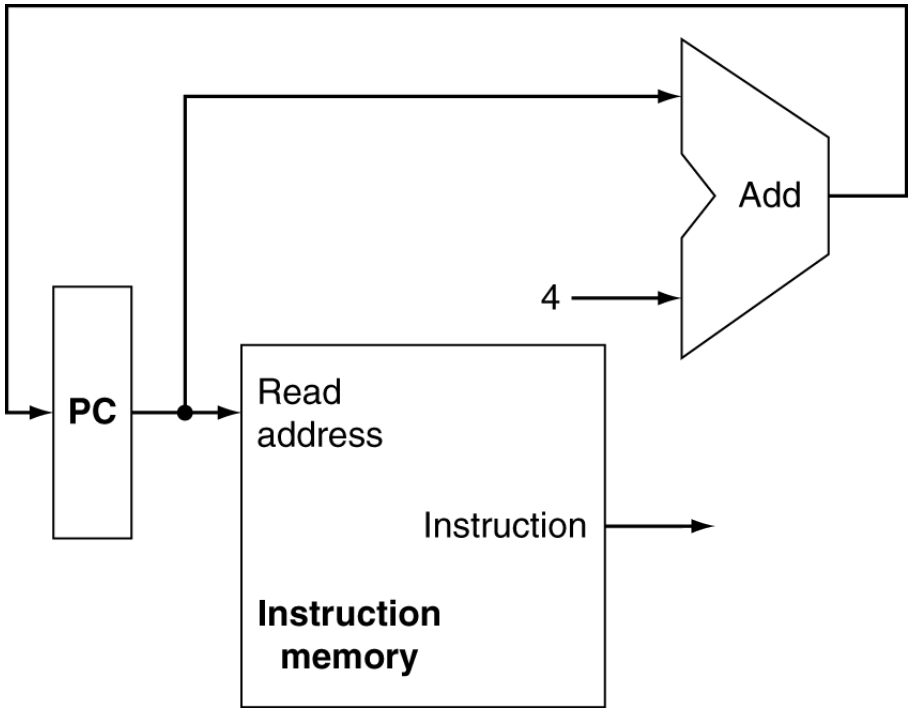
CSSE232
Computer Architecture I

Datapath Control

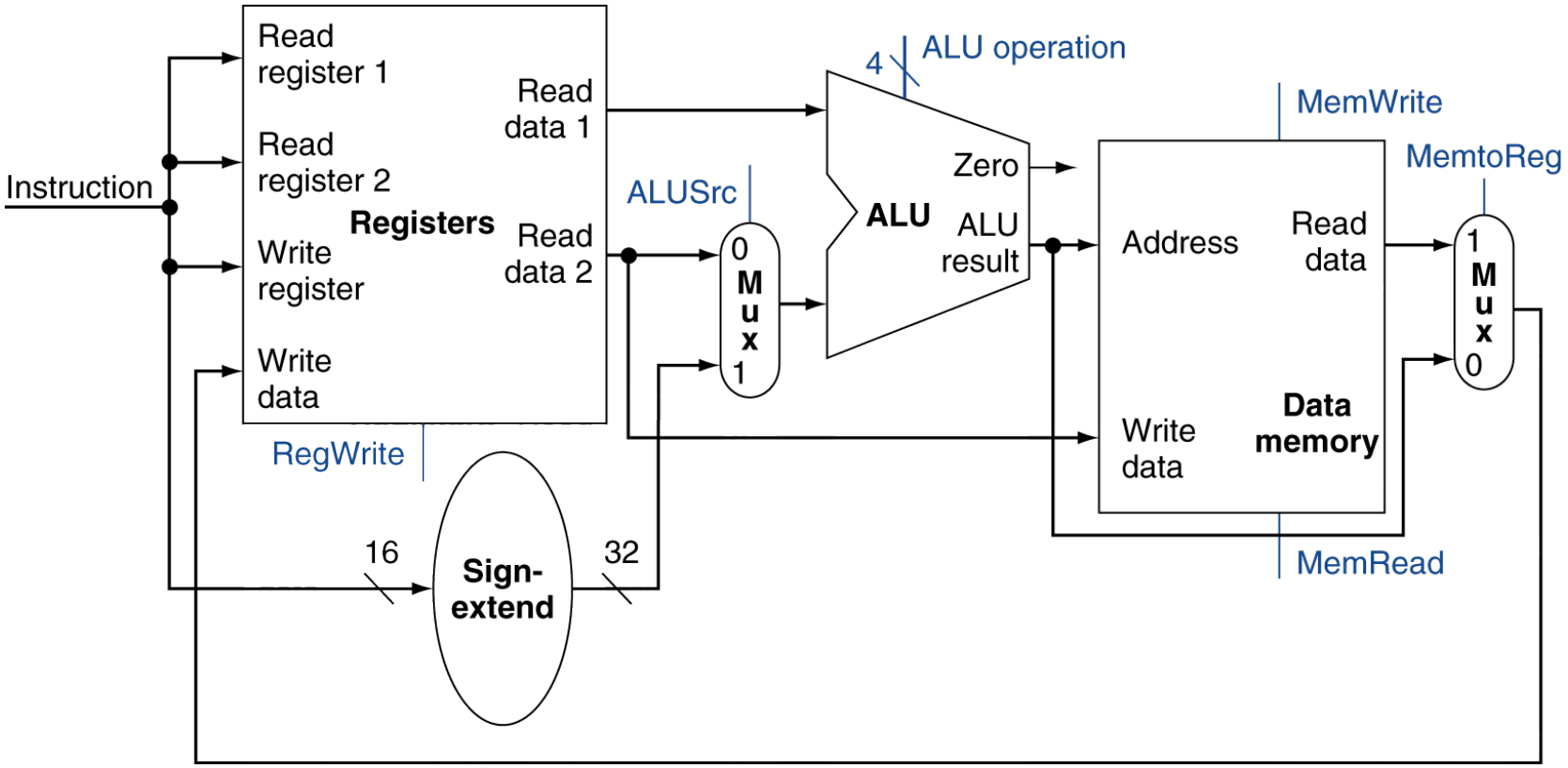
Outline

- Review of single-cycle datapath
- Performance
- Control

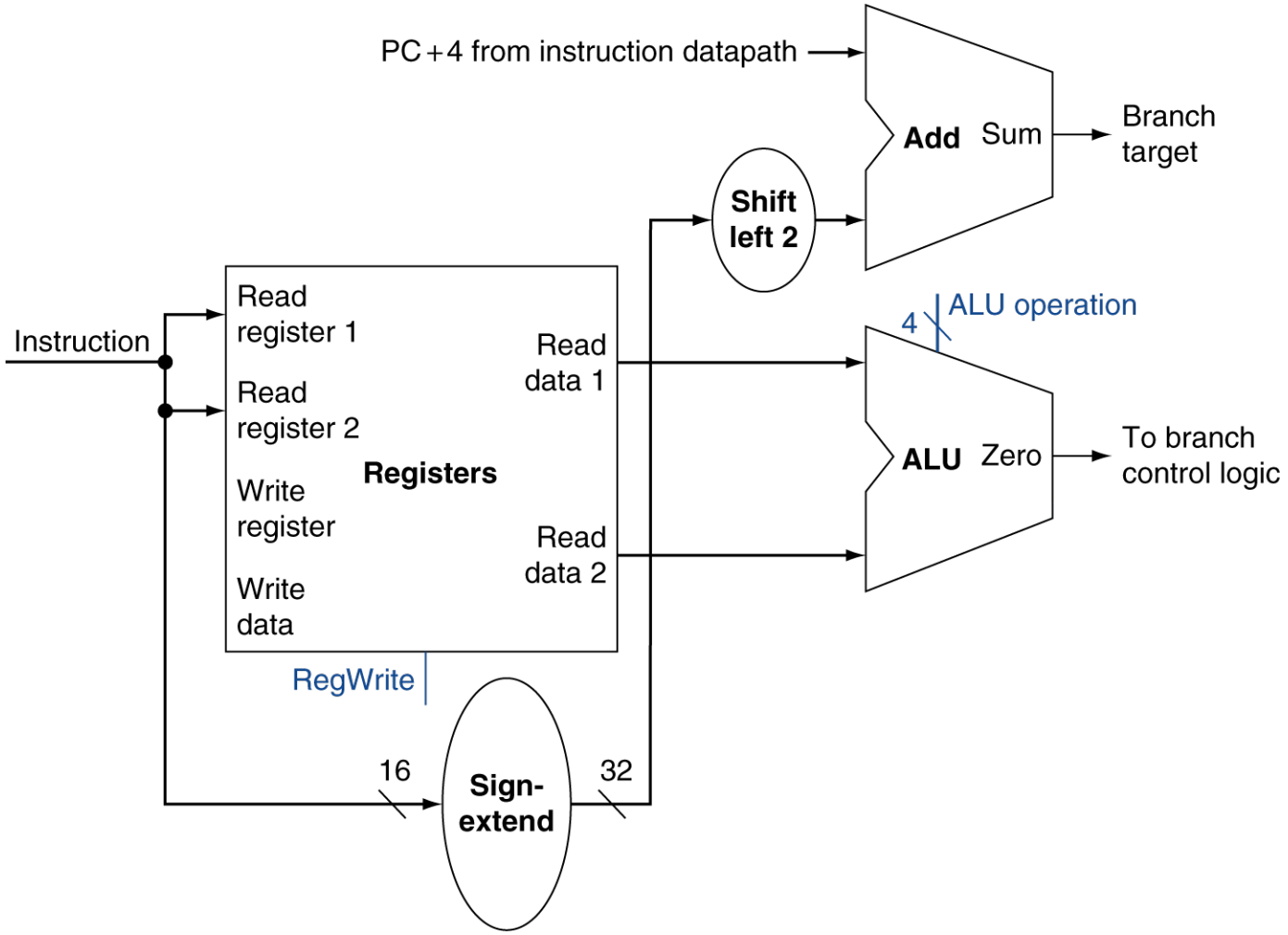
Fetch Datapath



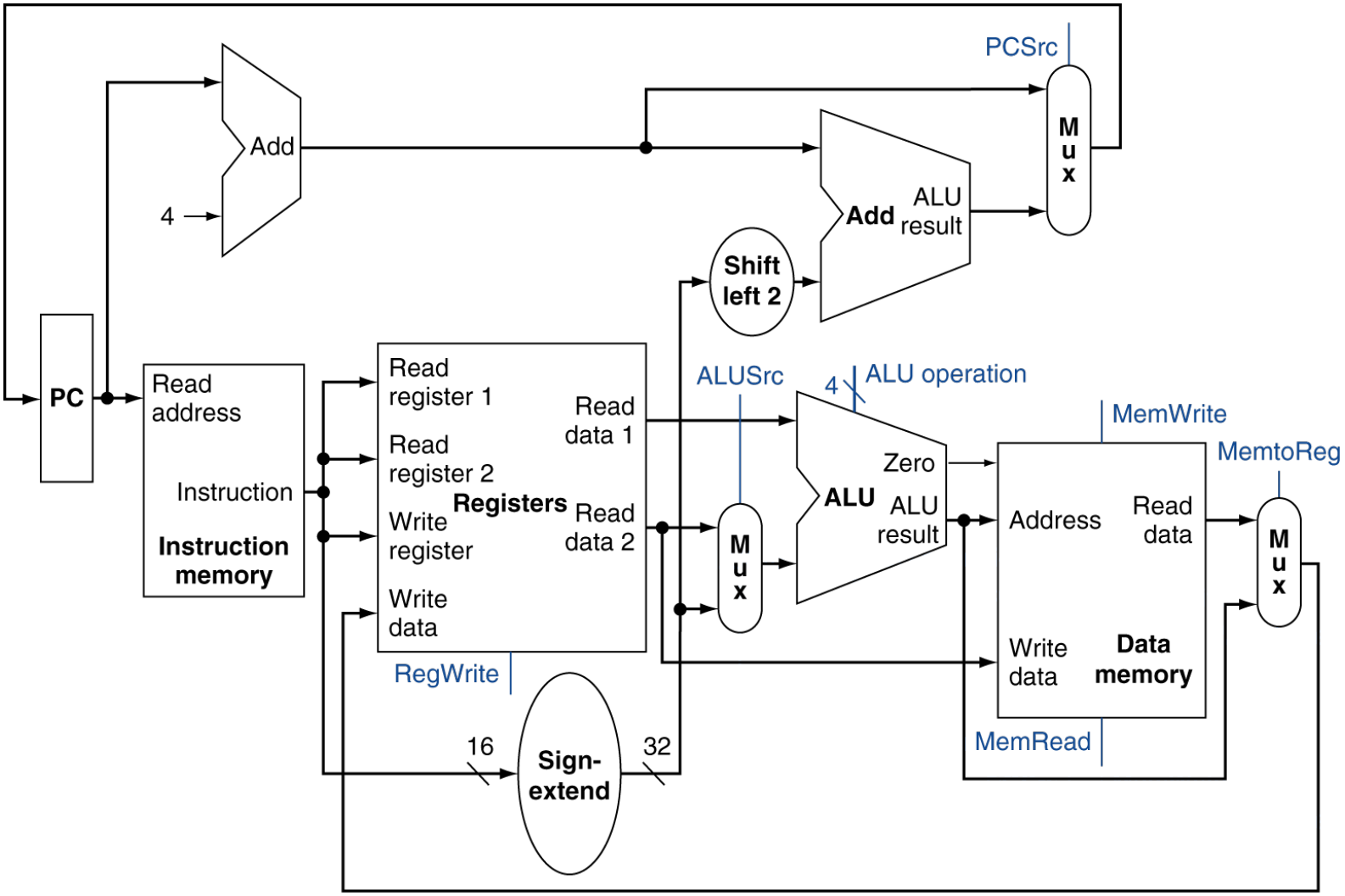
R-Type/Load/Store Datapath



Branch Instructions



Single-cycle Datapath



Different Subsets of Instructions

- Memory reference: `lw`, `sw`
- Arithmetic/logical: `add`, `sub`, `and`, `or`, `slt`
- Control transfer: `beq`

Two programs…

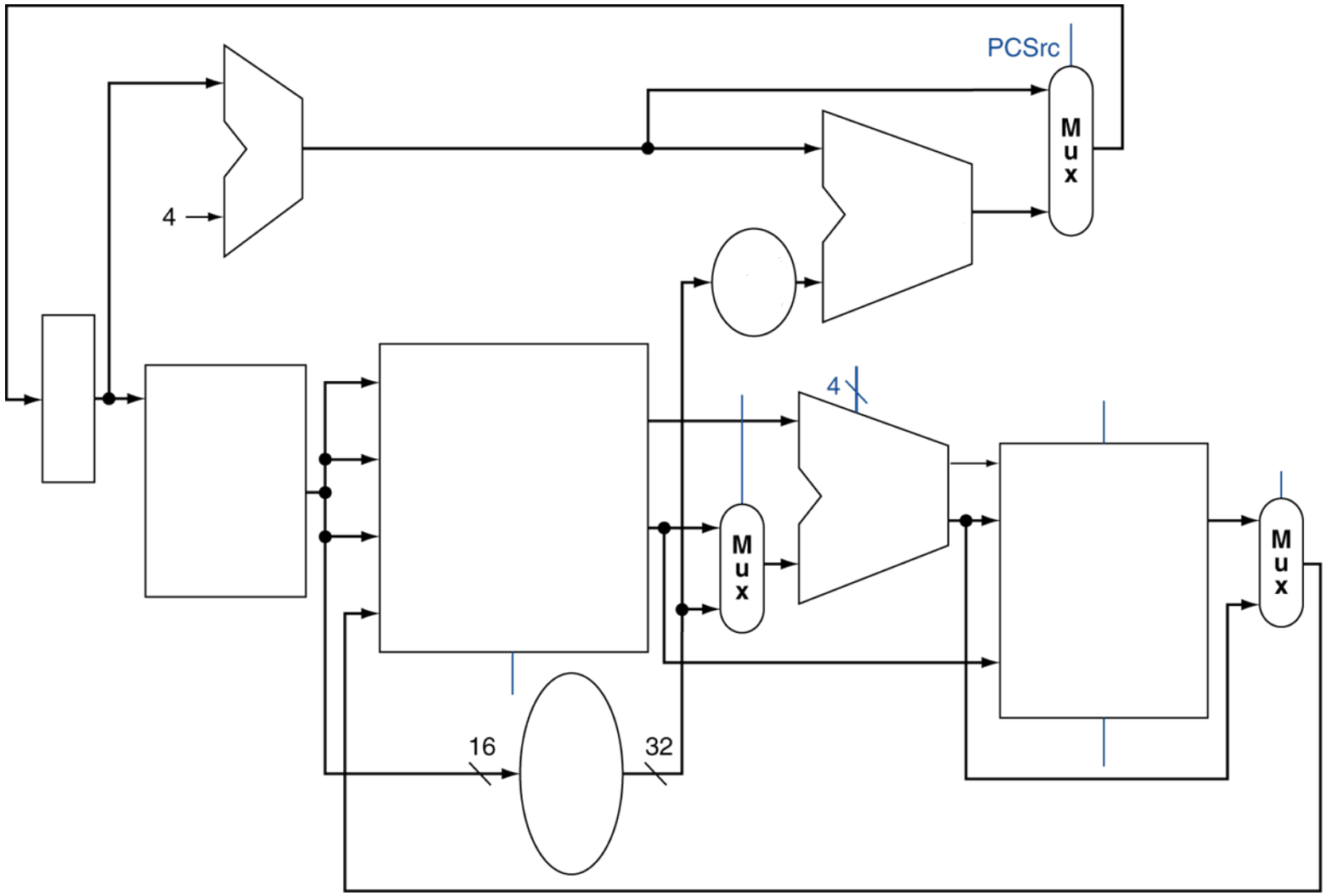
- Program 1

```
lw $t0 4($t1)
add $t0 $t0 $t2
beq $t0 $zero exit
```

- Program 2

```
sub $s1 $s1 $s2
sw $s1 0($t0)
beq $s1 $t2 loop
```

Blank Datapath



Datapath Exercise

- Trace instructions through datapath
  - Label used datapath components
  - Trace path through datapath

- Write your name at the top
- And the program you used

Control overview

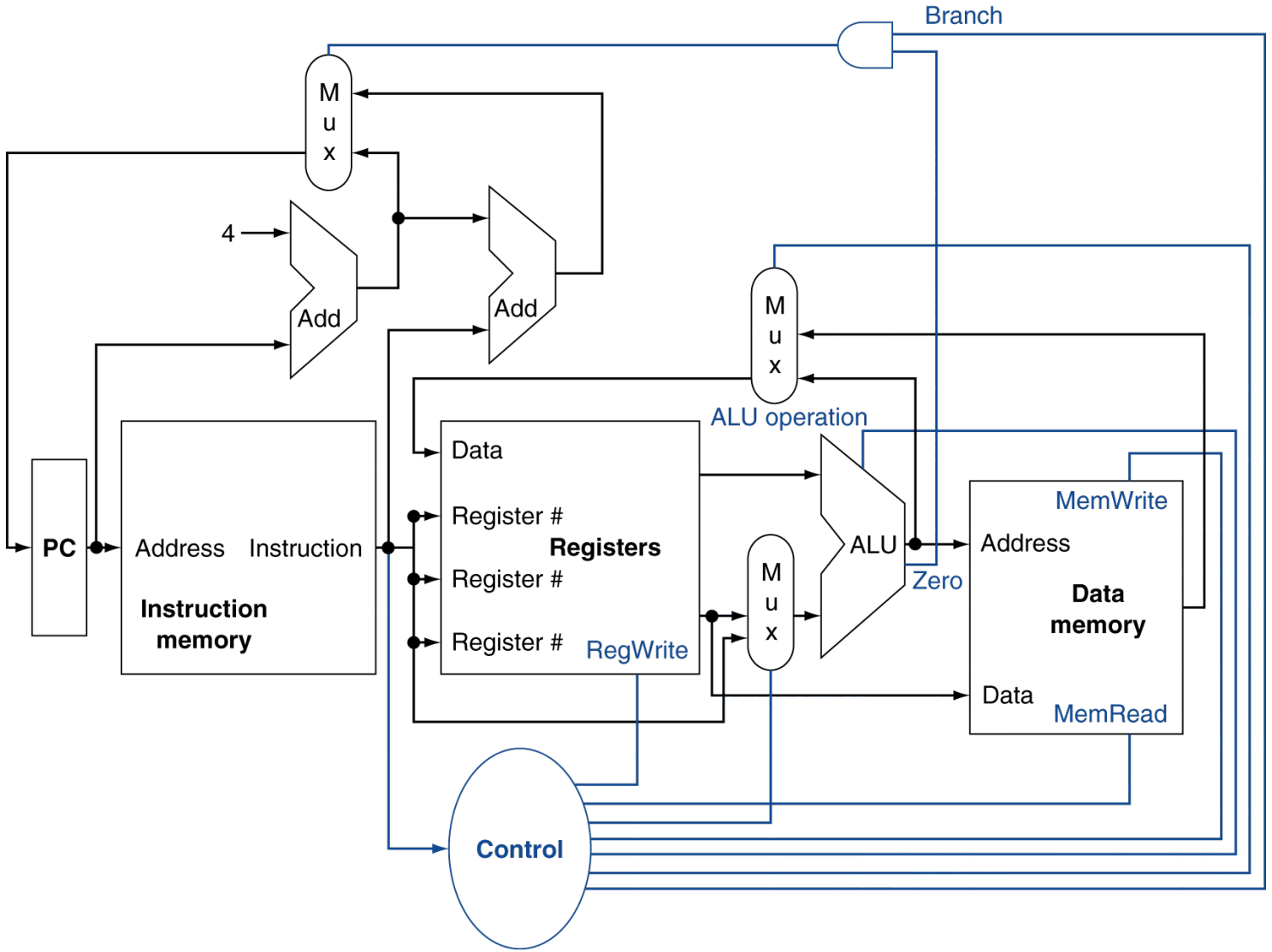
- What does control do?

- What directs control (what is its input)?

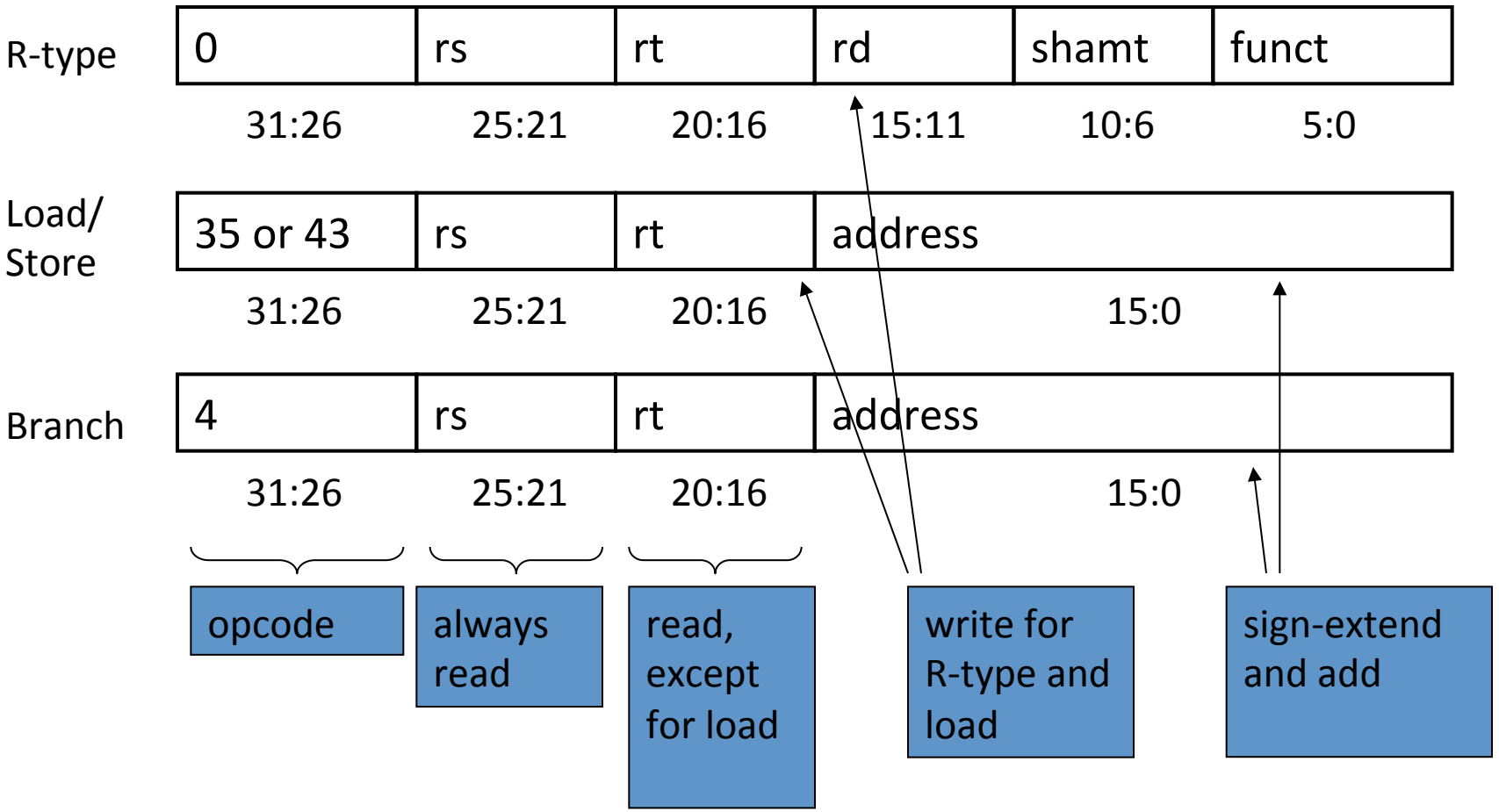
Control overview

- Directs processor operation
  - Controls muxes
  - Controls ALU input
  - Controls memory read/write
- Receives input from
  - Instruction opcode
  - Instruction function

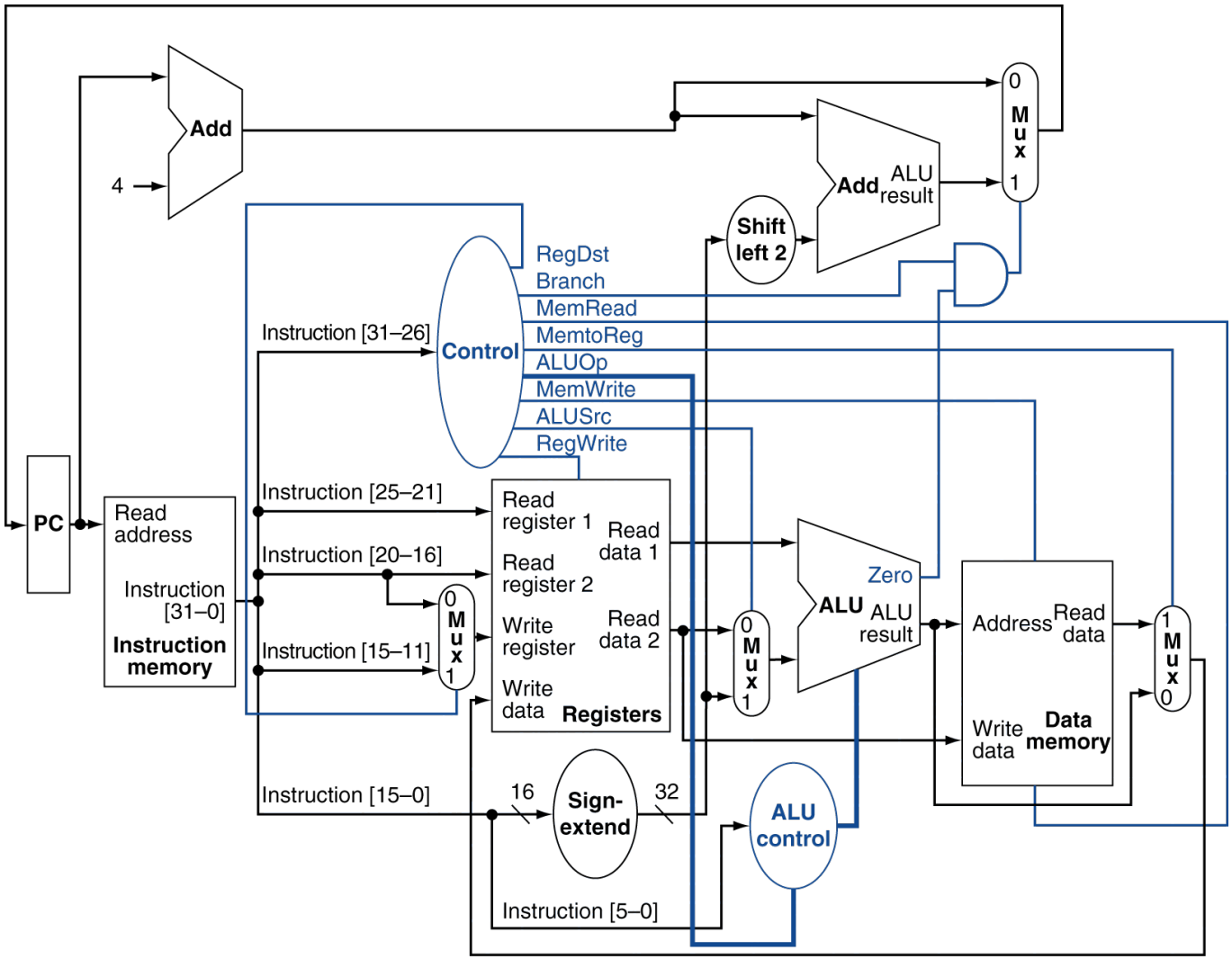
Control

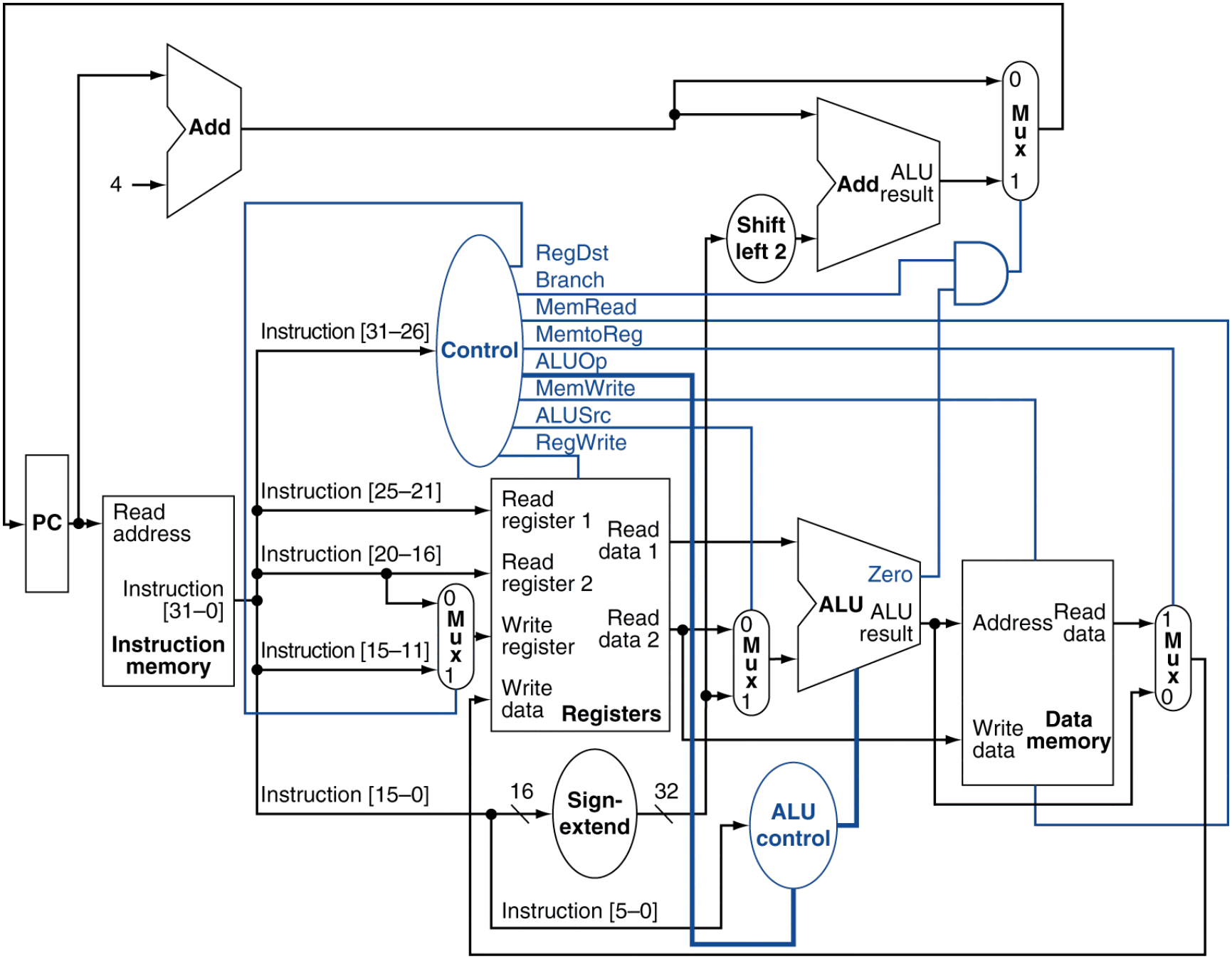

Branch
M
u
x
4
Add
Add
M
u
x
ALU operation
PC
Address
Instruction
Instruction
memory
Data
Register #
Registers
Register #
Register #
RegWrite
M
u
x
ALU
Zero
MemWrite
Address
Data
memory
Data
MemRead
Control

The Main Control Unit



Datapath With Control




Add
4
0
M
u
x
1
Add
ALU result
Shift left 2
RegDst
Branch
MemRead
MemtoReg
ALUOp
MemWrite
ALUSrc
RegWrite
Control
Instruction [31–26]
PC
Read address
Instruction [31–0]
Instruction memory
Instruction [25–21]
Instruction [20–16]
Instruction [15–11]
Instruction [15–0]
Instruction [5–0]
Read register 1
Read register 2
Write register
Write data
Read data 1
Read data 2
Registers
16
Sign-extend
32
ALU control
Zero
ALU
ALU result
Address
Read data
Write data
Data memory

Assignment

- Draw a table showing the values for the control lines for the three main types of instructions:
  - R-type – add, sub, or, etc
  - Data transfers – lw, sw
  - Branches – beq, bne

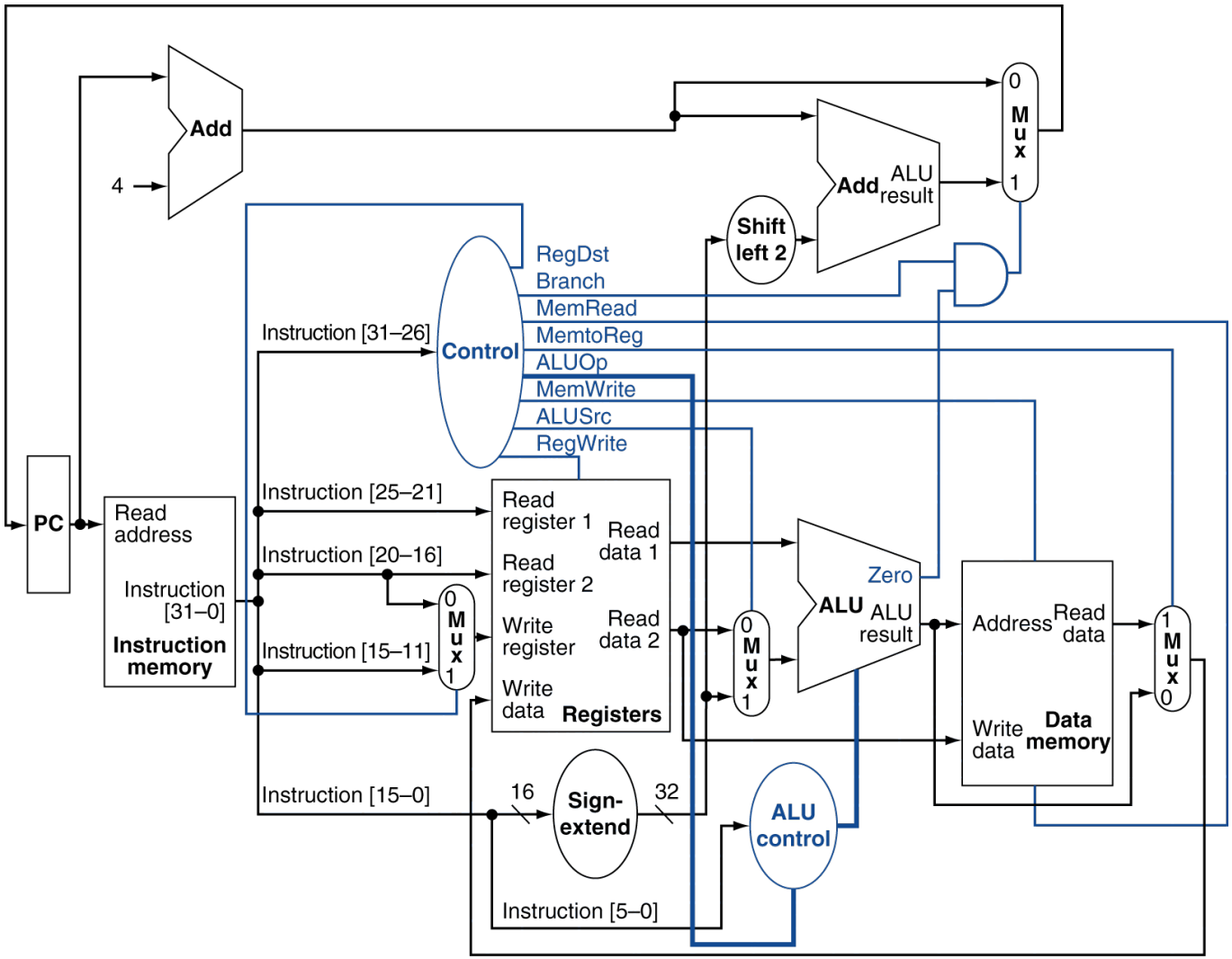
Inst/OPcode	RegDst	Branch	MemRead	MemWrite	MemToReg	ALUSrc	RegWrite
R-type (0)							
lw (35)							
sw (43)							
beq (4)							

Assignment

- Draw a table showing the values for the control lines for the three main types of instructions:
  - R-type – add, sub, or, etc
  - Data transfers – lw, sw
  - Branches – beq, bne

Inst/OPcode	RegDst	Branch	MemRead	MemWrite	MemToReg	ALUSrc	RegWrite
R-type (0)	1	0	0	0	0	0	1
lw (35)	0	0	1	0	1	1	1
sw (43)	X	0	0	1	X	1	0
beq (4)	X	1	0	0	X	0	0

Datapath With Control



ALU Control

- ALU used for
  - Load/Store: F = add
  - Branch: F = subtract
  - R-type: F depends on funct field

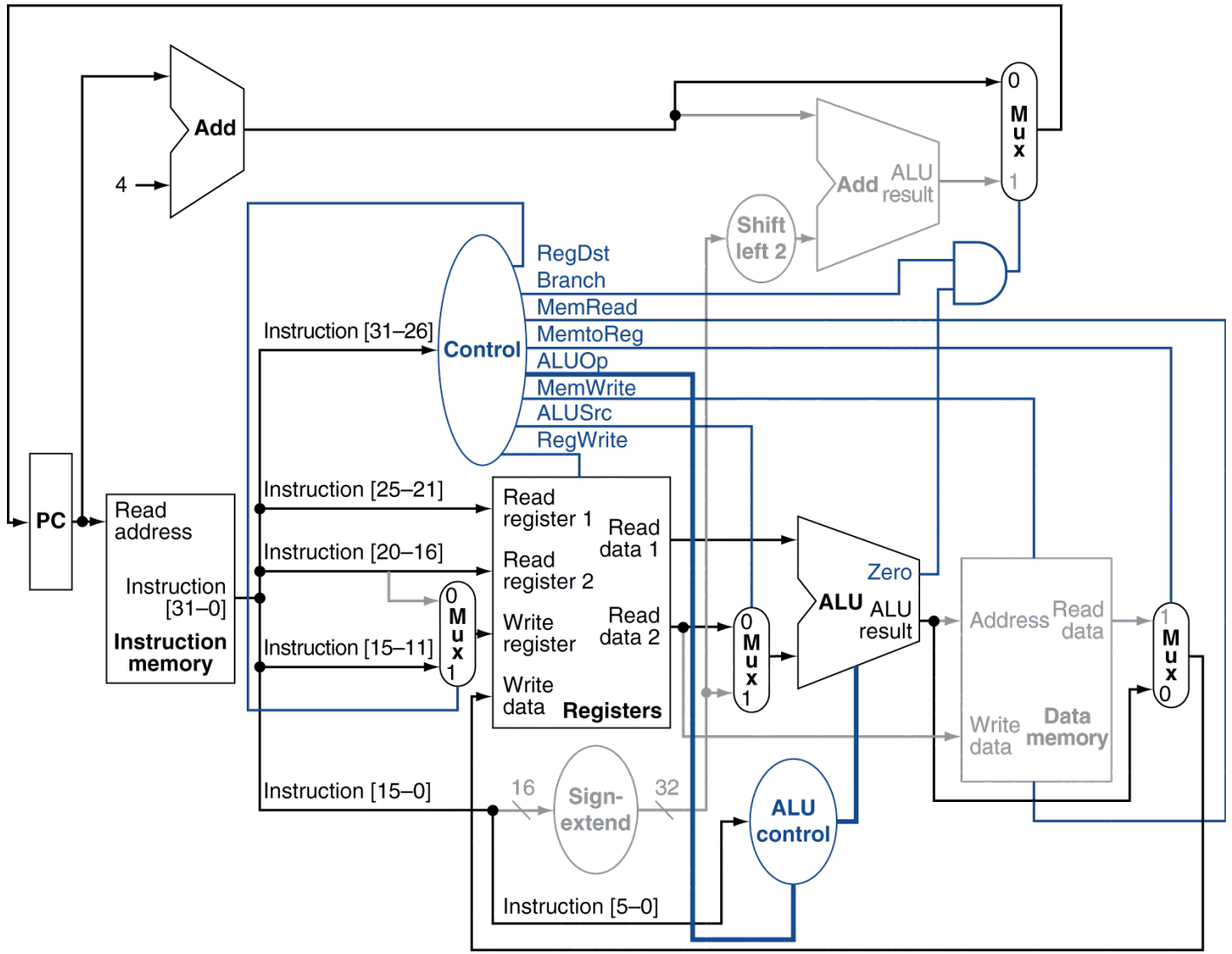
ALU control	Function
0000	AND
0001	OR
0010	add
0110	subtract
0111	set-on-less-than
1100	NOR

ALU Control

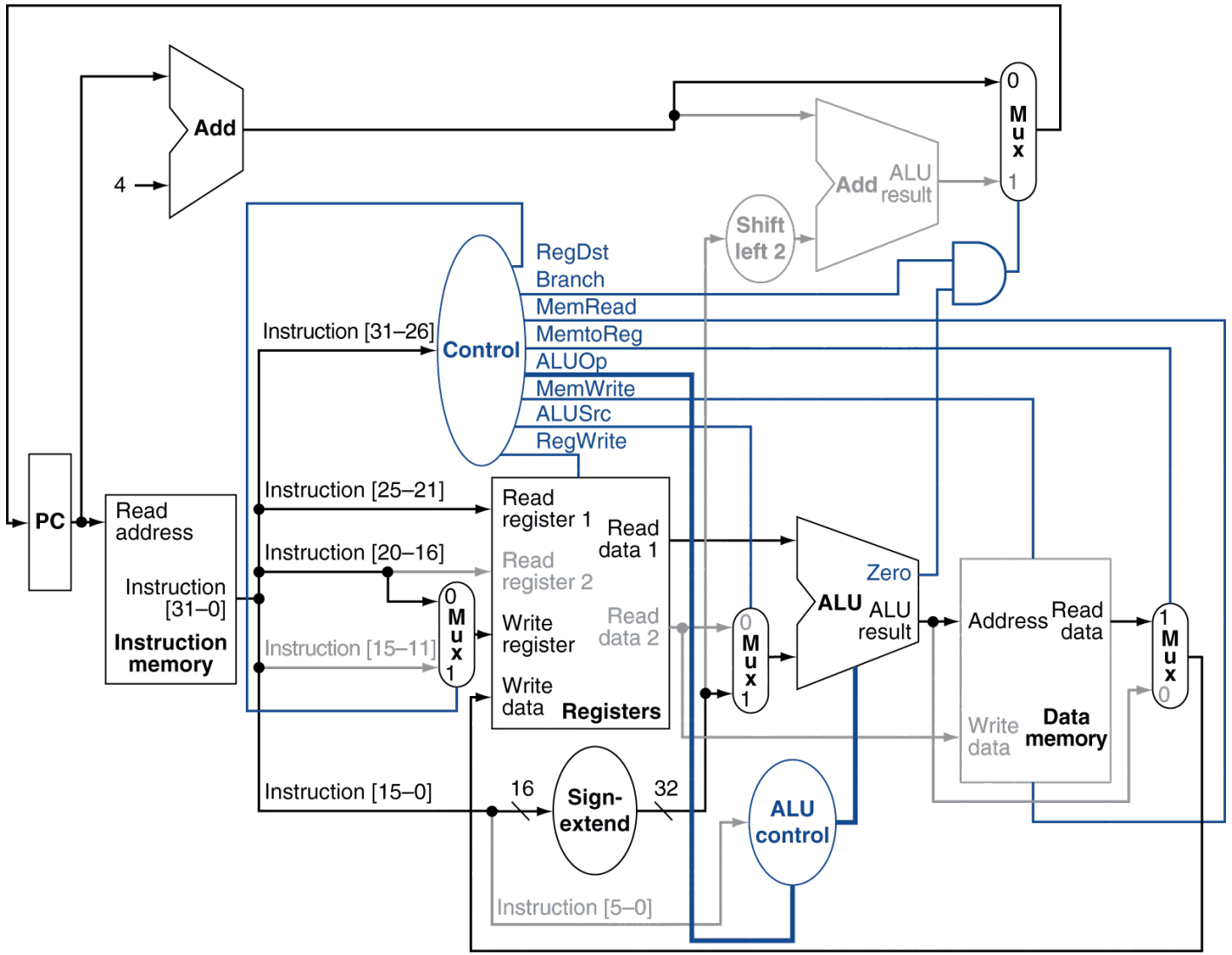
- Assume 2-bit ALUOp derived from opcode
  - Combinational logic derives ALU control

opcode	ALUOp	Operation	funct	ALU function	ALU control
lw	00	load word	XXXXXX	add	0010
sw	00	store word	XXXXXX	add	0010
beq	01	branch equal	XXXXXX	subtract	0110
R-type	10	add	100000	add	0010
		subtract	100010	subtract	0110
		AND	100100	AND	0000
		OR	100101	OR	0001
		set-on-less-than	101010	set-on-less-than	0111

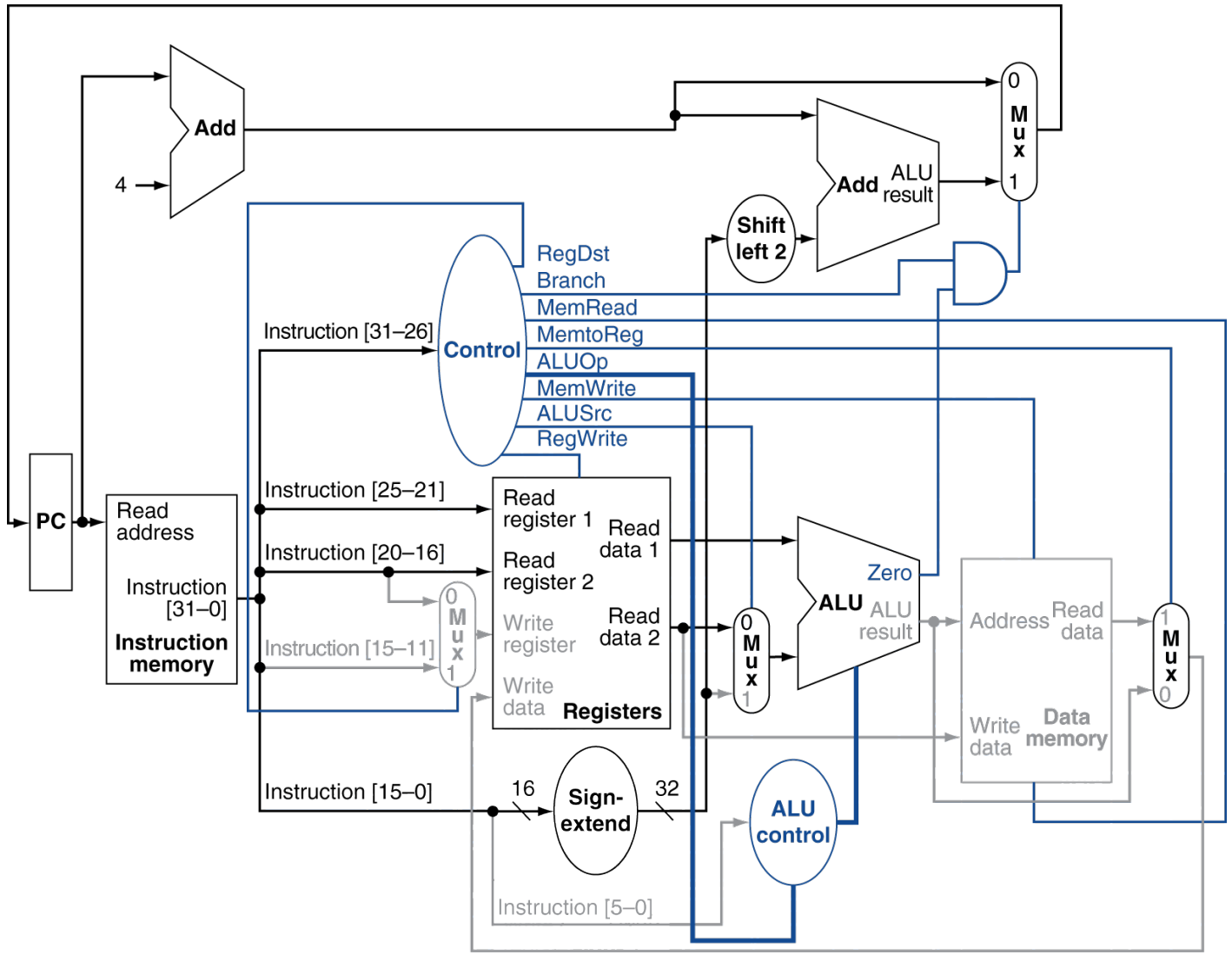
R-Type Instruction



Load Instruction



Branch-on-equal Instruction



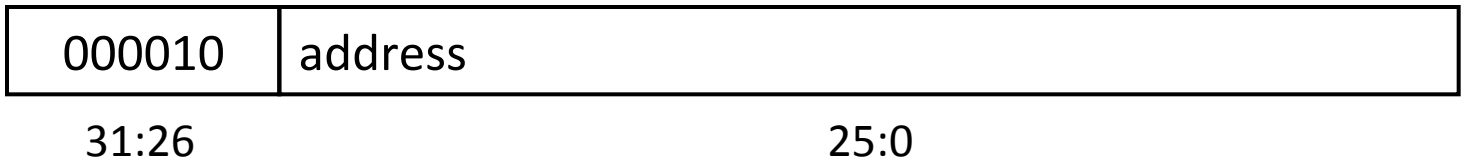
Implementing I-Types

- addi - similar to R-types except the immediate is sign extended
- ori, andi - similar to R-types except the immediate is zero extended
- RegDst is set to rt

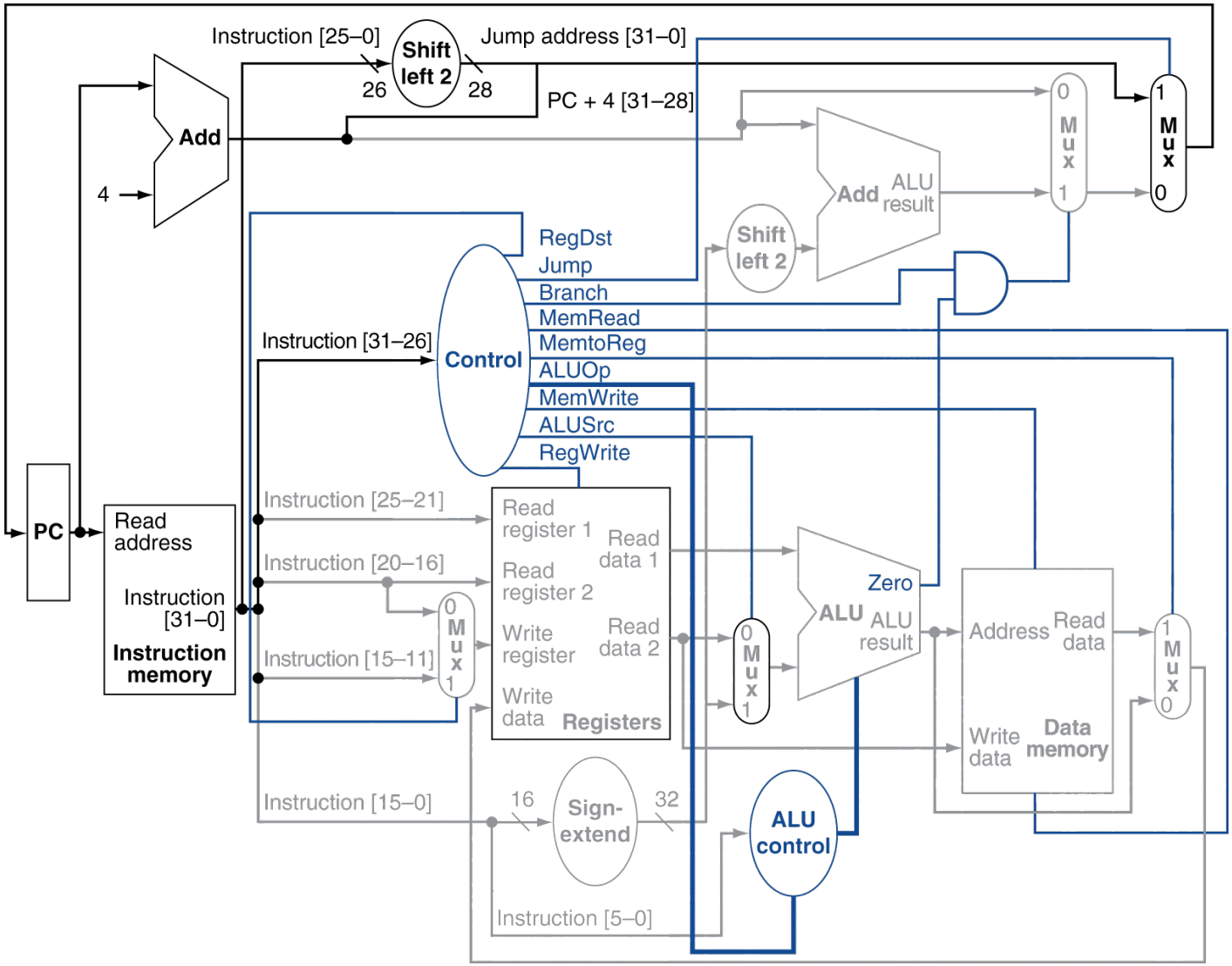
Implementing Jumps

- Jump uses word address
- Update PC with concatenation of
  - Top 4 bits of PC
  - 26-bit jump address
  - 00
- Need an extra control signal decoded from opcode

J-Type format



Datapath With Jumps Added



Review and Questions

- Datapath
- Control