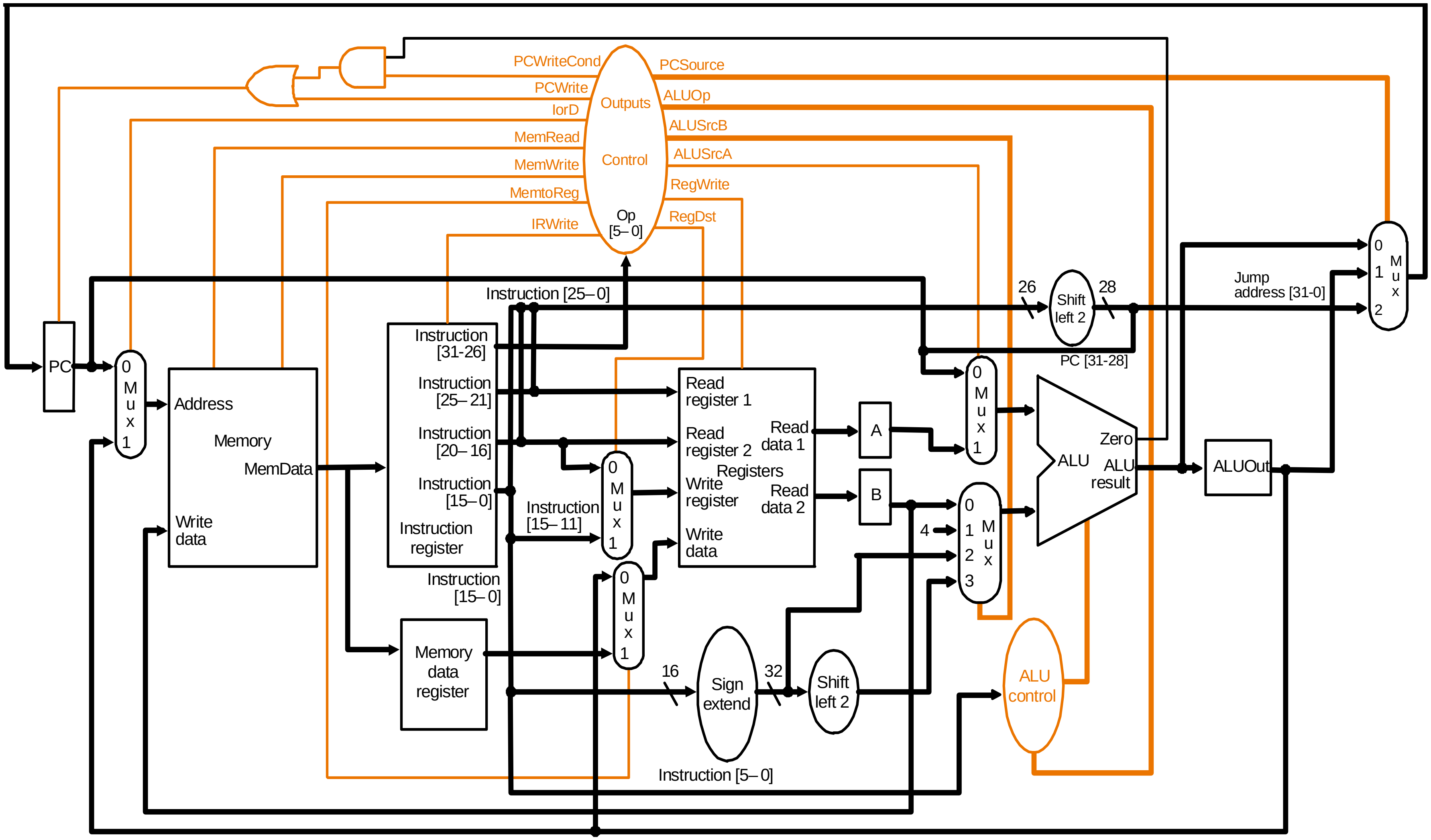
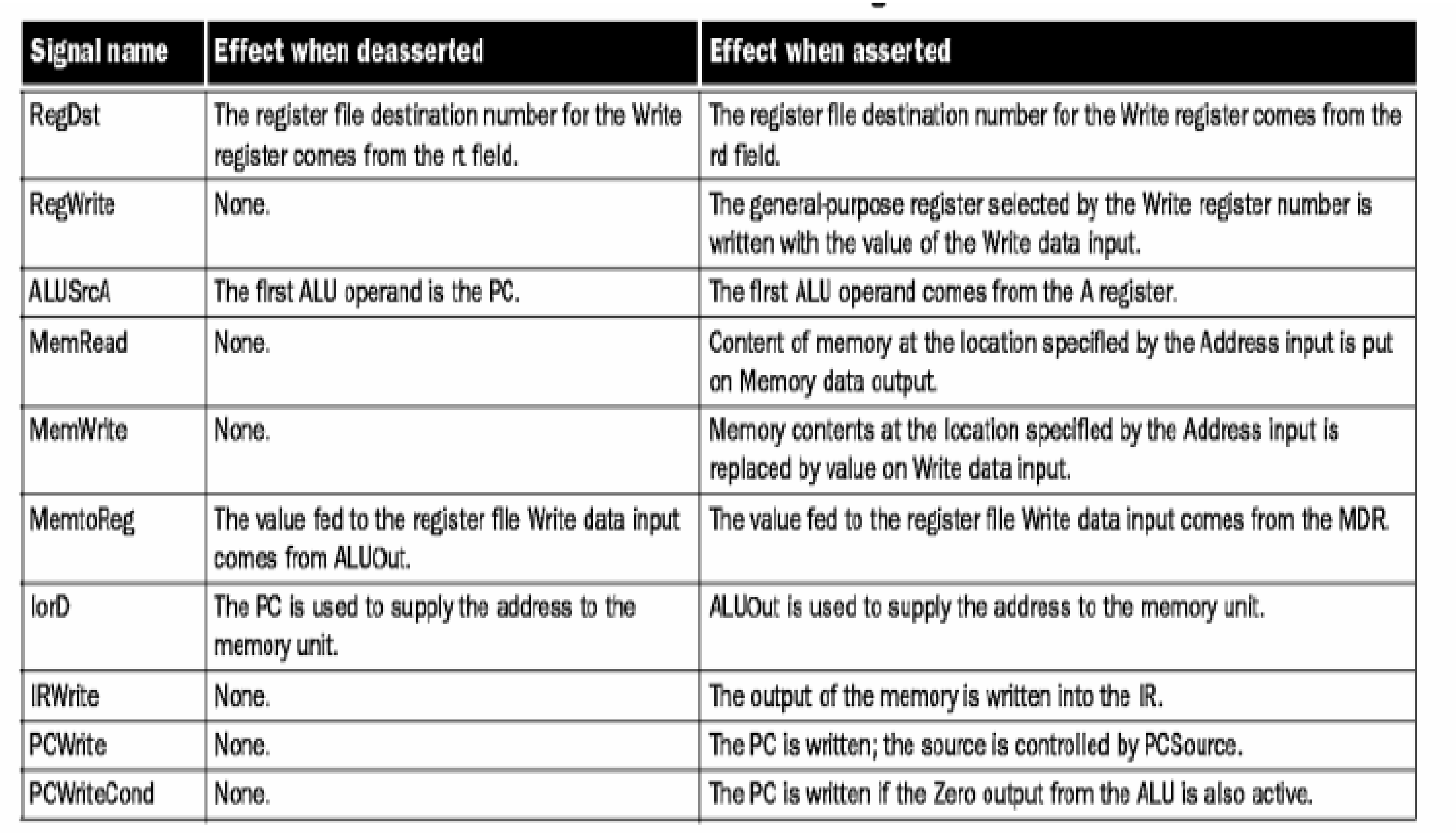
COMPUTER ORGANIZATION (IS F242)

LECT 32: MIPS ARCHITECTURE

Multicycle Datapath with Control

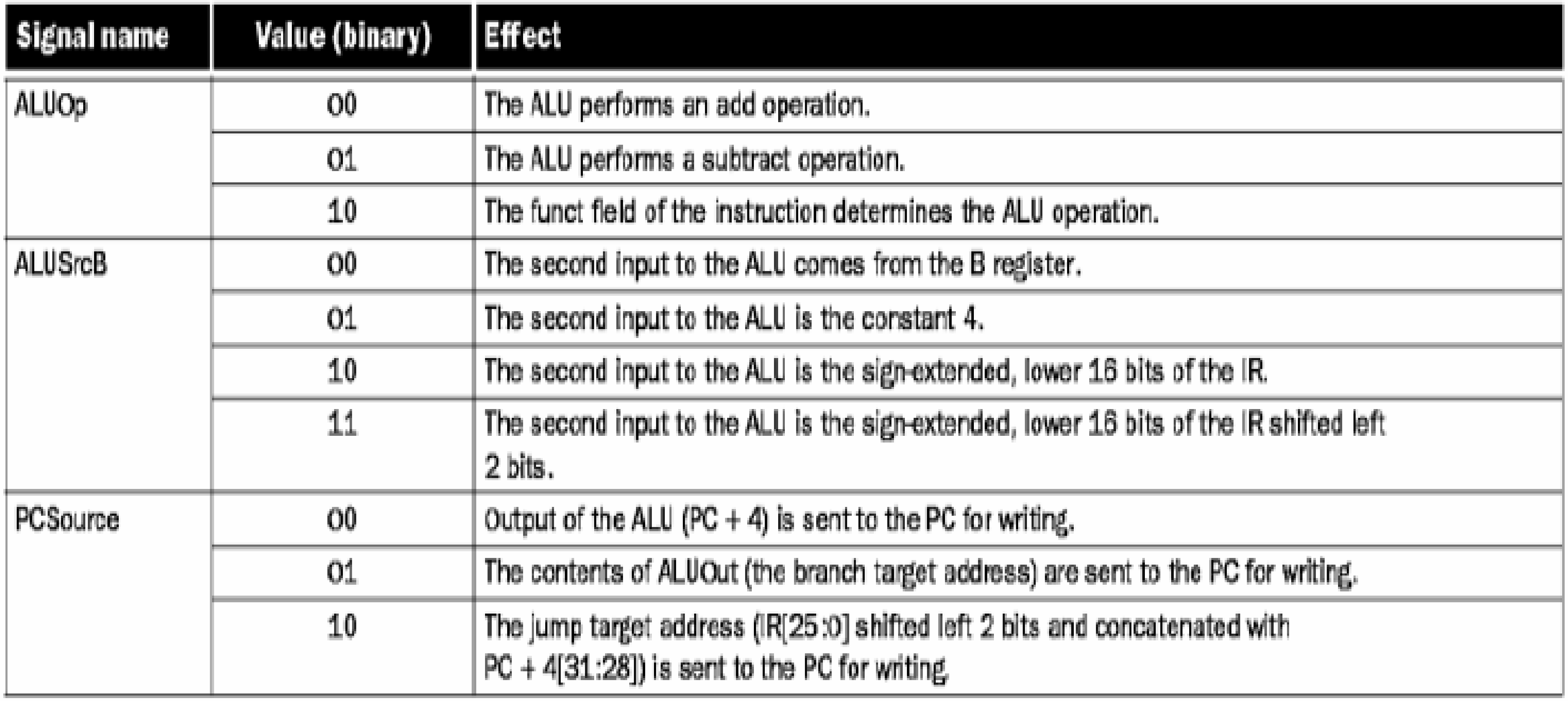


Actions of the 1 bit control signals



Signal name	Effect when deasserted	Effect when asserted
RegDst	The register file destination number for the Write register comes from the rt field.	The register file destination number for the Write register comes from the rd field.
RegWrite	None.	The general-purpose register selected by the Write register number is written with the value of the Write data input.
ALUSrcA	The first ALU operand is the PC.	The first ALU operand comes from the A register.
MemRead	None.	Content of memory at the location specified by the Address input is put on Memory data output
MemWrite	None.	Memory contents at the location specified by the Address input is replaced by value on Write data input.
MemtoReg	The value fed to the register file Write data input comes from ALUOut.	The value fed to the register file Write data input comes from the MDR.
IorD	The PC is used to supply the address to the memory unit.	ALUOut is used to supply the address to the memory unit.
IRWrite	None.	The output of the memory is written into the IR.
PCWrite	None.	The PC is written; the source is controlled by PCSource.
PCWriteCond	None.	The PC is written if the Zero output from the ALU is also active.

Actions of the 2 bit control signals



Signal name	Value (binary)	Effect
ALUOp	00	The ALU performs an add operation.
	01	The ALU performs a subtract operation.
	10	The funct field of the instruction determines the ALU operation.
ALUSrcB	00	The second input to the ALU comes from the B register.
	01	The second input to the ALU is the constant 4.
	10	The second input to the ALU is the sign-extended, lower 16 bits of the IR.
	11	The second input to the ALU is the sign-extended, lower 16 bits of the IR shifted left 2 bits.
PCSource	00	Output of the ALU (PC + 4) is sent to the PC for writing.
	01	The contents of ALUOut (the branch target address) are sent to the PC for writing.
	10	The jump target address (IR[25:0] shifted left 2 bits and concatenated with PC + 4[31:28]) is sent to the PC for writing.

Drawbacks of Multi Cycle Datapath

- Assume the following
  - Memory access = **2ns**
  - Register Read/Write = **1ns**
  - ALU function = **2ns**

Load	**5 cycles**
Add	**4 cycles**
Store	**4 cycles**
Branch	**3 cycles**

Cycle Time for Single cycle Implementation **8ns**

Cycle Time for Multi cycle Implementation **2ns**

Arithmetic	**48%**
Loads	**22%**
Stores	**11%**
Branch	**19%**

Multi cycle - Assume a program with a million instructions

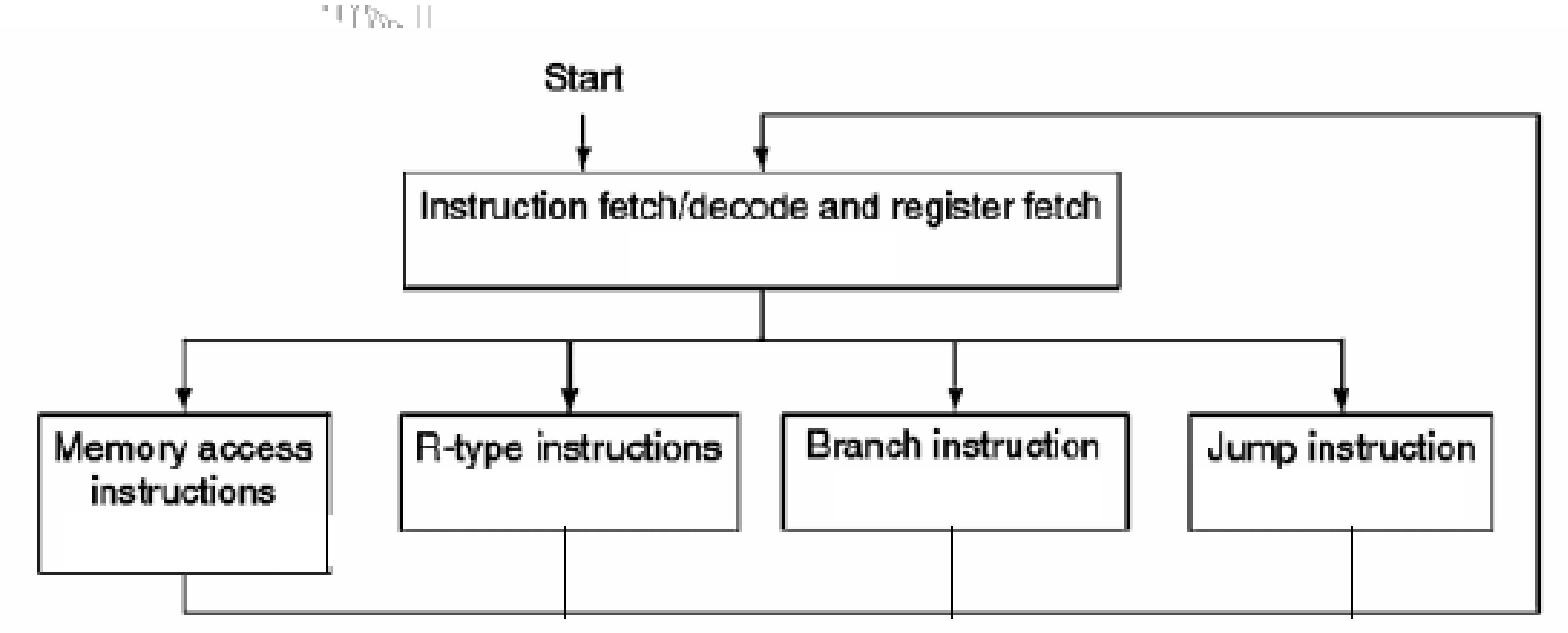
Avg. CPI = 0.48*4 + 0.22*5 + 0.11*4 + 0.19*3 = 4.03 cycles

T = I * CPI * Cycle Time = 10^6 * 4.03 * 2 = 8.06 ms

Defining the control

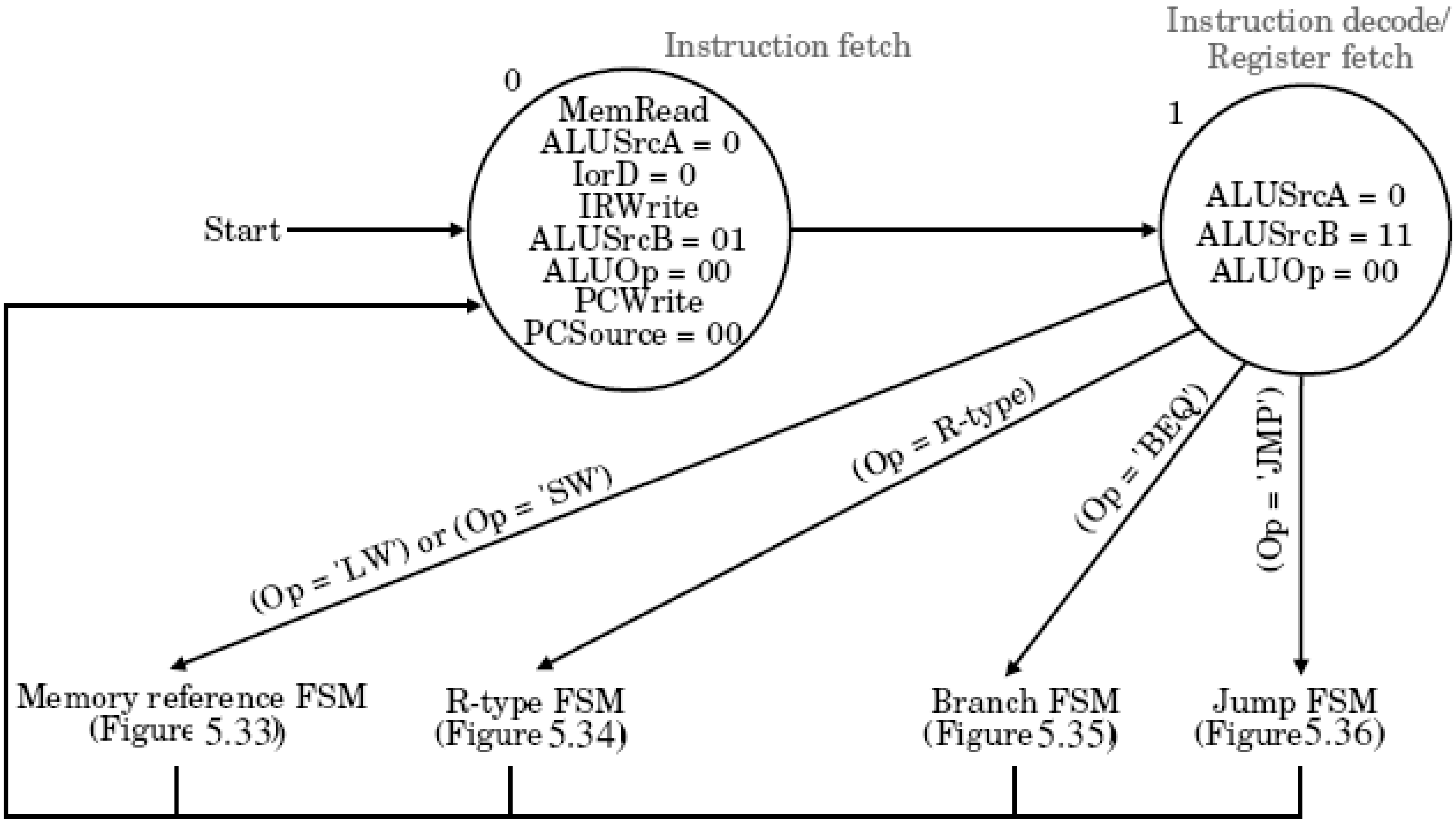
- Control unit design
  - Hardwired finite state machines
  - Microprogrammed ➔ A symbolic representation of control in the form of instructions that are executed on a simple machine
- Finite state machines
  - set of state + directions
- Next state function
  - combinational function that maps the current state and the inputs to the next state

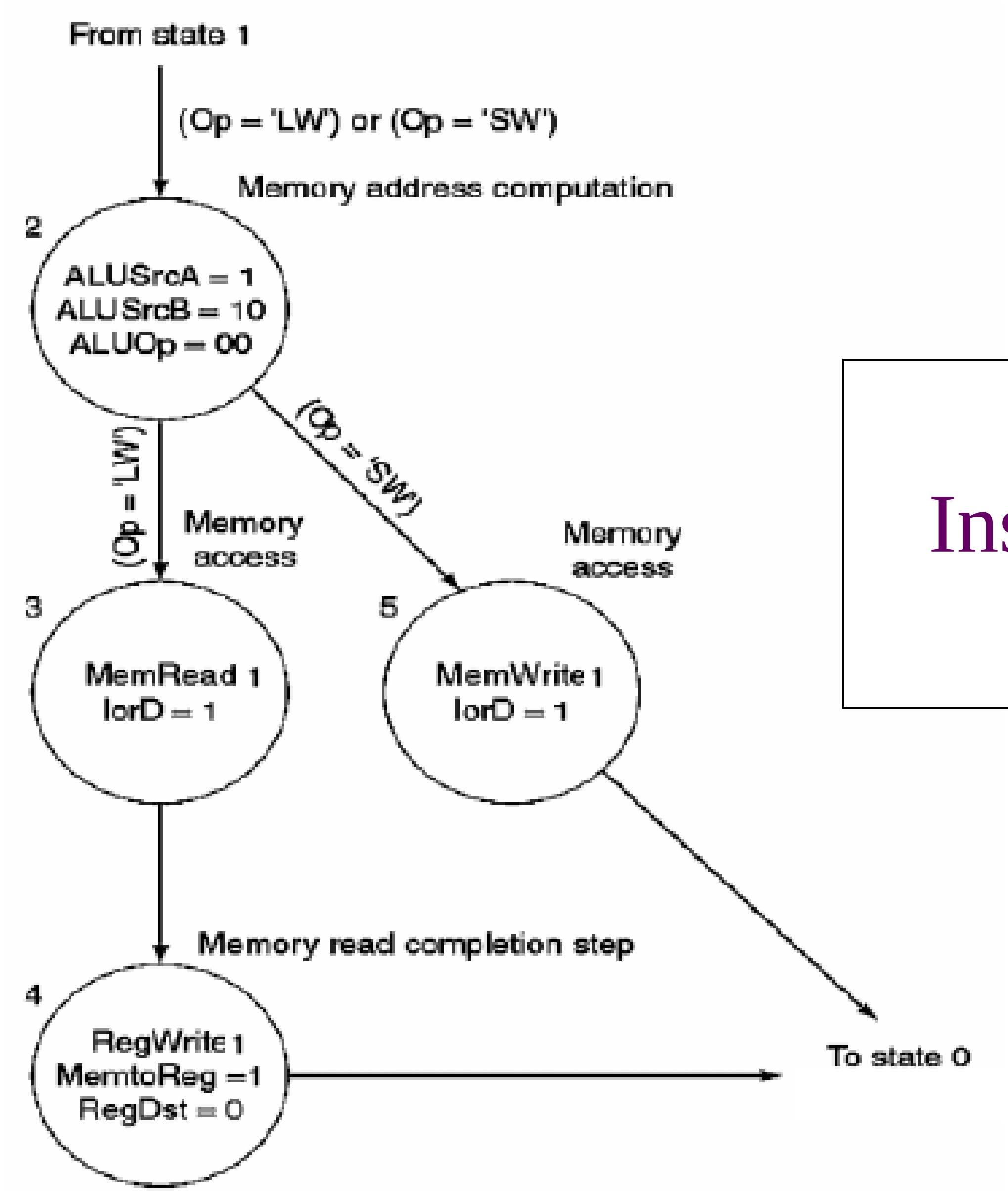
Control circuit



Start
Instruction fetch/decode and register fetch
Memory access instructions
R-type instructions
Branch instruction
Jump instruction

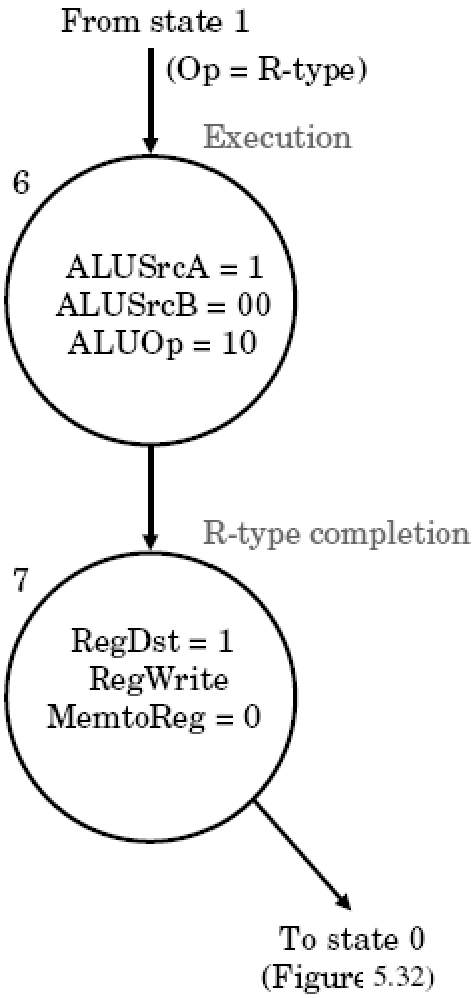
Instruction Fetch and Decode FSM States





Load / Store Instructions FSM States

R - type


From state 1

(Op = R-type)

Execution

6

ALUSrcA = 1
ALUSrcB = 00
ALUOp = 10

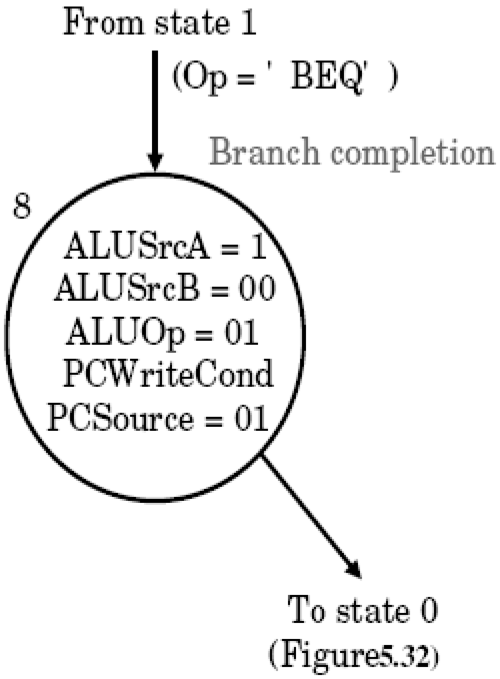
R-type completion

7

RegDst = 1
RegWrite
MemtoReg = 0

To state 0
(Figure 5.32)

Branch


From state 1

(Op = ' BEQ')

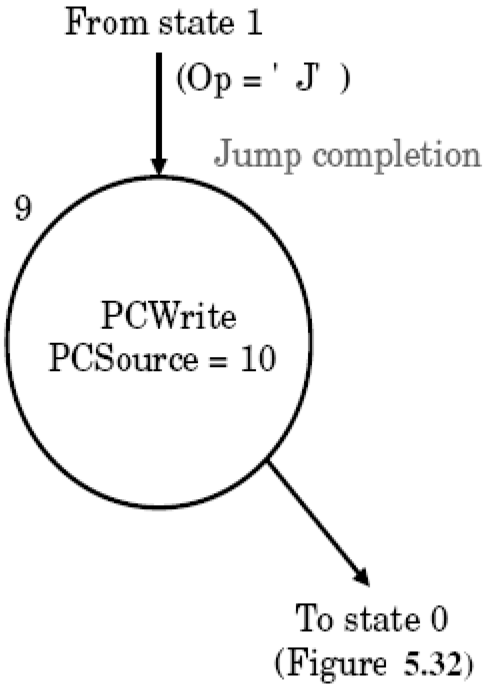
Branch completion

8

ALUSrcA = 1
ALUSrcB = 00
ALUOp = 01
PCWriteCond
PCSource = 01

To state 0
(Figure5.32)

Jump


From state 1

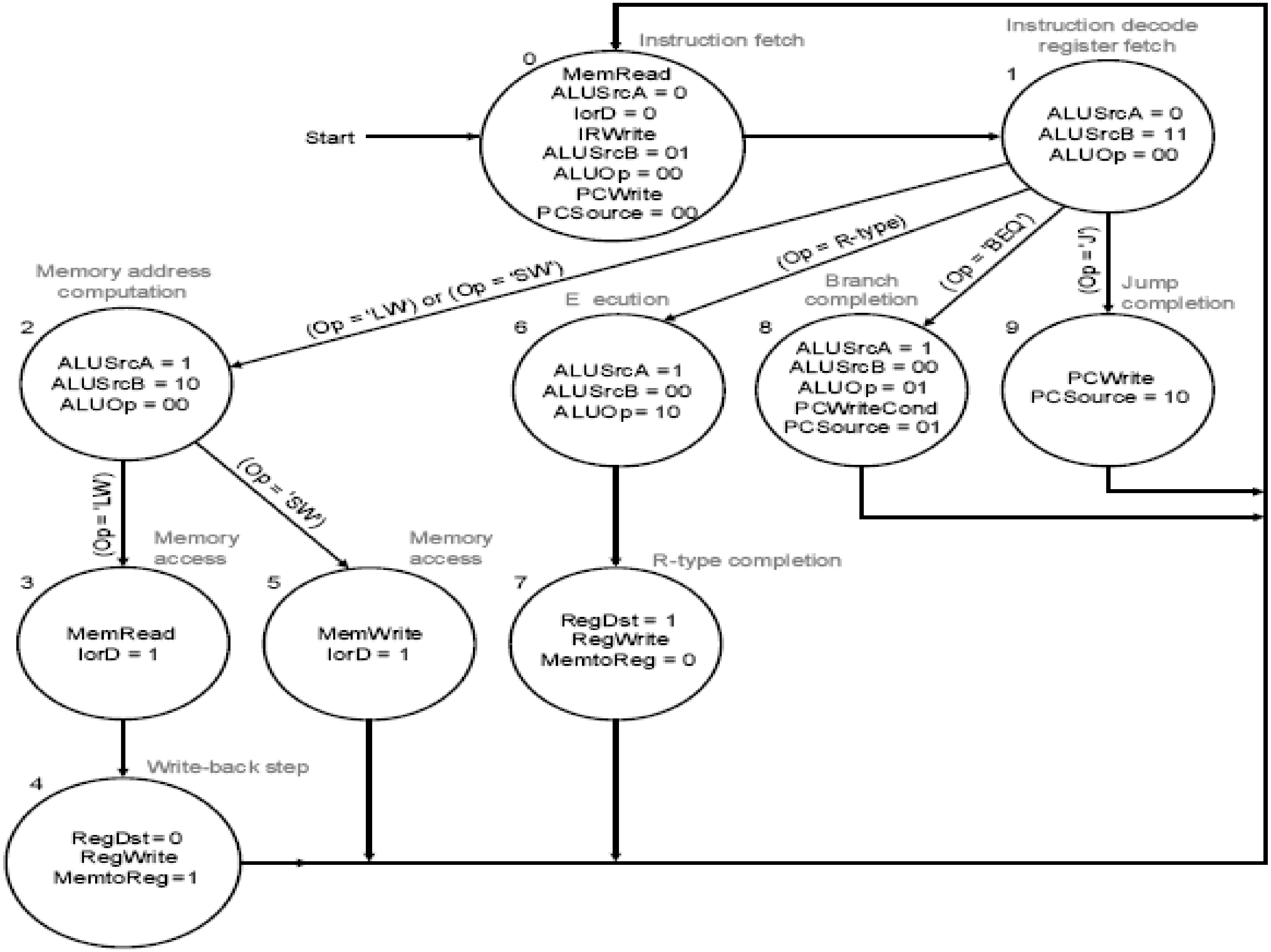
(Op = ' J')

Jump completion

9

PCWrite
PCSource = 10

To state 0
(Figure 5.32)


Start
Instruction fetch
0
MemRead
ALUSrcA = 0
IorD = 0
IRWrite
ALUSrcB = 01
ALUOp = 00
PCWrite
PCSource = 00
Instruction decode
register fetch
1
ALUSrcA = 0
ALUSrcB = 11
ALUOp = 00
(Op = 'LW') or (Op = 'SW')
(Op = R-type)
(Op = 'BEQ')
(Op = 'J')
Memory address
computation
2
ALUSrcA = 1
ALUSrcB = 10
ALUOp = 00
E ecution
6
ALUSrcA =1
ALUSrcB = 00
ALUOp= 10
Branch
completion
8
ALUSrcA = 1
ALUSrcB = 00
ALUOp= 01
PCWriteCond
PCSource = 01
Jump
completion
9
PCWrite
PCSource = 10
(Op = 'LW')
(Op = 'SW')
Memory
access
Memory
access
R-type completion
3
MemRead
IorD = 1
5
MemWrite
IorD = 1
7
RegDst = 1
RegWrite
MemtoReg = 0
Write-back step
4
RegDst=0
RegWrite
MemtoReg=1

Exceptions

- *Exception:*
  - is an anomalous event arising from within the processor, such as arithmetic overflow
- *Interrupt:*
  - is an event that causes an unexpected change in control flow. Interrupts are assumed to originate outside the processor, for example, an I/O request
- Patterson and Hennessey's convention
  - An interrupt is an externally caused event, and an exception one of all other events that cause unexpected control flow in a program

Interrupt Vs Exception in MIPS

Type of Event	From where?	MIPS terminology
I/O device request	External	Interrupt
Invoke OS from user program	Internal	Exception
Arithmetic overflow	Internal	Exception
Using an undefined instruction	Internal	Exception
Hardware malfunctions	Either	Exception or Interrupt