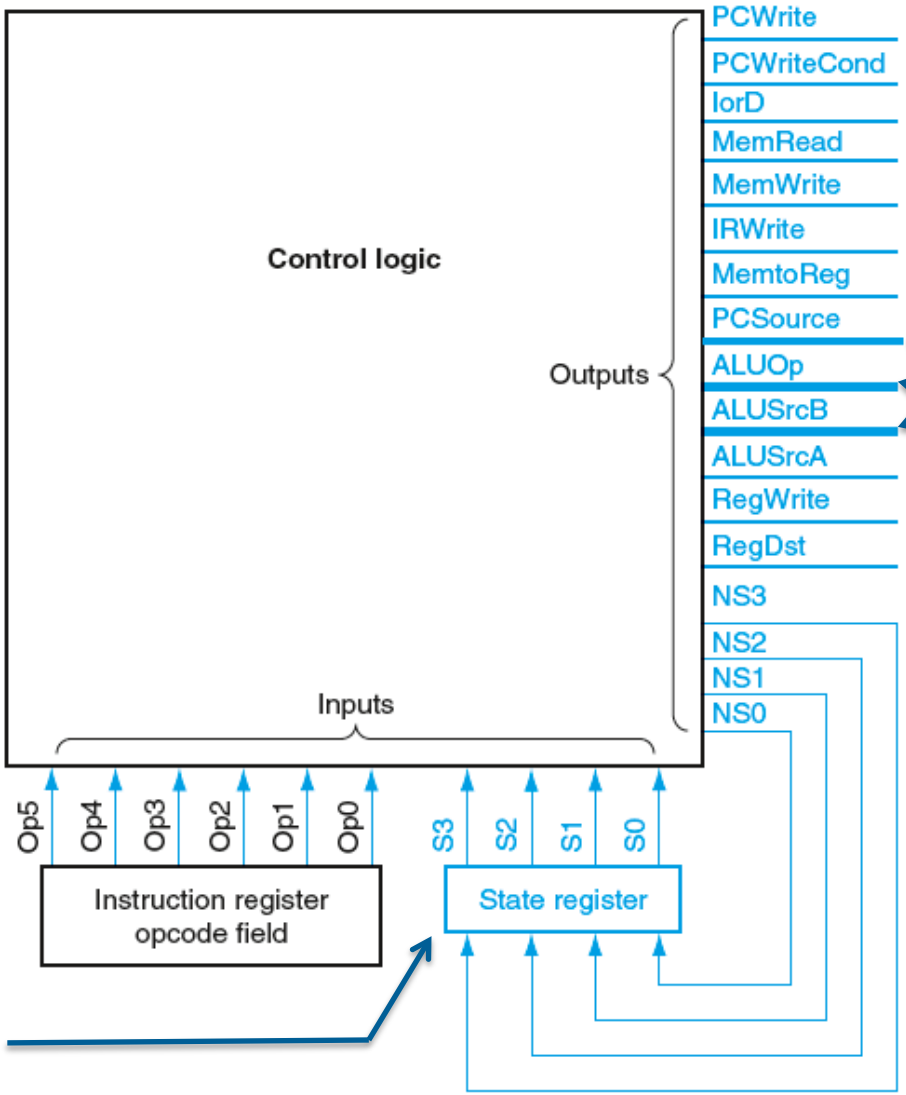
MIPS Control Unit



2-bit signals
others are 1 bit each

State register holds state number

MIPS Control Unit

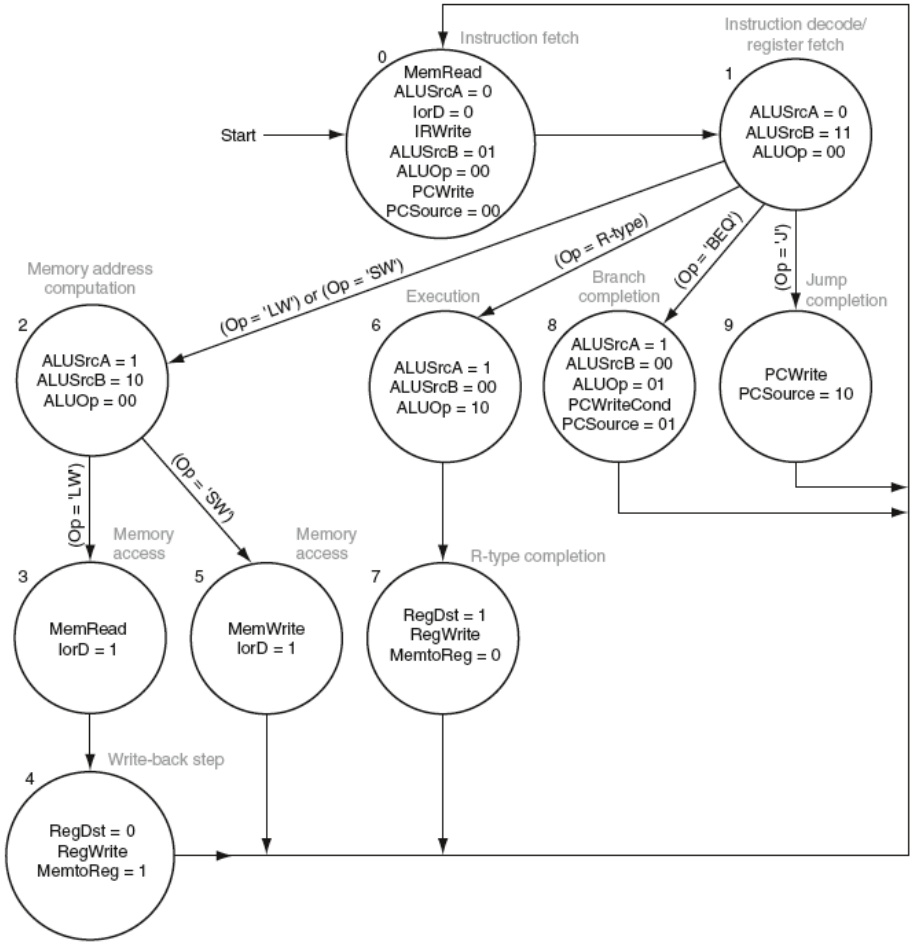
State alone is sufficient to determine output control signals

Output	Current states	Op
PCWrite	state0 + state9	
PCWriteCond	state8	
IorD	state3 + state5	
MemRead	state0 + state3	
MemWrite	state5	
IRWrite	state0	
MemtoReg	state4	
PCSource1	state9	
PCSource0	state8	
ALUOp1	state6	
ALUOp0	state8	
ALUSrcB1	state1 +state2	
ALUSrcB0	state0 + state1	
ALUSrcA	state2 + state6 + state8	
RegWrite	state4 + state7	
RegDst	state7	

MIPS Control Unit

State alone determines output control signals

Output	Current states
PCWrite	state0 + state9
PCWriteCond	state8
IorD	state3 + state5
MemRead	state0 + state3
MemWrite	state5
IRWrite	state0
MemtoReg	state4
PCSource1	state9
PCSource0	state8
ALUOp1	state6
ALUOp0	state8
ALUSrcB1	state1 +state2
ALUSrcB0	state0 + state1
ALUSrcA	state2 + state6 + state8
RegWrite	state4 + state7
RegDst	state7



MIPS Control Unit

Opcode as well as state are needed to determine next state

Output	Current states	Op
NextState0	state4 + state5 + state7 + state8 + state9	
NextState1	state0	
NextState2	state1	(Op = 'lw') + (Op = 'sw')
NextState3	state2	(Op = 'lw')
NextState4	state3	
NextState5	state2	(Op = 'sw')
NextState6	state1	(Op = 'R-type')
NextState7	state6	
NextState8	state1	(Op = 'beq')
NextState9	state1	(Op = 'jmp')

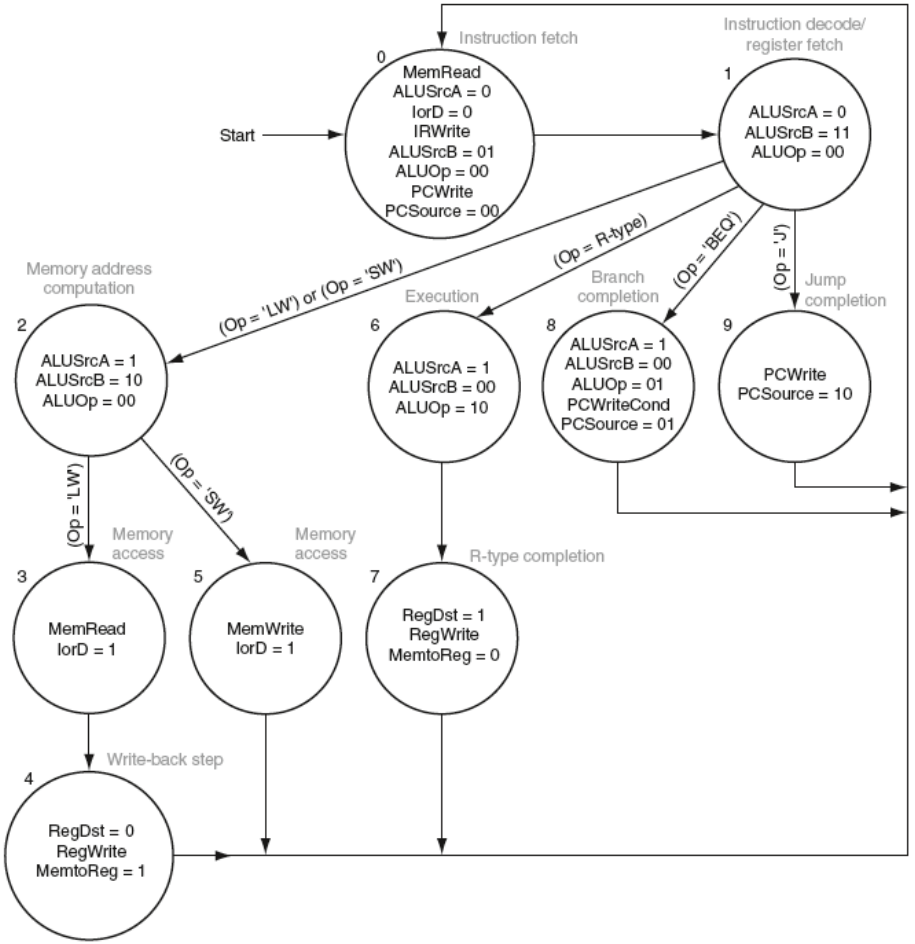
Four bits are needed for state number (since there are 10 states)

MIPS Control Unit

Output	Current states	Op
NextState0	state4 + state5 + state7 + state8 + state9	
NextState1	state0	
NextState2	state1	(Op = 'lw') + (Op = 'sw')
NextState3	state2	(Op = 'lw')
NextState4	state3	
NextState5	state2	(Op = 'sw')
NextState6	state1	(Op = 'R-type')
NextState7	state6	
NextState8	state1	(Op = 'beq')
NextState9	state1	(Op = 'jmp')

Opcode as well as state are needed to determine next state

Four bits are needed for state number (since there are 10 states)



Mapping Control to Gates

Outputs	Input values (S[3–0])									
	0000	0001	0010	0011	0100	0101	0110	0111	1000	1001
PCWrite	1	0	0	0	0	0	0	0	0	1
PCWriteCond	0	0	0	0	0	0	0	0	1	0
IorD	0	0	0	1	0	1	0	0	0	0
MemRead	1	0	0	1	0	0	0	0	0	0
MemWrite	0	0	0	0	0	1	0	0	0	0
IRWrite	1	0	0	0	0	0	0	0	0	0
MemtoReg	0	0	0	0	1	0	0	0	0	0
PCSource1	0	0	0	0	0	0	0	0	0	1
PCSource0	0	0	0	0	0	0	0	0	1	0
ALUOp1	0	0	0	0	0	0	1	0	0	0
ALUOp0	0	0	0	0	0	0	0	0	1	0
ALUSrcB1	0	1	1	0	0	0	0	0	0	0
ALUSrcB0	1	1	0	0	0	0	0	0	0	0
ALUSrcA	0	0	1	0	0	0	1	0	1	0
RegWrite	0	0	0	0	1	0	0	1	0	0
RegDst	0	0	0	0	0	0	0	1	0	0

Truth table gives the 16 control signals as function of 4-bit state

MIPS Control Unit

The 16 control signals could be read from a lookup table in ROM

Lower 4 bits of the address	Bits 19–4 of the word
0000	1001010000001000
0001	0000000000011000
0010	0000000000010100
0011	0011000000000000
0100	0000001000000010
0101	0010100000000000
0110	0000000001000100
0111	0000000000000011
1000	0100000010100100
1001	1000000100000000

Use state number as index or address in ROM LUT (lookup table)

MIPS Control Unit

Use state number (as row) and opcode (as column) to get next state number from second ROM LUT

	Op [5–0]					
Current state S[3–0]	**000000 (R-format)**	**000010 (jmp)**	**000100 (beq)**	**100011 (lw)**	**101011 (sw)**	**Any other value**
0000	0001	0001	0001	0001	0001	0001
0001	0110	1001	1000	0010	0010	Illegal
0010	XXXX	XXXX	XXXX	0011	0101	Illegal
0011	0100	0100	0100	0100	0100	Illegal
0100	0000	0000	0000	0000	0000	Illegal
0101	0000	0000	0000	0000	0000	Illegal
0110	0111	0111	0111	0111	0111	Illegal
0111	0000	0000	0000	0000	0000	Illegal
1000	0000	0000	0000	0000	0000	Illegal
1001	0000	0000	0000	0000	0000	Illegal

Undefined state/opcode combinations are “illegal”

MIPS Control Unit

- State number alone is sufficient to determine the 16 control bits
- The 4-bit state number and 6-bit opcode form a 10-bit address
- 10-bit address determines control bits as well as next state
- Instruction execution corresponds to a sequence of states
- Control bits output in each state direct datapath actions
- Each state is one step in the execution and takes 1 clock cycle
- Other control unit implementations will be described next.