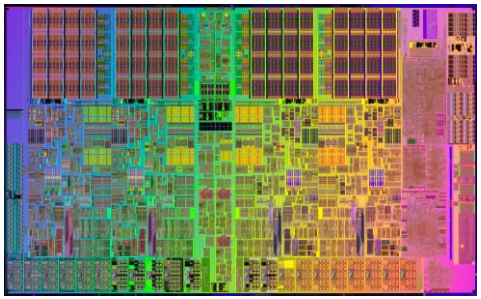
Computer Architecture

CH4 Processor Microarchitecture (I)

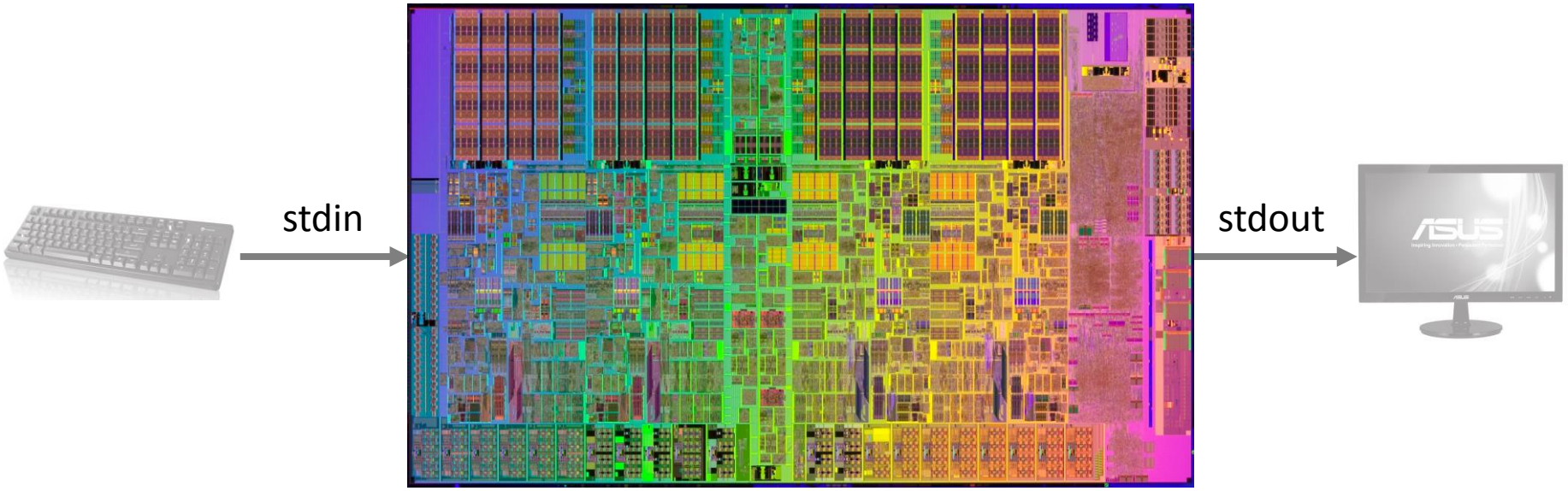
Prof. Ren-Shuo Liu
NTHU EE
Fall 2017



Outline

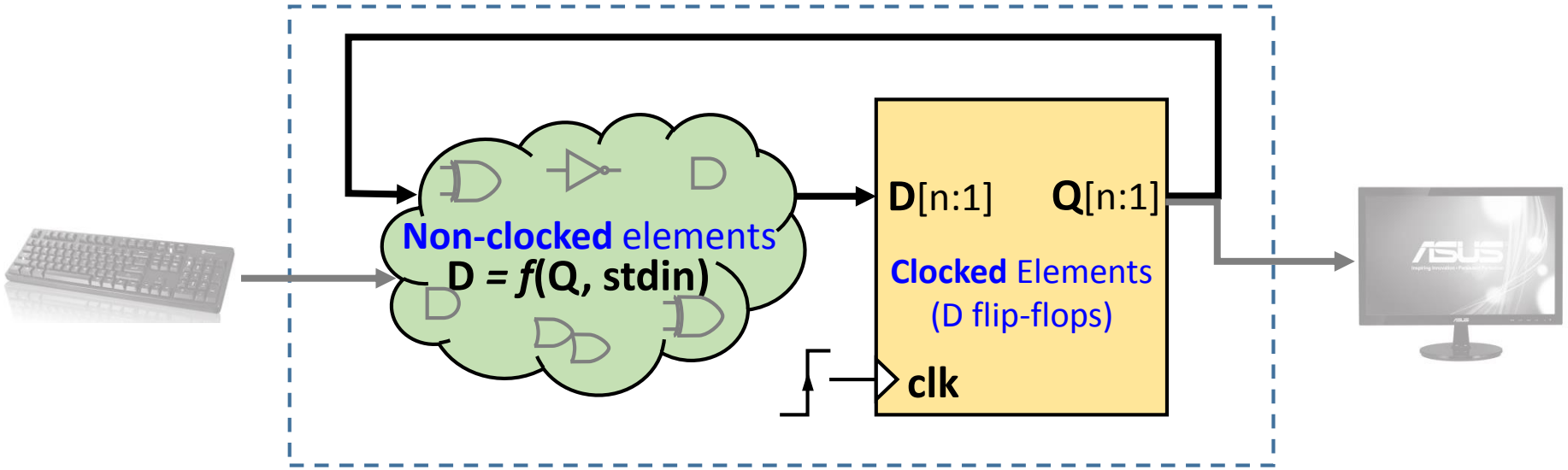
- Background
- Single-cycle design
- Pipelined design

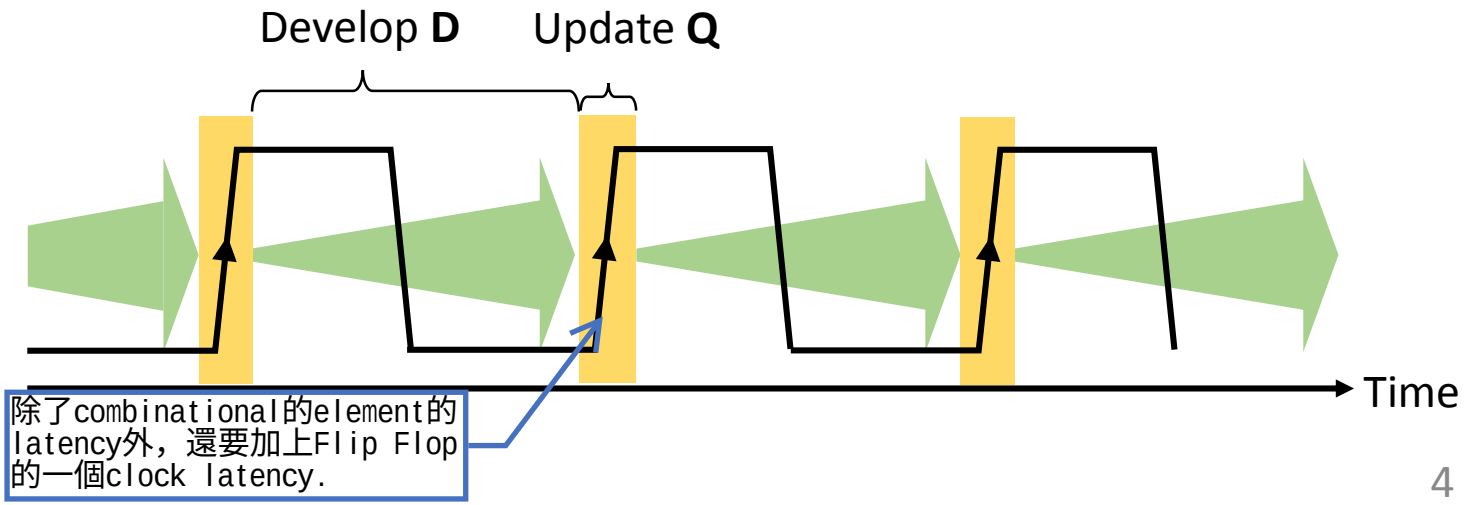
Processor



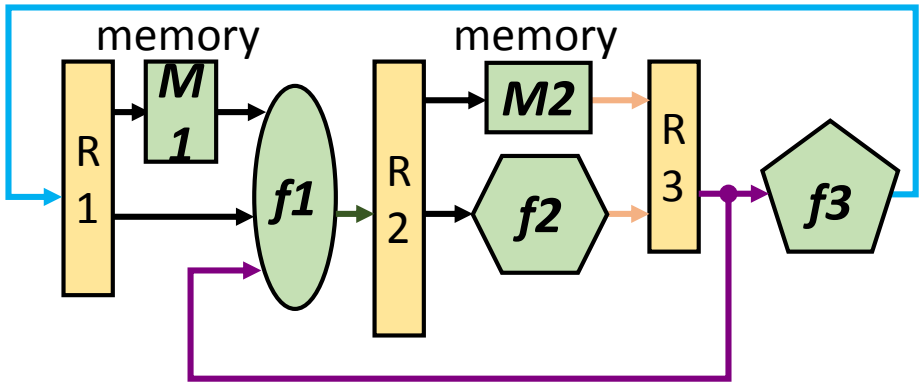
1~2 billion transistors in a modern processor

Processor

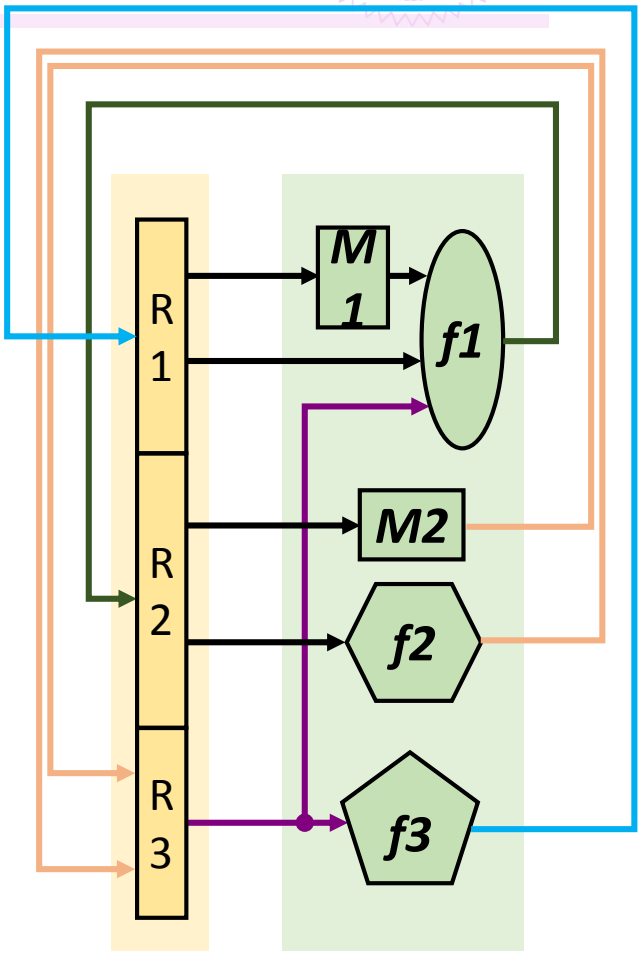

Non-clocked elements
D = f(Q, stdin)
D[n:1]
Q[n:1]
Clocked Elements
(D flip-flops)
clk


Develop D
Update Q
Time
除了combinational的element的latency外，還要加上Flip Flop的一個clock latency.

Processor

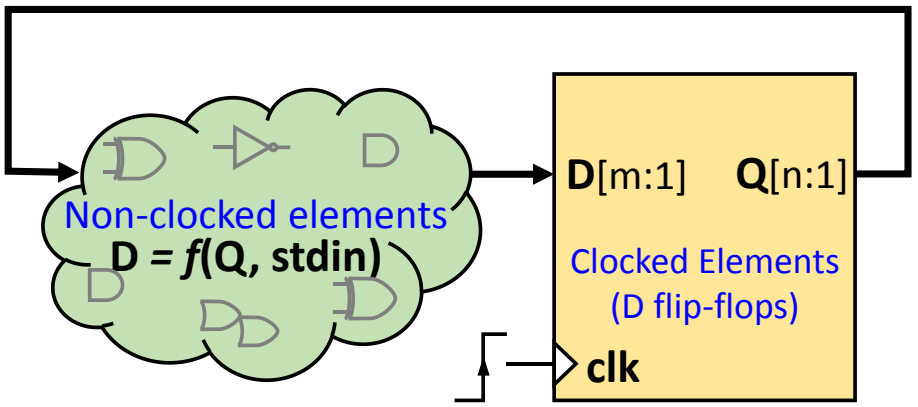

memory
memory
R 1
M 1
f1
R 2
M2
f2
R 3
f3

=


R 1
M 1
f1
R 2
M2
f2
R 3
f3

=

=


Non-clocked elements
D = f(Q, stdin)
D[m:1]
Q[n:1]
Clocked Elements
(D flip-flops)
clk

Simplified Processor Example

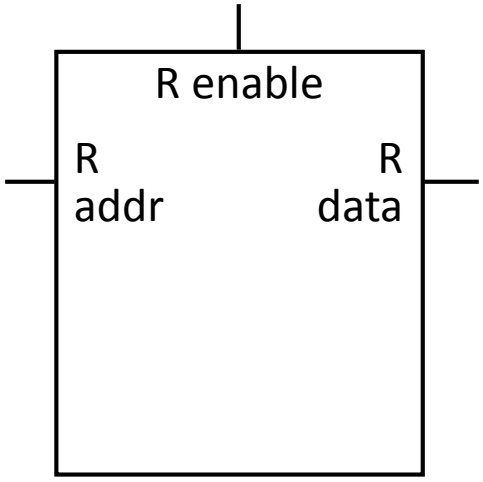
- Single-cycled
- Simplified instruction (sub)set
  - Memory access
    - lw, sw
  - Arithmetic/logical
    - add, sub, and, or, slt
  - Control transfer
    - beq, j

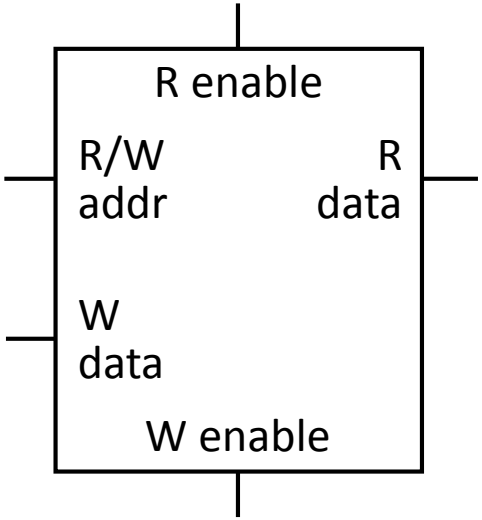
Building Blocks

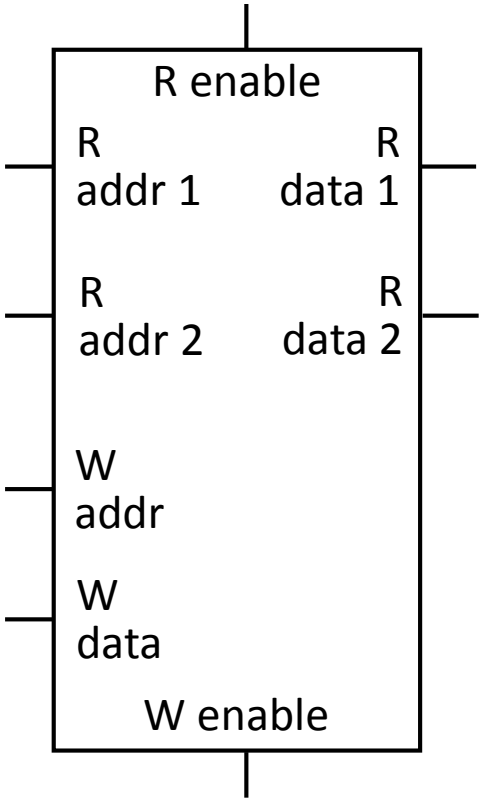
- Memory
  - Instruction memory
  - Data memory
  - Register ($0 ~ $31) memory
- Sign extender
- Shifter (fixed shift amount)
- ALU (as described in CH3)

Memory

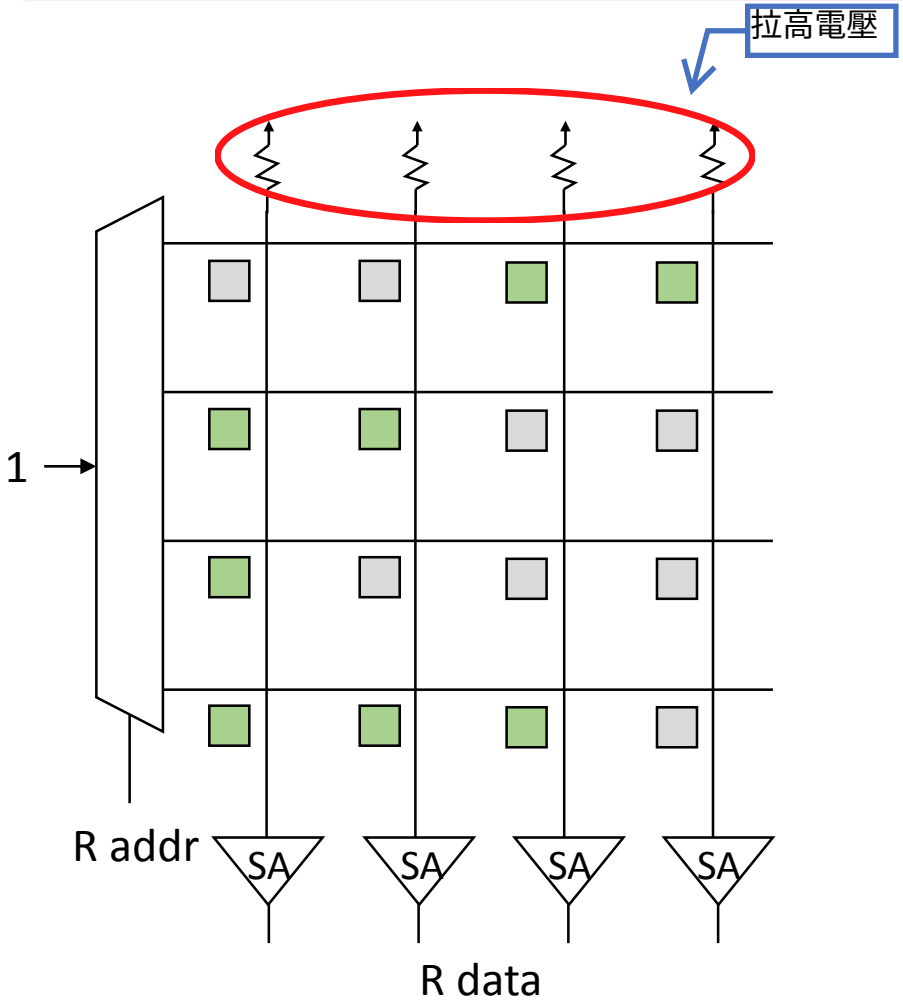
- Read-only memory (ROM) 用於Instruction Memory(事先燒死在記憶體內)
- Random access memory (RAM)
  用於Data Memory(允許做資料搬運)
- Multi-port RAM

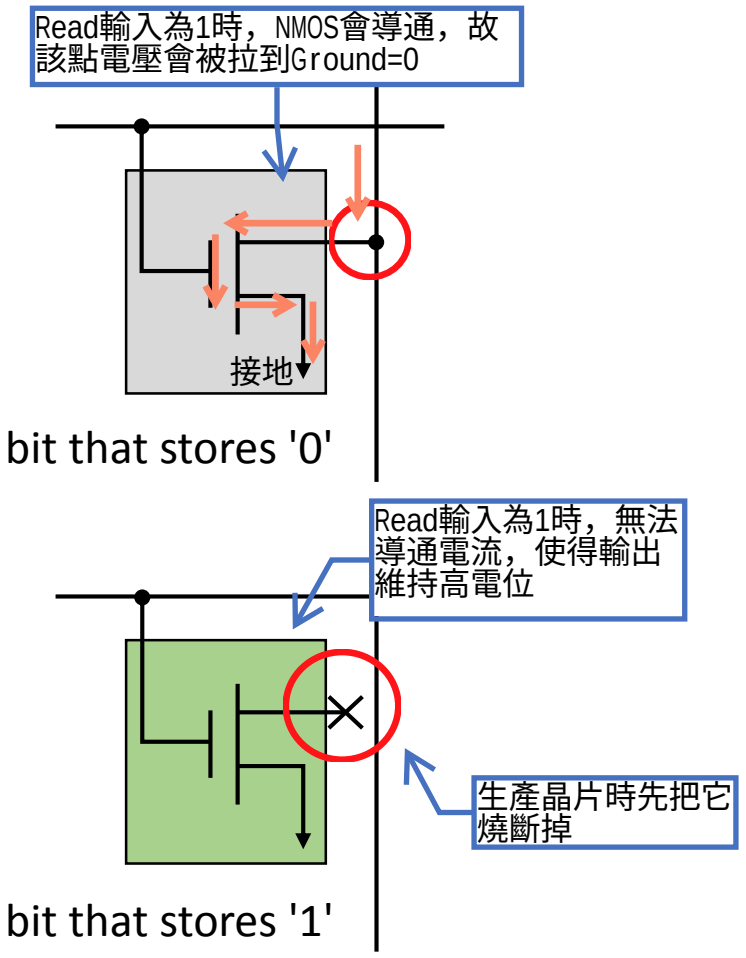






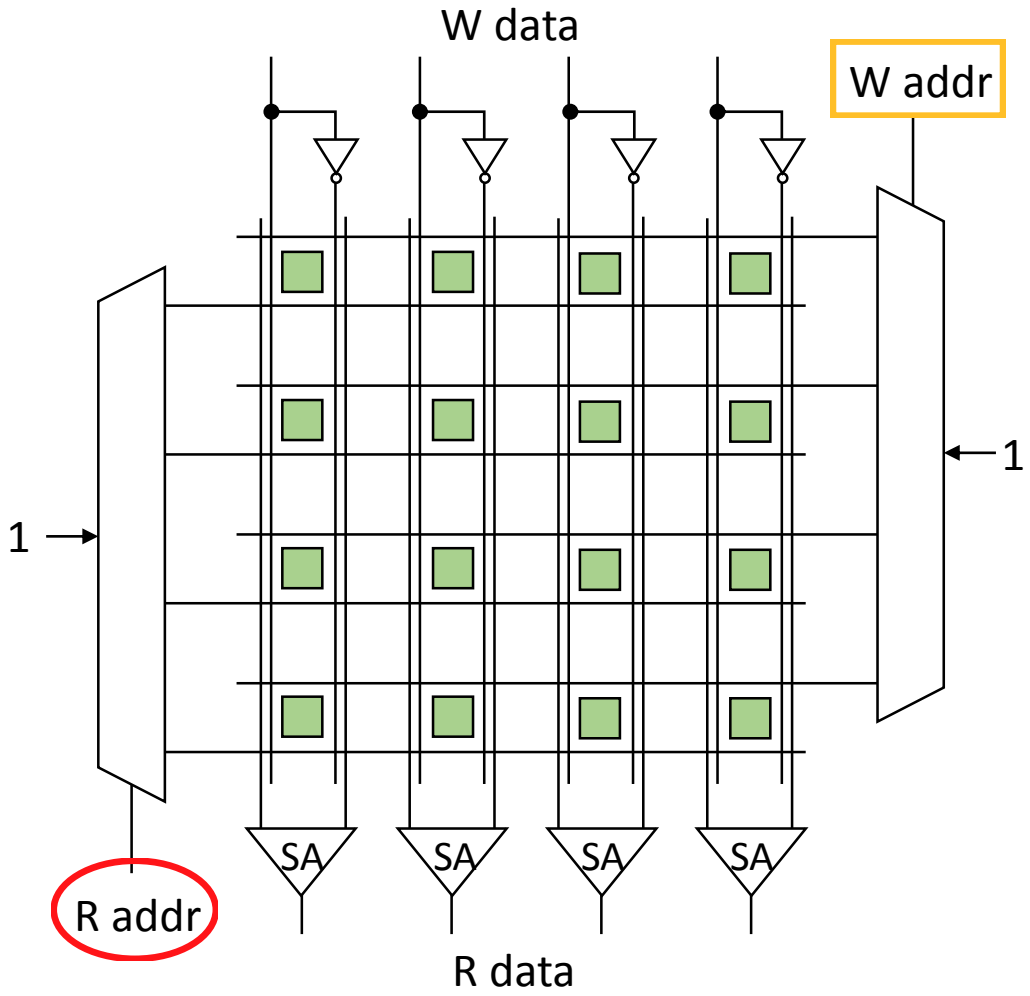
ROM Example

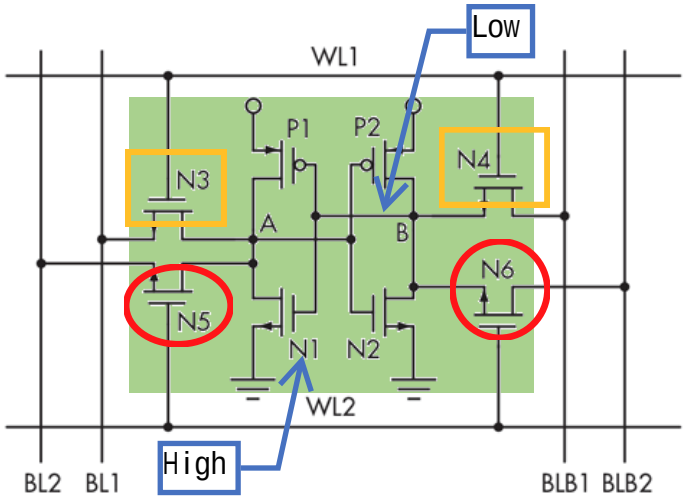




SA: sense amplifier

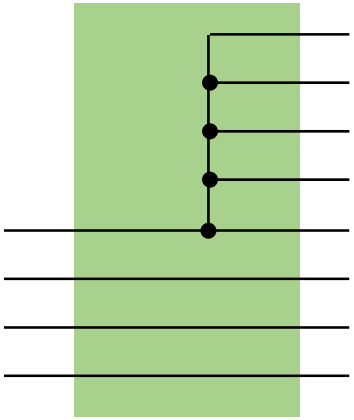
Multi (Dual) -Port RAM Example





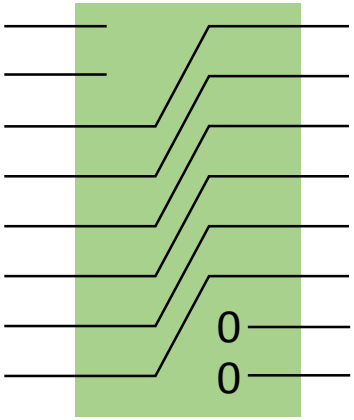
SA: sense amplifier

Sign Extender and Shifter



4-bit signed integer

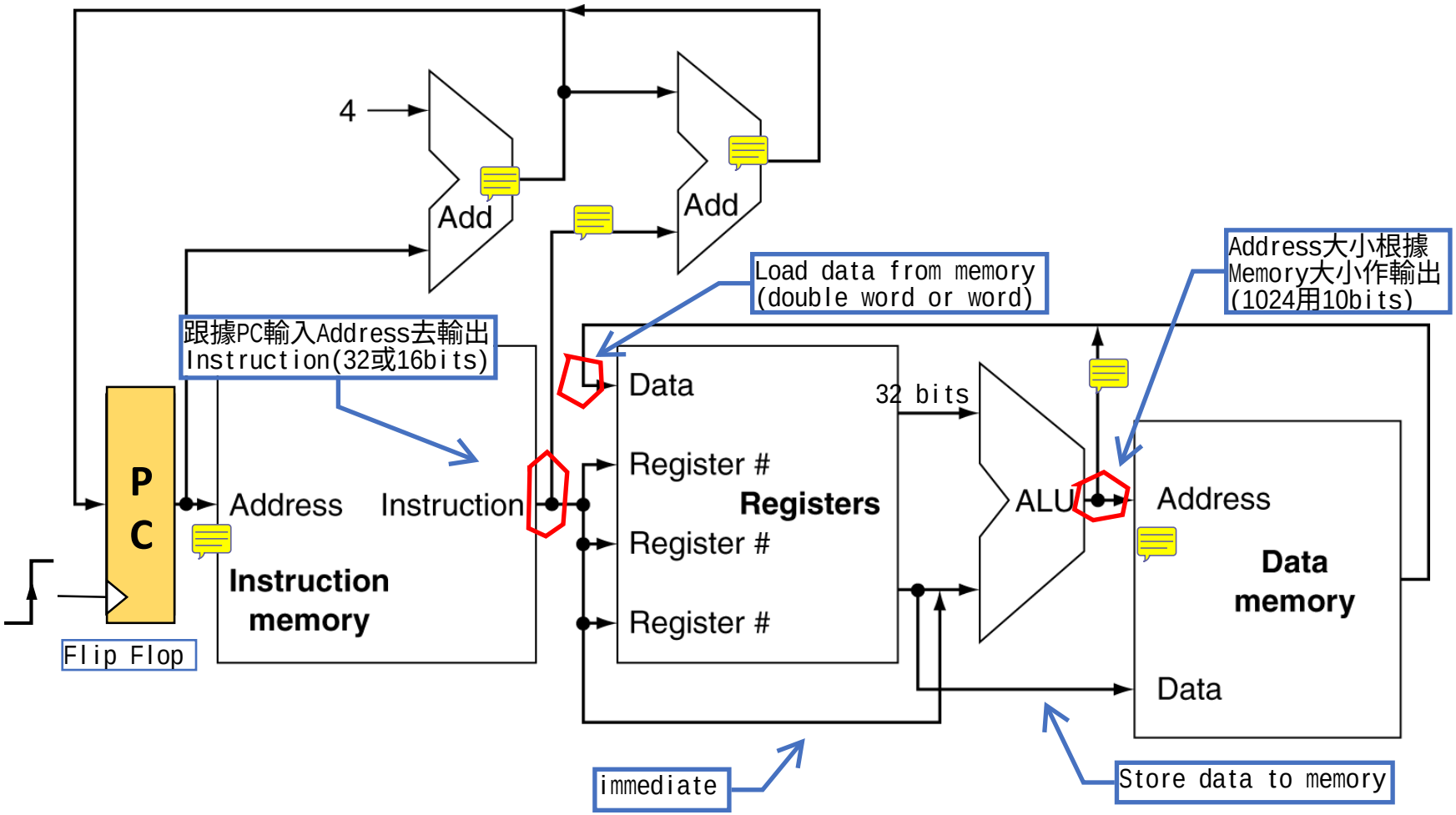
8-bit signed integer



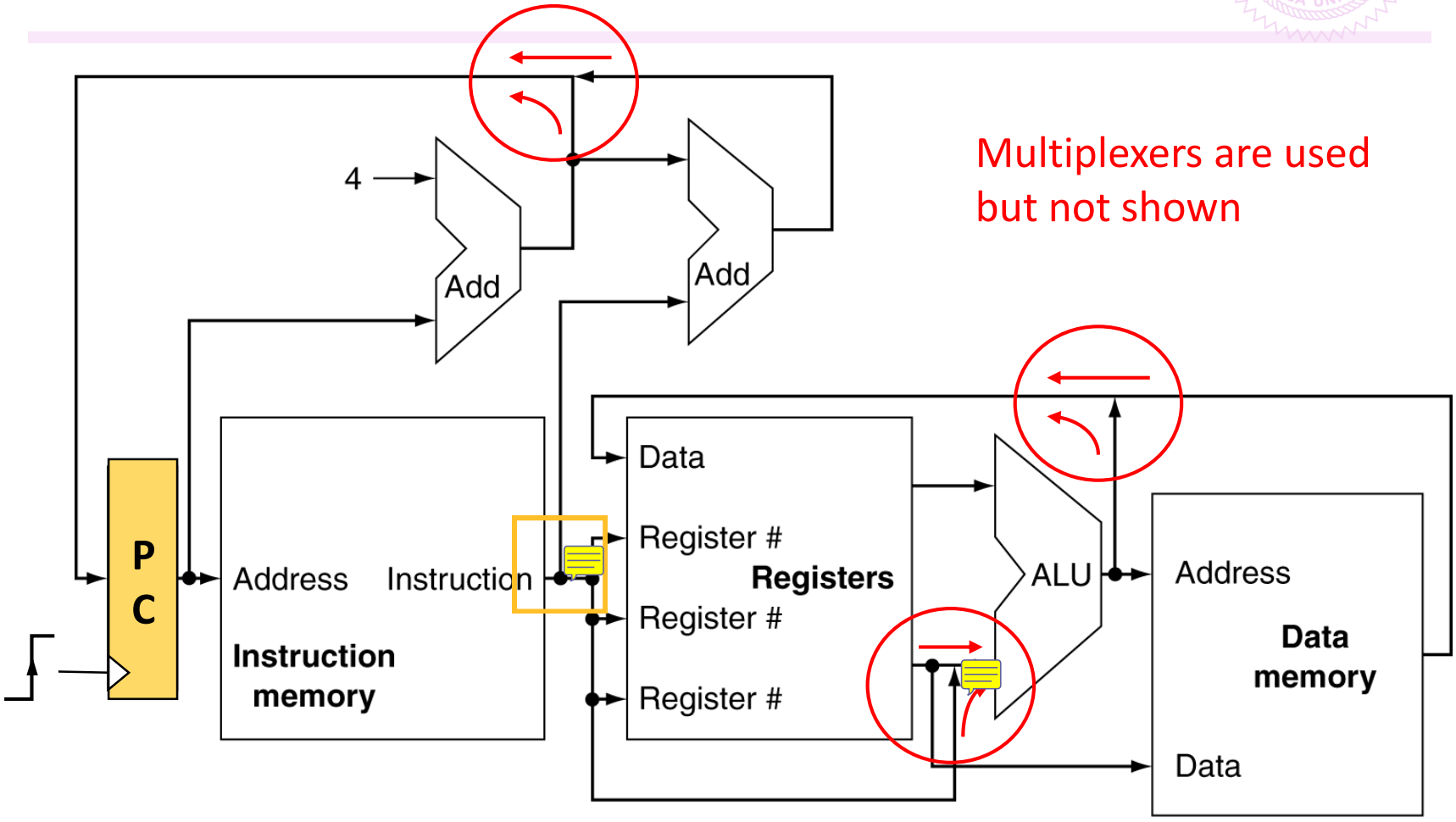
8-bit unsigned integer

8-bit unsigned integer
(fixed shift amount)

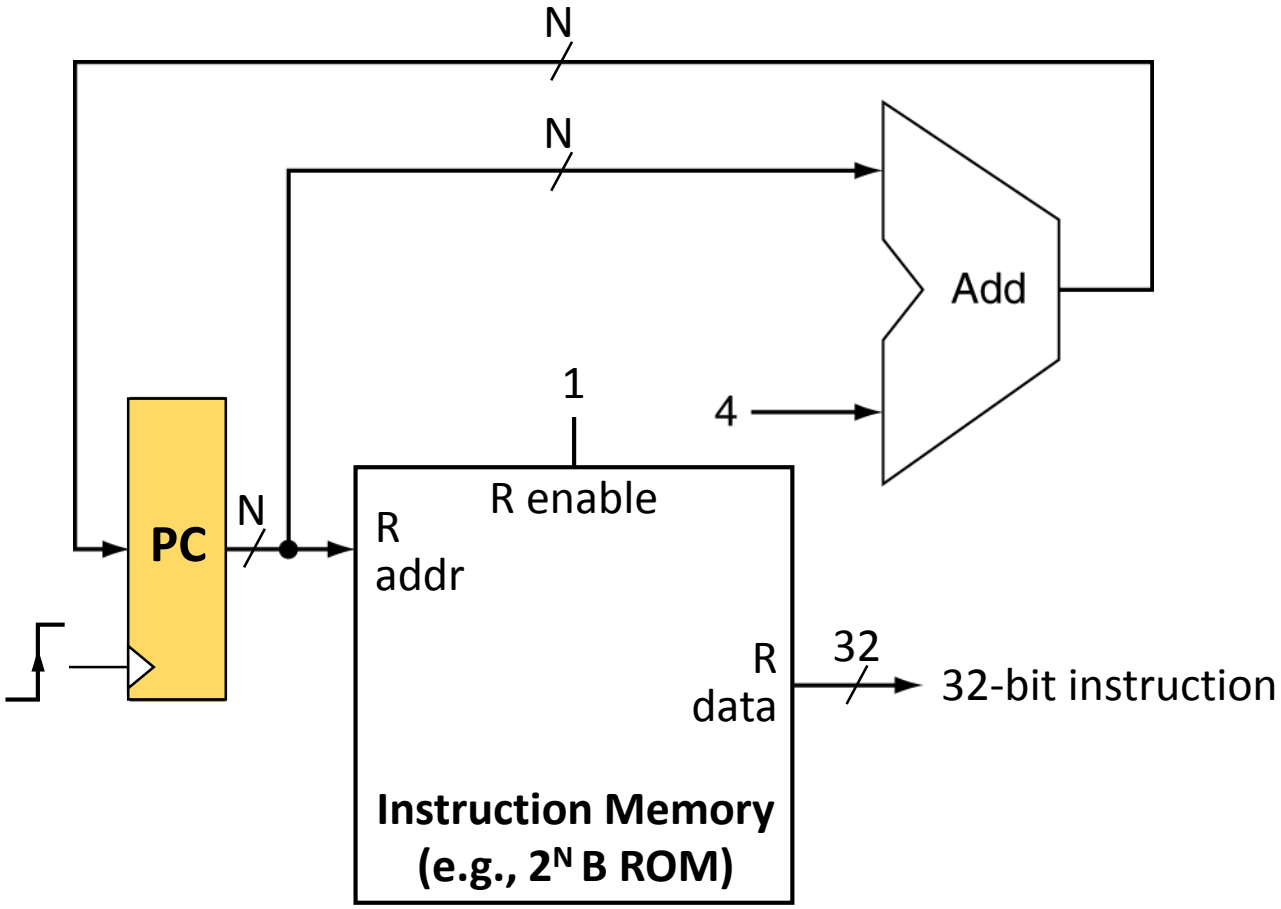
Rough Processor Design



Rough Processor Design (Cont'd)



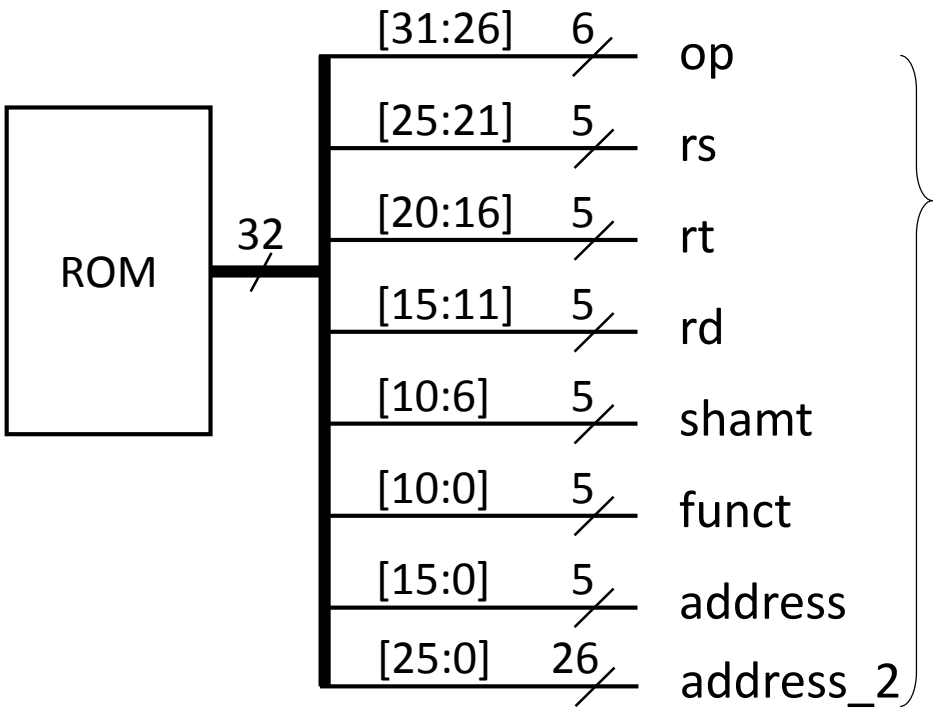
Instruction Fetching Hardware



N can be up to 32

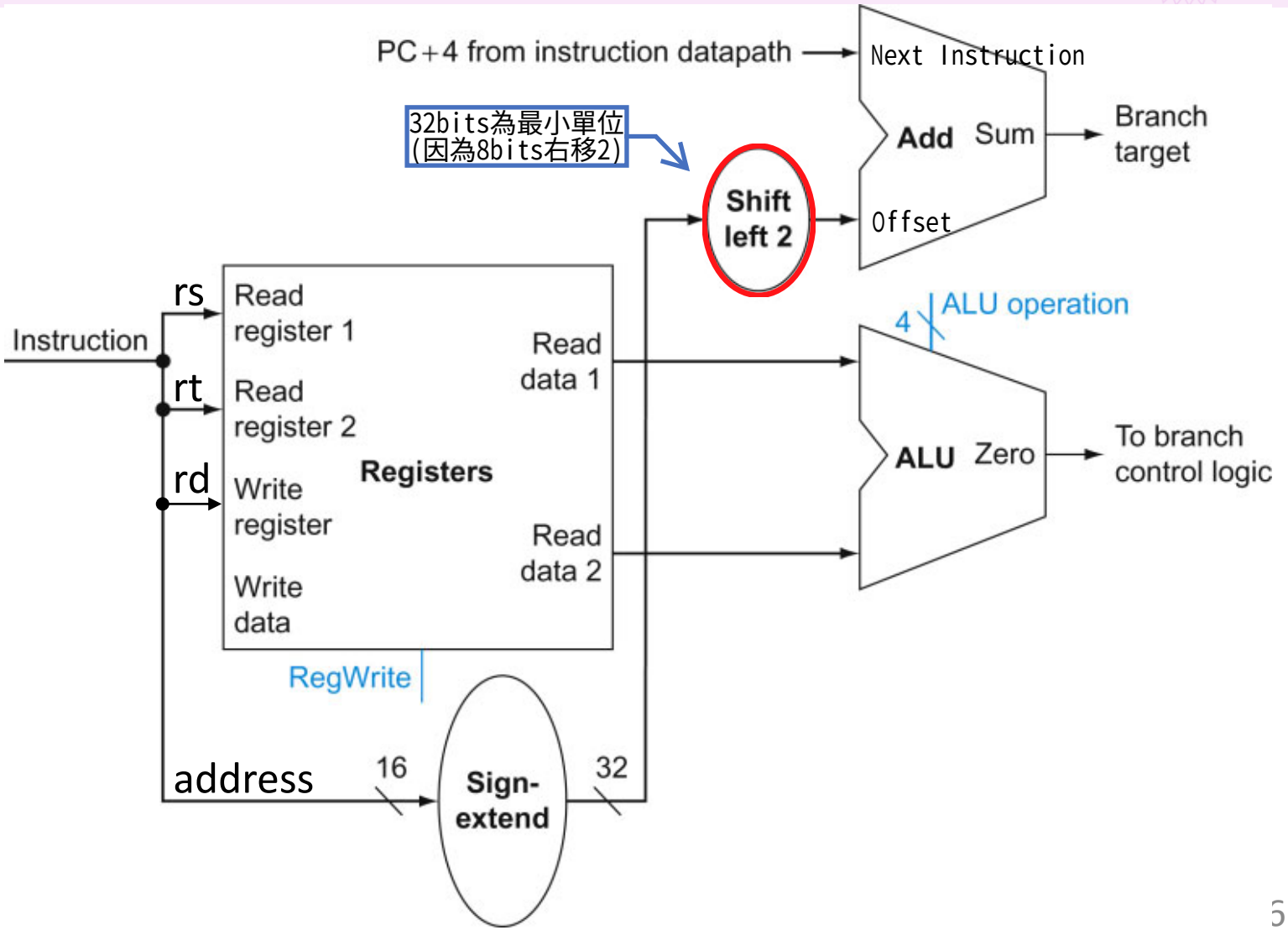
Instruction Fetching Hardware

	31:26	25:21	20:16	15:11	10:6	5:0
R-type	0	rs	rt	rd	shamt	funct
Load/Store	35 or 43	rs	rt	address		
Branch	4	rs	rt	address		
Jump	2	address_2				

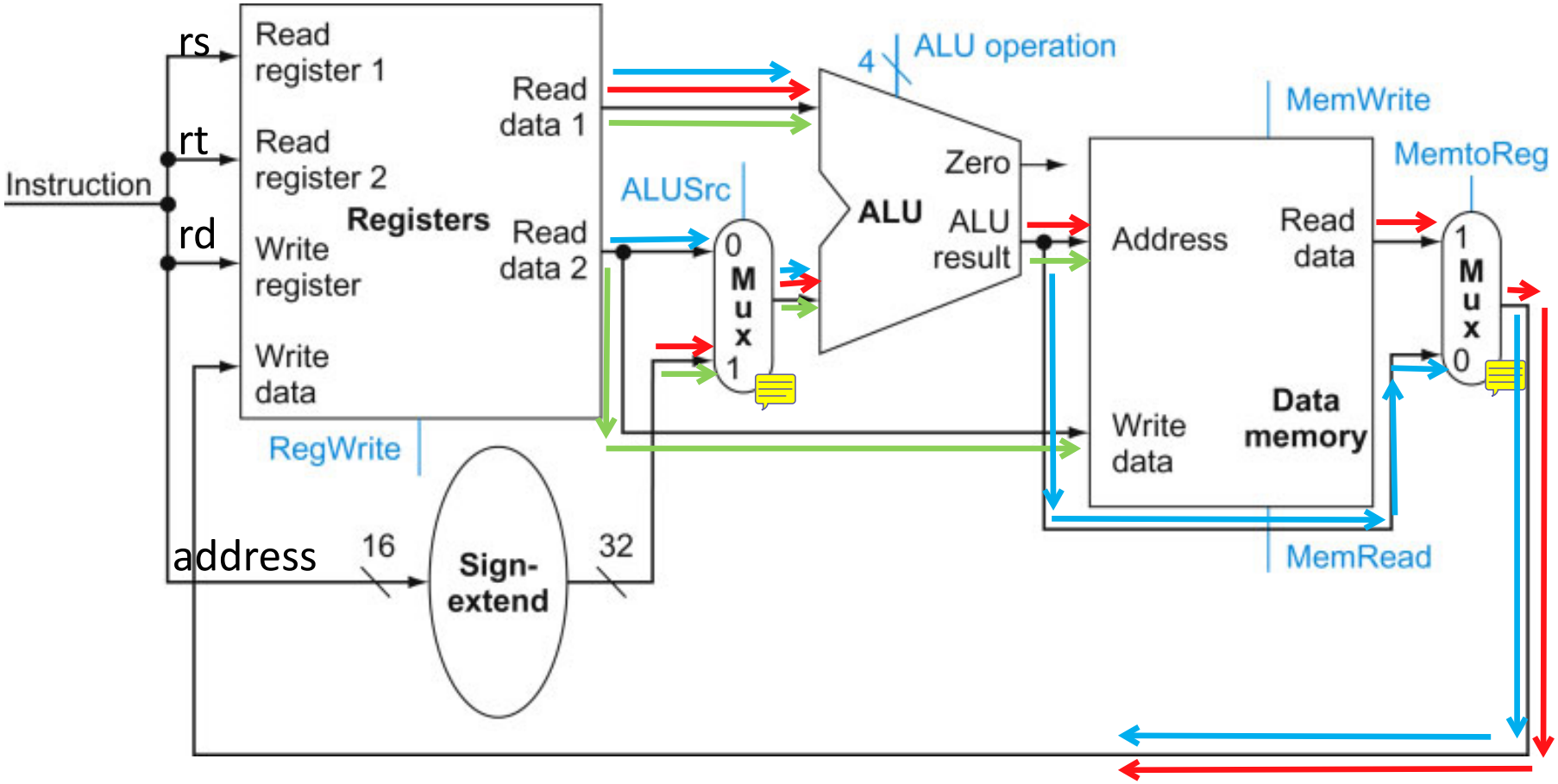


- All these fields are generated regardless of instruction types
- Depending on the op field, some fields turn out to be meaningless and are not actually used

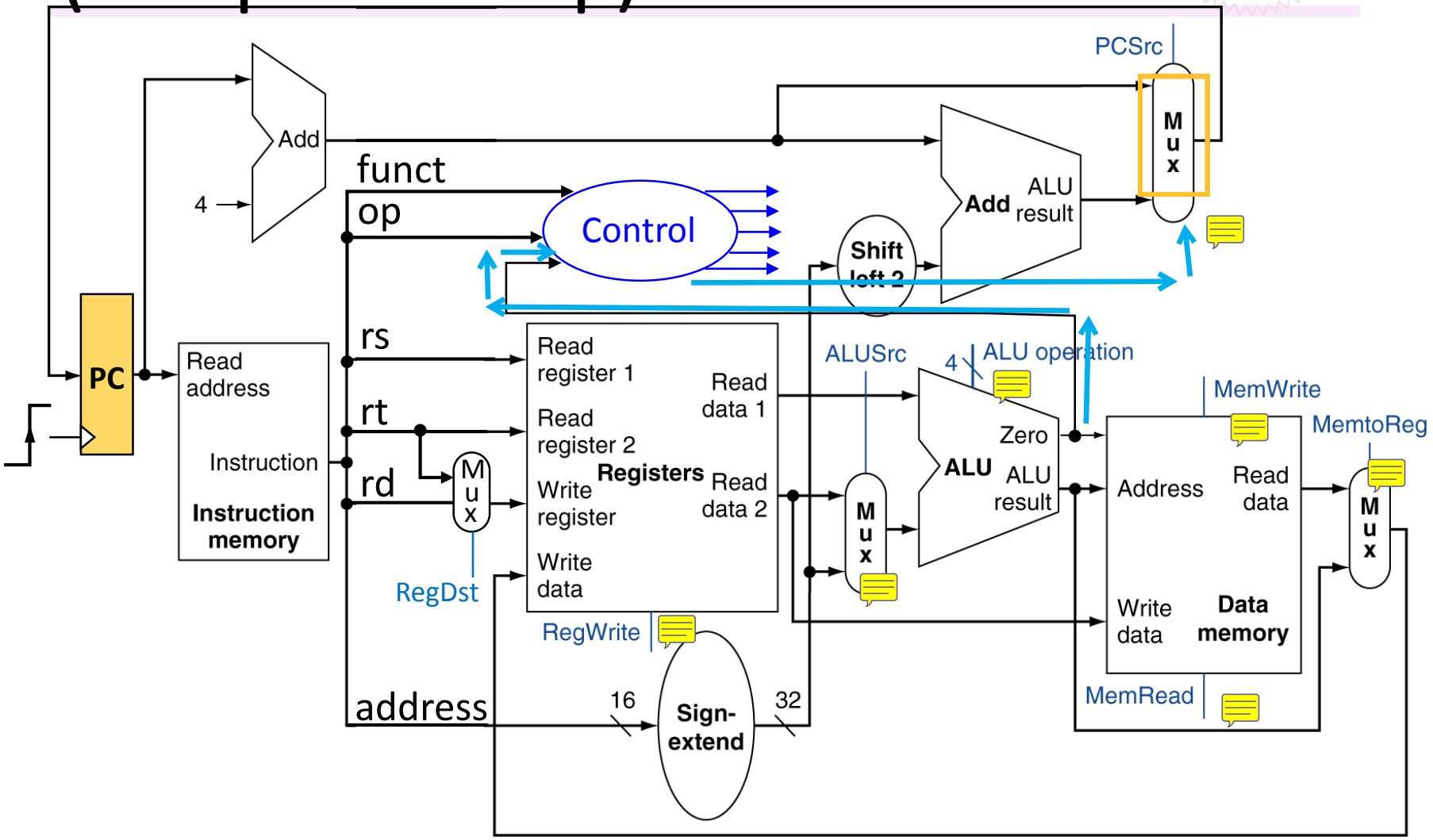
Branch Related Hardware



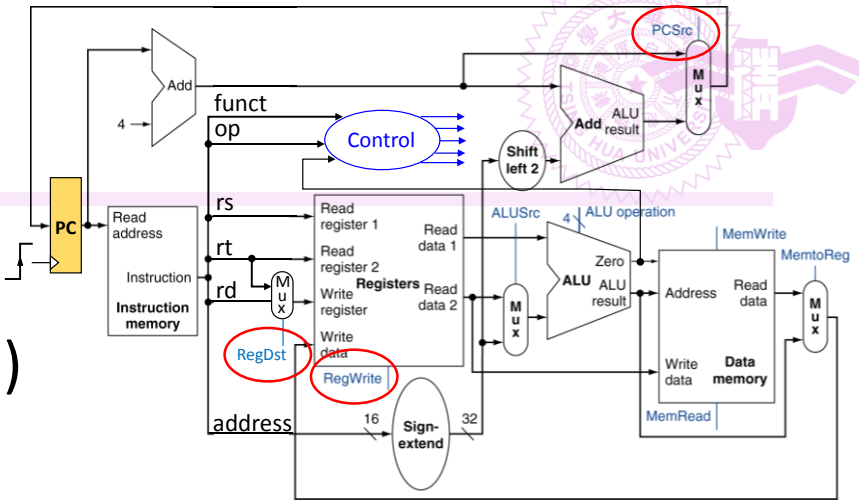
R-Type and Load/Store Related Hardware



Datapath with Control (Except for Jump)

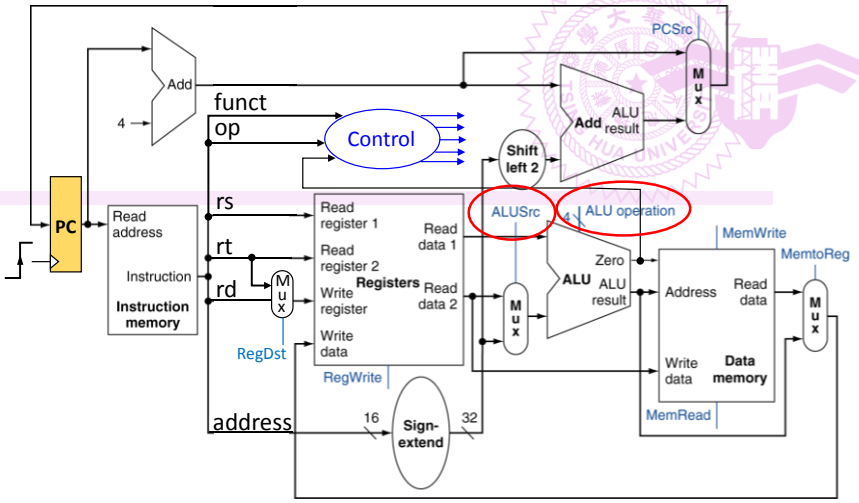


Control



- PCSrc
  - 1: if (op==beq) && (zero==1)
  - 0: otherwise
- RegDst
  - 0: if (op==sw) // rt is the write destination
  - 1: otherwise
- RegWrite
  - 1: if (op==sw) || (op==R-type)
  - 0: otherwise

Control (Cont'd)

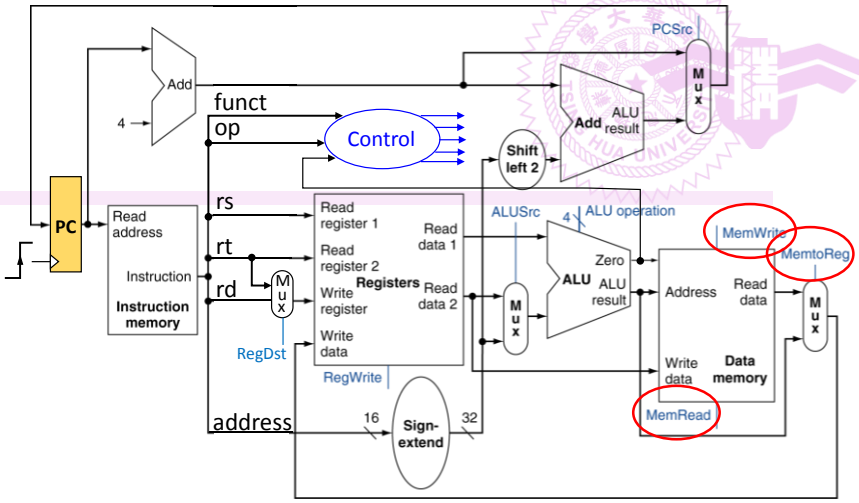


- ALUSrc
  - 1: if (op==lw || op==sw)
  - 0: otherwise
- ALU operation

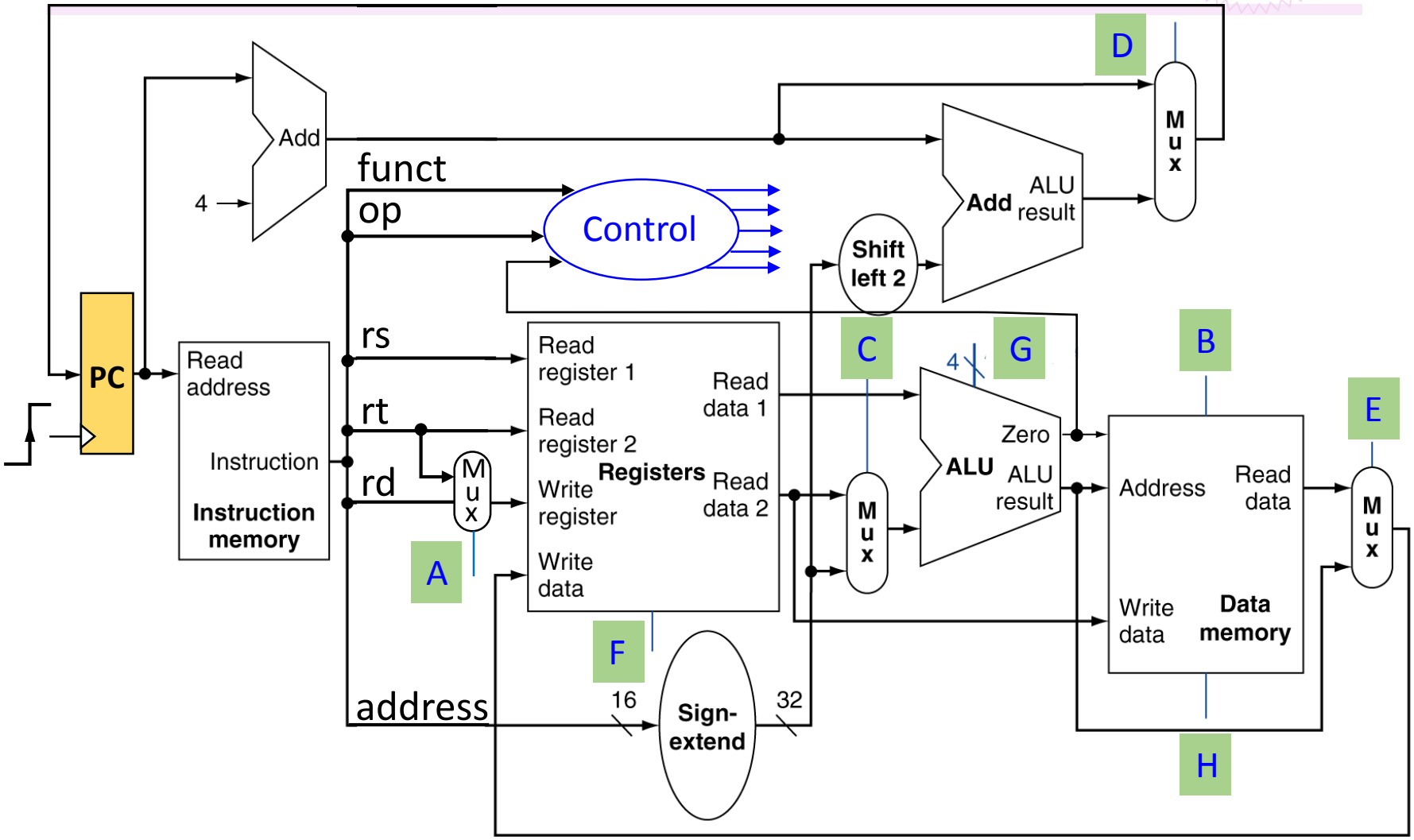
op	funct	Operation	ALU function
lw	--	load word	add
sw	--	store word	add
beq	--	branch equal	subtract
R-type	100000	add	add
	100010	subtract	subtract
	100100	AND	AND
	100101	OR	OR
	101010	set-on-less-than	set-on-less-than

Control (Cont'd)

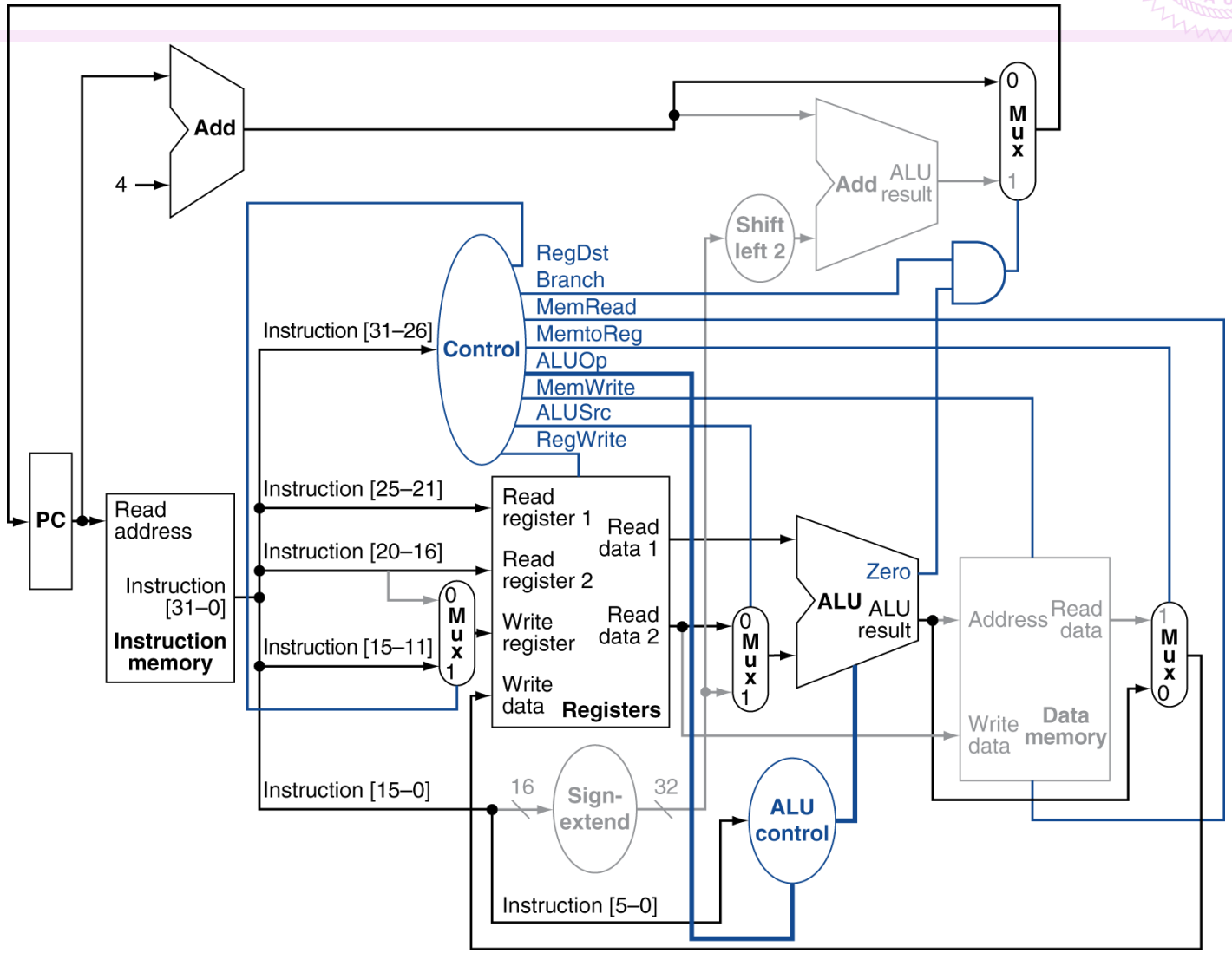
- MemtoReg
  - 1: if (op==lw)
  - 0: otherwise
- MemWrite
  - 1: if (op==sw)
  - 0: otherwise
- MemRead
  - 1: if (op==lw)
  - 0: otherwise



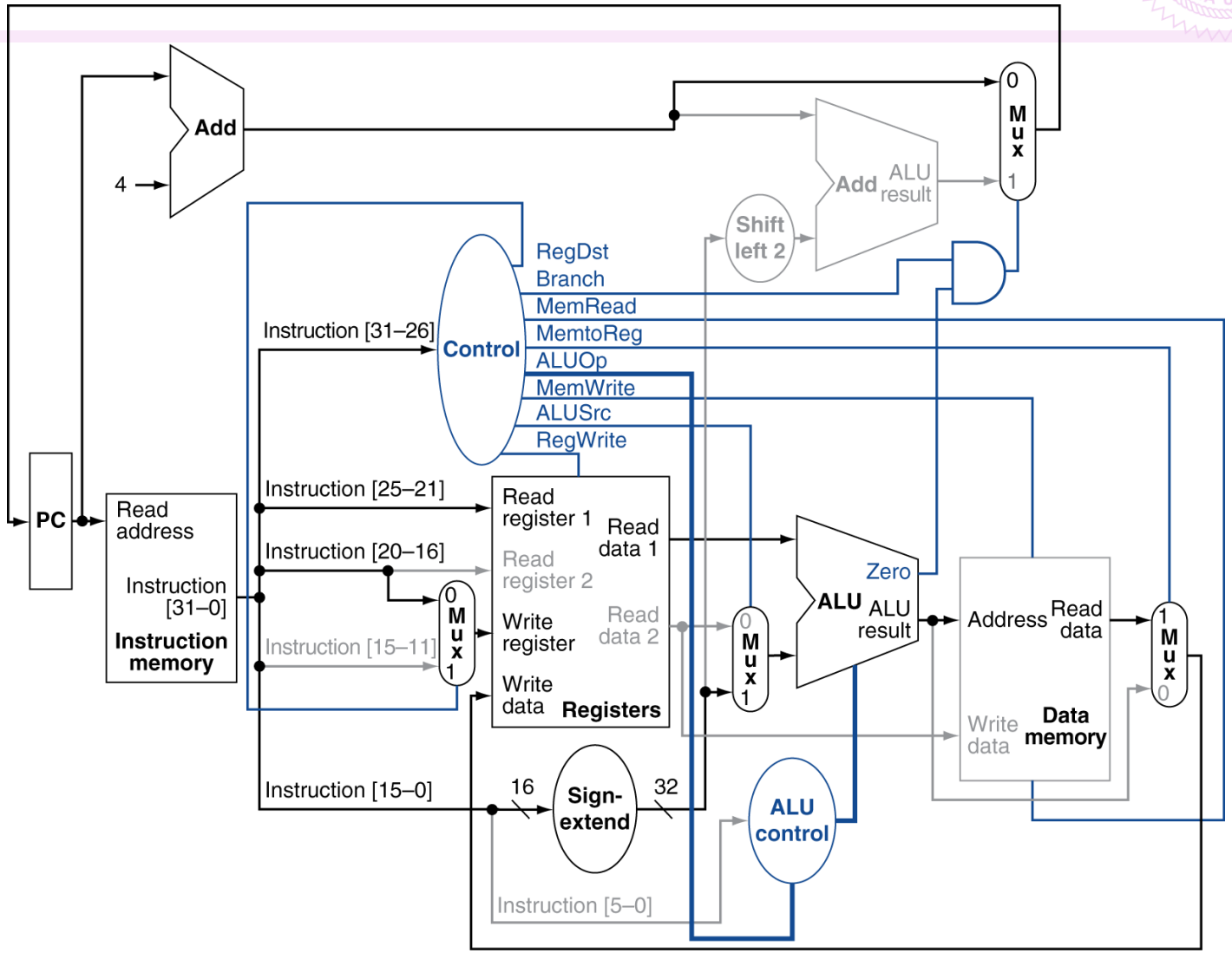
Recap


Add
4
funct
op
Control
Add
ALU result
Shift left 2
M u x
D
PC
Read address
Instruction
Instruction memory
rs
rt
rd
M u x
A
Read register 1
Read register 2
Registers
Write register
Write data
Read data 1
Read data 2
C
4
G
B
E
M u x
Zero
ALU
ALU result
Address
Read data
Write data
Data memory
M u x
F
address
16
Sign-extend
32
H

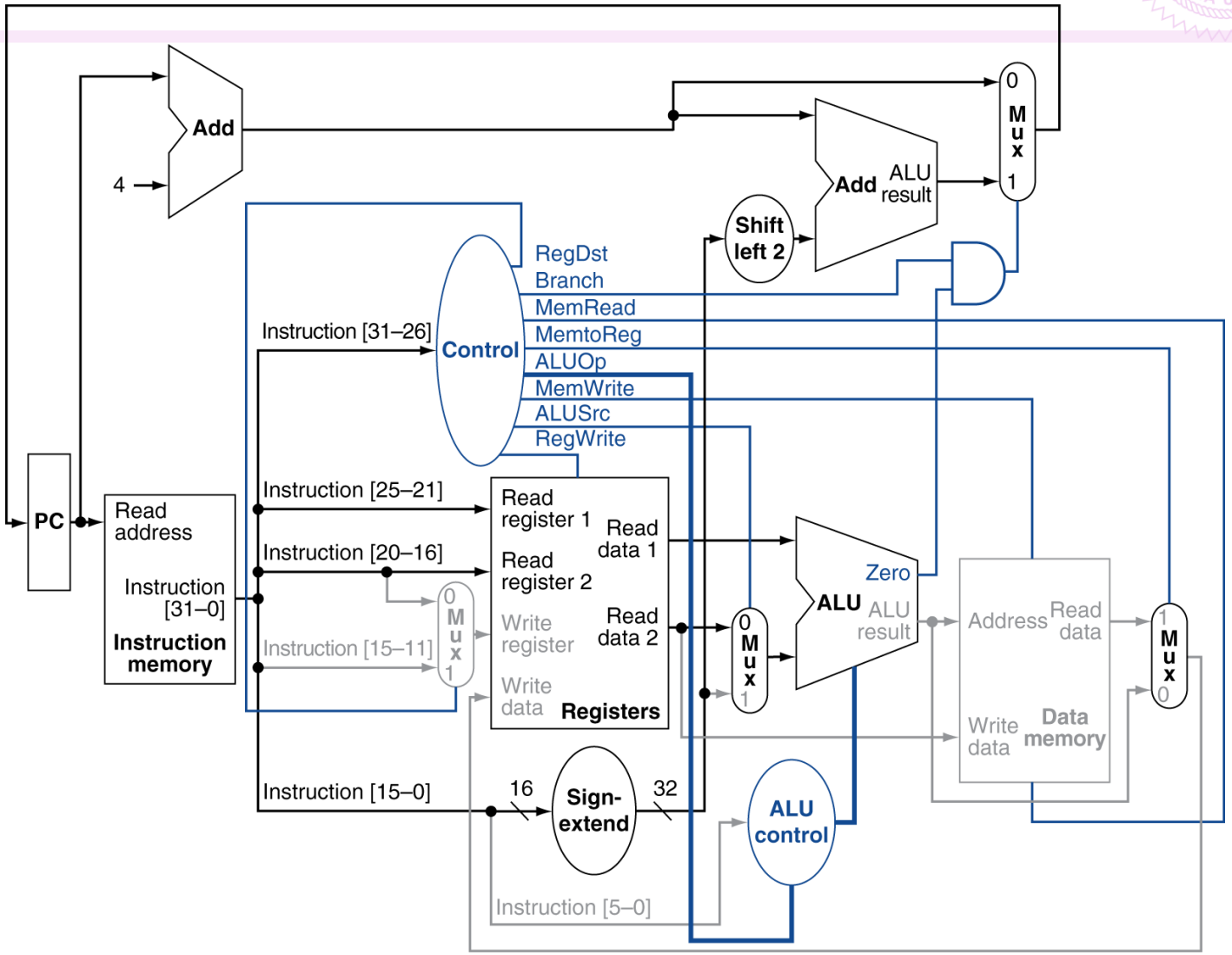
Executing R-Type Instructions



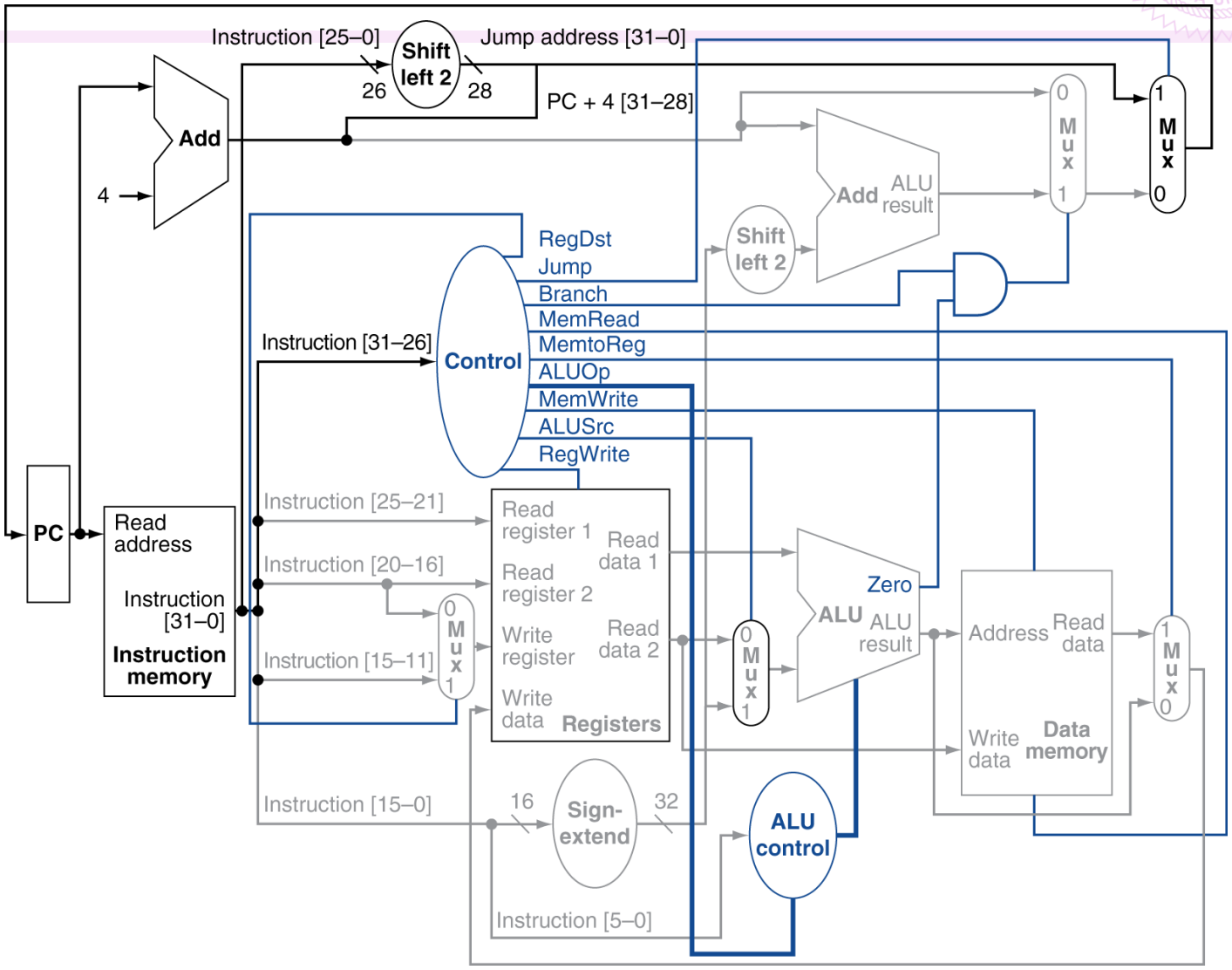
Executing a Load Instruction



Executing the BEQ Instruction

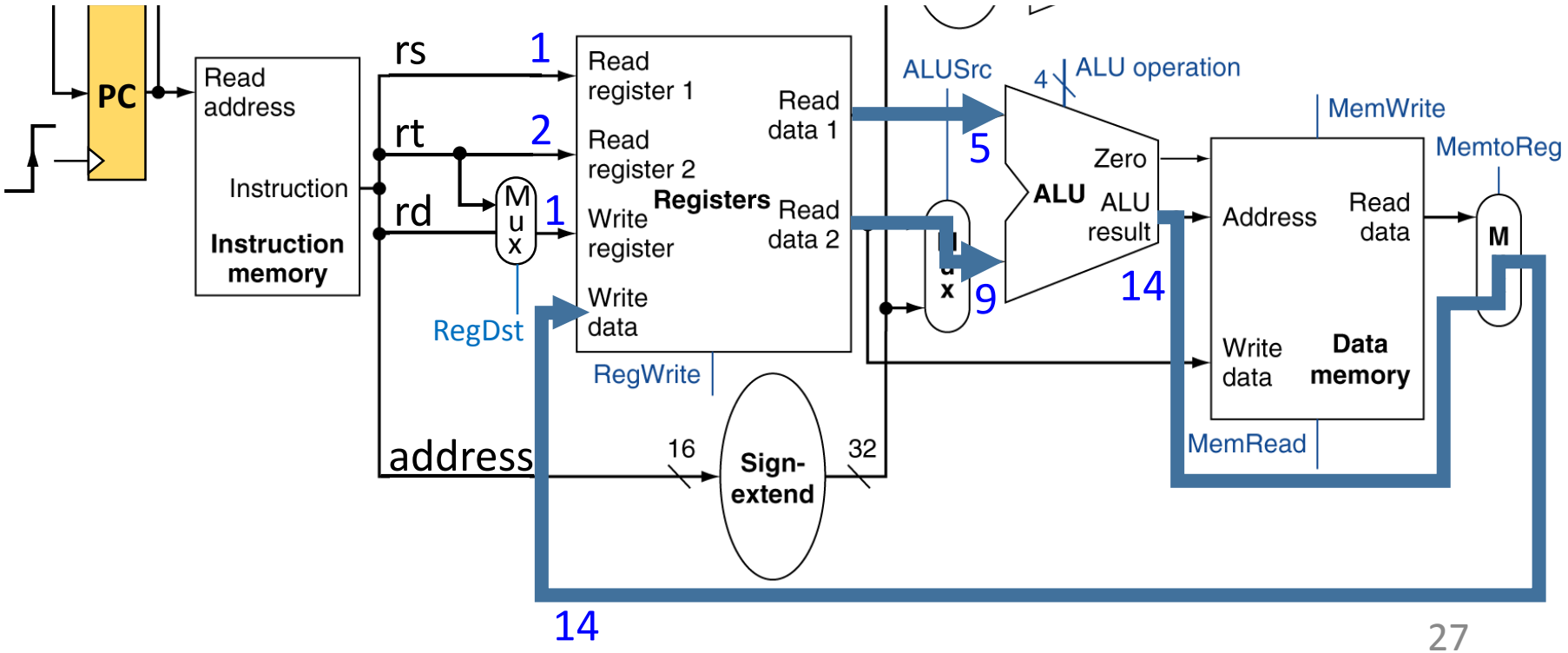


Full Datapath with Jump Added



Correctness Concern

- add $1, $1, $2 # $1 = $1 + $2
  - Suppose $1 = 5 and $2 = 9
- Is this situation a concern?



Performance Concerns

花費時間最長的Path

- Critical path determines clock period
  - Load instruction corresponds to the critical path
    - Instruction memory → register file → ALU → data memory → register file
  - Setting different clock periods for different instructions is not a feasible solution
  - Violates the "making the common case fast" principle
- We will improve performance by pipelining

Outline

- Background
- Single-cycle design
- Pipelined design