CSE3421
Computer Architecture

Homework 4

Xiaodong Zhang

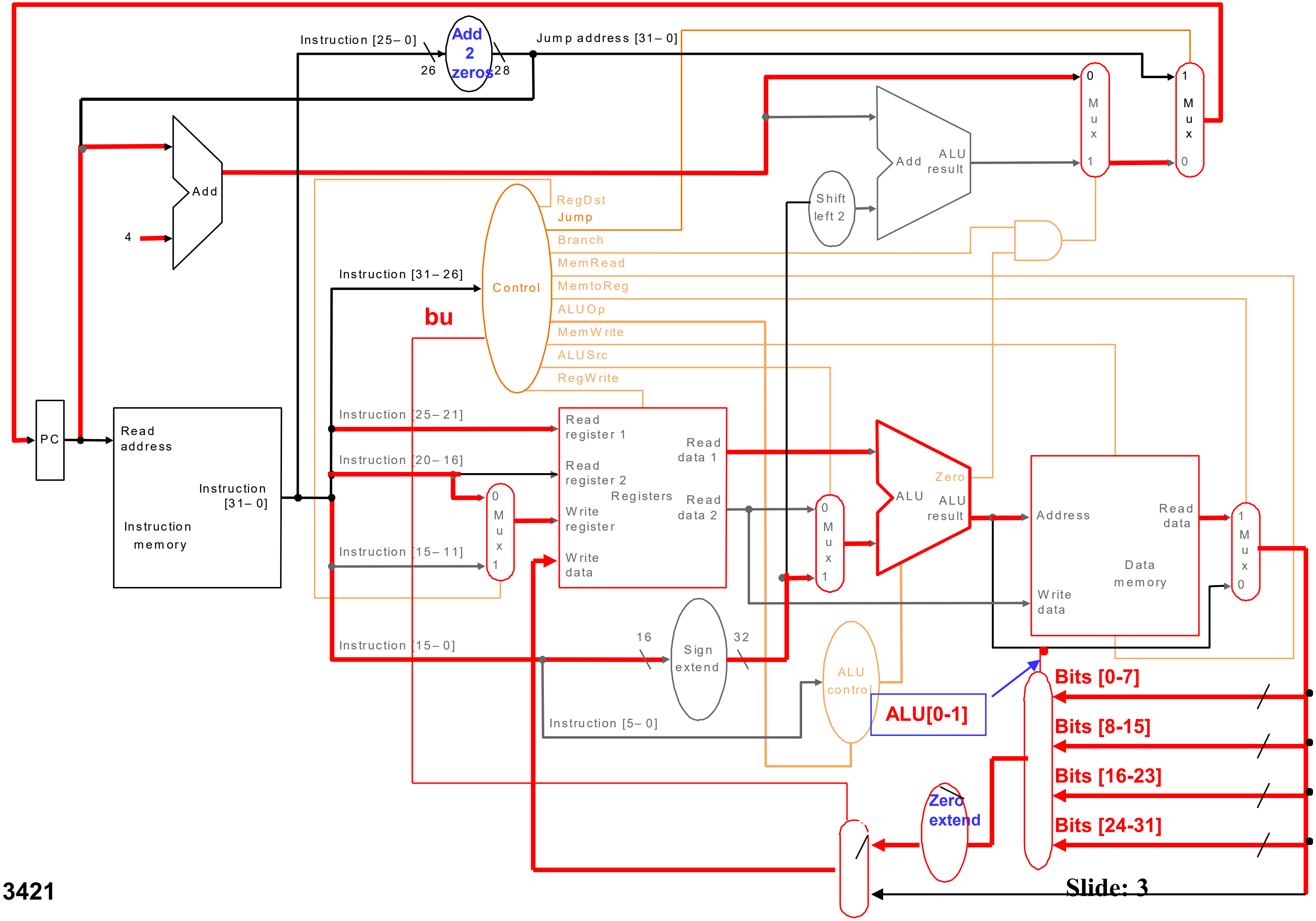
Problem 1

For a single-cycle design of a MIPS processor, how does instruction "lbu" work based on its data flow along with the control signals? Using the HomeWork4-slides figure ("Circuit for Instruction lbu") to explain all the related data flow and control signals within the single cycle. Please divide you single cycle into 5 stages: **(1)** instruction fetching, **(2)** instruction decoding, **(3)** ALU execution, **(4)** memory access, and **(5)** register writing.

Note: lbu: load byte unsigned from memory to register.

lbu $s1, 20($s2) # $s1 = Memory ([$s2 +20])

Circuit for Instruction lbu



Problem 1: Instruction Fetching

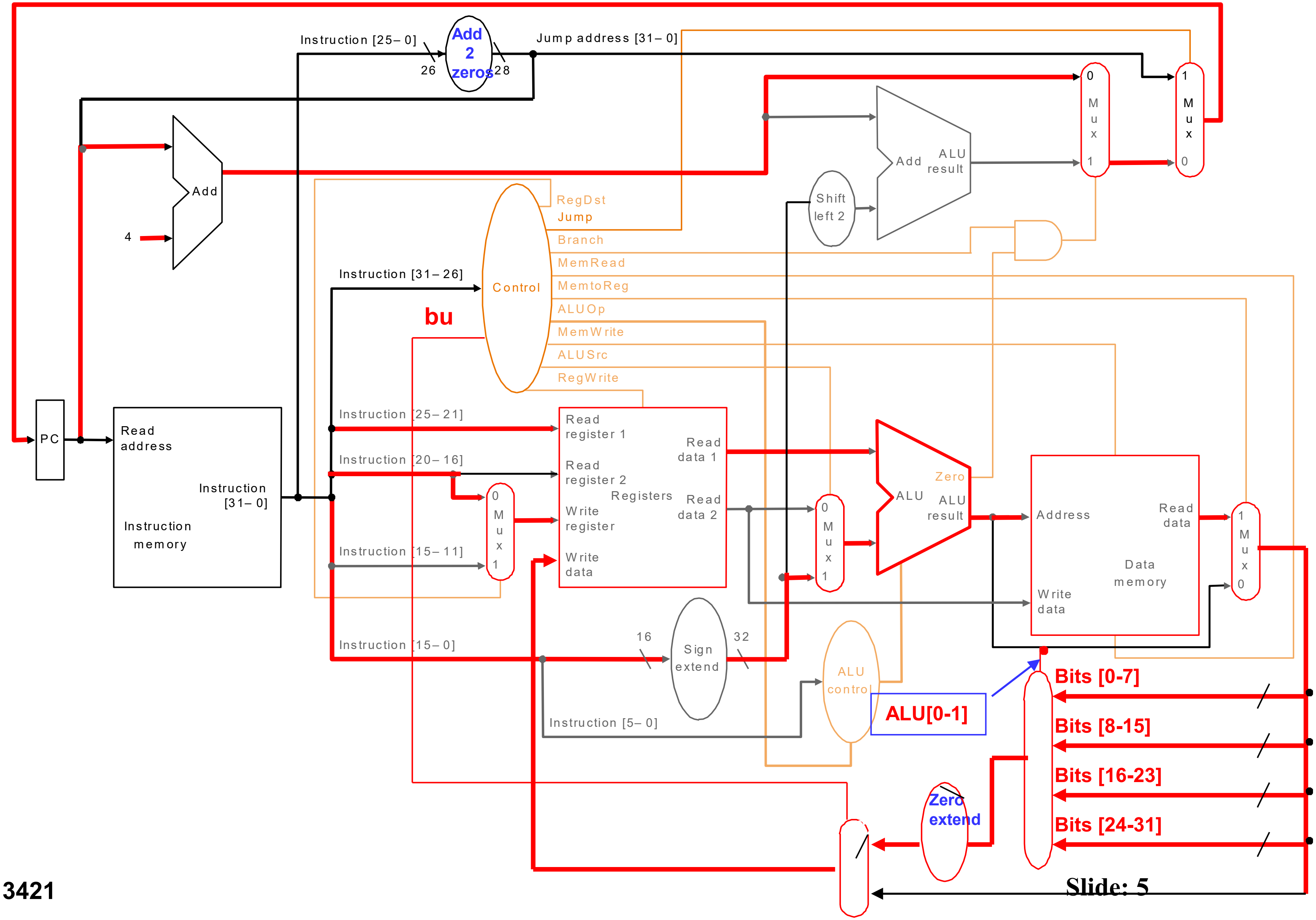
Stage 1:

PC+4 or go to branch address is controlled by the output of the AND gate. Two inputs to the AND gate:

a. Control signal branch = 0/1 and
b. Constant 0; For logical operation it is 0 (false) or 1 (true)

…

Circuit for Instruction lbu



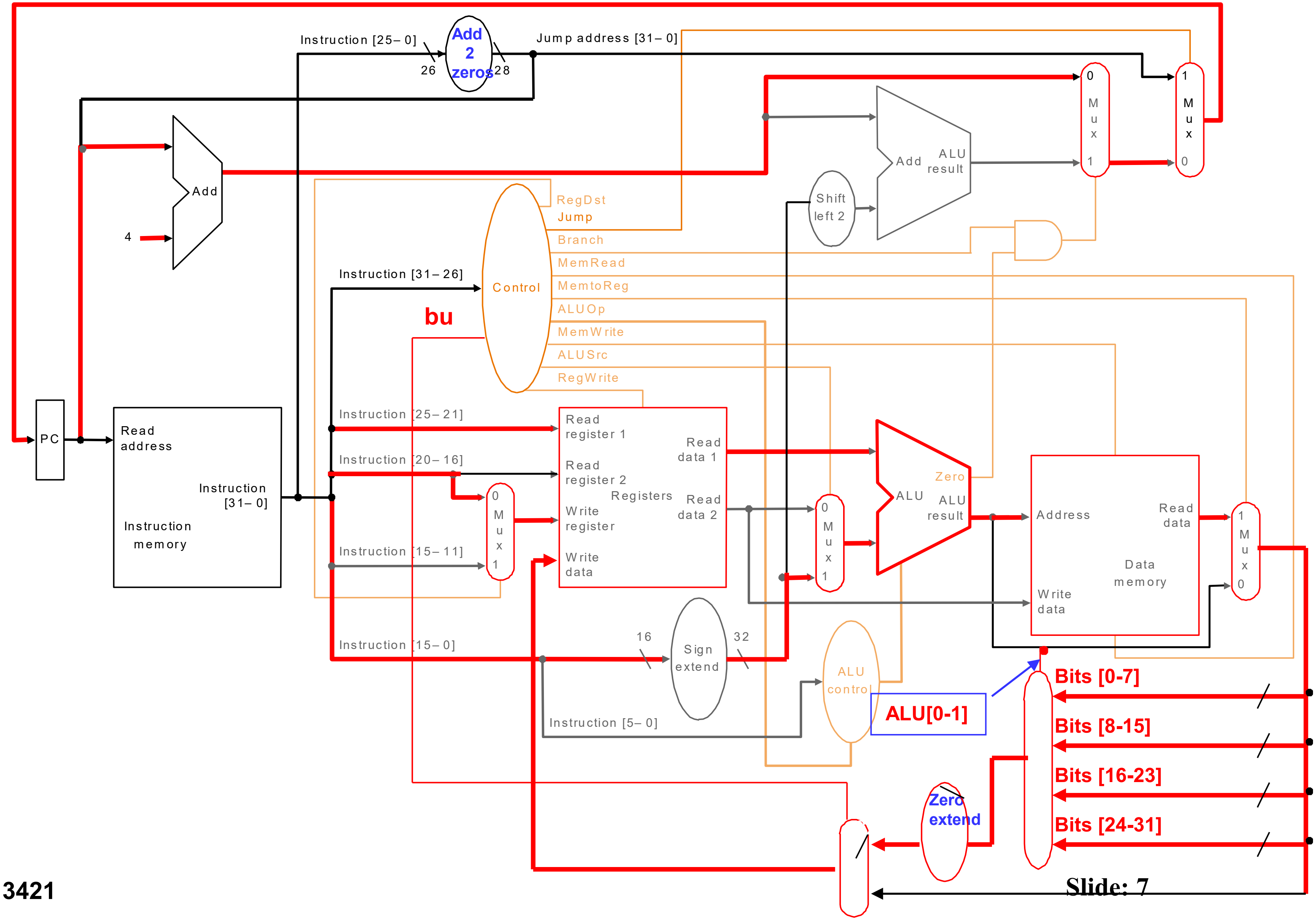
Problem 1: Instruction Decoding

Stage 2:

Preparing two inputs for ALU and one signal

Similar to the lw instruction we studied in the class for calculating the memory address.

Circuit for Instruction lbu



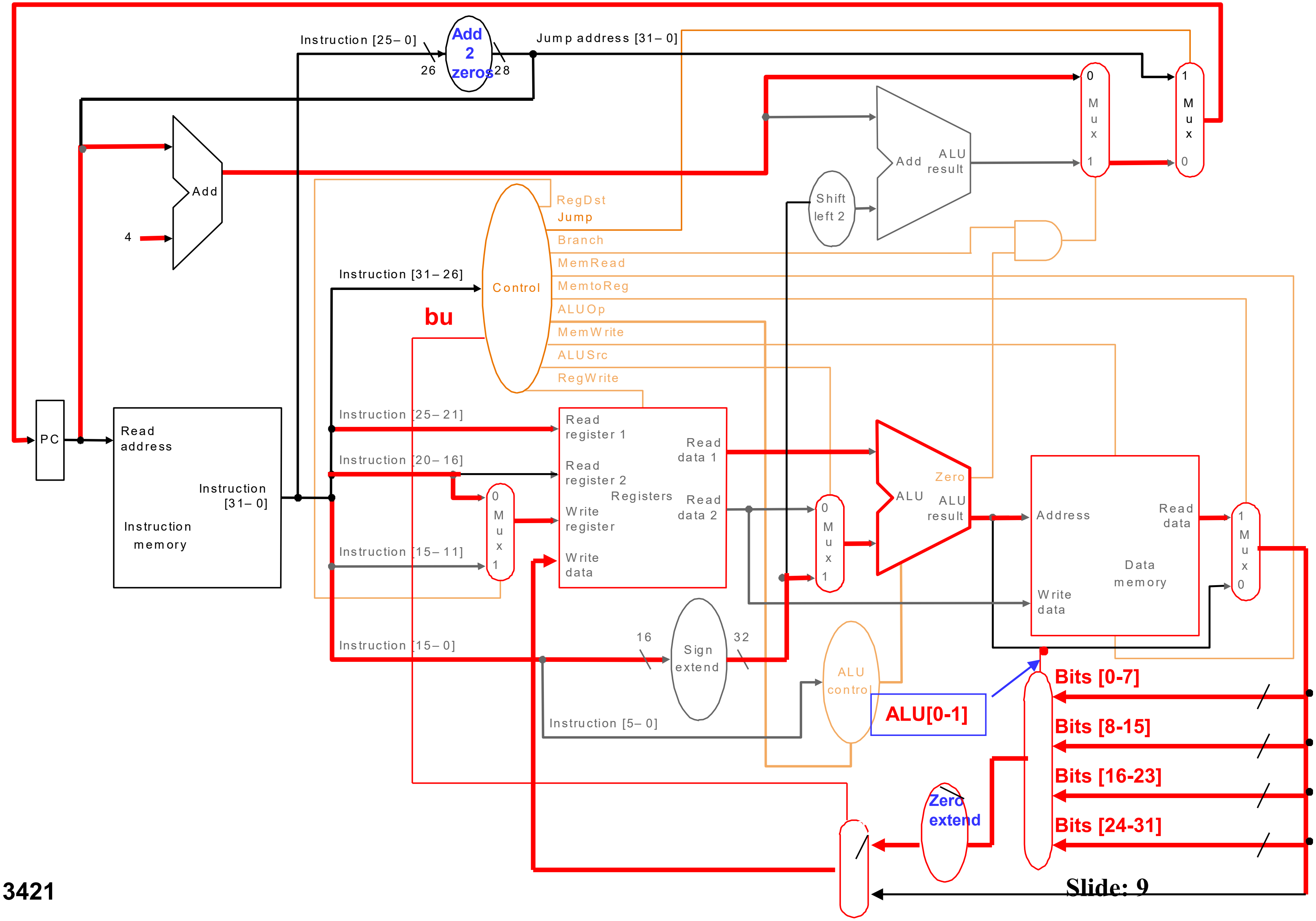
Problem 1: ALU Execution

Stage 3:

ALUop = add (determined by the instruction[5-0] for the following operation:

base register value + intermediate value = memory address (output from ALU)

Circuit for Instruction lbu



CSE 3421

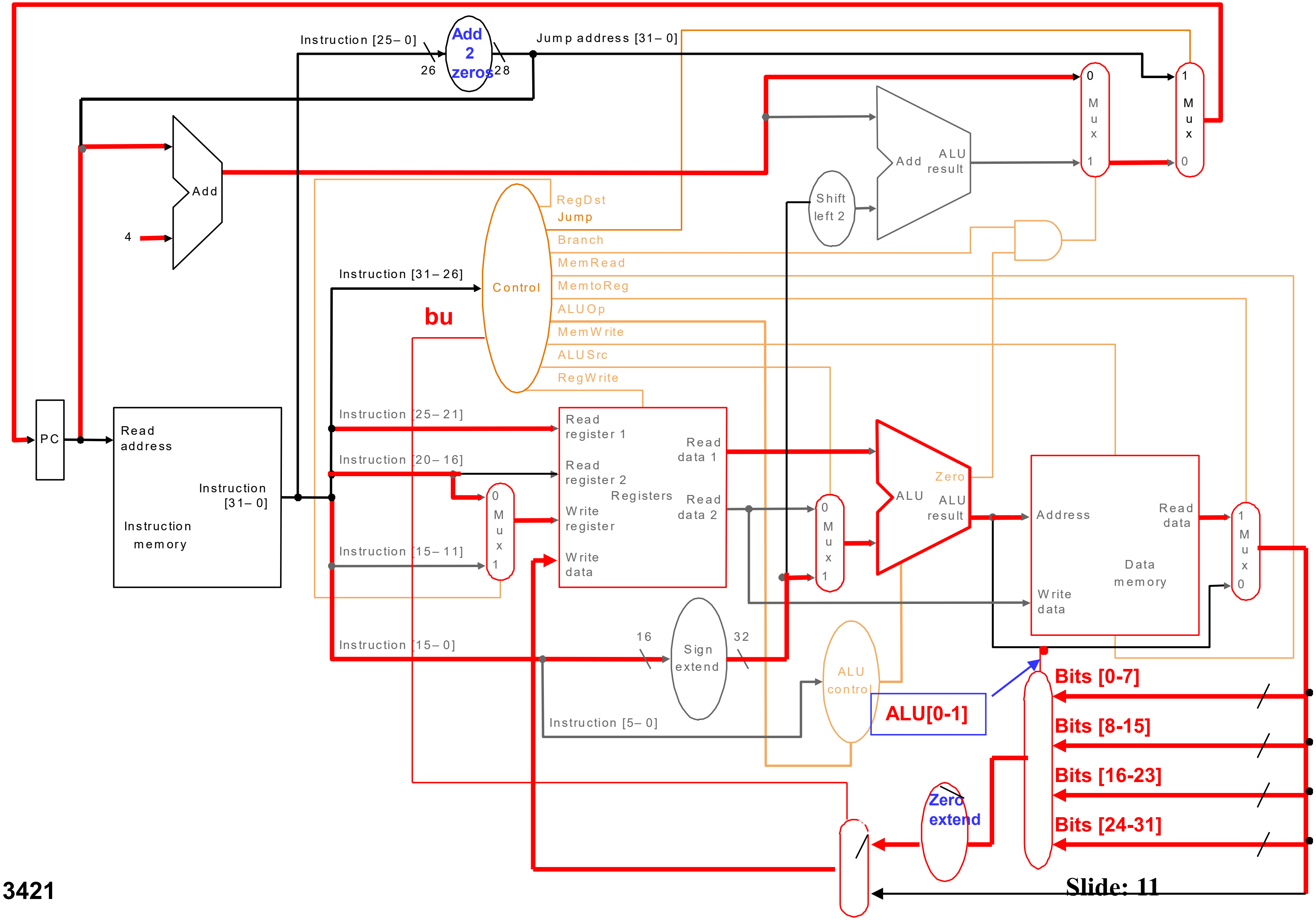
Problem 1: Memory Access

Stage 4:

Using the address from ALU,

- read the byte,
- signal MemRead = 1 and MemtoReg = 1
- to read and output the byte that becomes one of the 4 input of the multiplexor.

Circuit for Instruction lbu



CSE 3421

Problem 1: Register Writing

Stage 5:

- Since the ALU result identifies the targeted byte, ALU[0-1] selects the byte that is extended to 32 bits.
- Control signal bu=1 (byte unsigned) selects it to the register file port “Write data”.
- The byte write to the register is done by
  a. destination register (instruction [20-16]) is selected by RegDst=0
  b. RegWrite=1 is set in the register file

Problem 2

Considering the following sequence of MIPS code, your task is to use pipelining building blocks to execute this set of instructions.

```
add $R2, $R0, $R0

lw $R1, 0($R2)

addi $R3, $R1, 20

add $R1, $R2, $R0

or $R1, $R2, $R0
```

Problem 2

Questions 1

How many stall cycles would occur in a MIPS pipelining without Pipeline Forwarding? Using the pipelining building blocks to explain your result.

Make sure all the dependencies are detected.

Writing or reading register content only needs half cycle.

Problem 2

Questions 2

How many stall cycles would occur in a MIPS pipelining with Pipeline Forwarding? We assume that Pipeline Forwarding exists between DM-to-ALU, ALU-to-ALU, and DM-to-DM. Using the pipelining building blocks to explain your result.

Rely on the forwarding operations from both ALU results and memory results