FRISC-V naredbe

Naziv	*Asembler*	*Operandi*	*Operacija*
Aritmetičke naredbe			
add	add	rd, rs1, rs2	rd = rs1 + rs2
add immediate	addi	rd, rs1, imm	rd = rs1 + sign_ext(imm12)
subtract	sub	rd, rs1, rs2	rd = rs1 - rs2
Logičke naredbe			
bitwise AND	and	rd, rs1, rs2	rd = rs1 and rs2
bitwise AND immediate	andi	rd, rs1, imm	rd = rs1 and sign_ext(imm12)
bitwise OR	or	rd, rs1, rs2	rd = rs1 or rs2
bitwise OR immediate	ori	rd, rs1, imm	rd = rs1 or sign_ext(imm12)
bitwise exclusive OR	xor	rd, rs1, rs2	rd = rs1 xor rs2
bitwise XOR immediate	xori	rd, rs1, imm	rd = rs1 xor sign_ext(imm12)
Pomaci			
shift left logical	sll	rd, rs1, rs2	rd = rs1 << rs2[4:0]
shift left logical immediate	slli	rd, rs1, imm	rd = rs1 << imm[4:0]
shift right logical	srl	rd, rs1, rs2	rd = rs1 >> rs2[4:0]
shift right logical immediate	srli	rd, rs1, imm	rd = rs1 >> imm[4:0]
shift right arithmetic	sra	rd, rs1, rs2	rd = rs1 $>>_A$ rs2[4:0]
shift right arithmetic immediate	srai	rd, rs1, imm	rd = rs1 $>>_A$ imm[4:0]
Usporedbe			
set less than	slt	rd, rs1, rs2	if (rs1<rs2) rd=1 else rd=0
set less than immediate	slti	rd, rs1, imm	if (rs1<sign_ext(imm12)) rd=1 else rd=0
set less than unsigned	sltu	rd, rs1, rs2	if (rs1<rs2) rd=1 else rd=0
set less than immediate unsigned	sltiu	rd, rs1, imm	if (rs1<imm12) rd=1 else rd=0
Posebne AL naredbe			
load upper immediate	lui	rd,imm20	rd = [[imm20]000000000000]
add upper immediate to pc	auipc	rd,imm20	rd = pc + [[imm20]000000000000]
Prijenos podataka			
load	l[*b*|*bu*|*h*|*hu*|*w*]	rd, imm(rs1)	rd = MEM[rs1+sign_ext(imm)]
store	s[*b*|*h*|*w*]	rs2, imm(rs1)	MEM[rs1+sign_ext(imm)] = rs2
Upravljačke naredbe			
jump and link	jal	rd, label	rd = PC+4; PC = label
jump and link register	jalr	rd, imm(rs1)	rd = PC+4; PC =(rs1 + sign_ext(imm12)) & 0xFFFFFFFE
branch	b[*eq*|*ne*|*lt*|*ge*|*ltu*|*geu*]	rs1, rs2, label	if (rs1 *condition* rs2) PC=label
FRISCV naredbe			
halt	halt		stop the simulation

Asemblerske funkcije %hi i %lo
lui x1, %hi(const)
addi x1, x1, %lo(const)
} x1 = const — enables loading of 32b constant

Asemblerske funkcije %pcrel_hi i %pcrel_lo
auipc x1,%pcrel_hi(label)
addi x1,x1,%pcrel_lo(label)
} x1 = label — enables pc-relative address loading

FRISC-V registri

Ime	Alt.ime	Opis
x0	zero	hardwired zero
x1	ra	return address
x2	sp	stack pointer
x3	gp	global pointer
x4	tp	thread pointer
x5	t0	temporary / alternate link register
x6	t1	temporary
x7	t2	temporary
x8	s0/fp	saved register / frame pointer
x9	s1	saved register
x10	a0	function argument / return value
x11	a1	function argument / return value
x12	a2	function argument
x13	a3	function argument
x14	a4	function argument
x15	a5	function argument
x16	a6	function argument
x17	a7	function argument
x18	s2	saved register
x19	s3	saved register
x20	s4	saved register
x21	s5	saved register
x22	s6	saved register
x23	s7	saved register
x24	s8	saved register
x25	s9	saved register
x26	s10	saved register
x27	s11	saved register
x28	t3	temporary
x29	t4	temporary
x30	t5	temporary
x31	t6	temporary

Ime
pc