Opcode	Rs	Rt	Rd	SA	Funct	Assembly		
000000	*Rs*	*Rt*	*Rd*	00000	100000	add	Rd, Rs, Rt	# Rd = Rs + Rt, Exception: integer overflow
000000	*Rs*	*Rt*	*Rd*	00000	100001	addu	Rd, Rs, Rt	# Rd = Rs + Rt
000000	*Rs*	*Rt*	*Rd*	00000	100100	and	Rd, Rs, Rt	# Rd = Rs & Rt
000000	*Rs*	*Rt*	*Rd*	00000	100111	nor	Rd, Rs, Rt	# Rd = ~(Rs | Rt)
000000	*Rs*	*Rt*	*Rd*	00000	100101	or	Rd, Rs, Rt	# Rd = Rs | Rt
000000	00000	*Rt*	*Rd*	*sa*	000000	sll	Rd, Rt, sa	# Rd = Rt << sa
000000	00000	*Rt*	*Rd*	*sa*	000010	srl	Rd, Rt, sa	# Rd = Rt >>> sa, zero extended
000000	00000	*Rt*	*Rd*	*sa*	000011	sra	Rd, Rt, sa	# Rd = Rt >> sa, sign extended
000000	*Rs*	*Rt*	*Rd*	00000	101010	slt	Rd, Rs, Rt	# Rd = (Rs < Rt) ? 1 : 0, signed comparision
000000	*Rs*	*Rt*	*Rd*	00000	101011	sltu	Rd, Rs, Rt	# Rd = (Rs < Rt) ? 1 : 0, unsigned comparision
000000	*Rs*	*Rt*	*Rd*	00000	100010	sub	Rd, Rs, Rt	# Rd = Rs – Rt, Exception: interger overflow
000000	*Rs*	*Rt*	*Rd*	00000	100011	subu	Rd, Rs, Rt	# Rd = Rs – Rt
000000	*Rs*	*Rt*	*Rd*	00000	100110	xor	Rd, Rs, Rt	# Rd = Rs ^ Rt
001000	*Rs*	*Rt*	16 bit sign-extended			addi	Rt, Rs, imm	# Rt = Rs + sign_extend(imm), Exception: integer overflow
001001	*Rs*	*Rt*	16 bit sign-extended			addiu	Rt, Rs, imm	# Rt = Rs + sign_extend(imm)
001100	*Rs*	*Rt*	16 bit zero-extended			andi	Rt, Rs, imm	# Rt = Rs & zero_extend(imm)
000100	*Rs*	*Rt*	16 bit sign-extended			beq	Rs, Rt, imm	# PC = (PC+4)+(Rs==Rt)? sign_extend(imm)<<2 :0
000101	*Rs*	*Rt*	16 bit sign-extended			bne	Rs, Rt, imm	# PC = (PC+4)+(Rs!=Rt)? sign_extend(imm)<<2 :0
100000	*Rs*	*Rt*	16 bit sign-extended			lb	Rt, imm(Rs)	# Rt = sign_extend(MEM_B[Rs+sign_extend(imm)])
100100	*Rs*	*Rt*	16 bit sign-extended			lbu	Rt, imm(Rs)	# Rt = zero_extend(MEM_B[Rs+sign_extend(imm)])
001111	00000	*Rt*	16 bit			lui	Rt, imm	# Rt = imm << 16
100011	*Rs*	*Rt*	16 bit sign-extended			lw	Rt, imm(Rs)	# Rt = sign_extend(MEM_W[Rs+sign_extend(imm)])
001101	*Rs*	*Rt*	16 bit zero-extended			ori	Rt, Rs, imm	# Rt = Rs | zero_extend(imm)
101000	*Rs*	*Rt*	16 bit sign-extended			sb	Rt, imm(Rs)	# MEM_B[Rs+sign_extend(imm)] = lsb(Rt)
001010	*Rs*	*Rt*	16 bit sign-extended			slti	Rt, Rs, imm	# Rt = (Rs < sign_extend(imm)) ? 1 : 0, signed comparision
001011	*Rs*	*Rt*	16 bit sign-extended			sltiu	Rt, Rs, imm	# Rt = (Rs < sign_extend(imm)) ? 1 : 0, unsigned comparision
101011	*Rs*	*Rt*	16 bit sign-extended			sw	Rt, imm(Rs)	# MEM_W[Rs+sign_extend(imm)] = Rt
001110	*Rs*	*Rt*	16 bit zero-extended			xori	Rt, Rs, imm	# Rt = Rs ^ zero_extend(imm)
000010	26 bit					j	addr	# PC = ((PC+4) & 0xF0000000) | (addr << 2)
000011	26 bit					jal	addr	# $ra = PC+8; PC = ((PC+4) & 0xF0000000) | (addr << 2)
000000	*Rs*	00000 00000 00000			001000	jr	Rs	# PC = Rs