, rs neg rd, rs subr rd, x0, rs negw rd, rs sext.w rd, rs sext.w rd, rs sext.w rd, rs sext.w rd, rs sext.vrd, rs sext.vr	$s\{b h w d\}$ rd, symbol, rt	- *	Store global
Taylor	$fl\{w d\}$ rd, symbol, rt	$fl\{w d\}$ rd, symbol[11:0](rt)	Floating-point load global
li rd, immediate mv rd, rs addi rd, rs, 0 not rd, rs neg rd, rs sub rd, x0, rs reg rd, rs sub rd, x0, rs sext.w rd, rs sub rd, x0, rs sext.w rd, rs subw rd, x0, rs Sext.w rd Sign extend word Set fe zero Set if zero Set if zero Set if zero Set if zero Sop single-precision register Double-precision register Single-precision absolute value Single-precision absolute value Single-precision absolute value Single-precision absolute Single-precision absolute Single-precision	<pre>fs{w d} rd, symbol, rt</pre>	$fs\{w d\}$ rd, $symbol[11:0](rt)$	Floating-point store global
mv rd, rs  not rd, rs  not rd, rs  neg rd, rs  sub rd, x0, rs  regy rd, rs  subw rd, x0, rs  sext.w rd, rs  sext.w ref.xex  Sextife zero  Sextife zero  Single-precision absolute value  Double-precision register  Copy double-precision register  B	nop	addi x0, x0, 0	No operation
not rd, rs neg rd, rs neg rd, rs sub rd, x0, rs sub rd, x0, rs Two's complement Sign extend word Sign extend word Sign extend word Seq rd, rs seq rd, rs sltu rd, rs, 1 Set if = zero Set if ≥ zero Set if > ze	li rd, immediate	$Myriad\ sequences$	Load immediate
neg rd, rs         sub rd, x0, rs         Two's complement           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 ≠ zero           sltz rd, rs         slt rd, x0, rs         Set if > zero           fmv.s rd, rs         fsgnj.s rd, rs, rs         Copy single-precision register           fabs.s rd, rs         fsgnjx.s rd, rs, rs         Single-precision absolute value           fneg.s rd, rs         fsgnjx.s rd, rs, rs         Single-precision absolute value           fneg.s rd, rs         fsgnjx.s rd, rs, rs         Copy double-precision register           fabs.d rd, rs         fsgnjx.d rd, rs, rs         Copy double-precision absolute value           fneg.s rd, rs         fsgnjx.d rd, rs, rs         Double-precision absolute value           fneg.d rd, rs         fsgnjx.d rd, rs, rs         Double-precision negate           bed rs         s0, offset         Branch if = zero           bed rs         s0, offset         Branch if = zero           blez rs, offset         bge x0, rs, offset         Branch if ≤ zero           bgz rs, offset         bgr rs, x0, offset	mv rd, rs	addi rd, rs, 0	Copy register
negw rd, rs subw rd, x0, rs addiw rd, rs, 0 Sign extend word sext.w rd, rs subw rd, rs, 0 Sign extend word seqz rd, rs subw rd, x0, rs Set if = zero segz rd, rs subw rd, x0, rs Set if $\neq$ zero subw rd, rs, x0 Set if $\neq$ zero subw rd, rs subw rd, rs, x0 Set if $\neq$ zero segz rd, rs subw rd, x0, rs Set if $\neq$ zero segz rd, rs subw rd, x0, rs Set if $\neq$ zero segz rd, rs subw rd, x0, rs Set if $\neq$ zero segz rd, rs subw rd, x0, rs Set if $\neq$ zero segz rd, rs subw rd, x0, rs Set if $\neq$ zero segz rd, rs subw rd, x0, rs Set if $\neq$ zero segz rd, rs subw rd, x0, rs Set if $\neq$ zero segz rd, rs subw rd, x0, rs Set if $\neq$ zero segz rd, rs subw rd, x0, rs subw rd, x1, x2, x2, x3, x4, x5, x5, x5, x6, x5, x6, x5, x6, x5, x7, x5, x5, x6, x5, x7, x5, x5, x5, x6, x5, x7, x5, x5, x5, x5, x5, x5, x5, x5, x5, x5	not rd, rs	xori rd, rs, -1	One's complement
negw rd, rs subw rd, x0, rs addiw rd, rs, 0 Sign extend word sext.w rd, rs subw rd, rs, 0 Sign extend word seqz rd, rs subw rd, x0, rs Set if = zero segz rd, rs subw rd, x0, rs Set if $\neq$ zero subw rd, rs, x0 Set if $\neq$ zero subw rd, rs subw rd, rs, x0 Set if $\neq$ zero segz rd, rs subw rd, x0, rs Set if $\neq$ zero segz rd, rs subw rd, x0, rs Set if $\neq$ zero segz rd, rs subw rd, x0, rs Set if $\neq$ zero segz rd, rs subw rd, x0, rs Set if $\neq$ zero segz rd, rs subw rd, x0, rs Set if $\neq$ zero segz rd, rs subw rd, x0, rs Set if $\neq$ zero segz rd, rs subw rd, x0, rs Set if $\neq$ zero segz rd, rs subw rd, x0, rs Set if $\neq$ zero segz rd, rs subw rd, x0, rs subw rd, x1, x2, x2, x3, x4, x5, x5, x5, x6, x5, x6, x5, x6, x5, x7, x5, x5, x6, x5, x7, x5, x5, x5, x6, x5, x7, x5, x5, x5, x5, x5, x5, x5, x5, x5, x5	neg rd, rs	sub rd, x0, rs	Two's complement
sext.w rd, rs         addiw rd, rs, 0         Sign extend word           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           fmv.s rd, rs         fsgnj.s rd, rs, rs         Copy single-precision register           fabs.s rd, rs         fsgnjx.s rd, rs, rs         Single-precision absolute value           fneg.s rd, rs         fsgnjn.d rd, rs, rs         Copy double-precision register           fabs.d rd, rs         fsgnjx.d rd, rs, rs         Copy double-precision negate           fmv.d rd, rs         fsgnjx.d rd, rs, rs         Double-precision absolute value           fneg.d rd, rs         fsgnjx.d rd, rs, rs         Double-precision absolute value           fneg.d rd, rs         fsgnjx.d rd, rs, rs         Double-precision negate           bed rx         offset fsgnjx.d rd, rs, rs         Double-precision absolute value           fneg.d rd, rs         fsgnjx.d rd, rs, rs         Double-precision negate           bed rx         offset         Branch if = zero           bez rs, offset         beg rs, v0, offset         Branch if ≤ zero           bgz rs, offset         bgz r	_		
seqz rd, rs         sltiu rd, xo, rs         Set if ≠ zero           snez rd, rs         sltu rd, xo, rs         Set if ≠ zero           sltz rd, rs         slt rd, rs, xo         Set if > zero           sgtz rd, rs         slt rd, xo, rs         Set if > zero           fmv.s rd, rs         fsgnj.s rd, rs, rs         Copy single-precision register           fabs.s rd, rs         fsgnjx.s rd, rs, rs         Single-precision absolute value           fneg.s rd, rs         fsgnjx.s rd, rs, rs         Single-precision negate           fmv.d rd, rs         fsgnjx.d rd, rs, rs         Copy double-precision register           fabs.d rd, rs         fsgnjx.d rd, rs, rs         Double-precision absolute value           fneg.d rd, rs         fsgnjx.d rd, rs, rs         Copy double-precision register           fabs.d rd, rs         fsgnjx.d rd, rs, rs         Double-precision absolute value           fneg.d rd, rs         fsgnjx.d rd, rs, rs         Copy double-precision negate           fmv.d rd, rs         fsgnjx.d rd, rs, rs         Double-precision absolute value           fneg.d rd, rs         fsgnjx.d rd, rs, rs         Copy double-precision negate           bed rd, rs         fsgnjx.d rd, rs, rs         Double-precision absolute value           psero         fsergix.d         rs, rs         Branch if > zero <td>_</td> <td></td> <td>-</td>	_		-
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 fmv.s rd, rs fsgnj.s rd, rs, rs Copy single-precision register fabs.s rd, rs fsgnjx.s rd, rs, rs Single-precision absolute value fneg.s rd, rs fsgnj.s rd, rs, rs Single-precision negate fmv.d rd, rs fsgnj.d rd, rs, rs Copy double-precision absolute value fneg.s rd, rs fsgnj.d rd, rs, rs Copy double-precision negate fmv.d rd, rs fsgnj.d rd, rs, rs Copy double-precision absolute value fneg.d rd, rs fsgnj.d rd, rs, rs Double-precision absolute value fneg.d rd, rs fsgnj.d rd, rs, rs Double-precision absolute value fneg.d rd, rs fsgnj.d rd, rs, rs Double-precision negate beqz rs, offset beq rs, x0, offset Branch if $\neq$ zero bezz rs, offset bound rs, x0, offset Branch if $\neq$ zero blez rs, offset bge x0, rs, offset Branch if $\neq$ zero bgez rs, offset bge x0, rs, offset Branch if $\neq$ zero bgtz rs, offset blt rs, x0, offset Branch if $\neq$ zero bgtz rs, offset blt x0, rs, offset Branch if $\neq$ zero bgtz rs, offset blt rt, rs, offset Branch if $\neq$ zero btr rs, rt, offset blt rt, rs, offset Branch if $\neq$ bgtu rs, rt, offset blt rt, rs, offset Branch if $\neq$ bgtu rs, rt, offset bgu rt, rs, offset Branch if $\neq$ unsigned bleu rs, rt, offset bgu rt, rs, offset Branch if $\neq$ unsigned j offset jal x0, offset Jump and link jr rs jalr x0, rs, 0 Jump and link register ret jalr x0, x1, 0 Return from subroutine call offset jalr x1, x6, offset[11:0] auipc x6, offset[31:12] jalr x0, x6, offset[11:0] Tail call far-away subroutine			_
sltz rd, rs sgtz rd, rs sgtz rd, rs sgtz rd, rs sgtz rd, rs slt rd, x0, rs Sct if > zero  Sct if > zero  fmv.s rd, rs fsgnj.s rd, rs, rs fsgnjx.s rd, rs, rs fsgnjx.s rd, rs, rs fsgnjx.s rd, rs, rs Single-precision absolute value fneg.s rd, rs fsgnjx.d rd, rs, rs fsgnjx.d rd, rs, rs Copy double-precision negate fmv.d rd, rs fsgnjx.d rd, rs, rs fsgnjx.d rd, rs, rs Copy double-precision negate fmv.d rd, rs fsgnjx.d rd, rs, rs Double-precision absolute value fneg.d rd, rs beqz rs, offset beq rs, x0, offset beq rs, x0, offset bnez rs, offset bne rs, x0, offset bnez rs, offset bge x0, rs, offset bge x0, rs, offset bgez rs, offset bge x0, rs, offset bgez rs, offset bge rs, x0, offset bgez rs, offset blt rs, x0, offset bgtz rs, offset blt rs, x0, offset blt rs, rt, offset blt rt, rs, offset blt rt, rs, offset ble rs, rt, offset ble rs, rt, offset blu rt, rs, offset ble rs, rt, offset blu rt, rs, offset blu rs, rt, offset blu rt, rs, offset blu rt, rs, offset blu rs, rt, offset blu rt, rs, offs	_		Set if $\neq$ zero
sgtz rd, rsslt rd, x0, rsSet if > zerofmv.s rd, rsfsgnj.s rd, rs, rsCopy single-precision registerfabs.s rd, rsfsgnjx.s rd, rs, rsSingle-precision absolute valuefneg.s rd, rsfsgnjn.s rd, rs, rsSingle-precision absolute valuefmv.d rd, rsfsgnjn.d rd, rs, rsCopy double-precision registerfabs.d rd, rsfsgnjx.d rd, rs, rsDouble-precision negatefneg.d rd, rsfsgnjn.d rd, rs, rsDouble-precision negatebeqz rs, offsetbeq rs, x0, offsetBranch if = zerobnez rs, offsetbne rs, x0, offsetBranch if ≠ zeroblez rs, offsetbge x0, rs, offsetBranch if ≤ zerobgez rs, offsetbge rs, x0, offsetBranch if ≤ zerobltz rs, offsetblt rs, x0, offsetBranch if < zero			•
fmv.s rd, rs fsgnj.s rd, rs, rs Single-precision register fabs.s rd, rs fsgnjx.s rd, rs, rs Single-precision absolute value fneg.s rd, rs fsgnjn.s rd, rs, rs Single-precision absolute value fneg.s rd, rs fsgnjn.s rd, rs, rs Single-precision negate fmv.d rd, rs fsgnjx.d rd, rs, rs Copy double-precision register fabs.d rd, rs fsgnjx.d rd, rs, rs Double-precision absolute value fneg.d rd, rs fsgnjn.d rd, rs, rs Double-precision negate beqz rs, offset beq rs, x0, offset Branch if = zero bnez rs, offset bgx x0, rs, offset Branch if $\neq$ zero blez rs, offset bgx x0, rs, offset Branch if $\neq$ zero blez rs, offset bgx x0, rs, offset Branch if $\neq$ zero bltz rs, offset blt rs, x0, 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 bgtz rs, offset blt x0, rs, offset Branch if $\neq$ zero bgtz rs, offset blt x0, rs, offset Branch if $\neq$ zero bgtz rs, rt, offset blt x1, rs, offset Branch if $\neq$ zero bgtz rs, rt, offset blt x0, rs, offset Branch if $\neq$ zero bgtz rs, rt, offset blt x0, rs, offset Branch if $\neq$ zero bgtz rs, rt, offset blt x0, rs, offset Branch if $\neq$ zero bgtz rs, rt, offset blt x0, rs, offset Branch if $\neq$ zero bgtz rs, rt, offset blt x0, rs, offset Branch if $\neq$ zero bgtz rs, rt, offset blt x0, rs, offset Branch if $\neq$ zero bgtz rs, rt, offset blt x0, rs, offset Branch if $\neq$ zero bgtz rs, rt, offset blt x0, rs, offset Branch if $\neq$ zero bgtz rs, rt, offset blt x0, rs, offset Branch if $\neq$ zero bgtz rs, rt, offset blt x0, rs, offset Branch if $\neq$ zero bgtz rs, rt, offset blt x0, rs, offset Branch if $\neq$ zero bgtz rs, rt, offset blt x0, rs, offset Branch if $\neq$ zero bgtz rs, rt, offset blt x0, rs, offset Branch if $\neq$ zero bgtz rs, rt, offset blt x0, rs, offset Branch if $\neq$ zero bgtz rs, rt, offset blt x0, rs, offset Branch if $\neq$ zero bgtz rs, rt, offset blt x0, rs, offset Branch if $\neq$ zero bgtz rs, rt, offset blt x0, rs, offset Branch if $\neq$ zero bgtz rs, rt, offset blt x0, rs, offset blt x0, rs, offset blt x0, rs, offset blt			
fabs.s rd, rs fsgnjx.s rd, rs, rs fsgnjn.s rd, rs, rs fsgnjn.d rd, rs, rs fsgnjn.d rd, rs, rs fsgnjn.d rd, rs, rs pouble-precision absolute value fneg.d rd, rs beqz rs, offset beq rs, x0, offset bnez rs, offset bnez rs, offset blez rs, offset blez rs, offset bge x0, rs, offset bge x0, rs, offset bge rs, x0, offset bltz rs, offset blt rs, x0, offset blt rs, x0, offset blt rs, x0, offset blt rs, rt, offset blt rt, rs, offset blt rt, rs, offset blt rs, rt, offset blt rt, rs, offset blt rt, rs, offset blt rs, rt, offset blt rt, rs, offset blt rt, rs, offset blt rs, rt, offset blt rt, rs, offset blt rs, rt, offset blt rt, rs, offset blt rs, rt, offset blt rt, rs, offset blt rt, rs, offset blt rs, rt, offset blt rt, rs, offset blt rt, rs, offset blt rs, rt, offset blt rt, rs, offset blt rt, rs, offset blt rs, rt, offset blt rt, rs, offset blt rt, rs, offset blt rs, rt, offset blt rt, rs, offset			
fneg.s rd, rs fsgnjn.s rd, rs, rs fsgnj.d rd, rs, rs fsgnjs.d rd, rs, rs Double-precision absolute value fneg.d rd, rs beq rs, offset beq rs, x0, offset bnez rs, offset bnez rs, offset blez rs, offset blt rs, x0, offset blt rs, x0, offset blt rs, x0, offset blt rs, x0, offset blt rs, rt, offset blt rs, rt, offset blt rt, rs, offset ble rs, rt, offset blt rt, rs, offset blt rt, rs, offset blt rs, rt, offset blt rt, rs, offset blt rt, rs, offset blt rs, rt, offset blt rt, rs, offset blt rs, rt, offset blt rt, rs, offset blt rt, rs, offset blt rs, rt, offset blt rt, rs, offset blt rt, rs, offset blt rs, rt, offset blt rt, rs, offset blt rt, rs, offset blt rs, rt, offset blt rt, rs, offset blt rs, rt, offset blt rt, rs, offset blt rt, rs, offset blt rs, rt, offset blt rt, rs, offset blt rt, rs, offset blt rs, rt, offset blt rt, rs, offset blt rt, rs, offset blt rs, rt, offset blt rt, rs, offset blt rt, rs, offset blt rs, rt, offset blt rt, rs,	-		10 0 1
fmv.d rd, rs fsgnj.d rd, rs, rs fabs.d rd, rs fsgnjx.d rd, rs, rs pouble-precision absolute value fneg.d rd, rs fsgnjn.d rd, rs, rs beqz rs, offset beq rs, x0, offset blez rs, offset beg x0, rs, offset beg x0, rs, offset beg rs, x0, offset begr rs, rt, offset begr rs, rt, offset begr rt, rs, offset begr rs, rt, offset begr rs, x0, offset begr rs, x0, offset branch if ≤ zero begr rs, x1, x1, x1, x2, offset branch if ≤ zero begr rs, x1, x2, x3, x4, x4, x4, x4, x4, x4, x4, x4, x4, x4	-		
fabs.d rd, rs fsgnjx.d rd, rs, rs Double-precision absolute value fneg.d rd, rs beqz rs, offset beq rs, x0, offset Branch if = zero bnez rs, offset bge x0, rs, offset Branch if ≤ zero bgez rs, offset bge rs, x0, offset Branch if ≤ zero bgez rs, offset bge rs, x0, offset Branch if ≤ zero bgez rs, offset blt rs, x0, offset Branch if ≤ zero bgtz rs, offset blt rs, x0, offset Branch if > zero bgtz rs, offset blt rt, rs, offset Branch if > zero bgt rs, rt, offset blt rt, rs, offset Branch if > bernot bgt rs, rt, offset bge rt, rs, offset Branch if > bernot bgt rs, rt, offset bge rt, rs, offset Branch if > bernot bgt rs, rt, offset bgu rs, rt, offset bgu rs, rt, offset bgu rt, rs, offset Branch if > unsigned  Branch if >, unsigned  Branch if >, unsigned  Branch if >, unsigned  Jump jal offset jal x0, offset jal x0, offset Jump and link jr rs jalr x1, rs, 0 Jump and link register jalr rs jalr x0, x1, 0 Return from subroutine  call offset auipc x6, offset[31:12] jalr x1, x6, offset[11:0]  Tail call far-away subroutine	_		
fneg.d rd, rsfsgnjn.d rd, rs, rsDouble-precision negatebeqz rs, offsetbeq rs, x0, offsetBranch if = zerobnez rs, offsetbne rs, x0, offsetBranch if ≠ zeroblez rs, offsetbge x0, rs, offsetBranch if ≤ zerobgez rs, offsetbge rs, x0, offsetBranch if ≤ zerobltz rs, offsetblt rs, x0, offsetBranch if < zero			
bedz rs, offset bed 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 bltz 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 rt, rs, offset Branch if $>$ zero bgt rs, rt, offset blt rt, rs, offset Branch if $\geq$ blt rs, rt, offset bge rt, rs, offset Branch if $\leq$ bgtu rs, rt, offset bge rt, rs, offset Branch if $\geq$ 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 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 x6, offset [31:12] jalr x1, x6, offset [31:12] jalr x0, x6, offset [31:12] Tail call far-away subroutine			
bnez rs, offset bne rs, x0, offset Branch if ≠ zero blez rs, offset bge x0, rs, offset Branch if ≤ zero bgez rs, offset bge rs, x0, offset Branch if ≤ 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 > zero bgt rs, rt, offset blt rt, rs, offset Branch if > blt rt, rs, offset Branch if > blt rt, rs, offset Branch if > lt rt, rs, offset Branch if			
blez rs, offset bge x0, rs, offset Branch if ≤ zero bgez rs, offset bge rs, x0, offset Branch if ≥ 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 ≤ 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 and link jr rs jalr x1, rs, 0 Jump and link register ret jalr x0, x1, 0 Return from subroutine  call offset jalr x1, x6, offset[31:12] jalr x1, x6, offset[11:0]  tail offset jalr x0, x6, offset[11:0]  Tail call far-away subroutine	=		
bgez rs, offset bge rs, x0, offset Branch if ≥ 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 ≤ 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, x6, offset[31:12] jalr x1, x6, offset[11:0]  tail offset jalr x0, x6, offset[11:0]  Tail call far-away subroutine			· · · · · · · · · · · · · · · · · · ·
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 ≤, unsigned  bleu rs, rt, offset bgeu rt, rs, offset Branch if ≤, unsigned  j offset jal x0, 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 x6, offset[31:12]  jalr x1, x6, offset[11:0]  tail offset jalr x0, x6, offset[11:0]  Tail call far-away subroutine		_	
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 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 x6, offset[31:12] jalr x1, x6, offset[11:0]  tail offset jalr x0, x6, offset[11:0]  Tail call far-away subroutine	•	•	_
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, x6, offset[31:12]  jalr x0, x6, offset[11:0]  tail offset jalr x0, x6, offset[11:0]  Tail call far-away subroutine			
ble rs, rt, offset bgtu rs, rt, offset bltu rt, rs, offset bleu rs, rt, offset bleu rs, rt, offset bleu rs, rt, offset bgeu rt, rs, offset  j offset j al x0, offset j al x1, offset j al x1, offset j al x1, rs, 0 j alr x1, rs, 0 j alr x0, x1, 0  call offset j alr x1, x6, offset[31:12] j auipc x6, offset[31:12] j alr x0, x6, offset[11:0]  Tail call far-away subroutine  Tail call far-away subroutine			
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 auipc x6, offset[31:12]  jalr x1, x6, offset[11:0]  tail offset jalr x0, x6, offset[11:0]  Tail call far-away subroutine	•		
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, x6, offset[31:12]  jalr x1, x6, offset[11:0]  tail offset jalr x0, x6, offset[11:0]  Tail call far-away subroutine			
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, x6, offset[31:12] jalr x1, x6, offset[11:0]  tail offset jalr x0, x6, offset[11:0]  Tail call far-away subroutine	bleu rs, rt, offset	bgeu rt, rs, offset	
jal offset jal x1, offset Jump and link jr rs jalr x0, rs, 0 Jump register Jump and link register Jump and link register Jump and link register Return from subroutine  call offset auipc x6, offset[31:12] jalr x1, x6, offset[11:0]  tail offset jalr x0, x6, offset[11:0]  Tail call far-away subroutine			
jr rsjalr x0, rs, 0Jump registerjalr rsjalr x1, rs, 0Jump and link registerretjalr x0, x1, 0Return from subroutinecall offsetauipc x6, offset[31:12] jalr x1, x6, offset[11:0]Call far-away subroutinetail offsetauipc x6, offset[31:12] jalr x0, x6, offset[11:0]Tail call far-away subroutine	•	<del>-</del>	_
jalr x1, rs, 0  ret  jalr x0, x1, 0  auipc x6, offset[31:12]  jalr x1, x6, offset[11:0]  tail offset  jalr x0, x6, offset[31:12]  jalr x0, x6, offset[11:0]  Tail call far-away subroutine		<del>-</del>	
ret jalr x0, x1, 0 Return from subroutine  call offset auipc x6, offset[31:12] jalr x1, x6, offset[11:0]  tail offset jalr x0, x6, offset[31:12] jalr x0, x6, offset[11:0]  Tail call far-away subroutine	_	<del>-</del>	
call offset  auipc x6, offset[31:12]  jalr x1, x6, offset[11:0]  tail offset  auipc x6, offset[31:12]  jalr x0, x6, offset[11:0]  Call far-away subroutine  Tail call far-away subroutine	_		
tail offset  jalr x1, x6, offset[11:0]  auipc x6, offset[31:12]  jalr x0, x6, offset[11:0]  Tail call far-away subroutine		<u> </u>	
jalr x0, x6, offset[11:0]	call offset		Call far-away subroutine
fence fence iorw, iorw Fence on all memory and I/O	tail offset		Tail call far-away subroutine
	fence	fence iorw, iorw	Fence on all memory and I/O

Table 20.2: RISC-V pseudoinstructions.

Pseudoinstruction	Base Instruction	Meaning
rdinstret[h] rd	csrrs rd, instret[h], x0	Read instructions-retired counter
rdcycle[h] rd	csrrs rd, cycle[h], x0	Read cycle counter
rdtime[h] rd	csrrs rd, time[h], x0	Read real-time clock
csrr rd, csr	csrrs rd, csr, x0	Read CSR
csrw csr, rs	csrrw x0, csr, rs	Write CSR
csrs csr, rs	csrrs x0, csr, rs	Set bits in CSR
csrc csr, rs	csrrc x0, csr, rs	Clear bits in CSR
csrwi csr, imm	csrrwi x0, csr, imm	Write CSR, immediate
csrsi csr, imm	csrrsi x0, csr, imm	Set bits in CSR, immediate
csrci csr, imm	csrrci x0, csr, imm	Clear bits in CSR, immediate
frcsr rd	csrrs rd, fcsr, x0	Read FP control/status register
fscsr rd, rs	csrrw rd, fcsr, rs	Swap FP control/status register
fscsr rs	csrrw x0, fcsr, rs	Write FP control/status register
frrm rd	csrrs rd, frm, x0	Read FP rounding mode
fsrm rd, rs	csrrw rd, frm, rs	Swap FP rounding mode
fsrm rs	csrrw x0, frm, rs	Write FP rounding mode
fsrmi rd, imm	csrrwi rd, frm, imm	Swap FP rounding mode, immediate
fsrmi imm	csrrwi x0, frm, imm	Write FP rounding mode, immediate
frflags rd	csrrs rd, fflags, x0	Read FP exception flags
fsflags rd, rs	csrrw rd, fflags, rs	Swap FP exception flags
fsflags rs	csrrw x0, fflags, rs	Write FP exception flags
fsflagsi rd, imm	csrrwi rd, fflags, imm	Swap FP exception flags, immediate
fsflagsi imm	csrrwi xO, fflags, imm	Write FP exception flags, immediate

Table 20.3: Pseudoinstructions for accessing control and status registers.