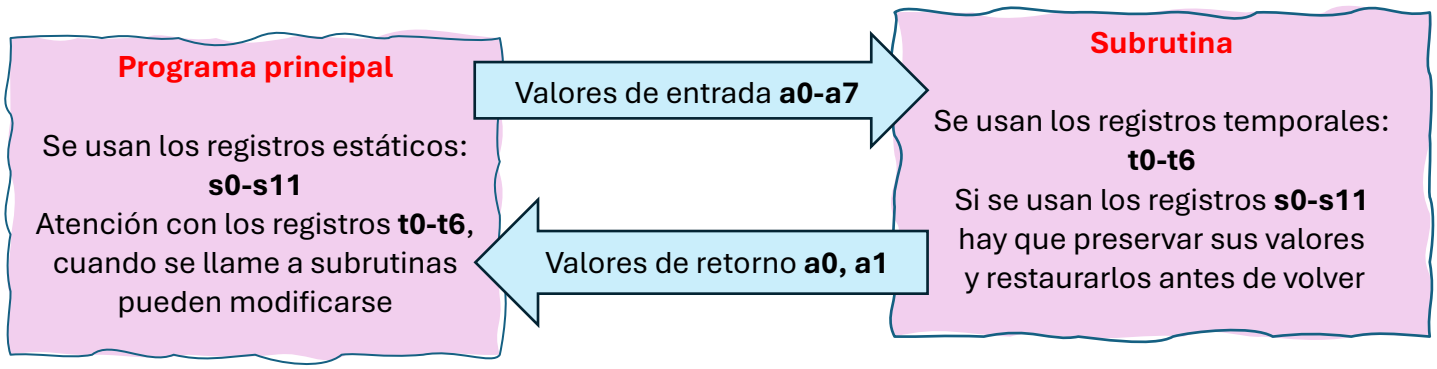
mnemónico	sintaxis	tipo	descripción
add	add rd, rs1, rs2	**R**	rd ← rs1 + rs2
sub	sub rd, rs1, rs2	**R**	rd ← rs1 – rs2
and	and rd, rs1, rs2	**R**	rd ← rs1 & rs2
or	or rd, rs1, rs2	**R**	rd ← rs1 | rs2
xor	xor rd, rs1, rs2	**R**	rd ← rs1 ^ rs2
addi	addi rd, rs1, imm12	**I**	rd ← rs1 + immext
andi	andi rd, rs1, imm12	**I**	rd ← rs1 & immext
ori	ori rd, rs1, imm12	**I**	rd ← rs1 | immext
xori	xori rd, rs1, imm12	**I**	rd ← rs1 ^ immext
sll	sll rd, rs1, rs2	**R**	rd ← rs1 << $rs2_{4:0}$
srl	srl rd, rs1, rs2	**R**	rd ← rs1 >> $rs2_{4:0}$
sra	sra rd, rs1, rs2	**R**	rd ← rs1 >>> $rs2_{4:0}$
slli	slli rd, rs1, imm12i	**I**	rd ← rs1 << imm5
srli	srli rd, rs1, imm12	**I**	rd ← rs1 >> imm5
srai	srai rd, rs1, imm12	**I**	rd ← rs1 >>> imm5
lb	lb rd, imm12(rs1)	**I**	rd ← sExt(MEM[rs1+inmext]$_{7:0}$)
lh	lh rd, imm12(rs1)	**I**	rd ← sExt(MEM[rs1+inmext]$_{15:0}$)
lw	lw rd, imm12(rs1)	**I**	rd ← MEM[rs1+inmext]
lbu	lbu rd, imm12(rs1)	**I**	rd ← zExt(MEM[rs1+inmext]$_{7:0}$)
lhu	lhu rd, imm12(rs1)	**I**	rd ← zExt(MEM[rs1+inmext]$_{15:0}$)
sw	sw rs2, imm12(rs1)	**S**	MEM[rs1+immext] ← rs2
sh	sh rs2, imm12(rs1)	**S**	MEM[rs1+immext] $_{15:0}$ ← $rs2_{15:0}$
sb	sb rs2, imm12(rs1)	**S**	MEM[rs1+immext] $_{7:0}$ ← $rs2_{7:0}$
lui	lui rd, inm20	**U**	rd ← {imm[31:12],12'b0}
auipc	auipc rd,imm20	**U**	rd ←PC + (imm20<<12)

mnemónico	Sintaxis	Tipo	Descripción
beq	beq rs1, rs2, imm13	**B**	if (rs1=rs2) => PC ← PC + sExt(imm12:1 <<1))
bne	bne rs1, rs2, imm13	**B**	if (rs1!=rs2) => PC ← PC + sExt(imm12:1 <<1))
blt	blt rs1, rs2, imm13	**B**	if (rs1<rs2) => PC ← PC + sExt(imm12:1 <<1))
bge	bge rs1, rs2, imm13	**B**	if (rs1≥rs2) => PC ← PC + sExt(imm12:1 <<1))
bltu	bltu rs1, rs2, imm13	**B**	if (rs1<rs2) => PC ← PC + sExt(imm12:1 <<1))
bgeu	bgeu rs1, rs2, imm13	**B**	if (rs1≥rs2) => PC ← PC + sExt(imm12:1 <<1))
jalr	jalr rd, rs1, imm12	**I**	PC ← rs1 +sExt(imm) rd ← PC + 4
jal	jal rd, imm21	**J**	PC ← PC +sExt($imm_{20:1}$<<1) rd ← PC + 4



Registro	**Nombre alternativo**	**Descripción**
x0	zero	Siempre vale 0 no se puede modificar
x1	ra	Dirección de retorno
x2	sp	Puntero de pila
x3	gp	Global pointer
x4	tp	Thread pointer
x5-x7	t0-t2	Temporales 0-2
x8	s0/fp	Saved register 0/frame pointer
x9	s1	Saved register 1
x10-x17	a0-a7	Argumentos de función
x18-x27	s2-s11	Saved register 2-11
x28-x31	t3-t6	Temporales 3-6
pc	pc	Contador de programa

Pseudoinstrucción	**Operación**	**Equivalente RISCV**
li x8,0x7fe	S8 ← 0x7fe	addi s8,x0,0x7fe (si dato <12b)
la t1, dato	t1← dir dato	auipc t1,imm20 addi t1, t1, imm12
li s8,0x56789def	S8 ← 0x56789def	lui s8,0x5678A addi s8, s8, 0xdef
mv x7,x2	x7 ← x2	addi x7,x2,0
not x7,x2	x7← Ca1(x2)	xori x7,x2,-1
neg x7,x2	x7← Ca2(x2)	sub x7,0,x2
b label	PC ← dir label	beq x0,x0,label
j label	PC ← dir label	jal x0,label
call label	PC← dir label, ra ← PC+4	jal x1,label
ret	PC ← ra	jalr x0,x1,0
nop	no operación	addi x0,x0,0
bgt x1,x3,label	if(x1>x3) PC ← dir label	blt x3,x1,label
ble x1,x3,label	if(x1≤x3) PC ← dir label	bge x3,x1,label
bgtu x1,x3,label	if(x1>x3) PC ← dir label	bltu x3,x1,label
bleu x1,x3,label	if(x1≤x3) PC ← dir label	bgeu x3,x1,label
beqz x1,label	if(x1=0) PC ← dir label	beq x1,x0,label
bnez x1,label	if(x1!=0) PC ← dir label	bne x1,x0,label