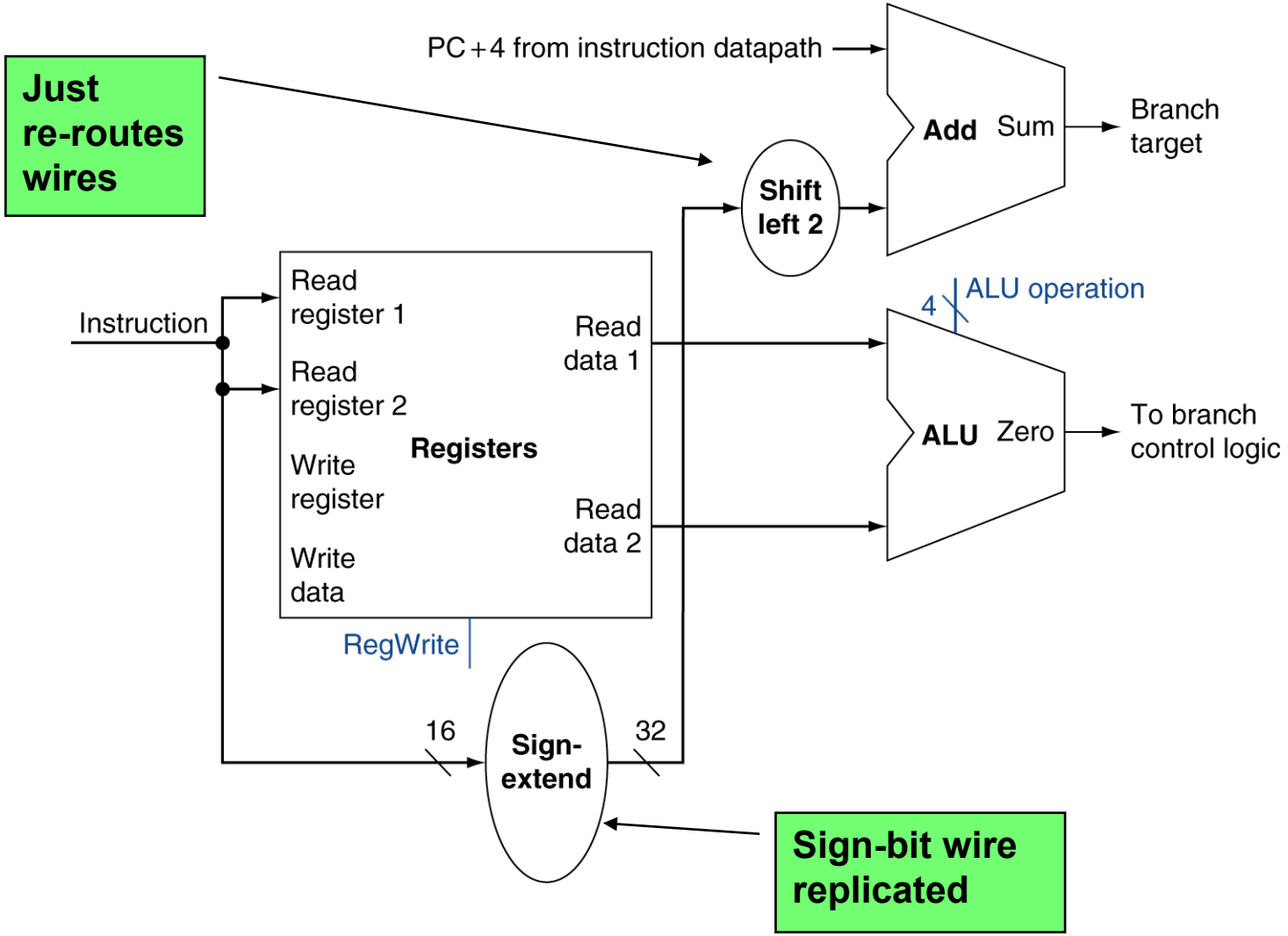
CPT_S 260 Intro to Computer Architecture
Lecture 29

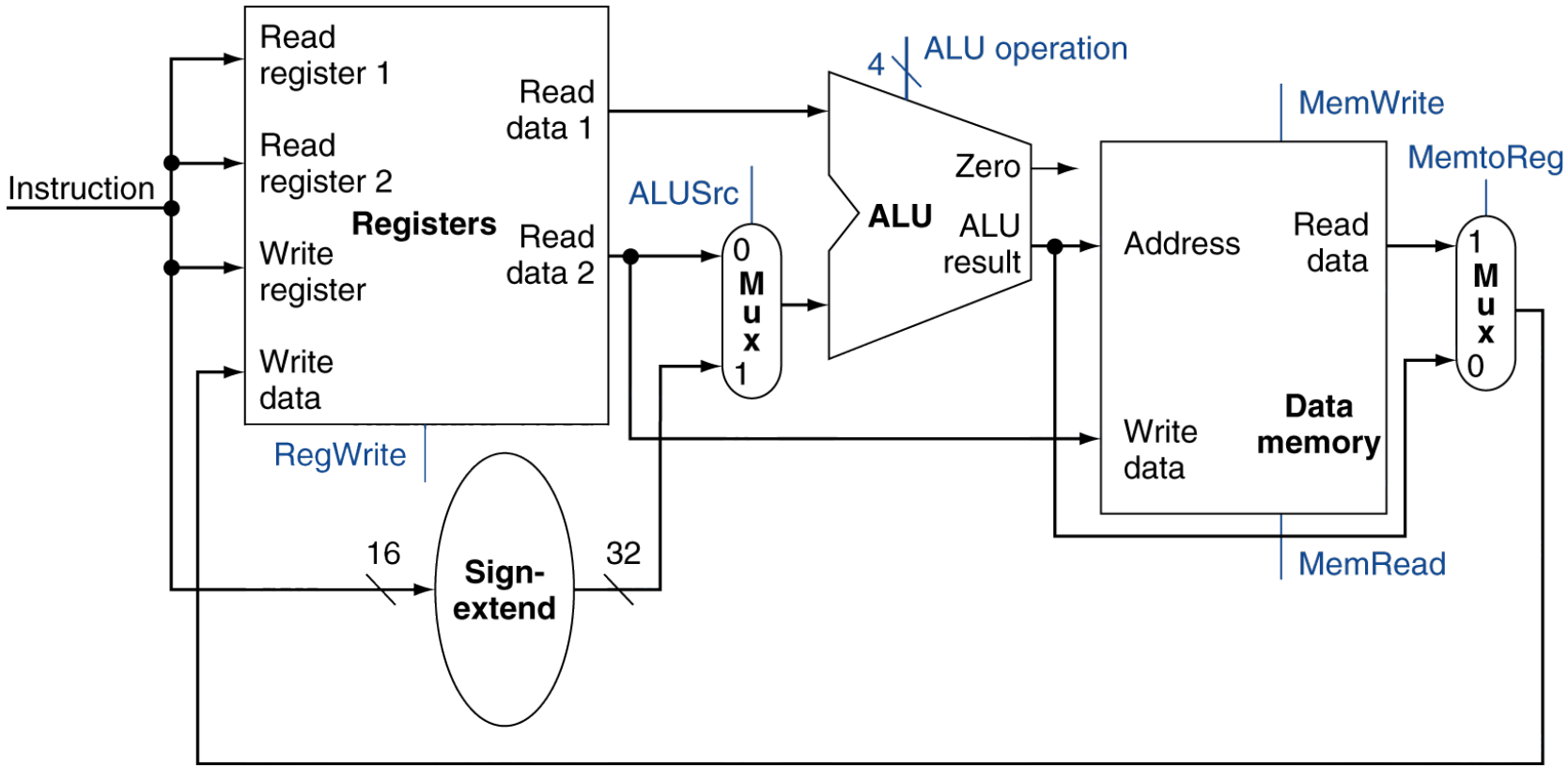
Single Cycle MIPS Control
March 28, 2022

Ganapati Bhat
School of Electrical Engineering and Computer Science
Washington State University

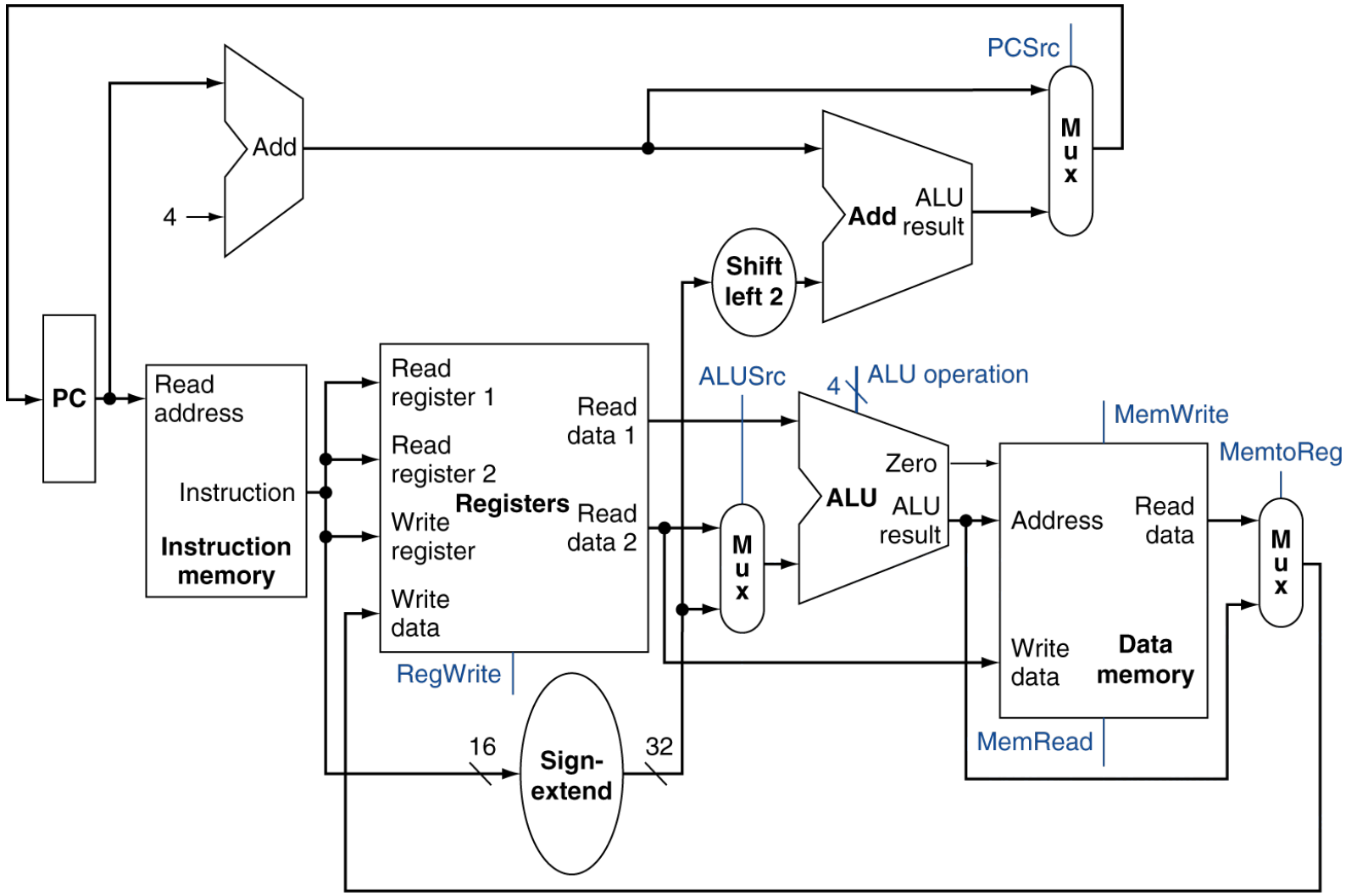
Recap: Branch Instructions



Recap: R-Type/Load/Store Datapath

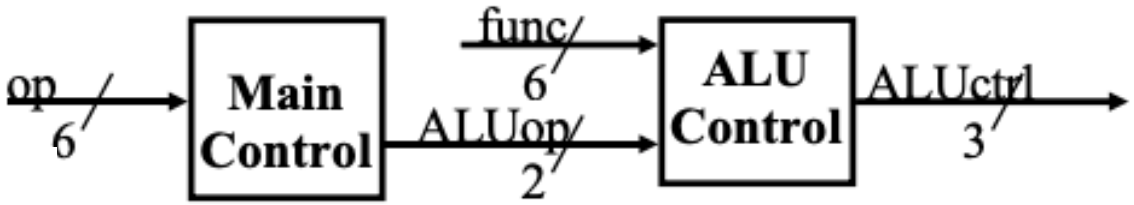


Recap: Full Datapath



Recap: ALU Control

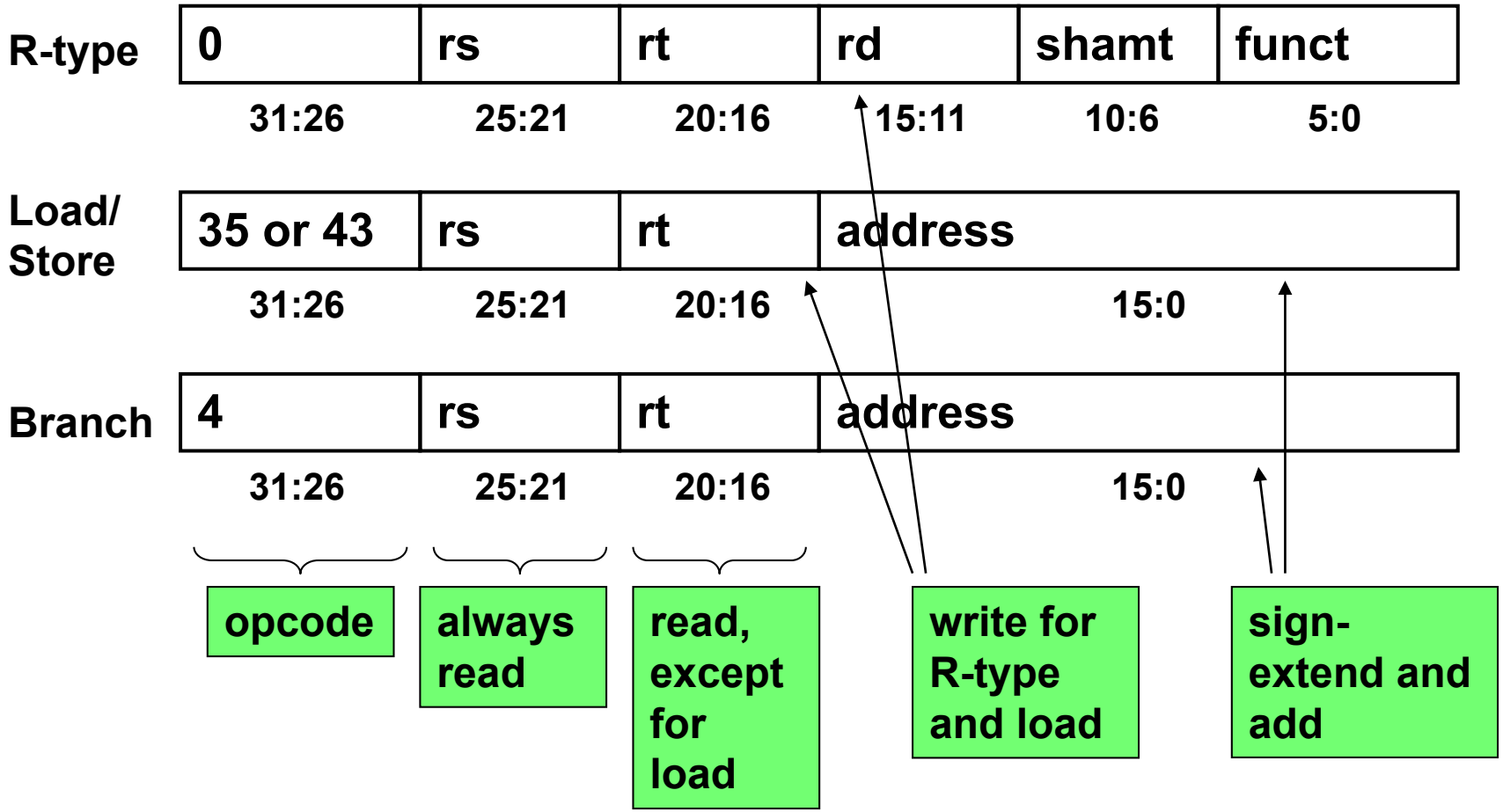
- **ALU used for**
  - Load/Store: F = add
  - Branch (e.g., beq): F = subtract
  - R-type: F depends on funct field



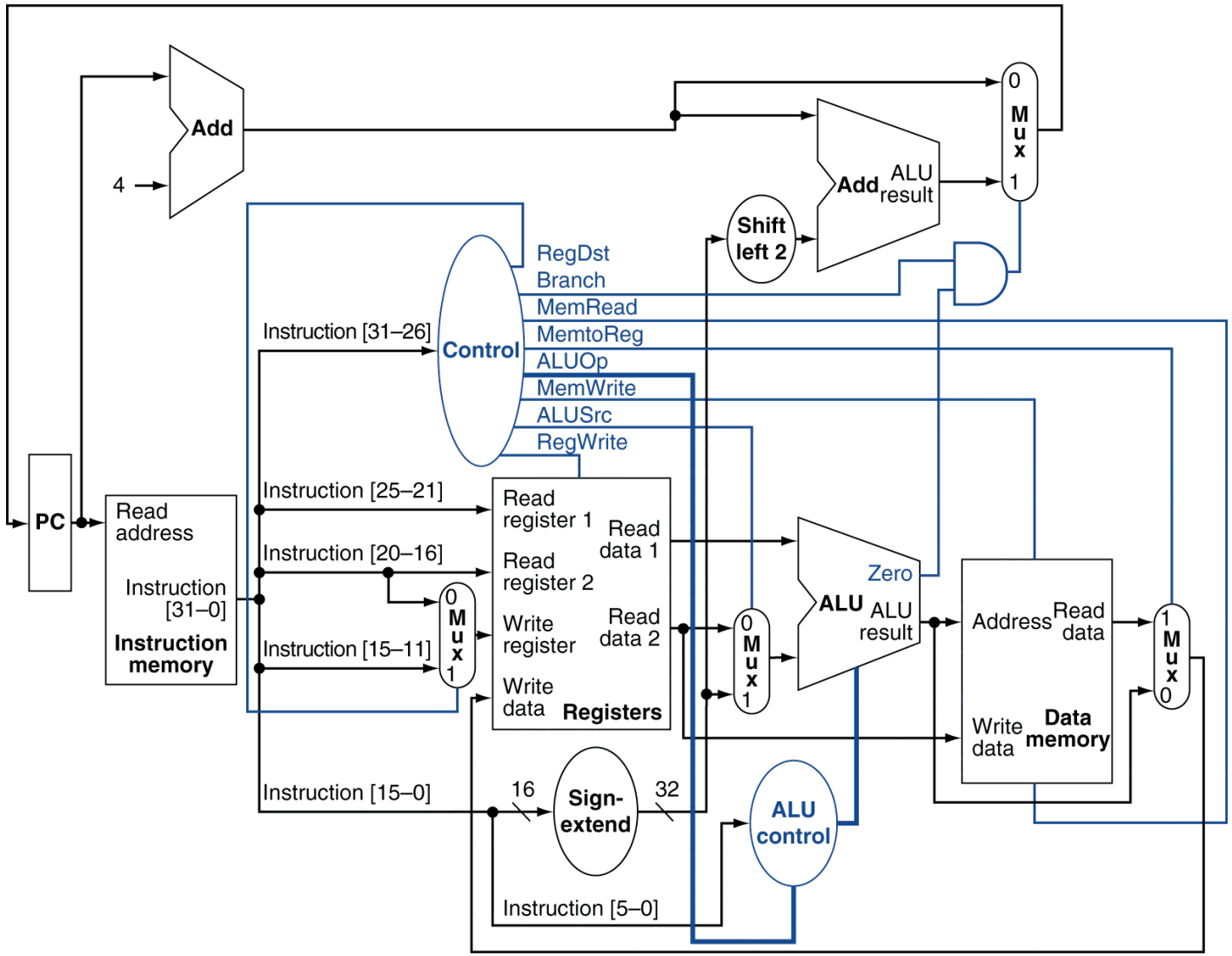
ALU control	Function
0000	AND
0001	OR
0010	add
0110	subtract
0111	set-on-less-than
1100	NOR

The Main Control Unit

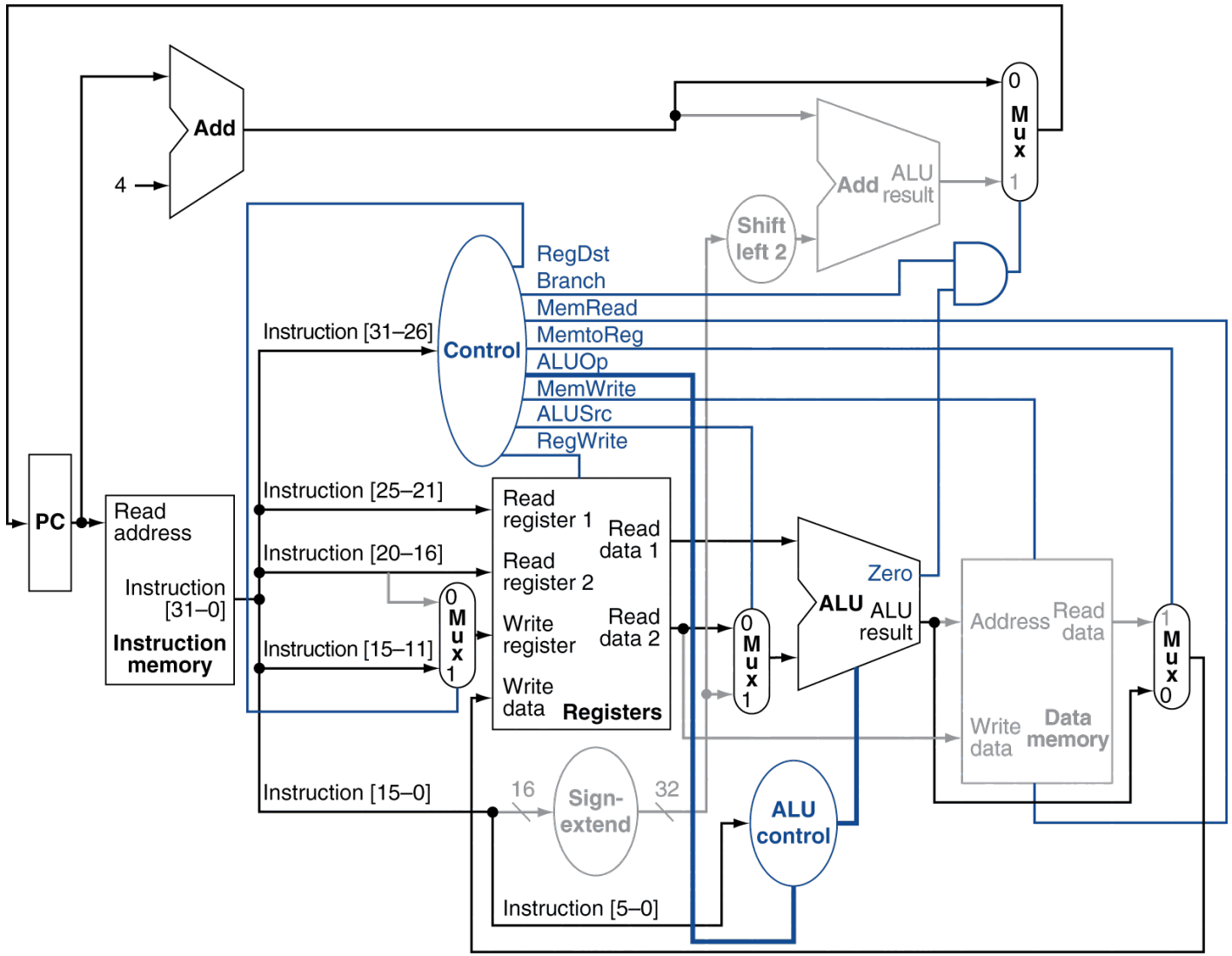
- Control signals derived from instruction



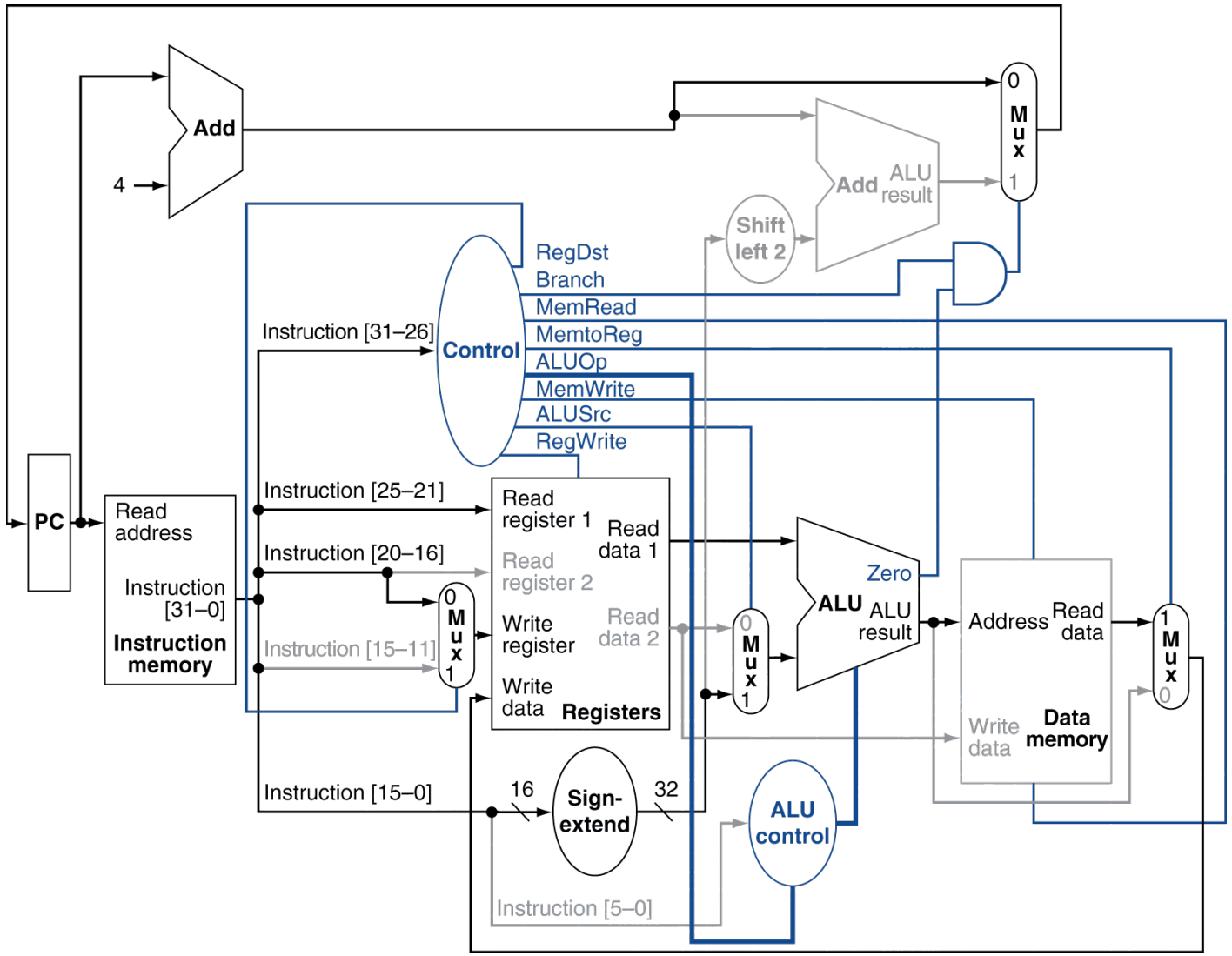
Datapath with Control



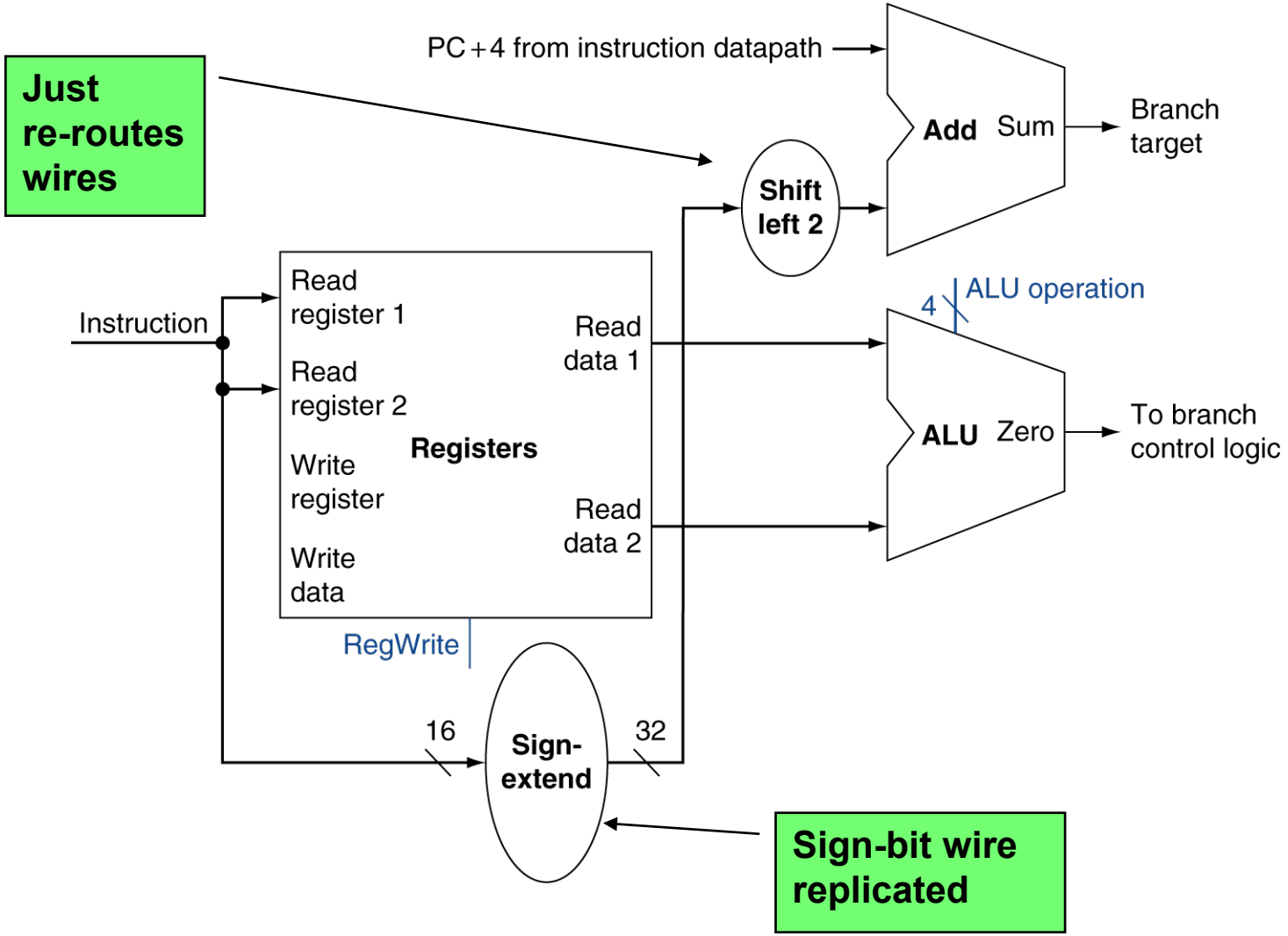
R-Type Instruction



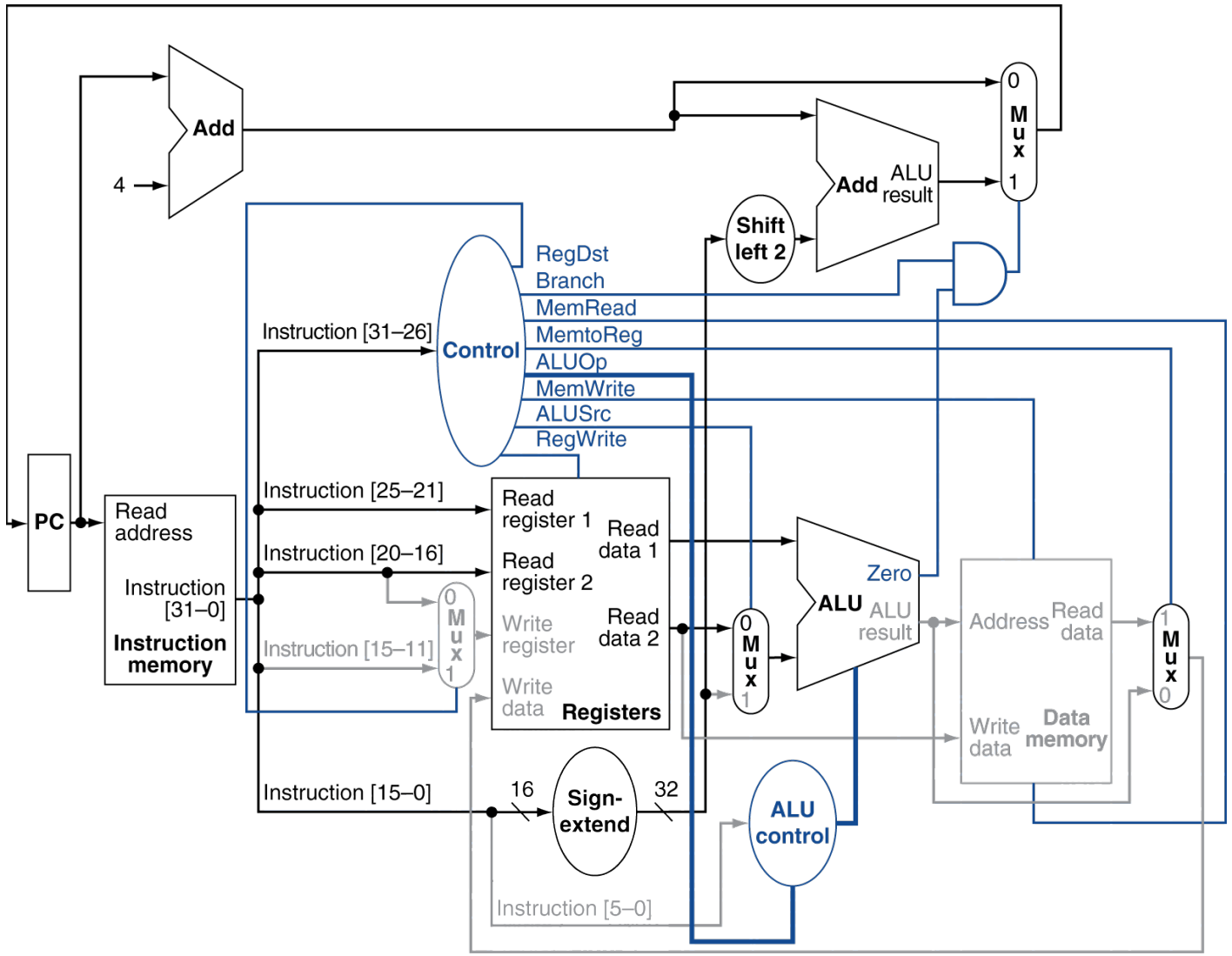
Load Instruction



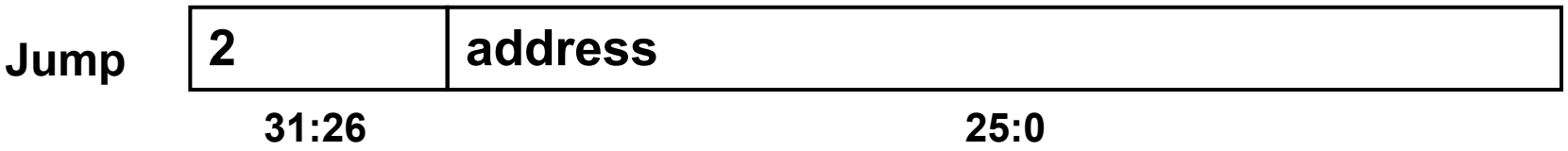
Branch Instructions



BEQ Instruction

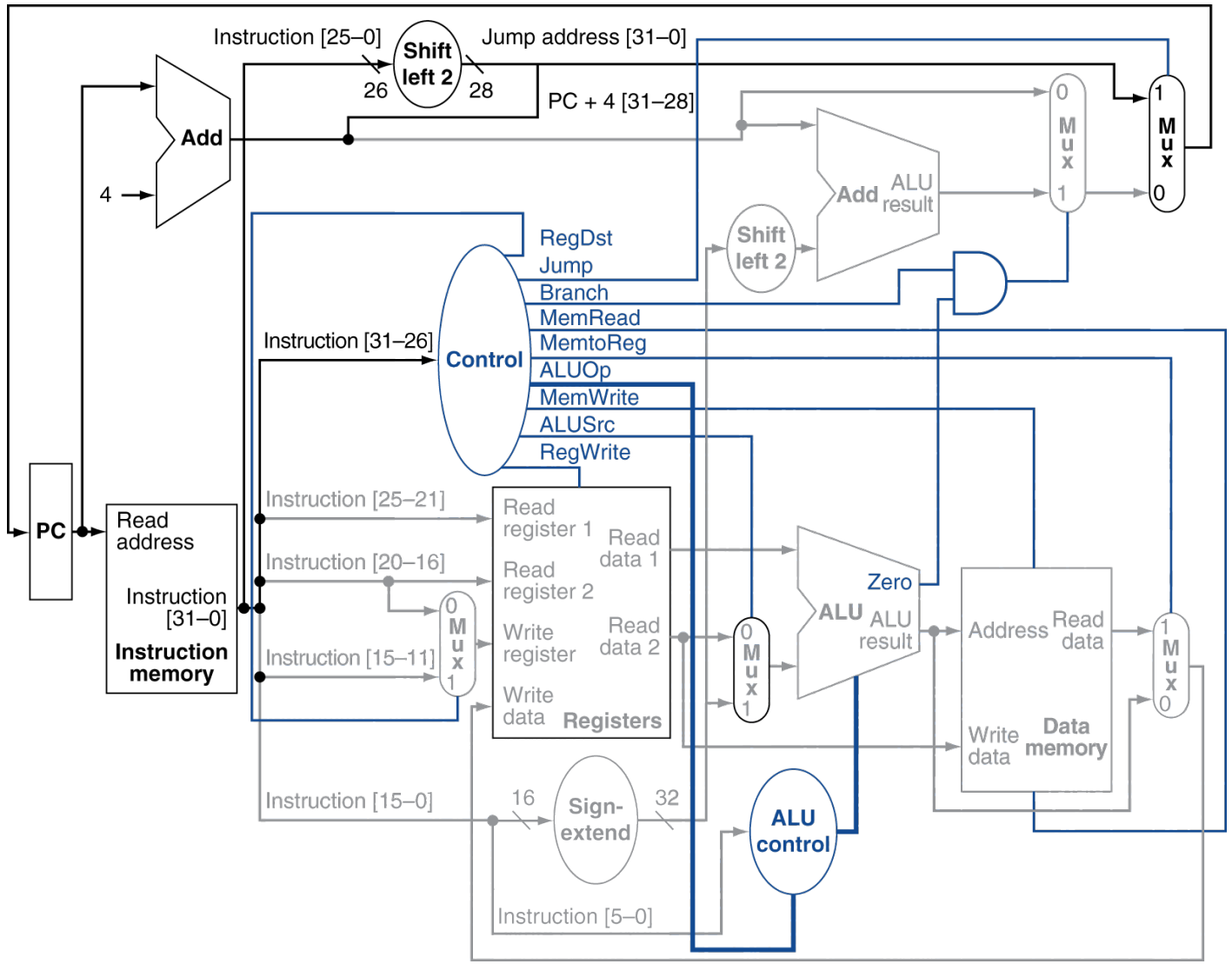


Implementing Jumps



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

Datapath with Jumps Added



Performance Issues

- **Longest delay determines clock period**
  - Critical path: load instruction
  - Instruction memory → register file → ALU → data memory → register file
- **Not feasible to vary period for different instructions**
- **Violates design principle**
  - Making the common case fast
- **We will improve performance by pipelining**

Example 1

8. [5 points] In the snippet of MIPS assembler code below, how many times is instruction memory accessed? How many times is data memory accessed? (Count only accesses to memory, not registers.)

```
lw $v1, 0($a0)
addi $v0, $v0, 1
sw $v1, 0($a1)
addi $a0, $a0, 1
```

Example #2: True/False Question

- The ‘beq’ instruction always modifies the program counter register (PC).
  - True
  - PC = PC+4 even if r1 ≠ r2
  - PC always advances (PC+4) during execution of this instruction.

Example #3: True/False Question

- The ‘add’ instruction does not modify the PC.
  - False
  - PC = PC+4
  - PC always advances (PC+4) during execution of this instruction.

Example #4: True/False Question

- **The ‘jal’ instruction always modifies the PC.**
  - **True**
  - **The ‘jal’ instruction is an ‘unconditional branch’. Thus, the PC should be modified to include target branch address.**

Example #5: True/False Question

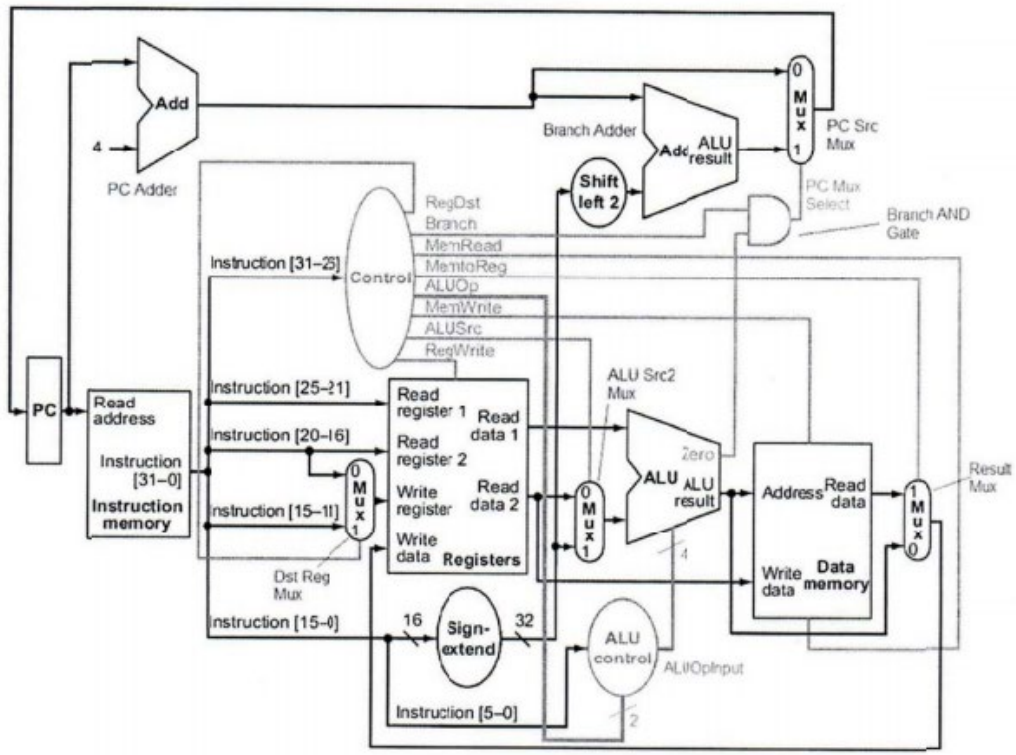
- The instruction ‘beq $1,$2,1’ will advance the PC by one word if [$1]=[$2]. (note: [] indicates content of the register)
  - False
  - PC = PC+4 + 4 x 1
  - The PC will change to PC+4 + 4 x 1 which is not essentially one word (4 bytes) change.

Example #6

This example focuses on MIPS single cycle architecture discussed before!

This example continues on next slide (parts c,d,e,f)!

1. Consider the following simple single cycle MIPS architecture, which is the same architecture discussed in the class.



a. What is the value (i.e., '0', '1', 'x') of each one of the following control signals during execution of 'ADD' instruction?

RegDst = 1
Branch = 0
MemtoReg = 0
AluSrc = 0

Jump = 0
MemRead = 0
MemWrite = 0
RegWrite = 1

b. What is the value (i.e., '0', '1', 'x') of each one of the following control signals during execution of 'J' instruction?

RegDst = 0
Branch = X
MemtoReg = 0
AluSrc = 0

Jump = 1
MemRead = 0
MemWrite = 0
RegWrite = 0

Example #6

Continued from previous slide! (single cycle MIPS architecture)

c. What is the value (i.e., '0', '1', 'x') of each one of the following control signals during execution of 'BEQ' instruction?

RegDst = X
Branch = 1
MemtoReg = X
AluSrc = 0
Jump = 0
MemRead = 0
MemWrite = 0
RegWrite = 0

d. What is the value (i.e., '0', '1', 'x') of each one of the following control signals during execution of 'SW' instruction?

RegDst = X
Branch = 0
MemtoReg = X
AluSrc = 1
Jump = 0
MemRead = 0
MemWrite = 1
RegWrite = 0

e. Assume instruction "J 1012d" is located in address 1000d (i.e., address 1000 decimal) in the instruction memory. What is the value of PC in decimal after this instruction is executed.

J address (Instruction[25-0]) : (26 bits)

J 1012d → 1012d = 00...01111111010 0 (16 bits)

We shift left the above number by 2 : (16 bits)

00...0111111010000

Finally we should concatenate the four most significant bit of this value:

PC+4 = 1004d = 00...0111111010 00 (22 bits) → four most significant bits are '0000'

PC new = 000000...0111111010000 = 4048d (20 bits)

f. Assume instruction "BEQ $t1, $t2, 10d" is located in address 1000d (i.e., address 1000 decimal) in the instruction memory. We also know that $t1 = 10d, and $t2 = 10d. What is the value of PC in decimal after this instruction is executed.

For BEQ, we should check the condition, if it is true (as it here), we should calculate the new value for PC using the relative PC addressing:

PC new = Sign-extend (branch address (Instruction(15-0))) << 2 + (PC old + 4) (next instruction)

branch address = 10d = 00...01010 (12 bits) → sign-extend → 00...0 1010 (28 bits)

shift left by 2 → 00...0 101000 (26 bits)

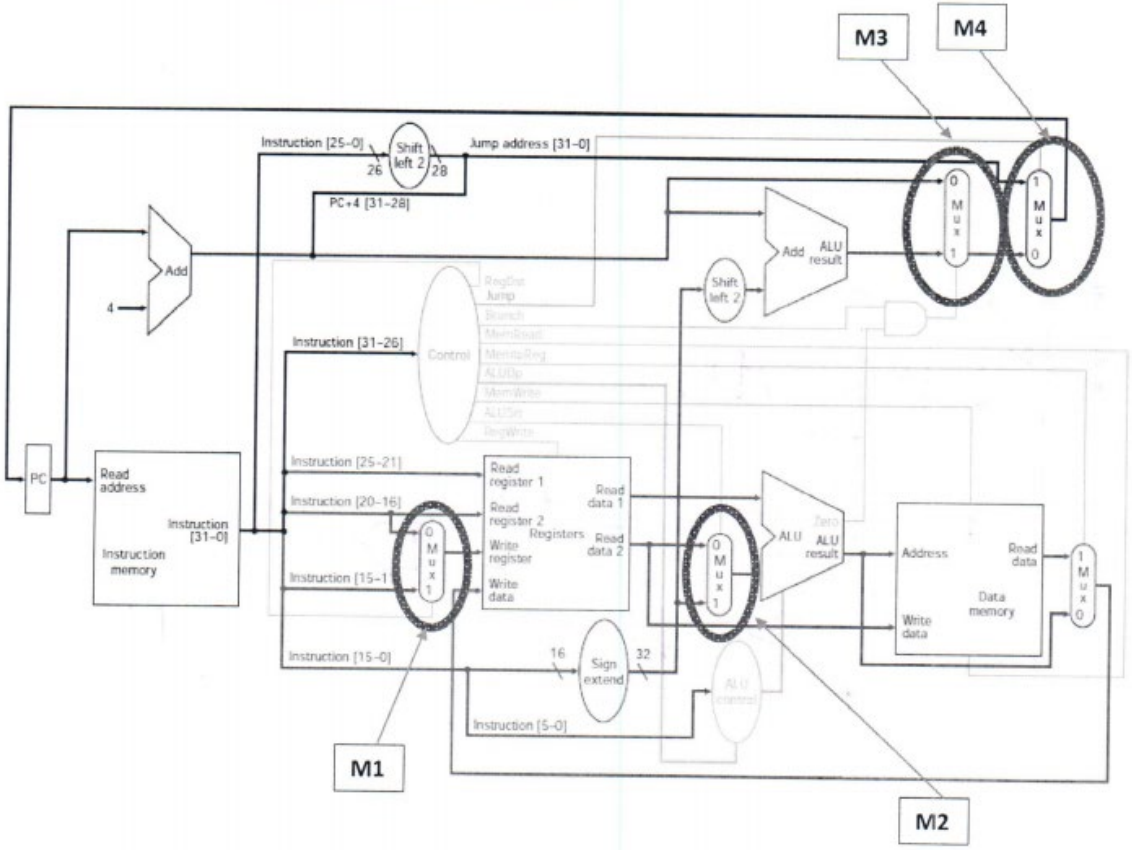
PC + 4 = 1004d = 00...0111111011 00 (22 bits)

PC new = 00...01010 + 00...0111111011 00 = 1044d

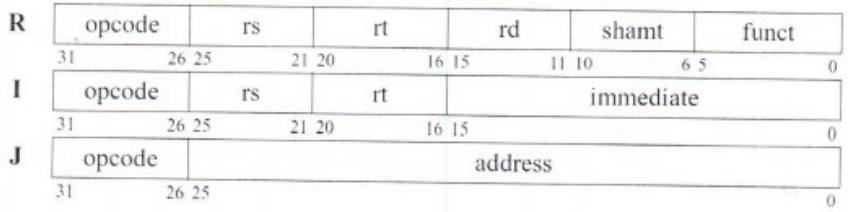
Example #7

This example continues on next slide!

3. Consider the following simple single cycle MIPS architecture and instruction format, which is the same architecture discussed in the class.



Instruction Format for R-Type, I-Type, and Jump instructions



Example #7

Continued from previous slide!

a. Assume that the following instruction is in the datapath to execute. The instruction is located in address 1000d (i.e., address 1000 decimal) in the instruction memory. We also know that $t1 = 10d, and $t2 = 24d.

BEQ $t1, $t2, 10

- What is the value of PC after this instruction is executed.
- What is the value of selection pin for each multiplexer in the datapath (0 or 1). The MUXs are labeled as M1, M2, M3, and M4.

Condition is not right ⇒ Branch is not taken ⇒

⇒ PC → PC + 4

1000d + 4d = 1004d = 1111101100 b

BEQ is an I-type Instruction

inst(25-20) = rs ⇒ M1 selection is 0

inst(20-16) = rt

ALU operation is substraction of $t1 and $t2 ⇒ M2 sel is 0

Branch is not taken ⇒ M3, M4 should take PC+4 as the next value of PC ⇒ M3, M4 sel's should be 0 (both)

b. Specify the value of each control signal listed in the following table during execution of each instruction listed in the first column.

Instruction	RegDst	Branch	MemRead	MemtoReg	MemWrite	AluSrc	RegWrite
add $9, $7, $8	1	0	0	0	0	0	1
lw $5, 10($8)	0	0	1	1	0	1	1
sw $4, 0($3)	X	0	0	X	1	1	0