CPU Datapath And Control I

Instructor: Dmitri A. Gusev

Fall 2007

CS 502: Computers and Communications

Lecture 6, September 24, 2007

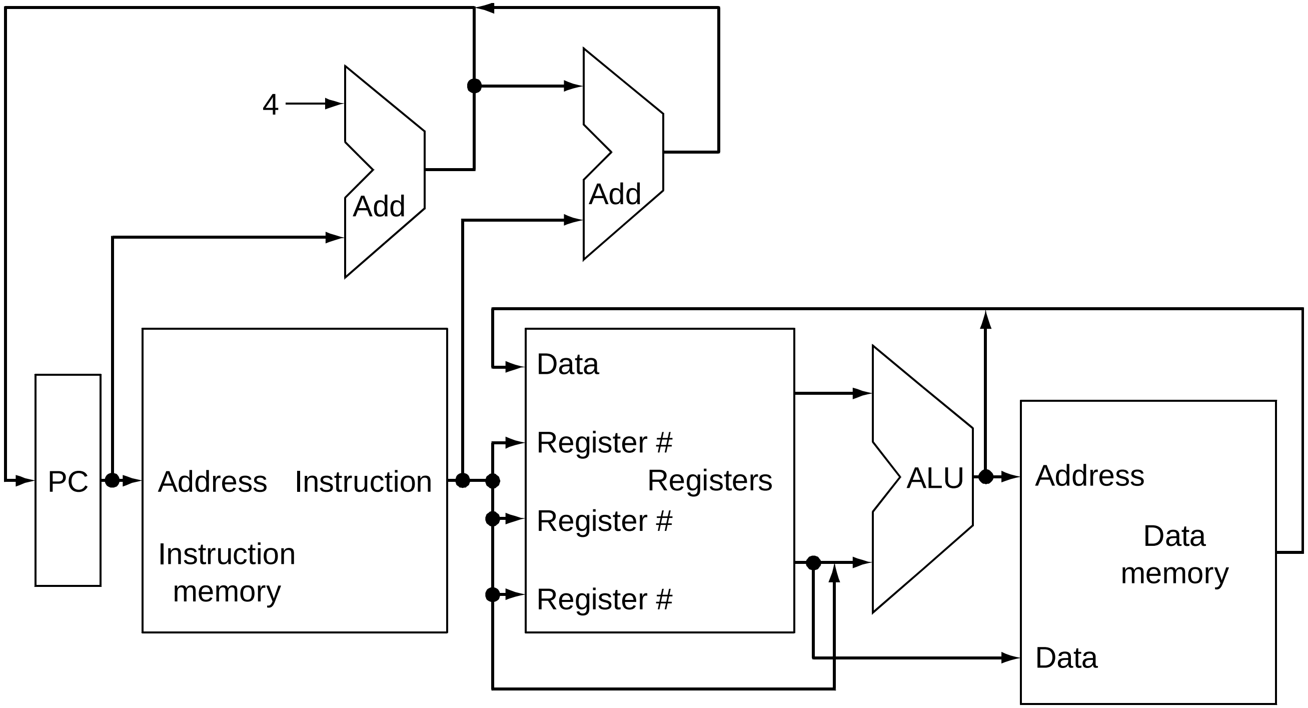
The Processor: Datapath & Control

- We're ready to look at an implementation of the MIPS
- Simplified to contain only:
  - 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 instruction address
  - get the instruction from memory
  - read registers
  - use the instruction to decide exactly what to do
- All instructions use the ALU after reading the registers

  Why? memory-reference? arithmetic? control flow?

More Implementation Details

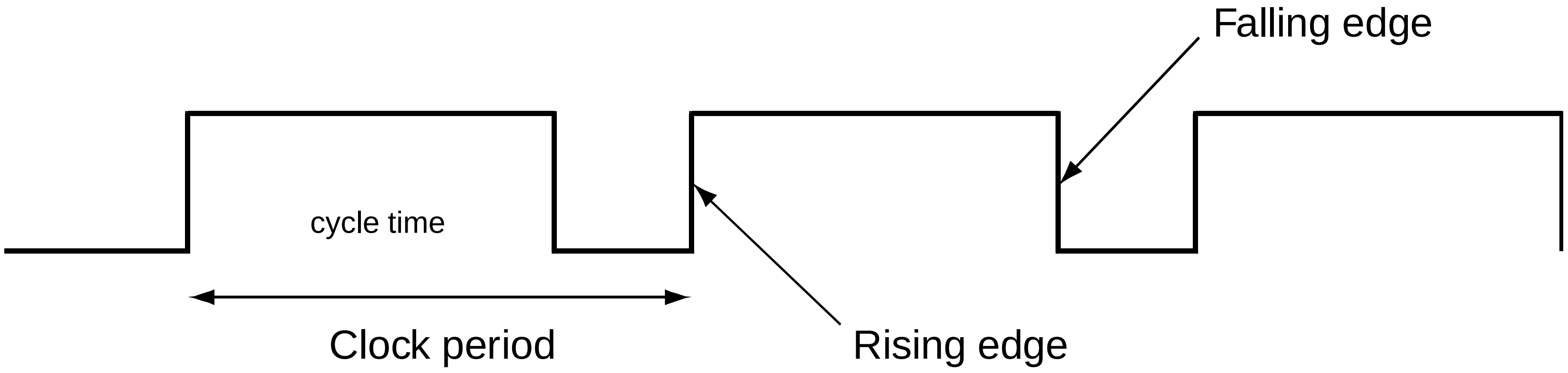
- Abstract / Simplified View:



- Two types of functional units:
  - elements that operate on data values (combinational)
  - elements that contain state (sequential)

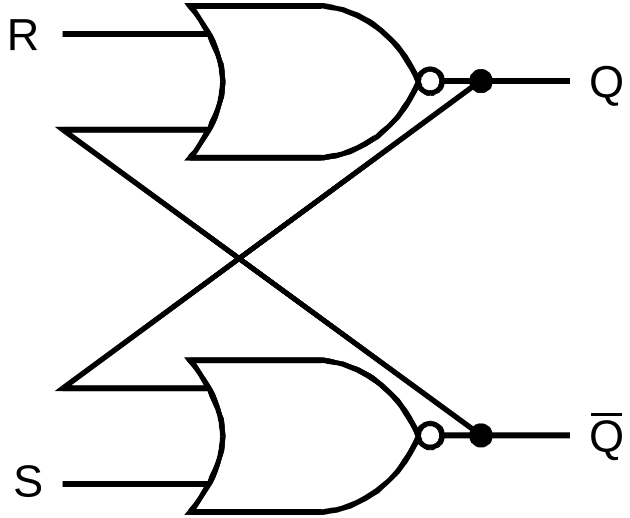
State Elements

- Unclocked vs. Clocked
- Clocks used in synchronous logic
  - – when should an element that contains state be updated?



An unclocked state element

- The set-reset latch (a.k.a. “S-R flip-flop”)
  - output depends on present inputs and also on past inputs



Latches and Flip-flops

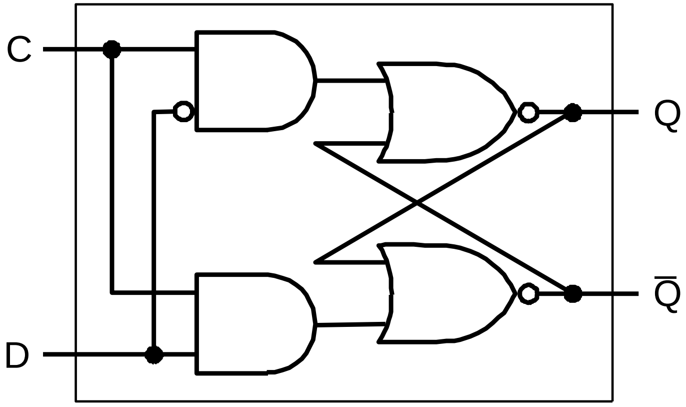
- Output is equal to the stored value inside the element
  (don't need to ask for permission to look at the value)
- Change of state (value) is based on the clock
- Latches: whenever the inputs change, and the clock is asserted
- Flip-flop: state changes only on a clock edge
  (edge-triggered methodology)

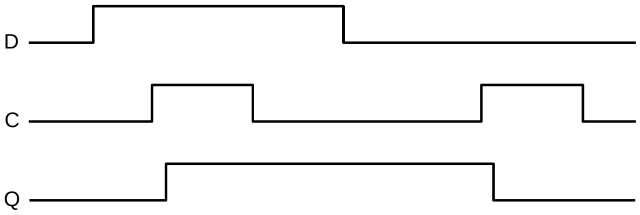
"logically true",
— could mean electrically low

A clocking methodology defines when signals can be read and written
— wouldn't want to read a signal at the same time it was being written

D-latch

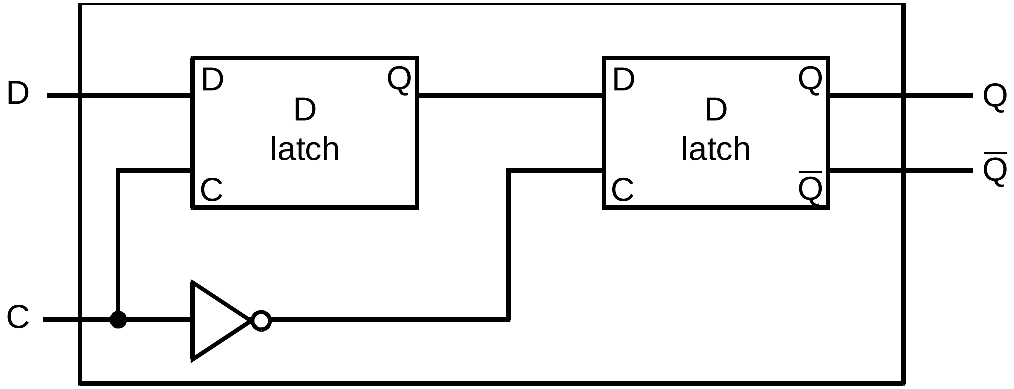
- Two inputs:
  - the data value to be stored (D)
  - the clock signal (C) indicating when to read & store D
- Two outputs:
  - the value of the internal state (Q) and it's complement

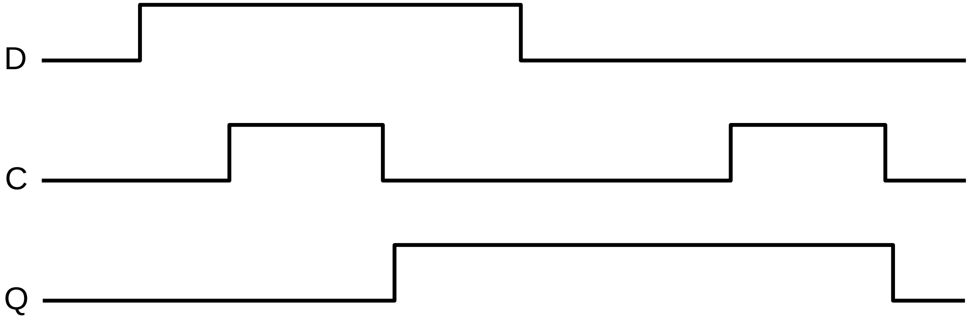




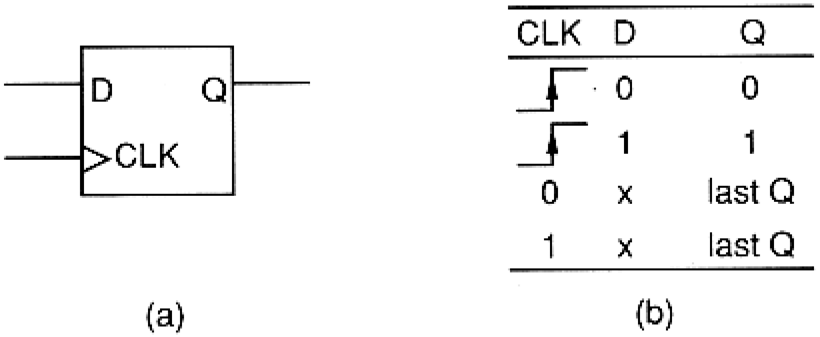
D flip-flop

- Output changes only on the clock edge





Circuits as Memory (cont’d)



(a)

CLK	D	Q
↑	0	0
↑	1	1
0	x	last Q
1	x	last Q

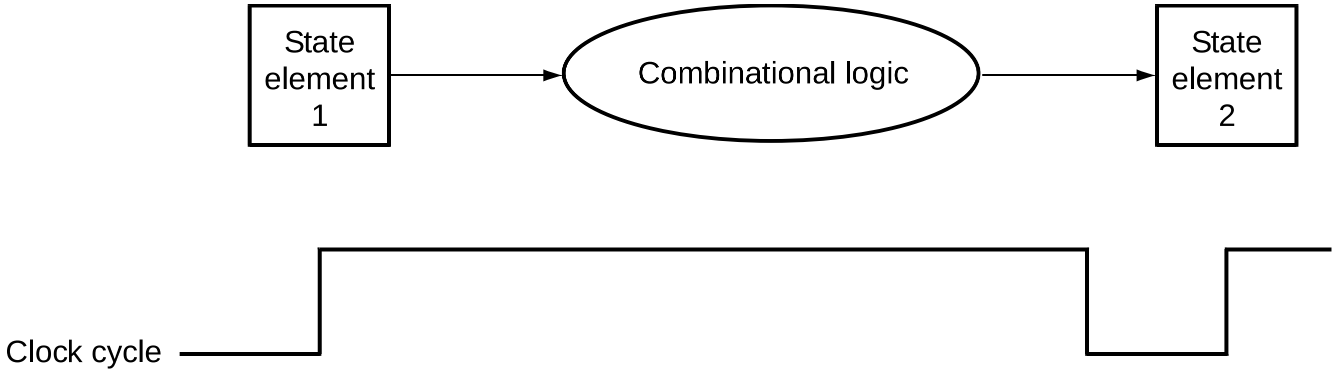
(b)

Positive edge-triggered D flip-flop: (a) symbol; (b) function table.

D flip-flops are grouped together into *registers* to store multi-bit quantities in a computer.

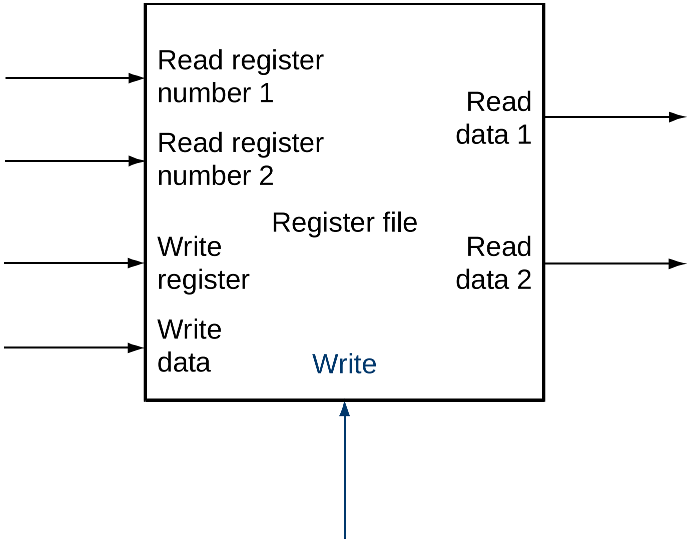
Our Implementation

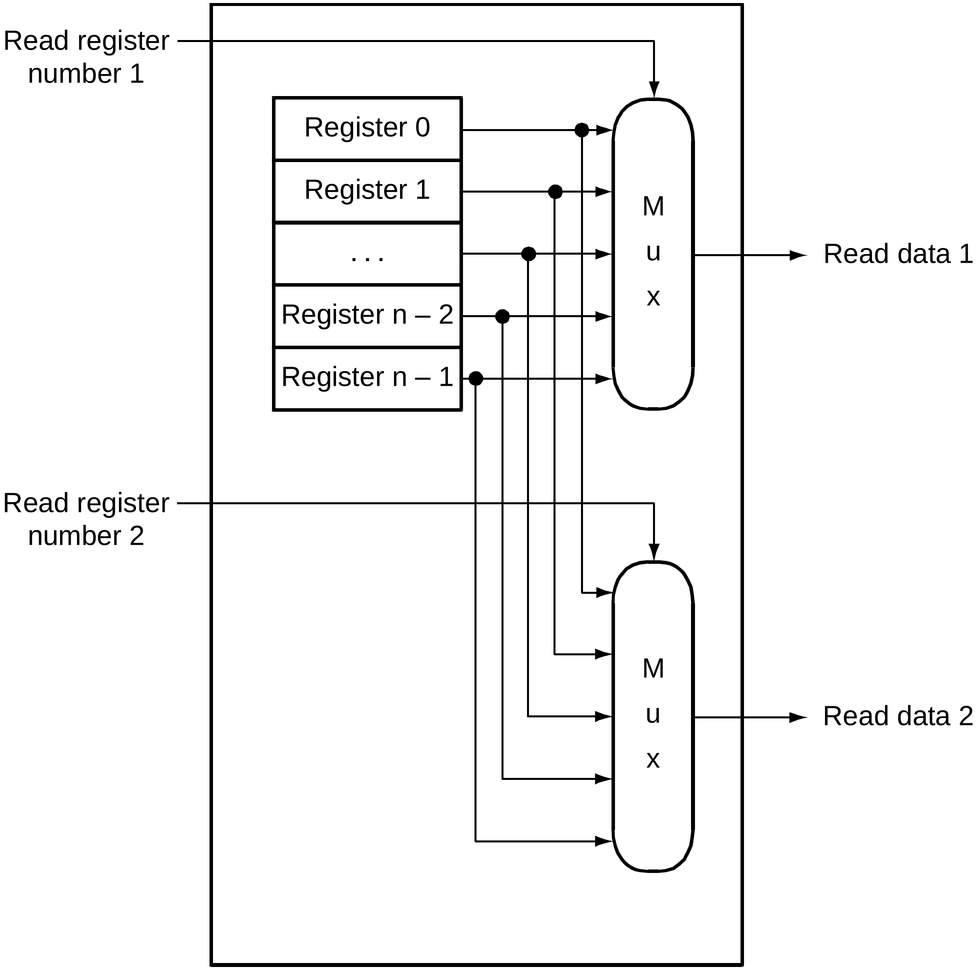
- An edge triggered methodology
- Typical execution:
  - read contents of some state elements,
  - send values through some combinational logic
  - write results to one or more state elements



Register File

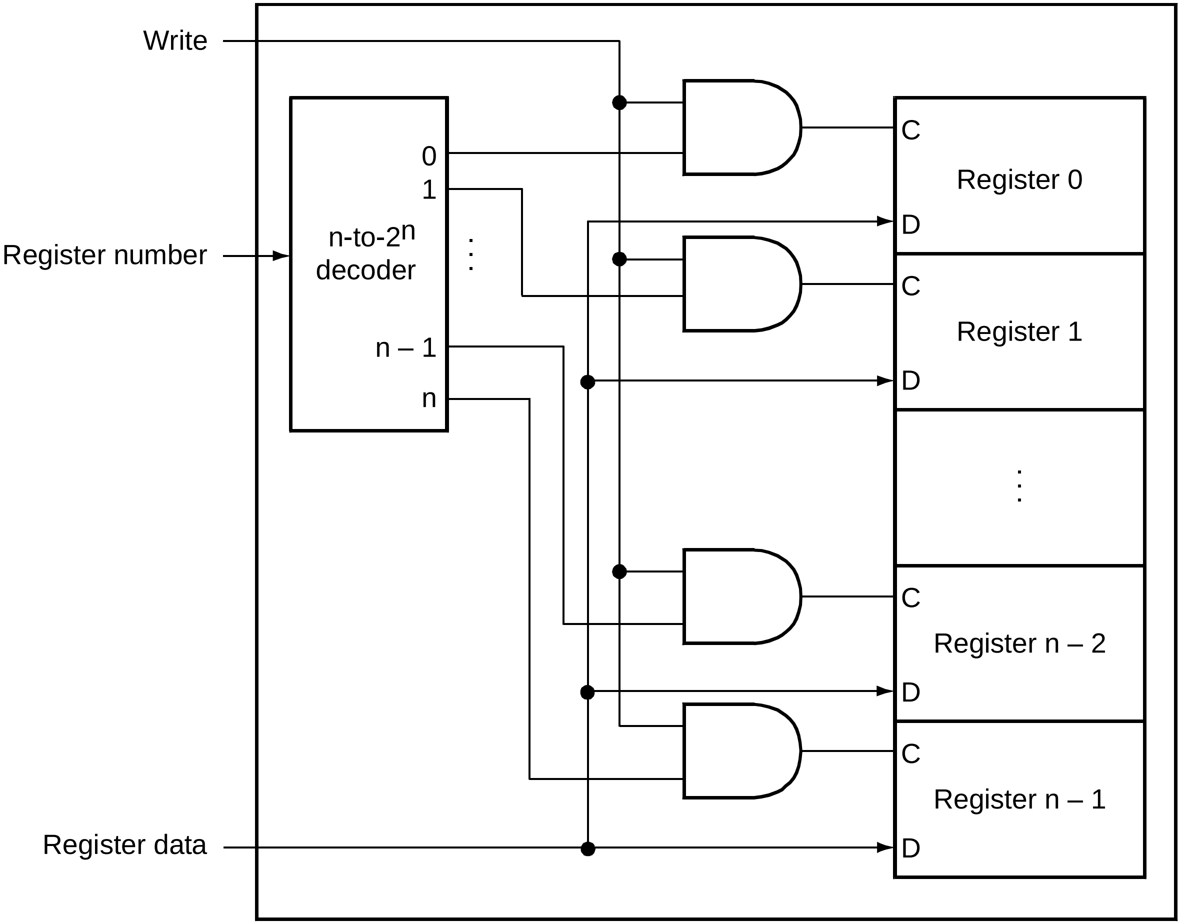
- Built using D flip-flops





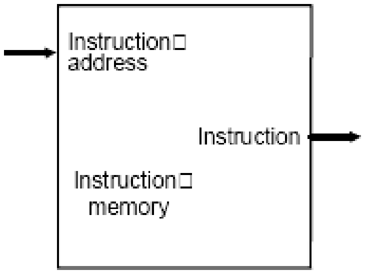
Register File

- Note: we still use the real clock to determine when to write

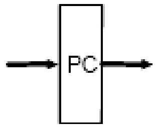


Simple Implementation

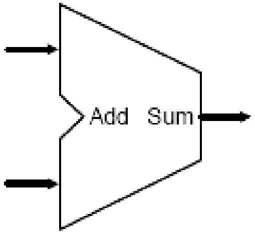
- Include the functional units we need for each instruction



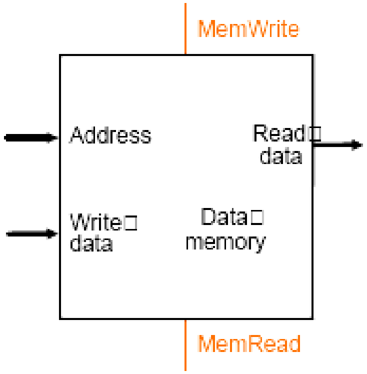
a. Instruction memory



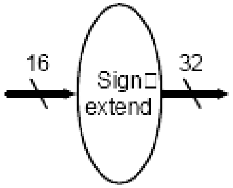
b. Program counter



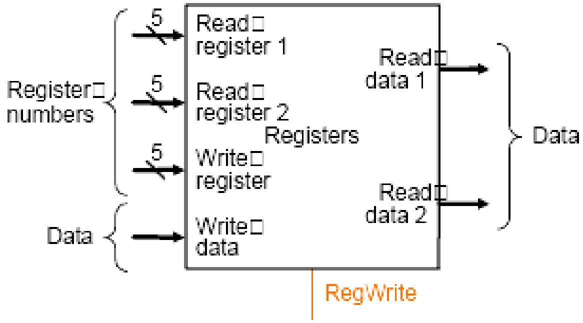
c. Adder



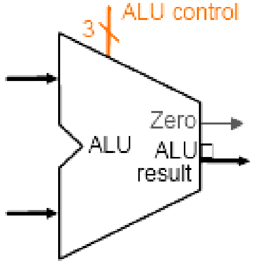
a. Data memory unit



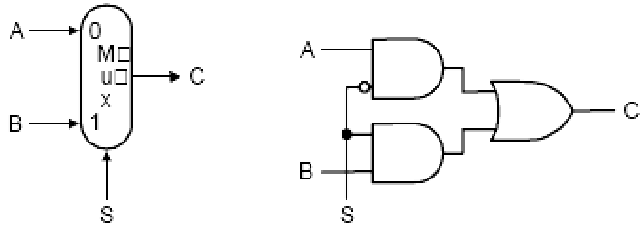
b. Sign-extension unit



a. Registers



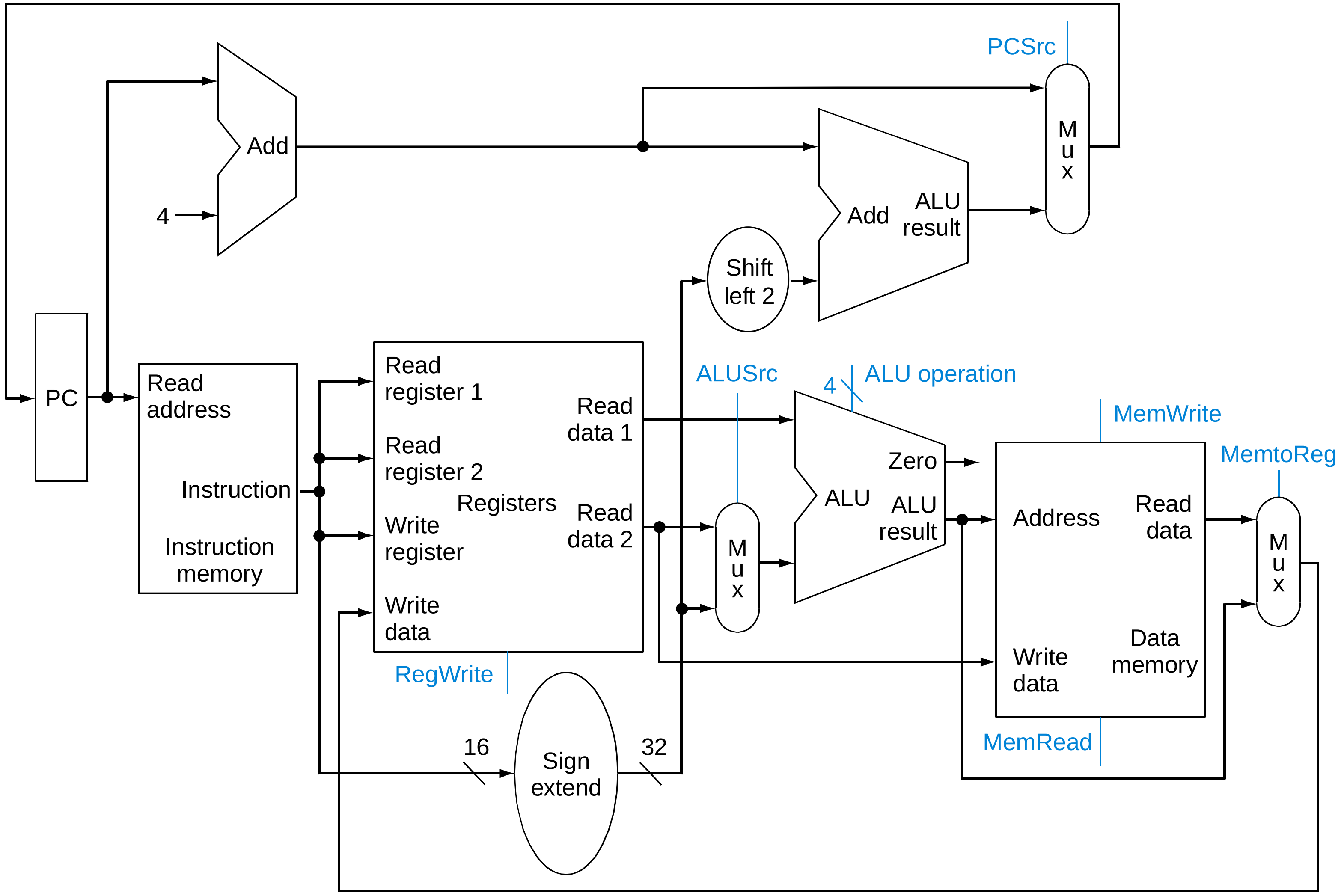
b. ALU



Multiplexor

Building the Datapath

- Use multiplexors to stitch them together


PCSrc
Add
4
M
u
x
Add
ALU
result
Shift
left 2
PC
Read
address
Instruction
Instruction
memory
Read
register 1
Read
register 2
Registers
Write
register
Write
data
Read
data 1
Read
data 2
RegWrite
ALUSrc
M
u
x
4
ALU operation
Zero
ALU
ALU
result
MemWrite
MemtoReg
Address
Read
data
Write
data
Data
memory
MemRead
M
u
x
16
Sign
extend
32