ICS 233
COMPUTER ARCHITECTURE

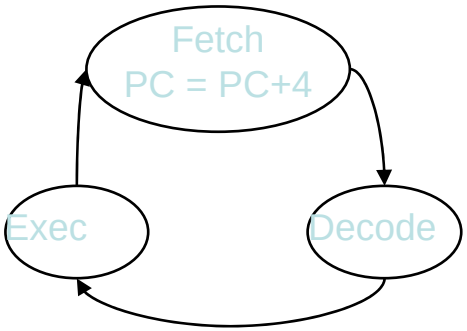
MIPS PROCESSOR
Datapath & Control

Lecture 19

MIPS Processor : Datapath & Control

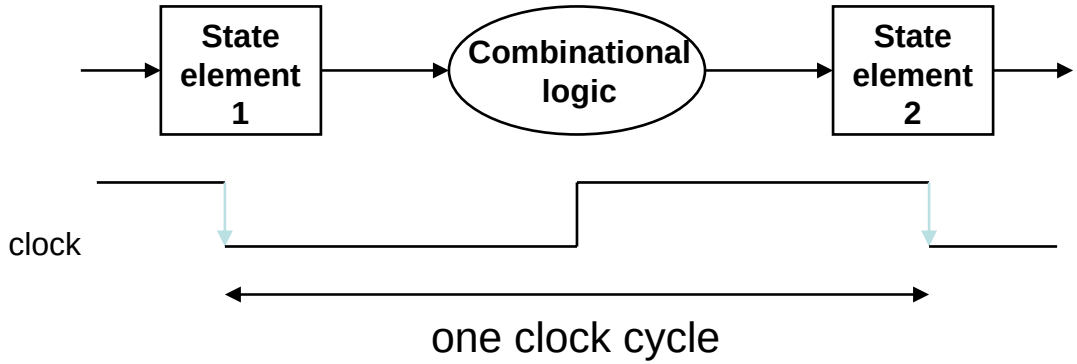
- **Our implementation of the MIPS is a subset of the core MIPS instruction set**
  - memory-reference instructions: **lw, sw**
  - arithmetic-logical instructions: **add, sub, and, or, slt**
  - control flow instructions: **beq, j**

- **Generic implementation**
  - use the program counter (PC) to supply the instruction address and fetch the instruction from memory (and update the PC)
  - decode the instruction (and read registers)
  - execute the instruction



Clocking Methodologies

- The clocking methodology defines when signals can be read and when they are written
  - An edge-triggered methodology
- Typical execution
  - read contents of state elements
  - send values through combinational logic
  - write results to one or more state elements

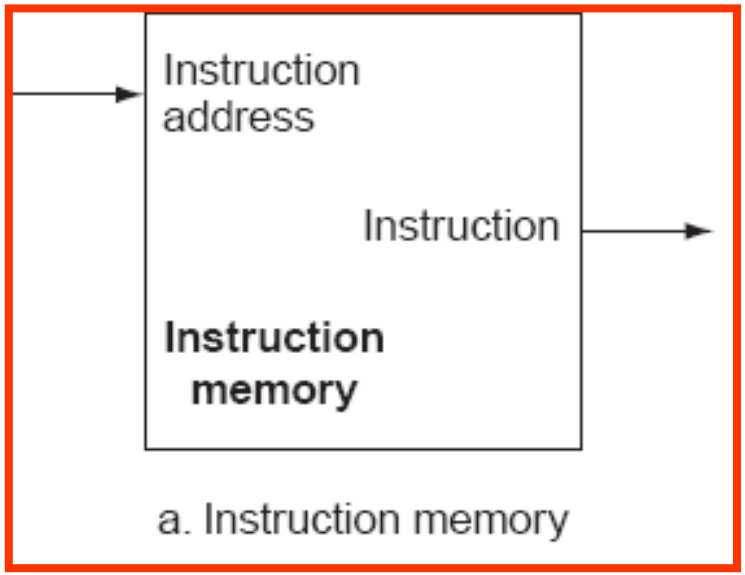


- Assumes state elements are written on every clock cycle; if not, need explicit write control signal
  - write occurs only when both the write control is asserted and the clock edge occurs

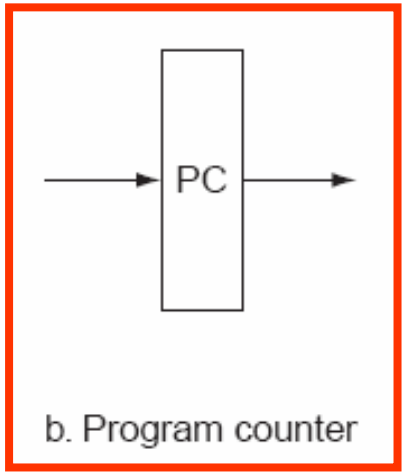
Building a Datapath

➢ Implementing Instruction Fetch Unit

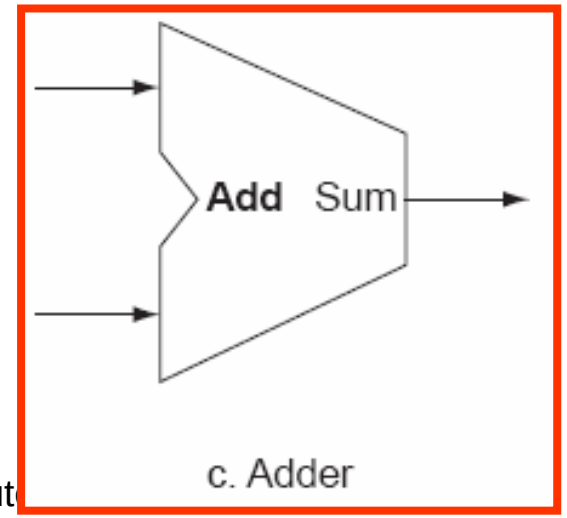
- **Instruction Memory** is used to hold and supply instructions, given an address



- **Program Counter (PC)** register is required to store the address of the next instruction to be executed

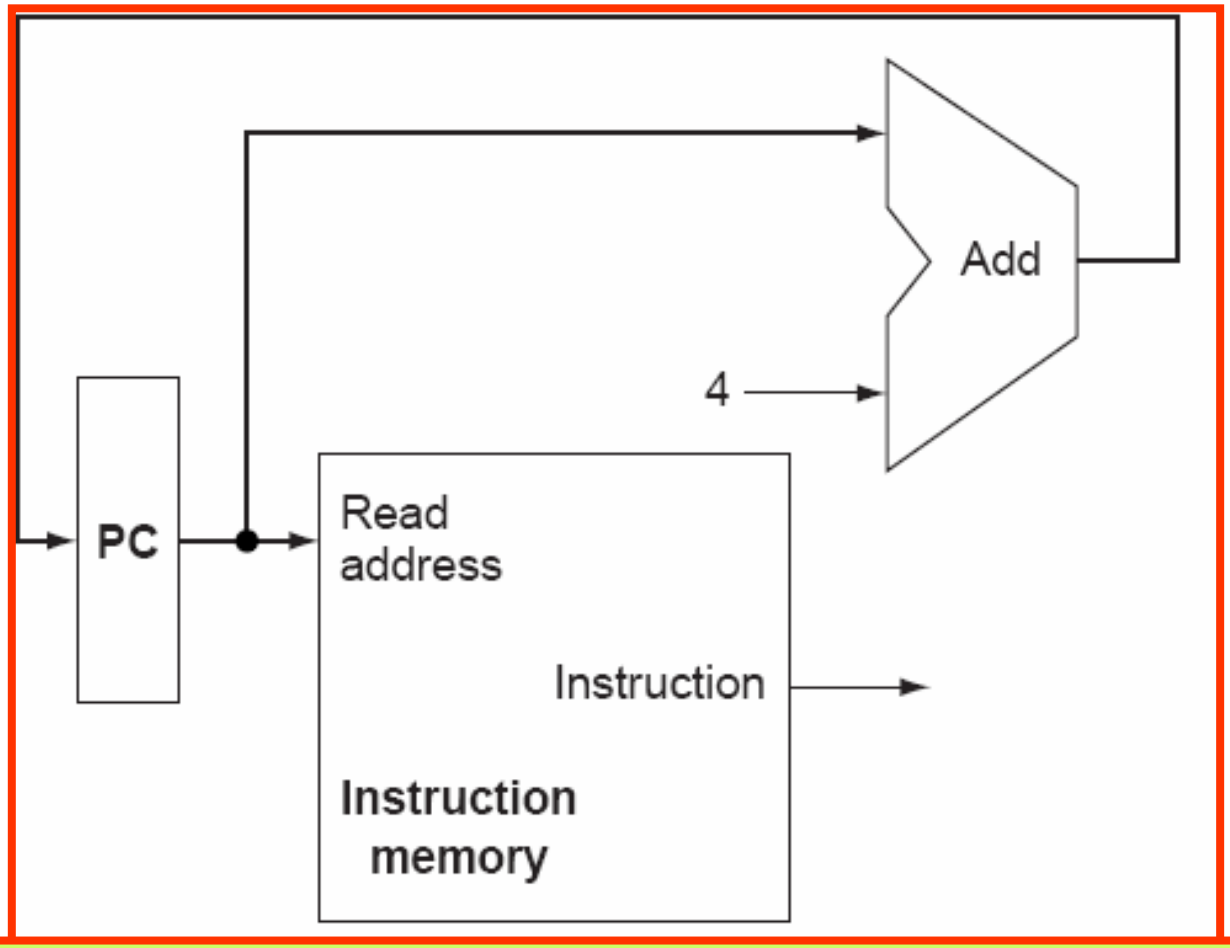


- **Adder** is required to increment the PC to the address of the next instruction



Building a Datapath

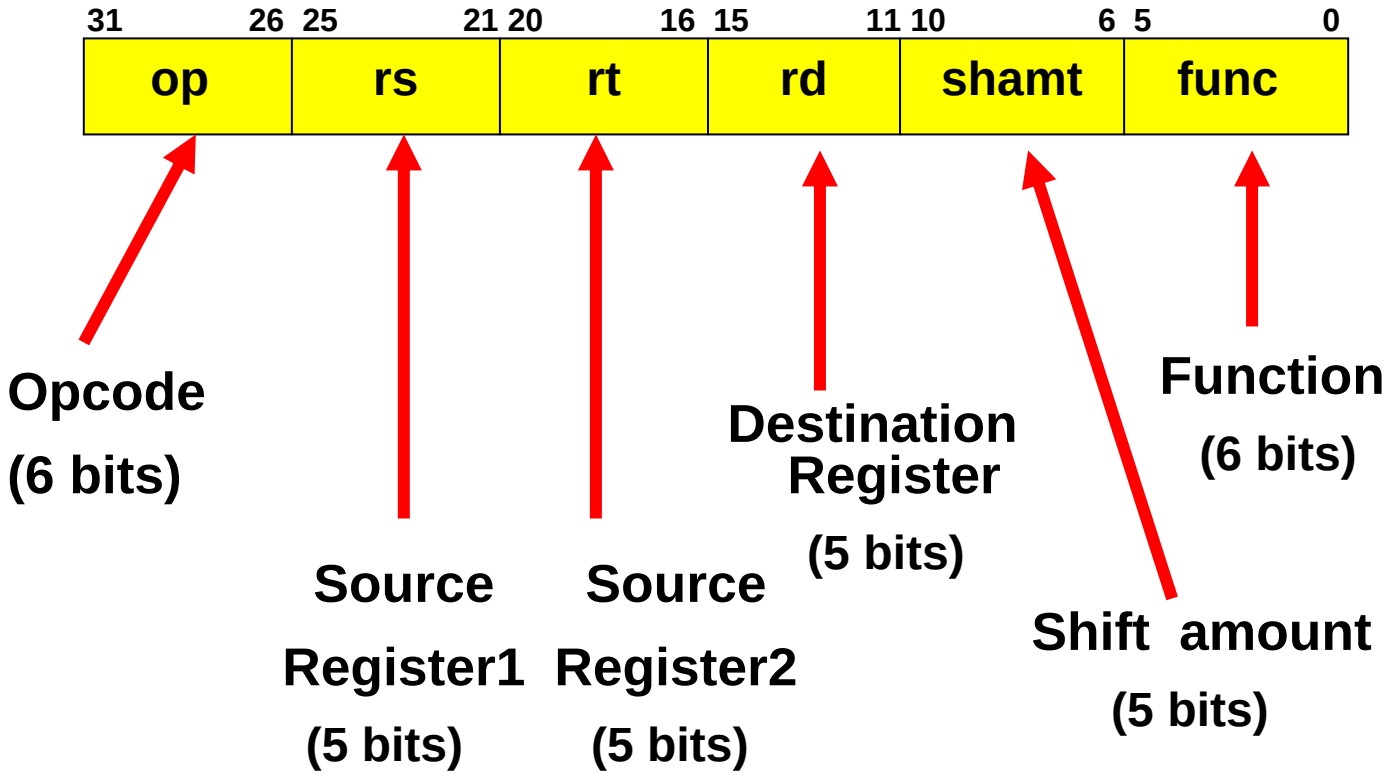
➢ Implementing Instruction Fetch Unit



A portion of the datapath used for fetching instructions and incrementing the Program Counter

R-type Instruction format

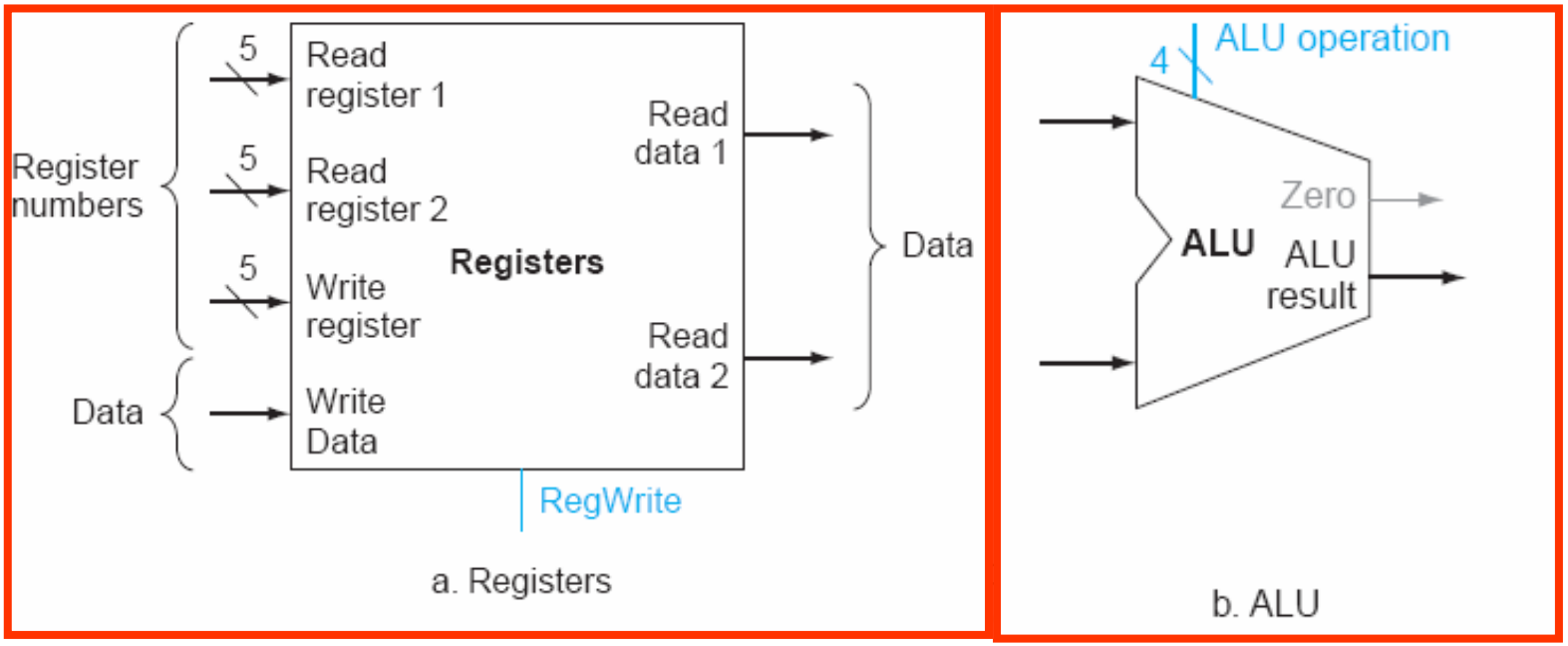
add instruction : add rd, rs, rt
sub instruction : sub rd, rs, rt
and instruction : and rd, rs, rt
or instruction : or rd, rs, rt
slt instruction : slt rd, rs, rt



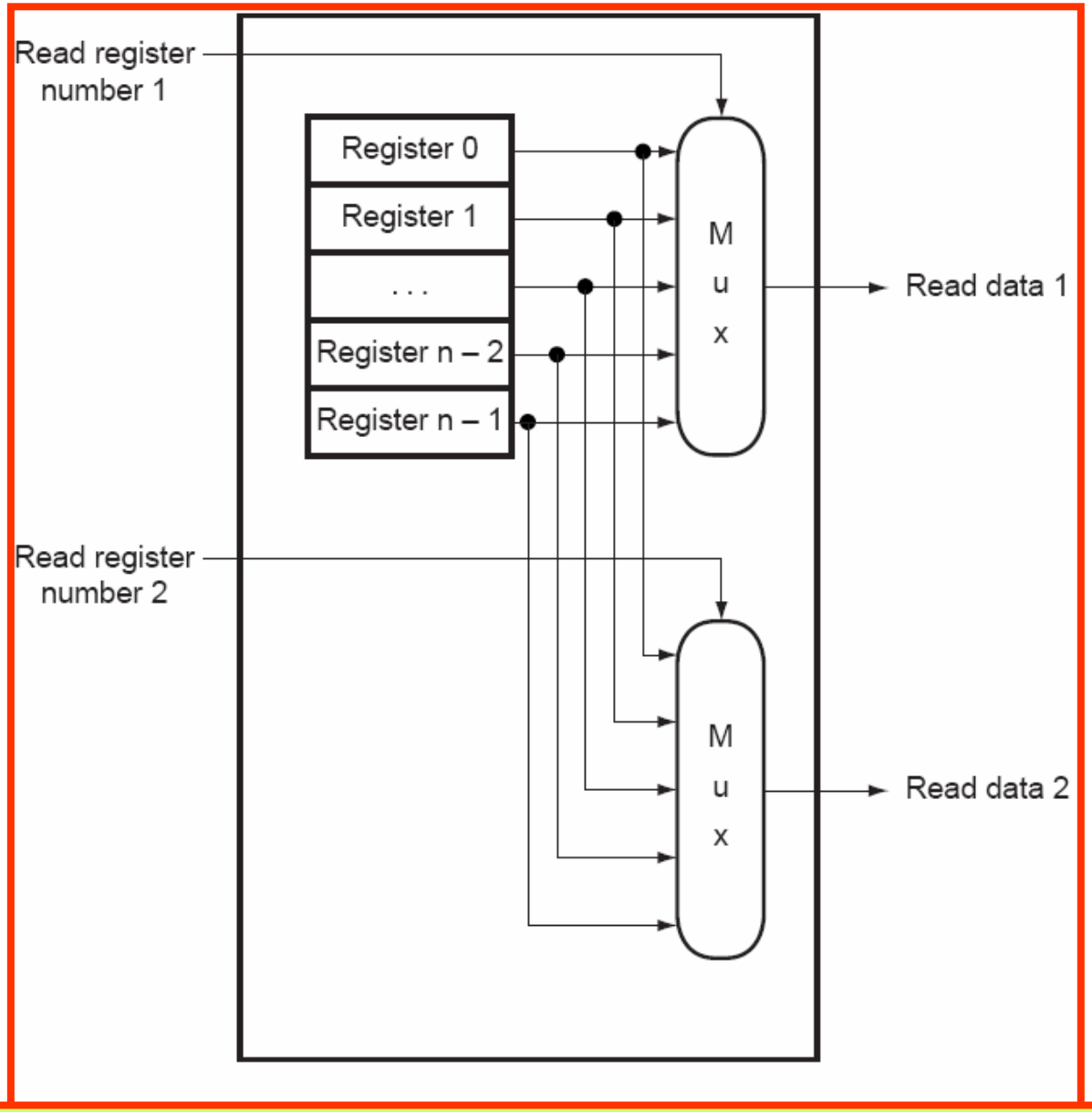
Building a Datapath

➢ Implementing R-format operations (arithmetic & logical)

- **R-format instructions have three register operands – read operands from the two registers, perform an ALU operation on these operands and write the result to the third register**
- **32 registers in MIPS processor are stored in a structure called a register file.**
- **A register file is a collection of registers in which any register can be read or written by specifying the number of the register in the file.**
- **An ALU is required to operate on the values read from the registers**

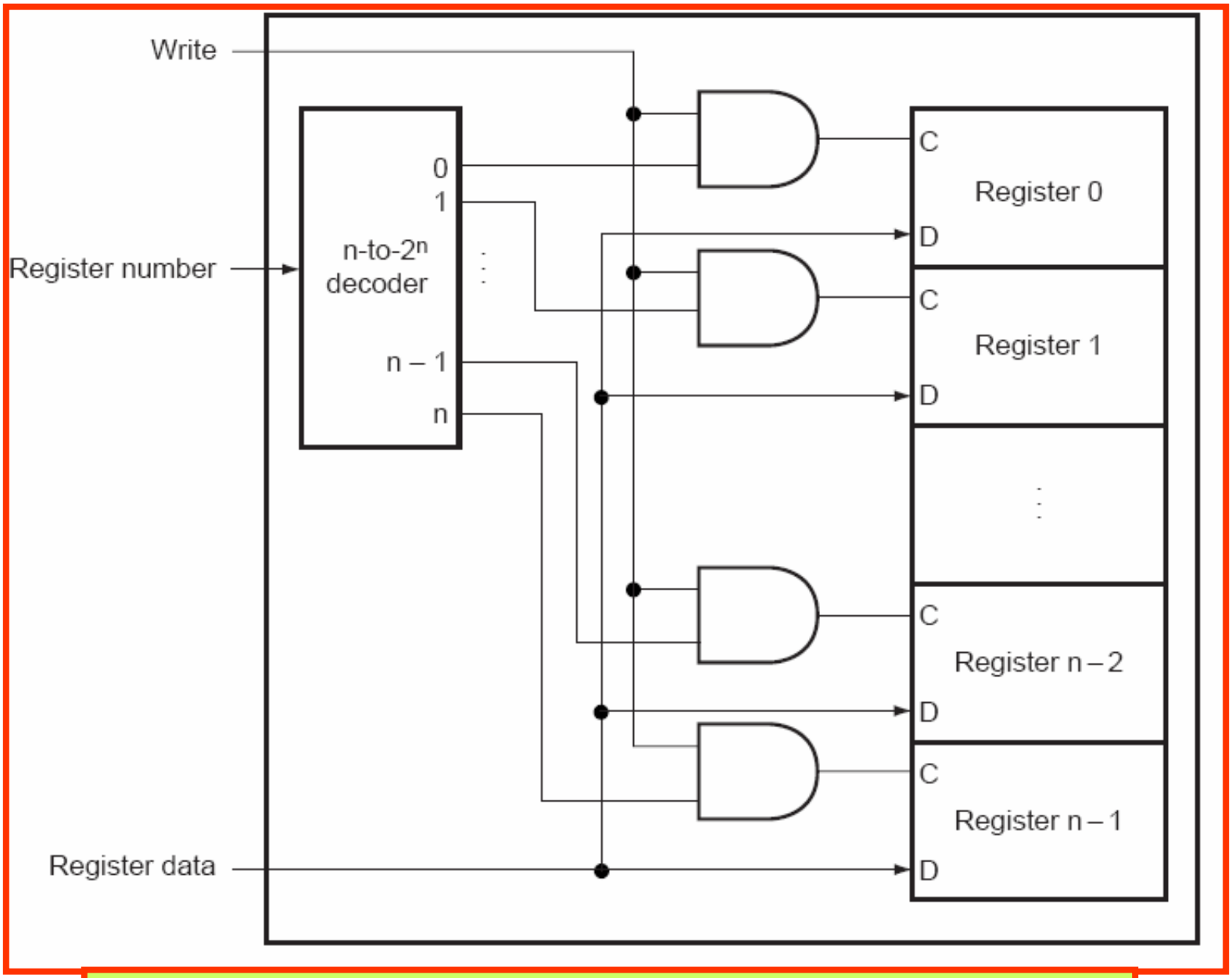


Implementing a Register File



Implementation of Two Read ports

Implementing a Register File

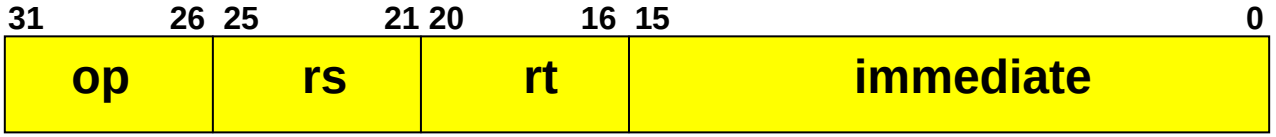


Implementation of One write port

I-type Instruction format

load instruction : **lw rt, offset (rs)**

store instruction : **sw rt, offset (rs)**



Opcode (6 bits)

Base address Register (5 bits)

Destination Register for load (lw) Or Source Register for store (sw) (5 bits)

Offset (16 bits)

Effective address = contents of rs + offset sign-extended to 32 bits

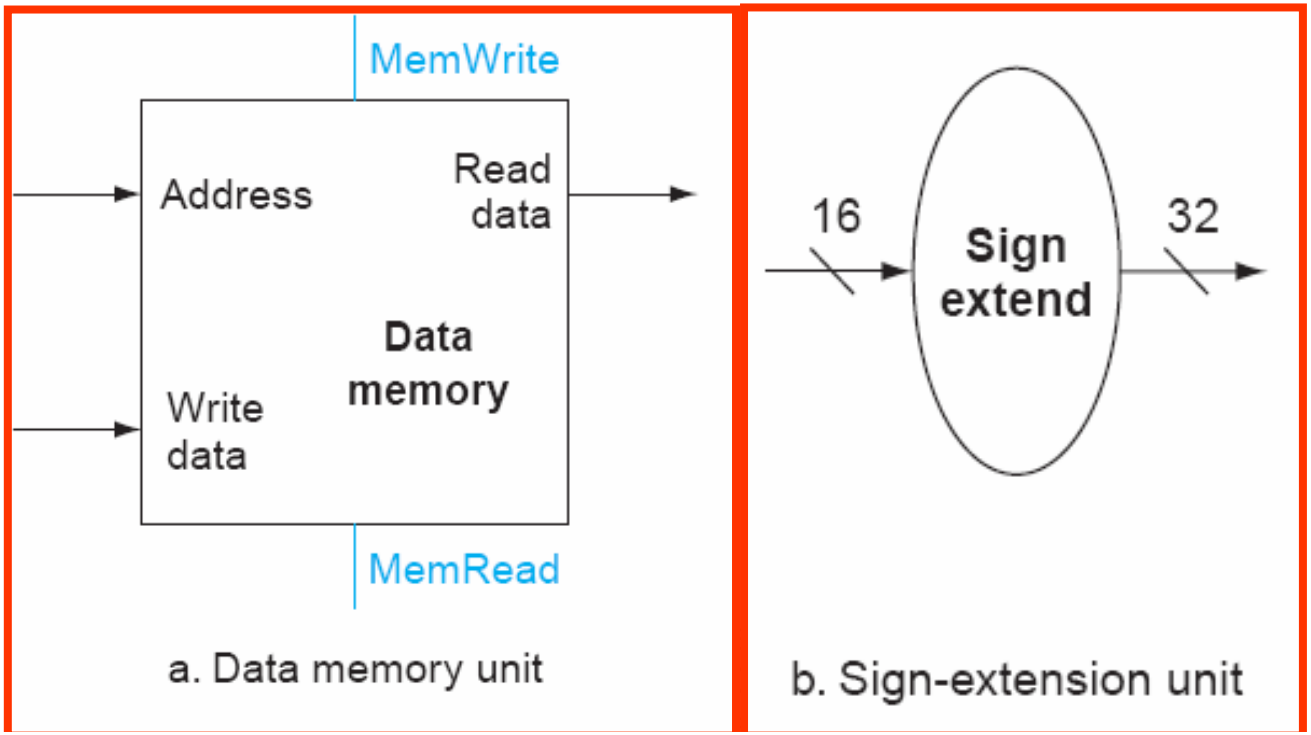
Building a Datapath

➢ Implementing Load and Store operations

- These instructions compute a memory address by adding the base register to the 16-bit signed offset field contained in the instruction
- Components needed

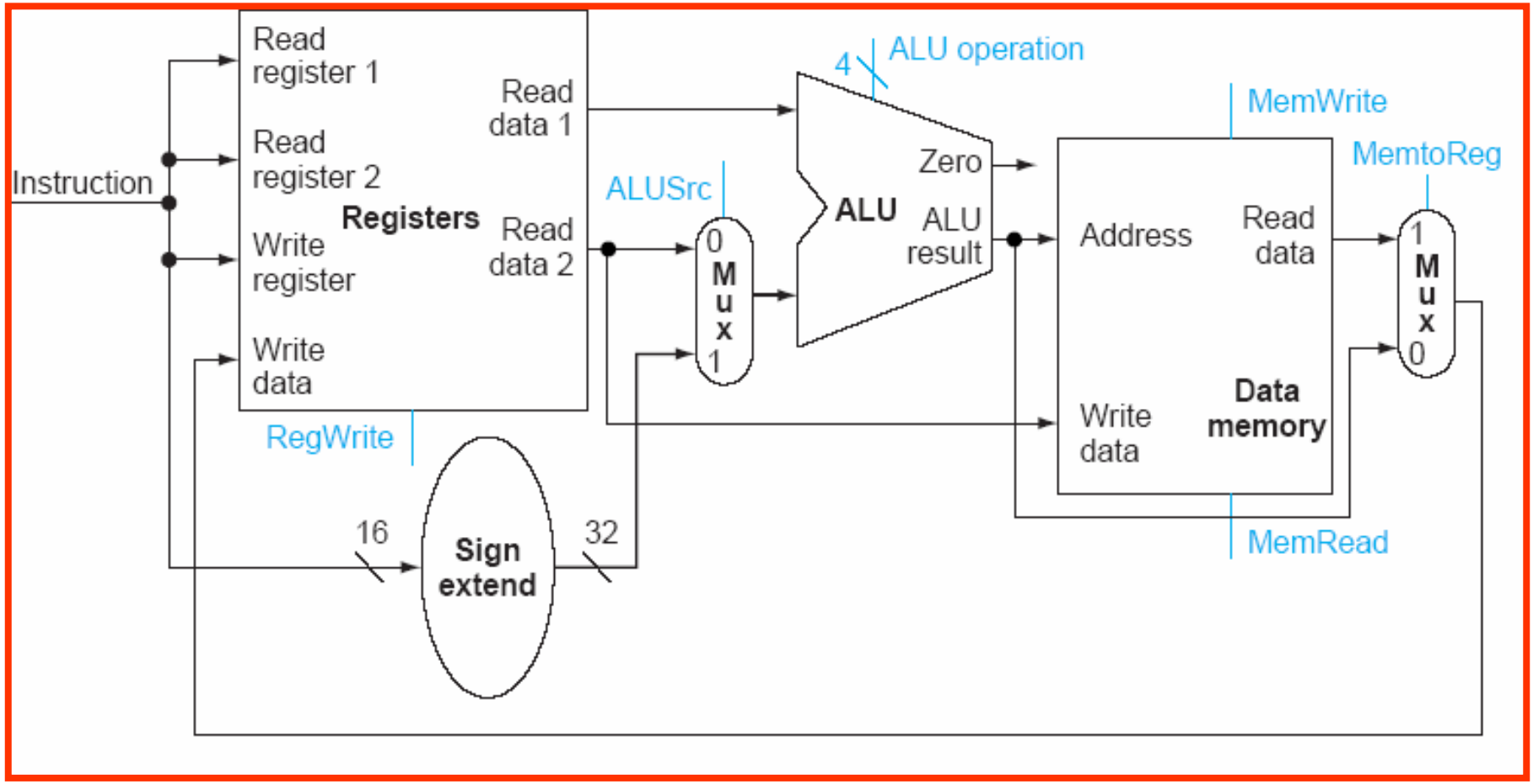
-A **sign-extension unit** to sign-extend the 16-bit offset field in the instruction to a 32-bit signed value

-A **Data Memory unit** to read from or write to. Data memory must be read on load instructions and written on store instructions



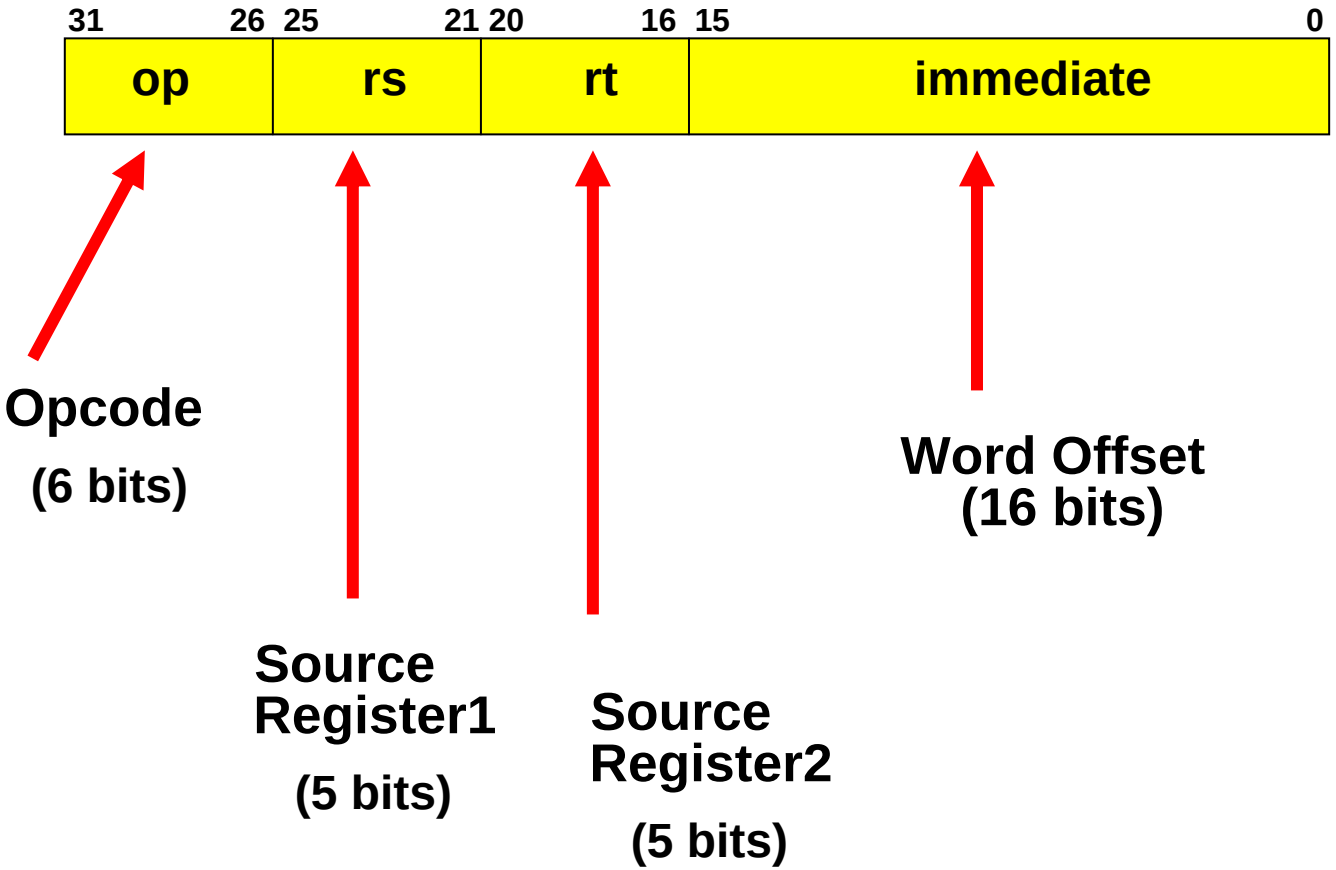
Building a Datapath

- **Combining the datapaths for the memory instructions and the R-type instructions**



I-type Instruction format

Branch instruction : beq rs, rt, target

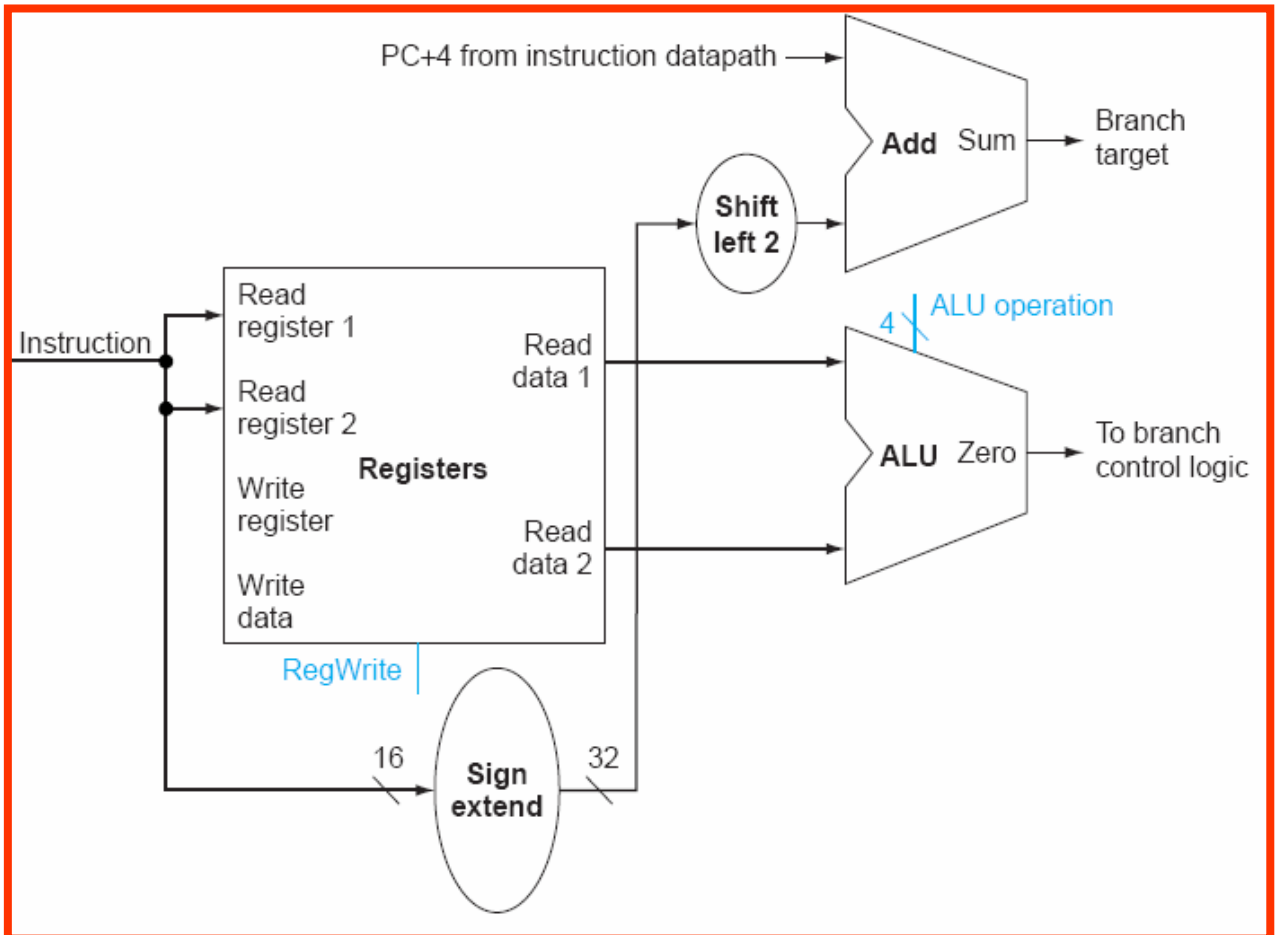


Target address = (Current PC + 4) + 4*offset sign extended to 32 bits

Building a Datapath

➢ Implementing branch operations

- Branch instruction (e.g., beq) has three operands, two registers that are compared for equality, and a 16-bit offset used to compute the branch target address relative to the branch instruction address.
- Compute the branch target address by adding the sign-extended offset field of the instruction to the PC.
- Before adding, the offset field is shifted left by 2 bits so that it is a word offset.



Building a Datapath

- **Simple Datapath for the MIPS Architecture combining the elements required by different instruction classes**