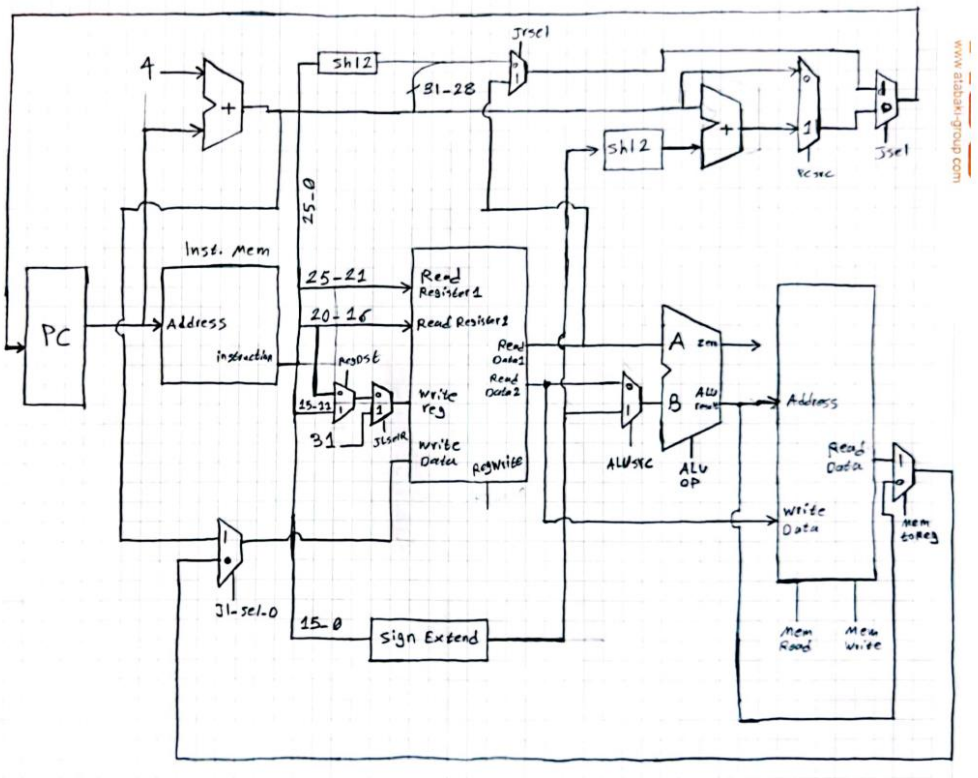
Single-cycle Mips Processor

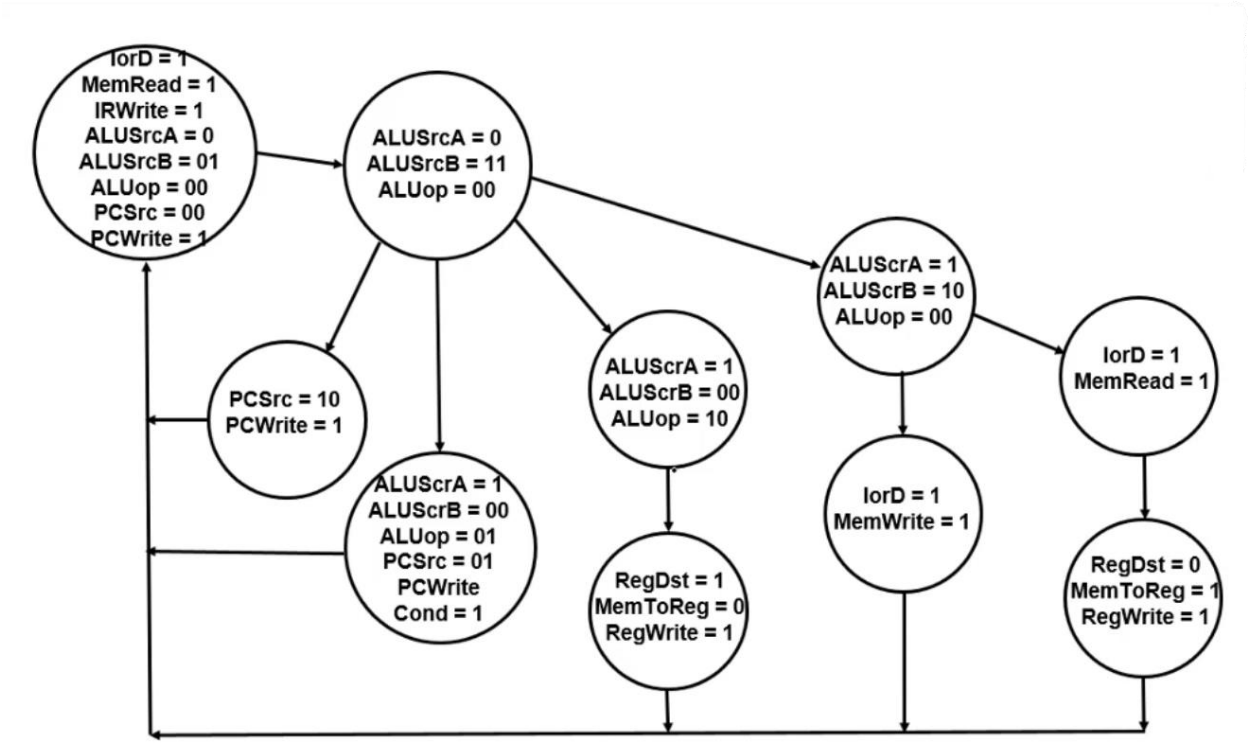
Instructions Format:

Instruction Type	Example		Instruction Coding						
	ALU Usage								
Non-Jump R-Type	add rd, rs, rt	R	31	26 25	21 20	16 15	11 10	6 5	
			op	rs	rt	rd	sa	fn	
	The ALU performs the operation indicated by the mnemonic, which is coded into the fn field.								
Immediate	addi rt, rs, imm	I	31	26 25	21 20	16 15			
			op	rs	rt	imm			
	The ALU performs the operation indicated by the mnemonic, which is coded into the op field.								
Branch	beq $rs, $rt, imm	I	31	26 25	21 20	16 15			
			op	rs	rt	imm			
	The ALU subtracts rt from rs for comparison.								
Load	lw rt, imm(rs)	I	31	26 25	21 20	16 15			
			op	rs	rt	imm			
	The ALU adds rs and imm to get the address.								
Store	sw rt, imm(rs)	I	31	26 25	21 20	16 15			
			op	rs	rt	imm			
	The ALU adds rs and imm to get the address.								
Non-Register Jump	jal target	J	31	26 25					
			op	target					
	The ALU is not used.								
Jump Register	jalr rd, rs	R	31	26 25	21 20	16 15	11 10	6 5	
			op	rs	rt	rd	sa	fn	
	The ALU is not used.								

Datapath Schematic:



Controller state machine:



Controller Signal Guides:

Instruction/signal	Jsel0	Jselr	Jsel	jrsel	Alusrc	RegDst	RegWrite	MemRead	MemWrite	PCsrc	MemtoReg	ALUop
Add	0	0	0	-	0	1	1	-	-	0	0	000
Addi	0	0	0	-	1	0	1	-	-	0	0	000
Sub	0	0	0	-	0	1	1	-	-	0	0	001
Slt	0	0	0	-	0	1	1	-	-	0	0	100
Slti	0	0	0	-	1	0	1	-	-	0	0	100
and	0	0	0	-	0	1	1	-	-	0	0	010
Or	0	0	0	-	0	1	1	-	-	0	0	011
Load Word	0	0	0	-	1	0	1	1	0	0	1	000
Store Word	0	0	0	-	1	0	0	0	1	0	-	000
Jump	-	-	1	0	-	-	-	-	-	-	-	-
Jump & link	-	1	1	0	-	-	-	-	-	-	-	-
Jump Register	-	-	1	1	-	-	-	-	-	-	-	-
Branch equal	0	0	0	-	0	-	0	-	-	is_zero	-	001

0 : signal 0
1 : signal 1
- : Don't care