Instruction	Usage	Description	Opcode(6)	R1 (5)	R2 (5)	Shift amount(5)	Immediate value (21)	Opcode Extension(11)
R-type instructions								
Add	add rs,rt	rs <- (rs) + (rt)	000000	rs	rt	XXXXX	*XXXXXXXXXXXXXXXXXXXXX*	00000000000
Complement	comp rs,rt	rs <- 2's complement (rt)	000000	rs	rt	XXXXX	*XXXXXXXXXXXXXXXXXXXXX*	00000000001
And	and rs,rt	rs <- (rs) & (rt)	000000	rs	rt	XXXXX	*XXXXXXXXXXXXXXXXXXXXX*	00000000010
Xor	xor rs,rt	rs <- (rs) ⊕ (rt)	000000	rs	rt	XXXXX	*XXXXXXXXXXXXXXXXXXXXX*	00000000011
Shift left logical	shll rs,sh	rs <- (rs) left shifted by sh	000000	rs	XXXXX	shift amount sh	*XXXXXXXXXXXXXXXXXXXXX*	00000000100
Shift right logical	shrl rs,sh	rs <- (rs) right shifted by sh	000000	rs	XXXXX	shift amount sh	*XXXXXXXXXXXXXXXXXXXXX*	00000000101
Shift left logical variable	shllv rs,rt	rs <- (rs) left shifted by (rt)	000000	rs	rt	XXXXX	*XXXXXXXXXXXXXXXXXXXXX*	00000000110
Shift right logical variable	shrlv rs,rt	rs <- (rs) right shifted by (rt)	000000	rs	rt	XXXXX	*XXXXXXXXXXXXXXXXXXXXX*	00000000111
Shift right arithmetic	shra rs, sh	rs <- (rs) arithmetic right shifted by sh	000000	rs	XXXXX	shift amount sh	*XXXXXXXXXXXXXXXXXXXXX*	00000001000
Shift right arithmetic variable	shrav rs,rt	rs <- (rs) arithmetic right shifted by (rt)	000000	rs	rt	XXXXX	*XXXXXXXXXXXXXXXXXXXXX*	00000001001
I-type instructions								
Load Word	lw rt,imm(rs)	rt <- mem[(rs) + imm]	000001	rs	rt	XXXXX	immediate constant value (16 bits)	XXXXXXXXXXX
Store Word	sw rt, imm(rs)	mem[(rs) + imm] <- rt	000010	rs	rt	XXXXX	immediate constant value (16 bits)	XXXXXXXXXXX
Add immediate	addi rs,imm	rs <- (rs) + imm	001111	rs	XXXXX	XXXXX	immediate constant value	XXXXXXXXXXX
Complement immediate	compi rs,imm	rs <- 2's complement of imm	010000	rs	XXXXX	XXXXX	immediate constant value	XXXXXXXXXXX
Branch Register	br rs	goto contents(rs)	000100	rs	XXXXX	XXXXX	*XXXXXXXXXXXXXXXXXXXXX*	XXXXXXXXXXX
Bit Position(Allocation)			31..26	25..21	20..16	15..11	*20..0*	10..0

Instruction	Usage	Description	Opcode(6)	Offset (26)
Unconditional branch	b L	goto L	000011	*offset*
Branch on zero	bz L	if (zflag == 1) then goto L	000101	*offset*
Branch on not zero	bnz L	if(zflag == 0) then goto L	000110	*offset*
Branch on Carry	bcy L	if (carryflag == 1) then goto L	000111	*offset*
Branch on No Carry	bncy L	if (carryflag == 0) then goto L	001000	*offset*
Branch on Sign	bs L	if (signflag == 1) then goto L	001001	*offset*
Branch on Not Sign	bns L	if (signflag == 0) then goto L	001010	*offset*
Branch on Overflow	bv L	if (overflowflag == 1) then goto L	001011	*offset*
Branch on No Overflow	bnv L	if (overflowflag == 0) then goto L	001100	*offset*
Call	call L	ra ← (PC)+4 ; goto L	001101	*offset*
Return	Ret	goto contents(ra)	001110	*XXXXXXXXXXXXXXXXXXXXXXXXXX*
Bit Position(Allocation)			31..26	25..0