15 : 13	12 : 10	9	8 : 6	5 : 3	2 : 0	Mnemonic	Functional description
000	000	g	rs2	rs1	rd	AND	rd = rs1 & rs2
	001					OR	rd = rs1 | rs2
	010					XOR	rd = rs1 ^ rs2
	011					ADD	rd = rs1 + rs2
	100					SUB	rd = rs1 - rs2
	101					SLL	rd = rs1 << rs2
	110					SRA	rd = rs1 >> rs2 (arithmetic)
	111					SRL	rd = rs1 >> rs2 (logical)
001000		g	000	rs1	rd	NOT	rd = !rs1
			001			COM	rd = ~rs1 (complement)
		0	010			MVHL	g0.rd = g1.rs1 (move register from high group to low group)
		1				MVLH	g1.rd = g0.rs1 (move register from low group to high group)
		1	011			MVH	g1.rd = g1.rs1 (move register in high group)
001001		g	offset	rs1	rd	LH	rd = MEM[rs1 + offset] (load half-word)
010	imm1	g	imm2	imm3	rd	LI	rd = {imm1, imm2, imm3} (load immediate)
001010		g	rs2	rs1	offset	SH	MEM[rs1 + offset] = rs2 (store half-word)
100000		g	rs2	rs1	000	SLT	set COMS to 1 when rs1 < rs2 (set on less than)
					001	SOE	set COMS to 1 when rs1 == rs2 (set on equal)
100	001	offset				BOZ	next_pc = current_pc + offset, when COMS is zero
	010					BONZ	next_pc = current_pc + offset, when COMS is not zero
100	100	offset				JAL	next_pc = current_pc + offset, a15 = current_pc + 2
	101	g	offset0	rs1	offset1	JALR	next_pc = rs1 + {offset0, offset1}, a15 = current_pc + 2
111111		1111		XXX	XXX	HALT	next_pc = current_pc, all memory and register status cannot be changed