

Computer Architecture

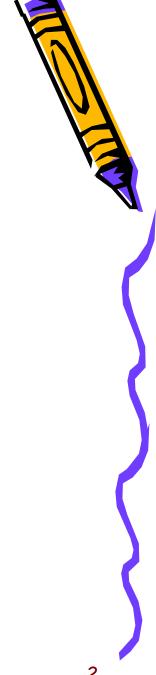
Hossein Asadi
Department of Computer Engineering
Sharif University of Technology
asadi@sharif.edu



Lecture 5

Today's Topics

Data Path Design





Copyright Notice

- · Parts (text & figures) of this lecture adopted from:
 - Computer Organization & Design, The Hardware/Software Interface, 3rd Edition, by D. Patterson and J. Hennessey, MK publishing, 2005.
 - "Intro to Computer Architecture" handouts, by Prof. Hoe, CMU, Spring 2009.
 - "Computer Architecture & Engineering" handouts, by Prof. Kubiatowicz, UC Berkeley, Spring 2004.
 - "Intro to Computer Architecture" handouts, by Prof. Hoe, UWisc, Fall 2009.
 - "Computer Arch I" handouts, by Prof. Garzarán, UIUC, Spring 2009.

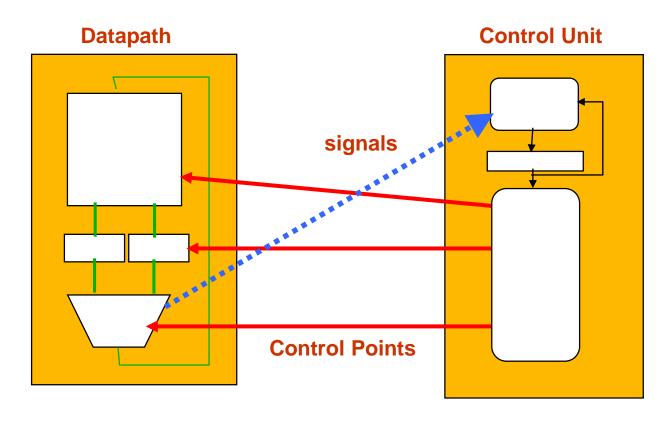


Datapath

- Datapath?
 - A functional unit used to operate on or hold data within a processors
- Datapath Elements in MIPS
 - Instruction memory
 - Data memory
 - Register file
 - ALU
 - Adders
- · Control Unit
 - Schedule data movements in datapath

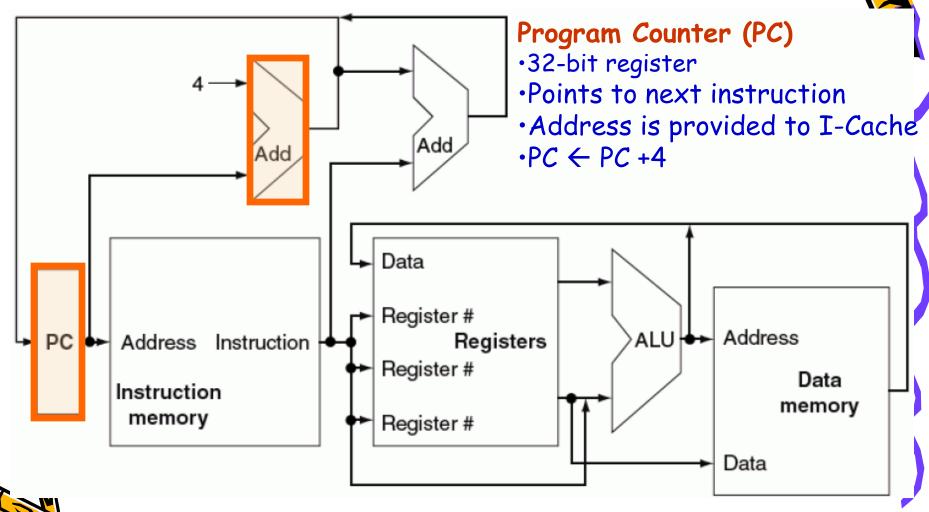


Datapath (cont.)



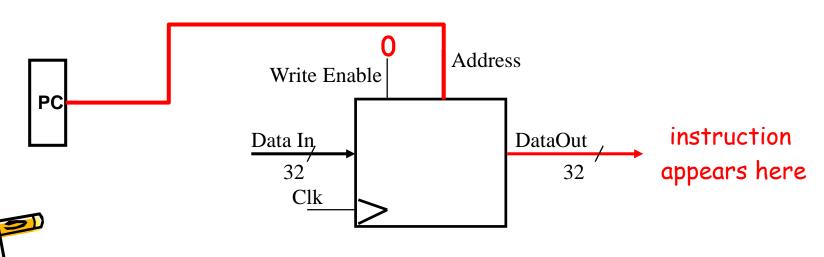


Abstract View of MIPS Implementation

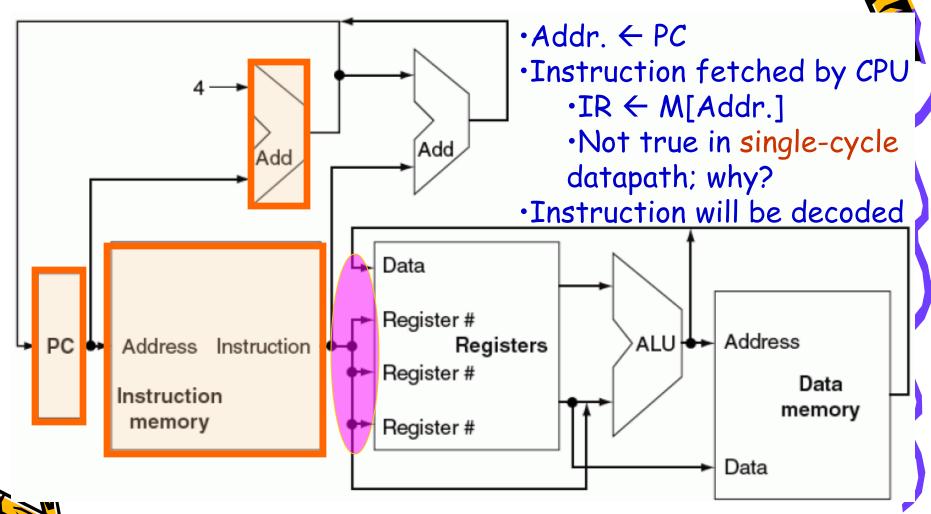


Instruction Fetch

- Program Counter (PC)
 - Supplies instruction address
 - Get instructions from memory

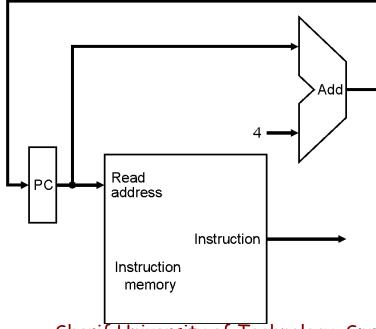


Instruction Memory (I-Cache)



Instruction Fetch Unit

- Updating PC for Next Instruction
 - Sequential Code: PC ← PC + 4
 - Branch and Jump: PC ← New Address
 - we'll worry about this later





Sharif University of Technology, Spring 2019

Instruction Encoding

MIPS Instruction Format

	6 bits	5 bits	5 bits	5 bits	5 bits	6 bits
r format	OP	rs	rt	rd	sa	funct
i format [OP	rs	rt	imm	ediate	
j format [OP		targe	et		

- · OP: opcode
- rs: first register source operand
- rt: second register source operand
- · rd: register destination operand
- sa: shift amount

funct: function code







- sub, and, or, slt 6 bits 5 bits 5 bits 5 bits 6 bits

· LOAD / STORE

- lw rt, rs, imm
- sw rt, rs, imm

0		16	21	26	<u>31</u>
	immediate	rt	rs	op	
	16 bits	5 bits	5 bits	6 bits	

· BRANCH

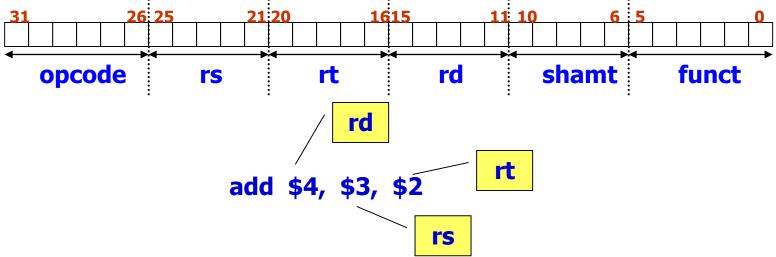
beq rs, rt, imm

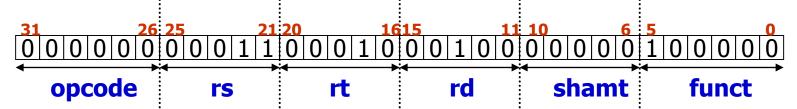
81 26 21 16 op rs rt

oprsrtdisplacement6 bits5 bits5 bits16 bits



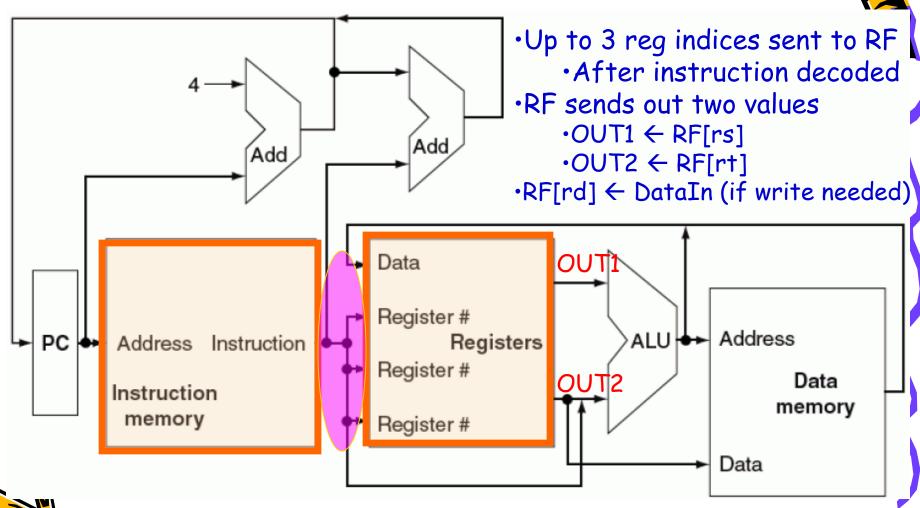






Encoding = 0x00622020

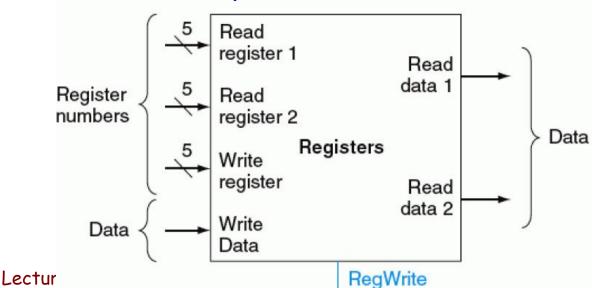
Register File



Lecture 5

Datapath Elements

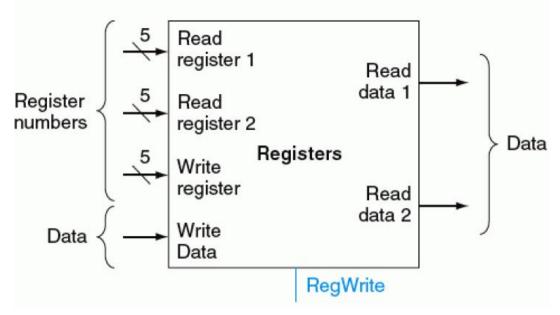
- Register File (RF)
 - Also called, General Purpose Registers (GPR)
 - Up to three indices as inputs
 - 32-bit write input data
 - Two 32-bit outputs





Datapath Elements (cont.)

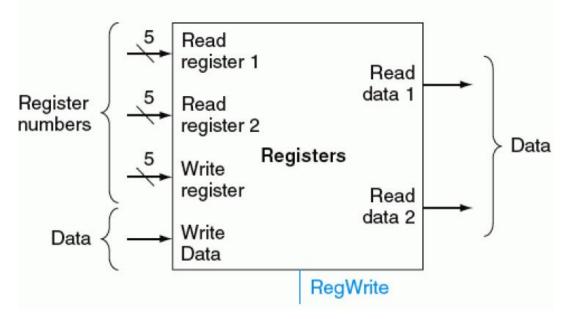
- Question 1:
 - How read & write can be accomplished in one clock cycle without data contention?





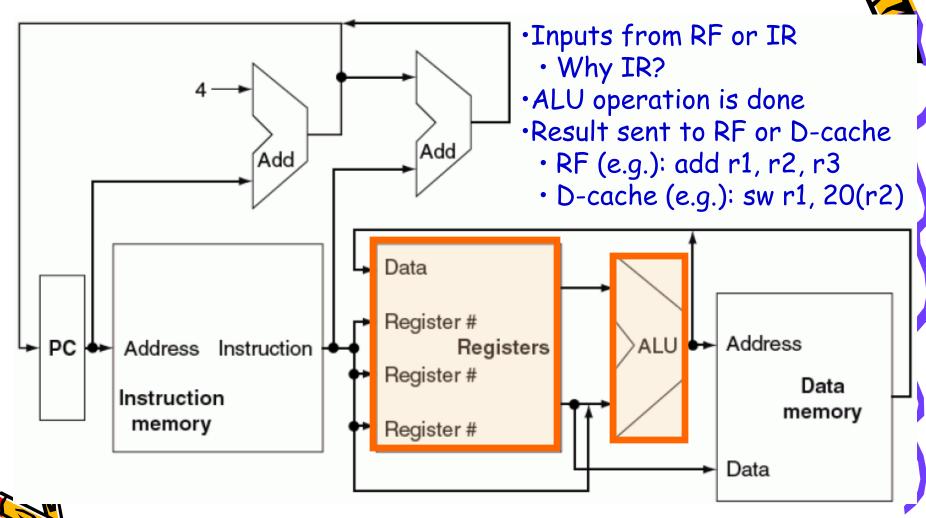
Datapath Elements (cont.)

- Question 2:
 - How two simultaneous read operations possible?





ALU



Reminder: Addressing Modes

Operand is constant

op rs rt Immediate

Register

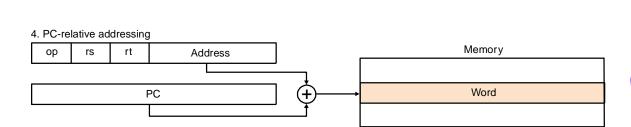
2. Register addressing

3. Base addressing

Operand is in register

lb \$t0, 48(\$s0)

bne \$4, \$5, Label (label will be assembled into a distance)



funct

Address

Registers

Register

Memory

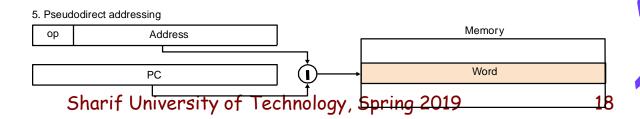
Word

Halfword



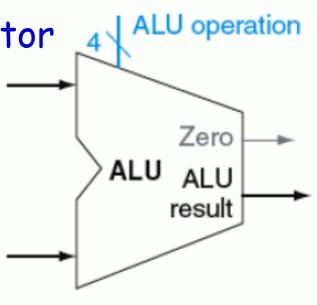
j Label

Lecture 5



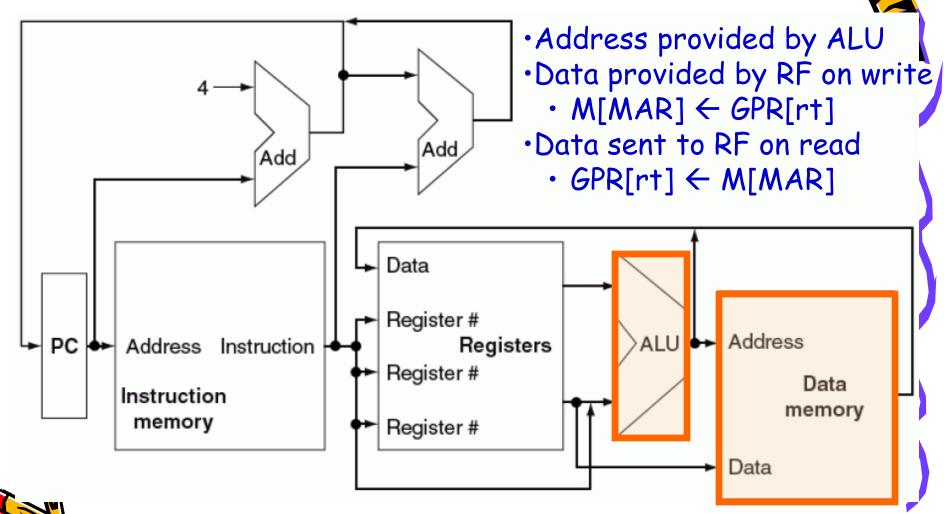
Datapath Elements (cont.)

- · ALU
 - Two 32-bit inputs
 - One 32-bit output
 - 4-bit operation selector
 - Zero?





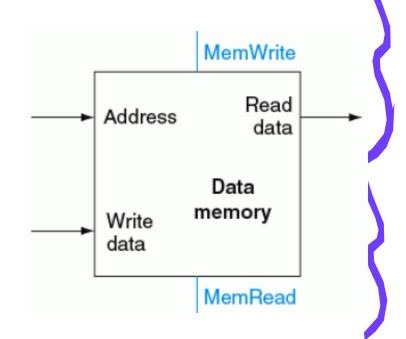
Data Memory (D-Cache)



Lecture 5

Datapath Elements (cont.)

- Data Memory Unit
 - Address
 - Loads/store: MAR = GPR[rs] + imm16
 - 32-bit write data
 - Write data ← GPR[rt]
 - 32-bit read data
 - GPR[rt] ← Read data
 - MemRead signal
 - MemWrite signal



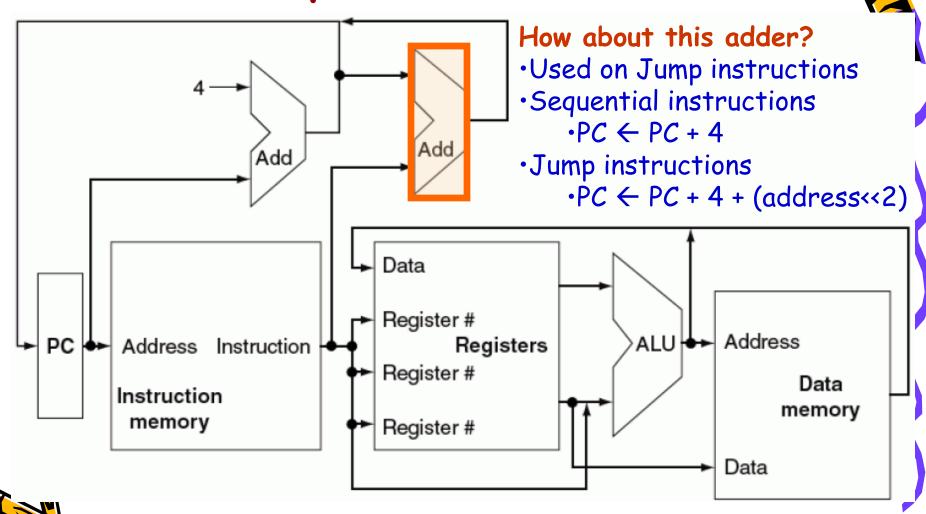


Discussion

- Compare MemRead and MemWrite Activation Process in Following Cases
 - Case A: separate I-cache & D-cache with separate data bus
 - Case B: separate I-cache & D-cache with common data bus
 - Case C: unified I-cache & D-cache

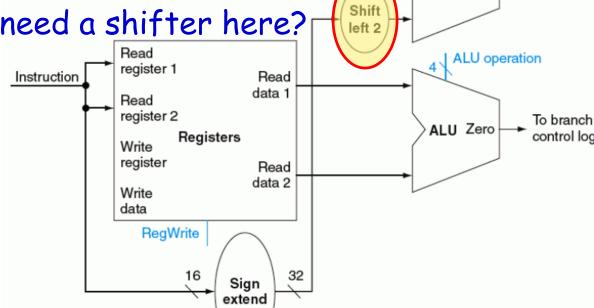


Abstract View of MIPS Implementation



Datapath Elements (cont.)

- Branch Unit
 - Sign extend unit
 - Adder
 - Shifter
 - Do we need a shifter here?



PC+4 from instruction datapath -



Lecture 5

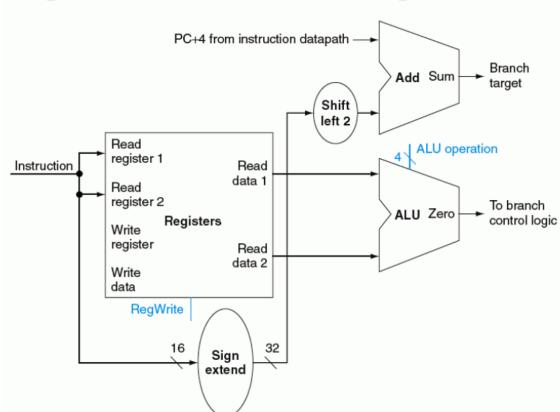
Branch

target

Add Sum

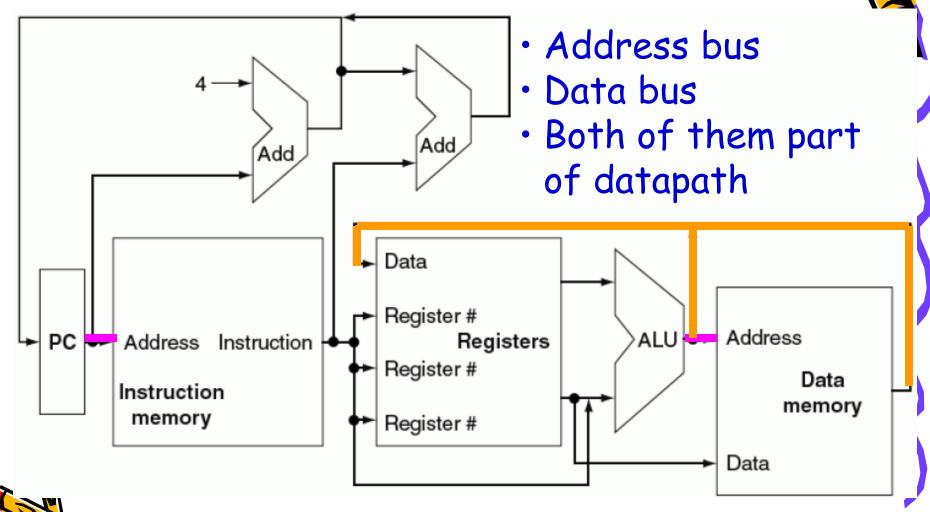
Practice

- In Following Circuit:
 - Draw sign-extend & shift logic



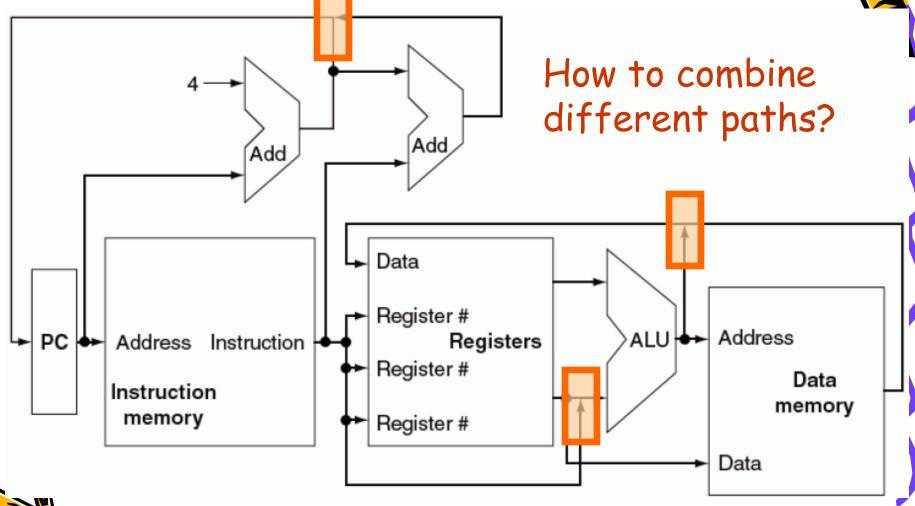


Address Bus & Data Bus



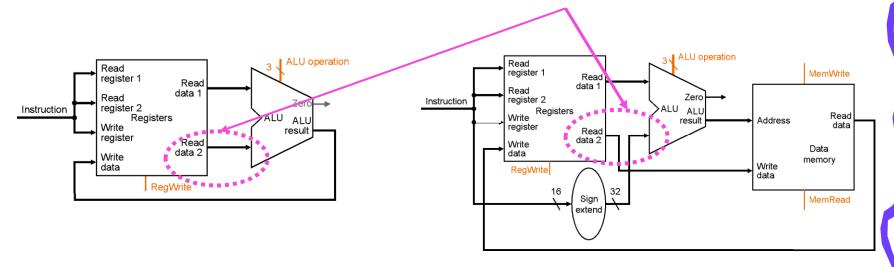
Lecture 5

Combining Datapaths



Combining Datapaths (cont.)

 How to have different datapaths for different instruction?





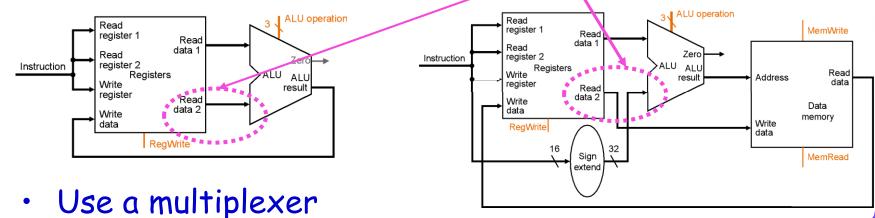
Store

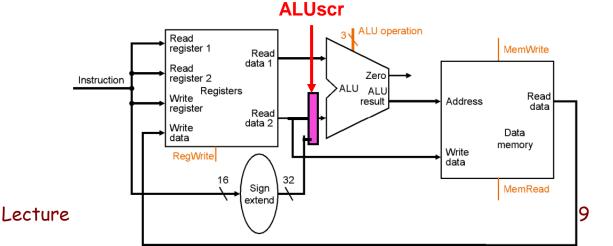


Combining Datapaths (cont.)

· How to have different datapaths for different

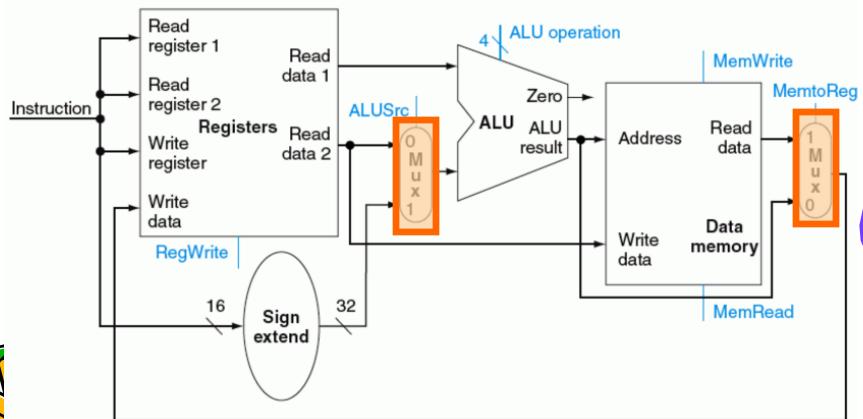
instruction?





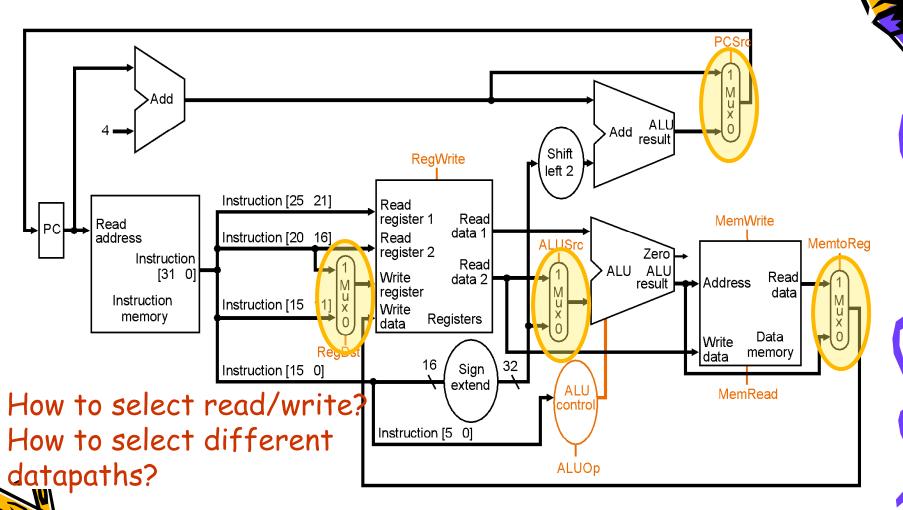
Combining Datapaths (cont.)

 Merging R-type datapath with load/store datapath using multiplexer





All Together: Single Cycle Datapath



Practice

- · Question:
 - Could we swap rs, rt, & rd bits to have easier datapath design?
 - In other words, can we remove RegDst MUX?

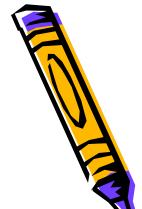


Practice: Answer

- R-type
 - rs & rt: read index bits
 - rd: write index bits
- lw:
 - rs: read index bits
 - rt: write index bits
- SW:
 - rs & rt: read index bits



Reminder: Instruction Encoding



- R-TYPE
 - add rd, rs, rt

	31 20	21	10	11	0	
	op	rs	rt	rd	shamt	funct
10.						

5 bits

- sub, and, or, slt
- 6 bits
- 5 bits

- 5 bits
- 5 bits
- 6 bits

- LOAD / STORE
 - lw rt, rs, imm
 - sw rt, rs, imm

0		. 16	21	<u>31 26</u>	<u>31</u>
diate	immedia	rt	rs	op	
bits	16 bit	5 bits	5 bits	6 bits	

BRANCH



- **26** rs
 - 21
- **16** rt



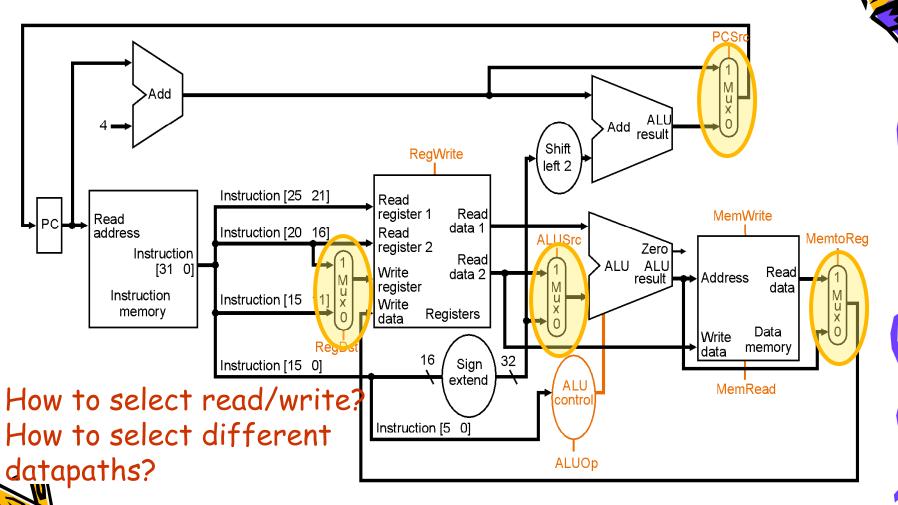
16 bits

- 6 bits

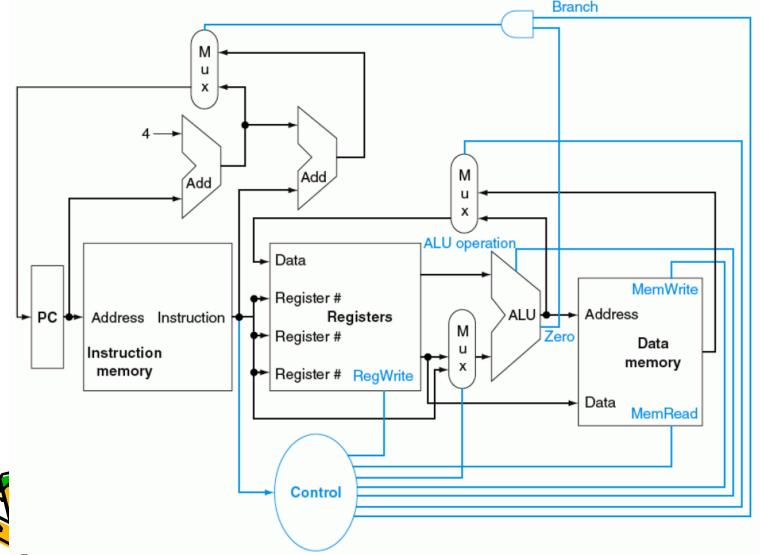
5 bits

displacement

All Together: Single Cycle Datapath



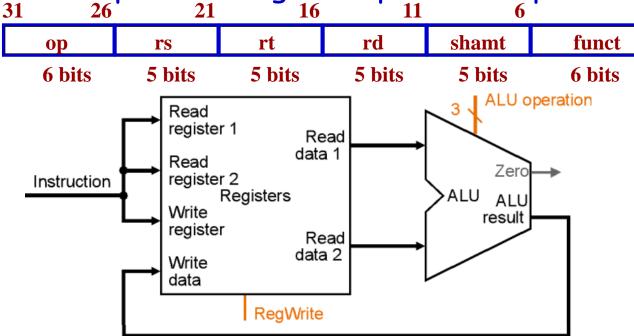
Abstract View of MIPS Implementation: Control





Datapath for Reg-Reg Operations

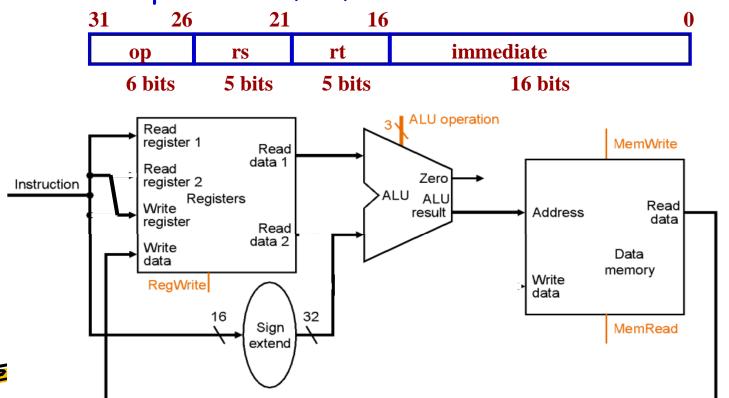
- GPR[rd] ← GPR[rs] op GPR[rt]
 - Example: add rd, rs, rt
 - ALUoperation signal depends on op and funct



ALU operation and RegWrite: control logic after decoding instruction

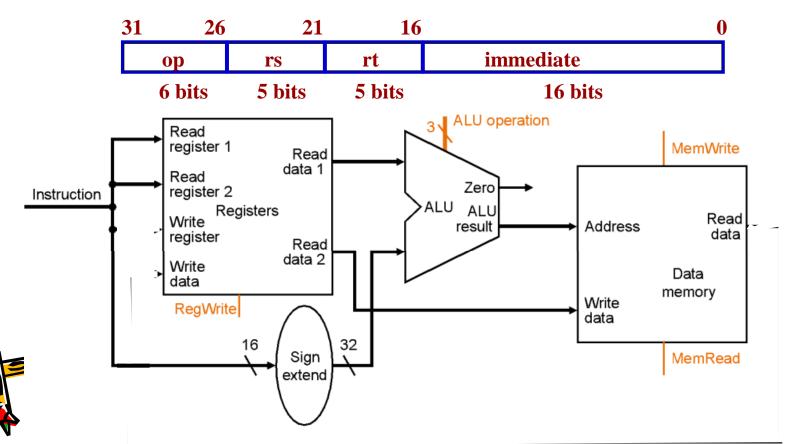
Datapath for Load Operations

- GPR[rt] ← Mem[GPR[rs] + SignExt[imm16]]
 - Example: Iw rt, rs, imm16



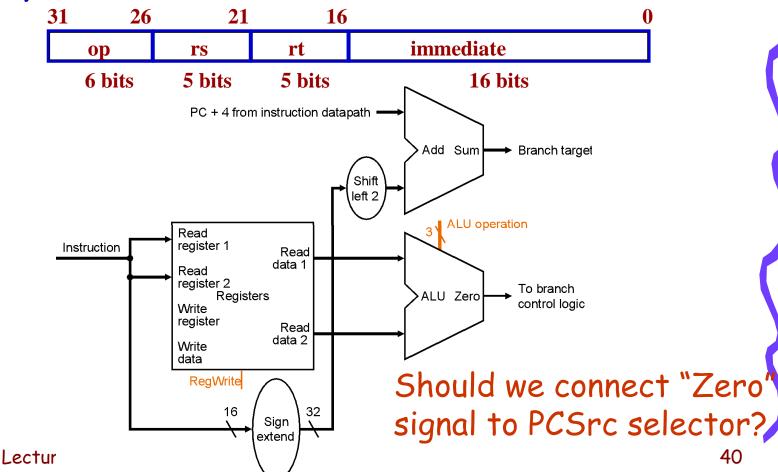
Datapath for Store Operations

- Mem[GPR[rs] + SignExt[imm16]] ← GPR[rt]
 - Example: sw rt, rs, imm16

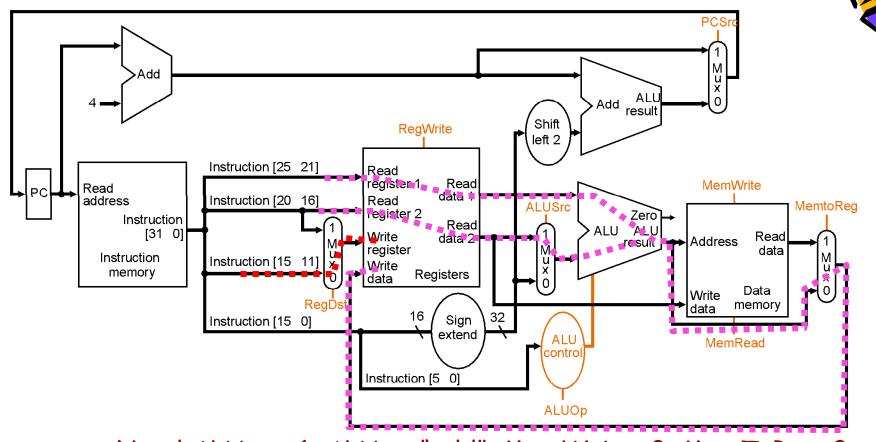


Datapath for Branch Operations

• beq rs, rt, imm16



R-Format Datapath (e.g. add)



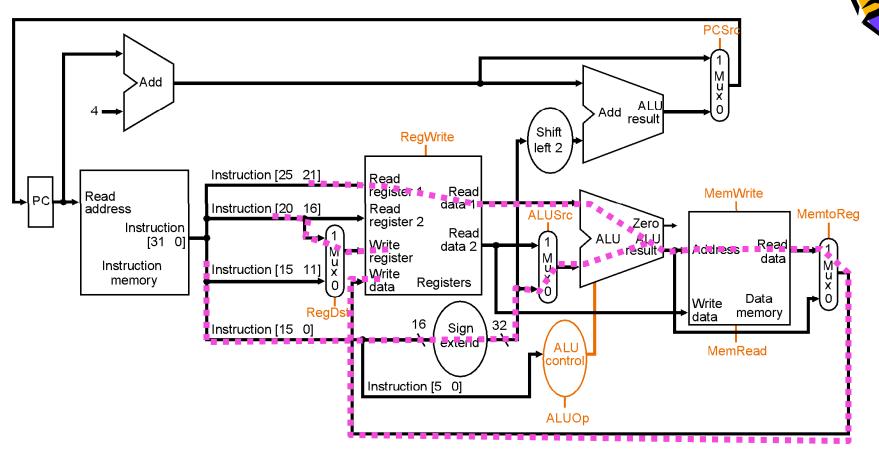


Need ALUsrc=1, ALUop="add", MemWrite=0, MemToReg=0, RegDst = 0, RegWrite=1 and PCsrc=1.

Lecture 5

Sharif University of Technology, Spring 2019

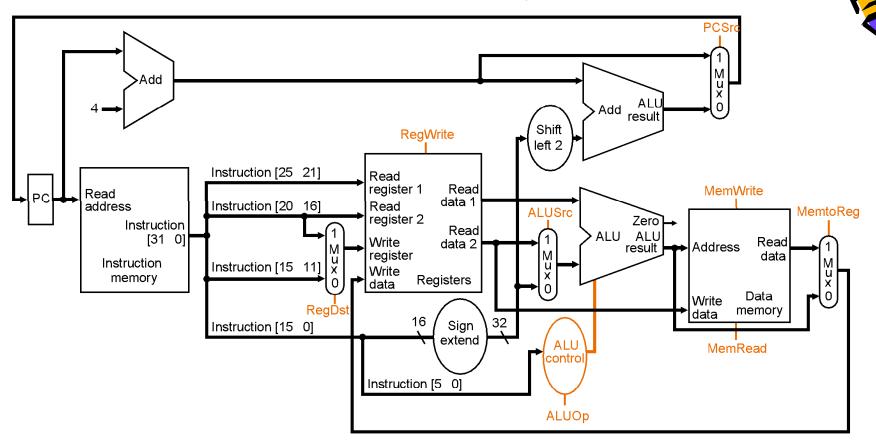
Load Datapath





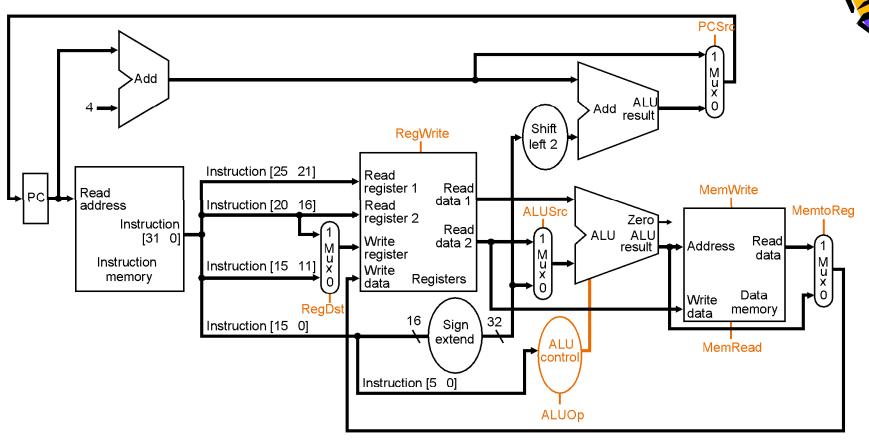
What control signals do we need for load??

Store Datapath



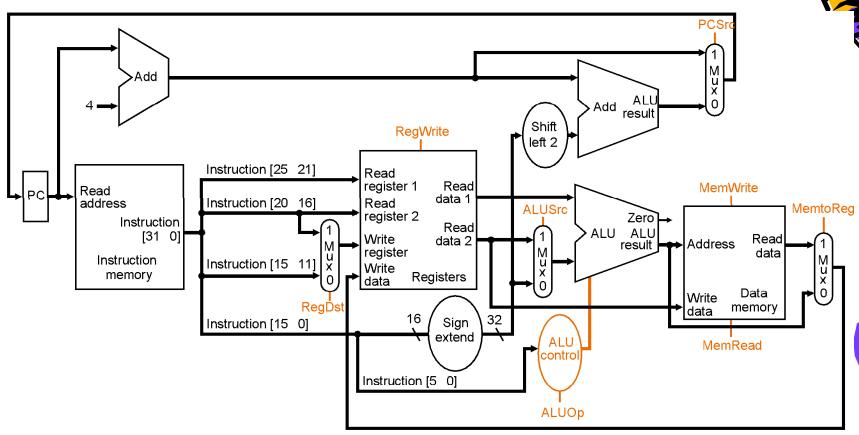


beq Datapath





Putting it All Together



We have everything except details for generating control signals

Backup



