

# Lecture 7: Datapath

COS 471a, COS 471b / ELE 375

Computer Architecture and Organization

Princeton University  
Fall 2004

Prof. David August

1

## The Quiz We Might Have Had

1. Write -2.5 in our IEEE 754 Wimpy Precision (8 bits total, 4-bit exponent, 3-bit mantissa).  
( $-1.01 \times 2^1$  à bias = 7 à 11000010)
2. How is two's-complement subtraction typically implemented in HW?  
(carry\_in<sub>0</sub> = 1, invert subtrahend)
3. Give one reason why floating point is better than fixed point.  
(greater range)

2

## Datapath

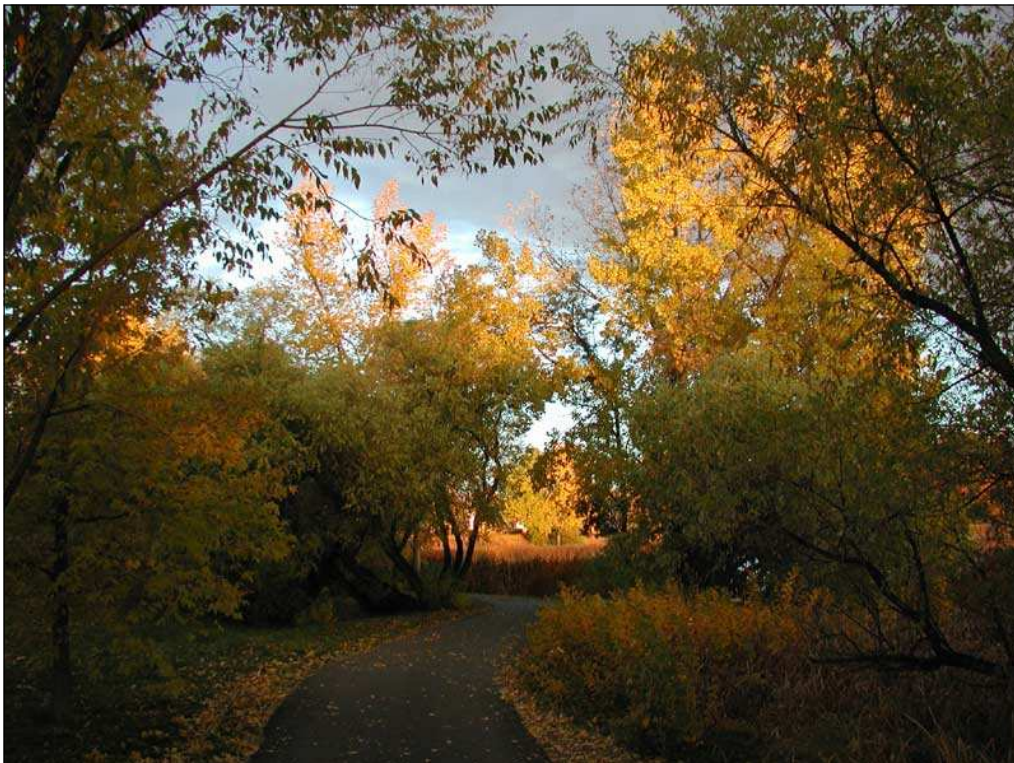
### Datapath

"The component of the processor that performs arithmetic operations" - H&P

### Datapath

The collection of state elements, computation elements, and interconnections that together provide a conduit for the flow and transformation of data in the processor during execution. - DIA

4



## Datapath - Part of the Microarchitecture

### Architecture

- The ISA - the programmer's view of the machine
- Implementation independent, **an interface**



### Microarchitecture

- The lower-level implementation of the ISA
- Design specific, **an implementation**



### Example use of terminology

- Architectural state: **Register r5**
- Microarchitectural state: **Carry bit on the 5<sup>th</sup> 1-bit ALU**

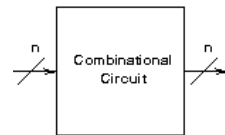
6

## Datapath Elements

- ALUs are just one datapath building block
- What about the other elements?

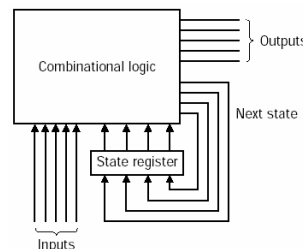
### Computational Elements

- Combination Circuits
- Outputs follow inputs
- Familiar Example: ALU



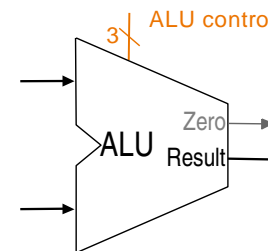
### State Elements

- Sequential Circuits
- Outputs change on clock edge
- Familiar Example: A Register



## Computation Element: ALU

- Combinational - you had better know how to design it by now!!!
- Refine for MIPS
  - Zero equality test on all results - why?
  - Set on less than for `slt` instruction

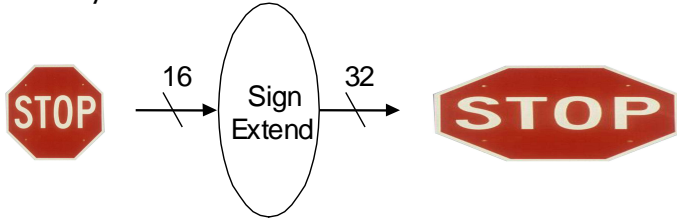


ALU Control	Function
000	AND
001	OR
010	add
110	subtract
111	set on less than

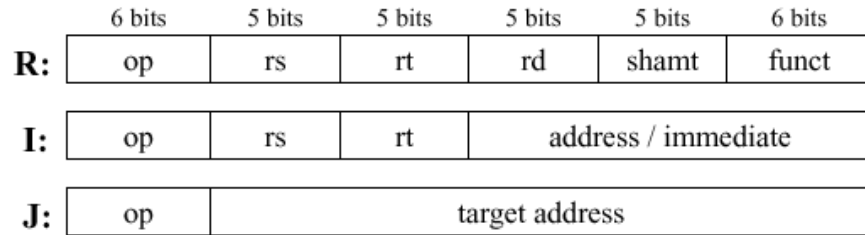
7

## Computation Element: Sign Extender

- 16 à 32 bit Sign extender
- Why is this necessary in MIPS?



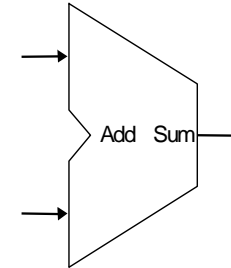
• Hint:



Implementation?

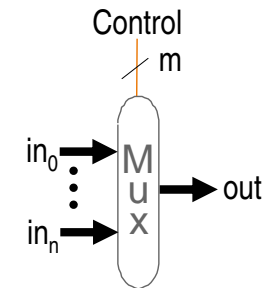
## Computation Element: Adder

- Not an ALU, just add
- Why would we need this in MIPS to execute instructions?

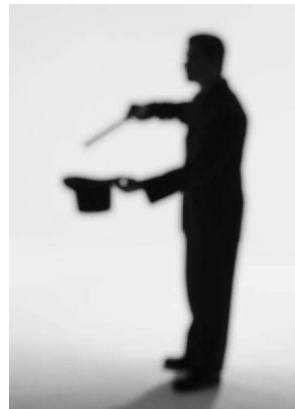


## Computational Element: The Magical Mux

- Mux is short for Multiplexer (Think: selector)
- n input lines (of any common width)
- m control wires to select
- $n = 2^m$

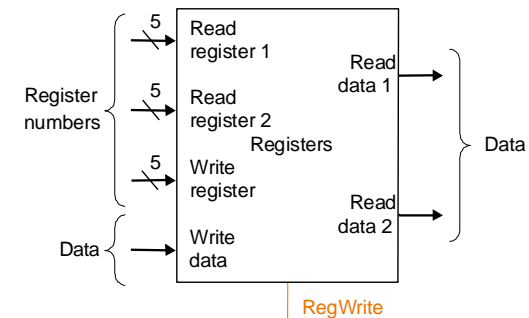


Implementation?



## State Element: Register File

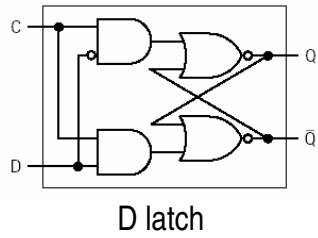
- Microarchitecture to implement architectural state
- Built using D flip-flops
- MIPS:
  - Need to be able to read two operands at once
  - 2 source operands per instruction



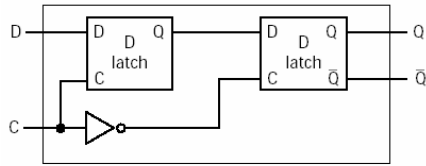
5-bits? 2 Reads? 1 Write?

## State Element: Register File

### Register Implementation



D latch

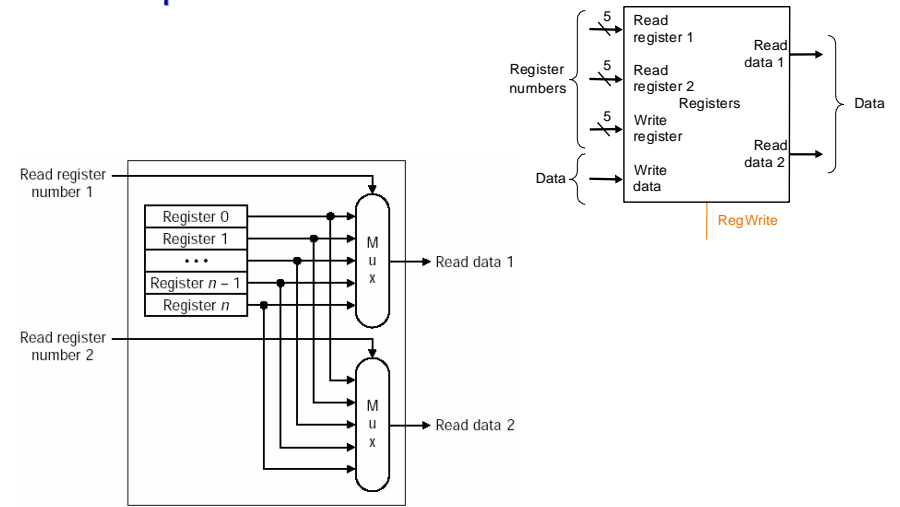


Falling edge triggered D flip-flop

13

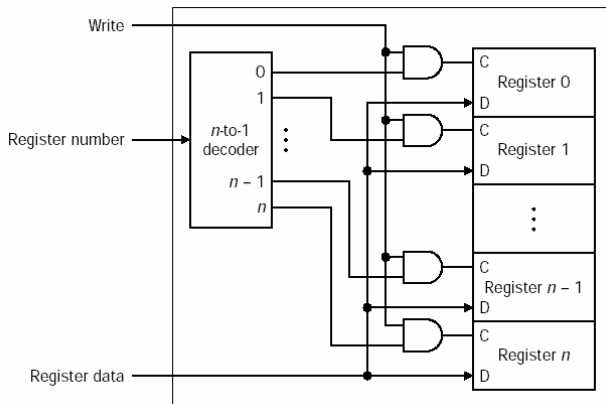
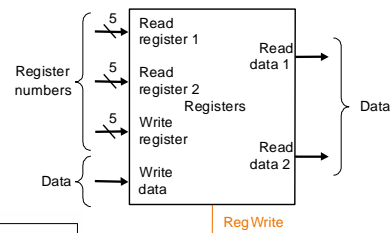
## State Element: Register File

### Read Implementation



## State Element: Register File

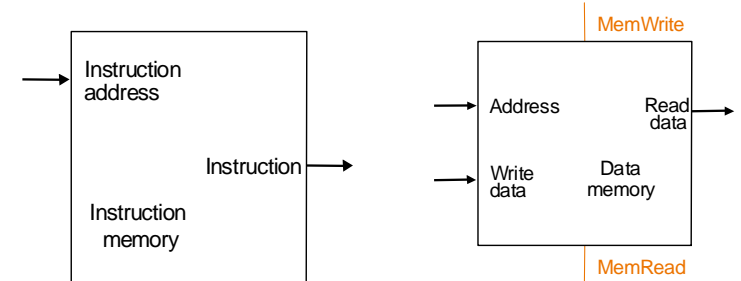
### Write Implementation



Know decoders

## State Element: Data and Instruction Memory

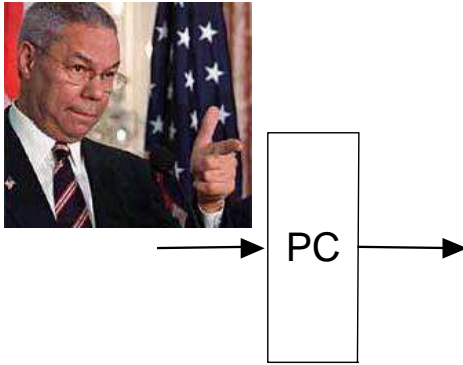
- Microarchitectural element to hold the architectural memory state
- See Appendix B for implementation details



16

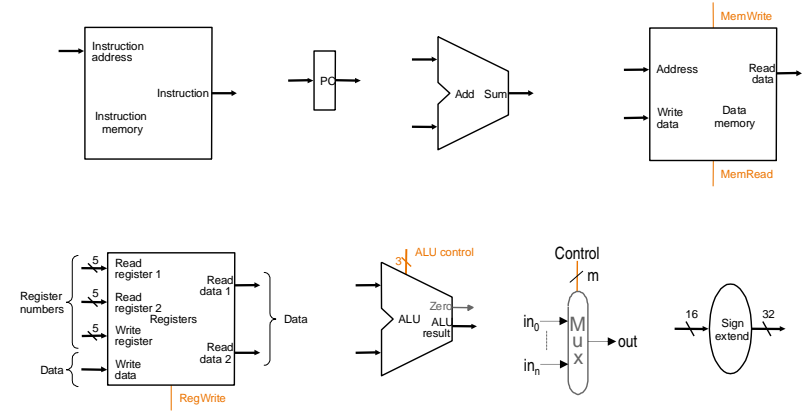
## State Element: The Program Counter

- To hold the architectural PC state
- Just like a single register

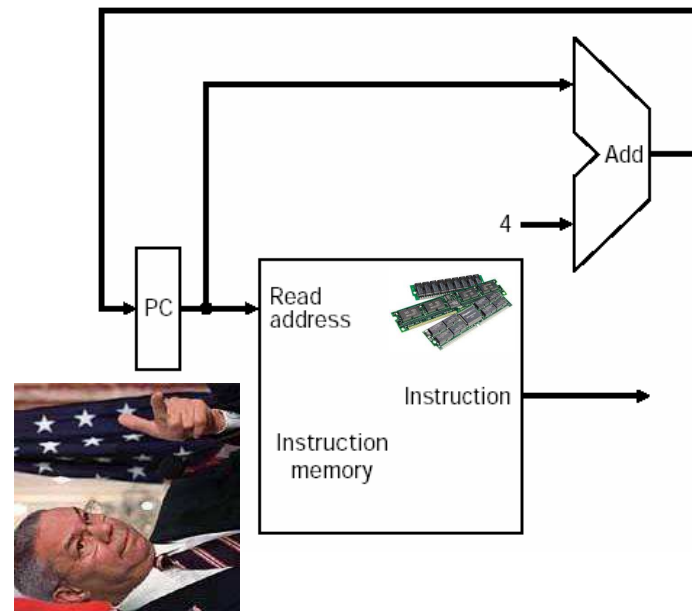


17

## Our Complete Line of Products! There may be others, but this is good for MIPS



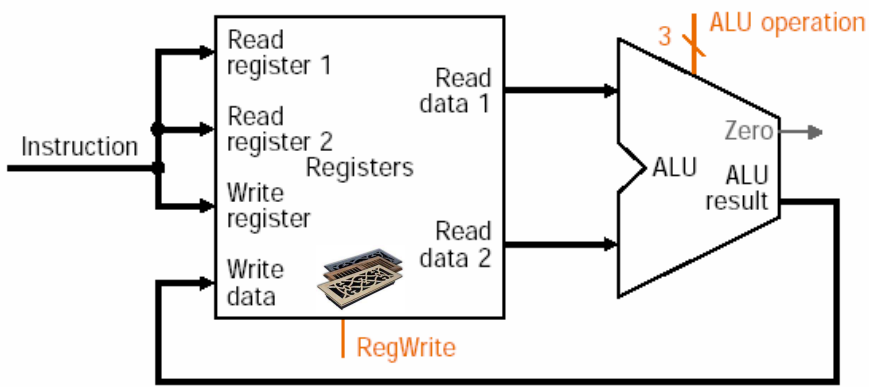
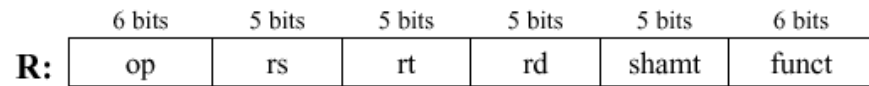
## Fetching Instructions (no branching)



20



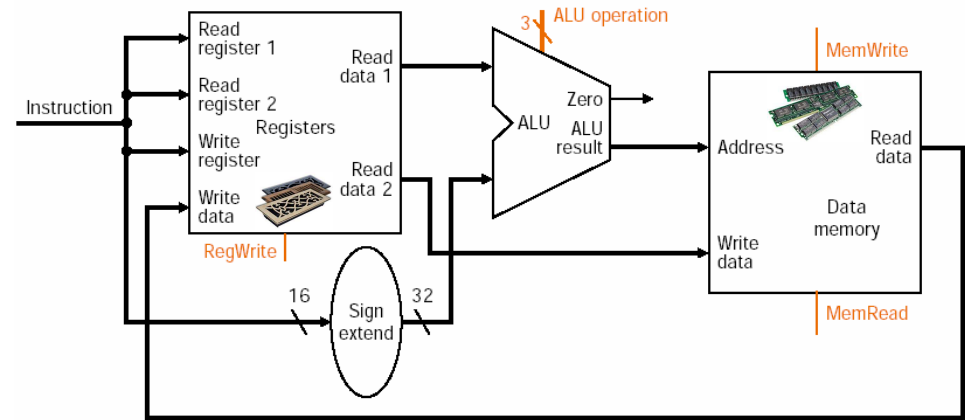
## The ALU (R-Type) Instructions



Consider:  $r1 = r2 - r3$

21

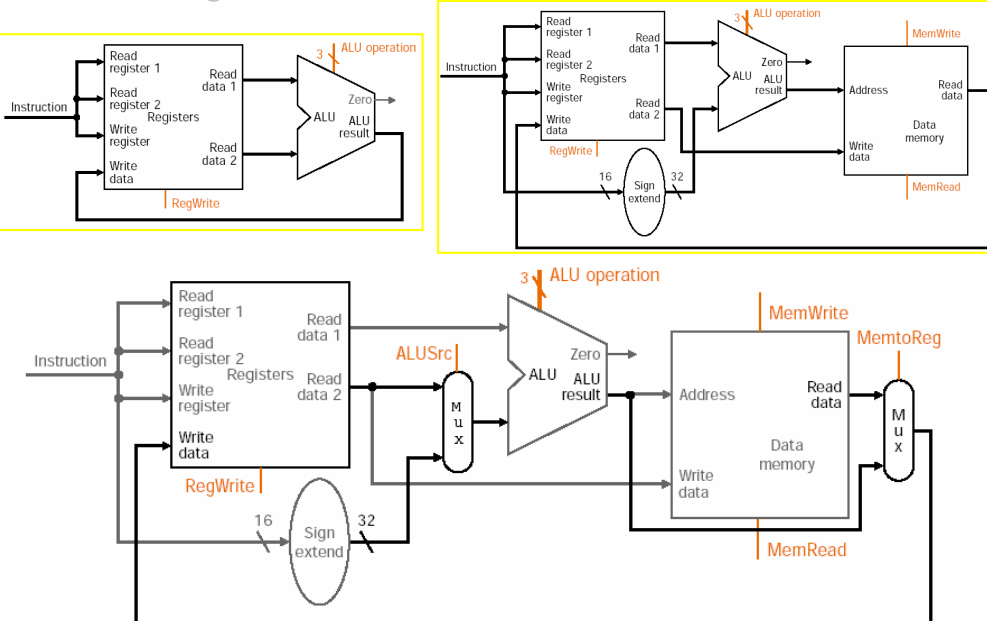
## Load and Store Instructions



Consider:  $r1 = M[r2 - 3]$

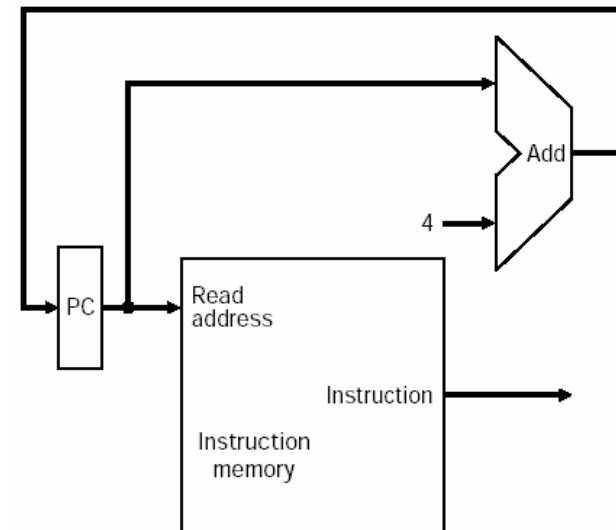
22

## Composition of Memory and R-Type Datapath The Magic of the Mux



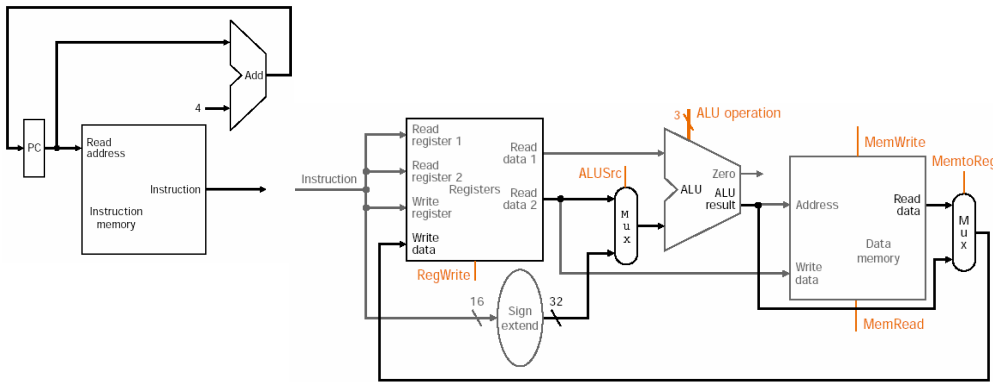
23

## Recall Fetch



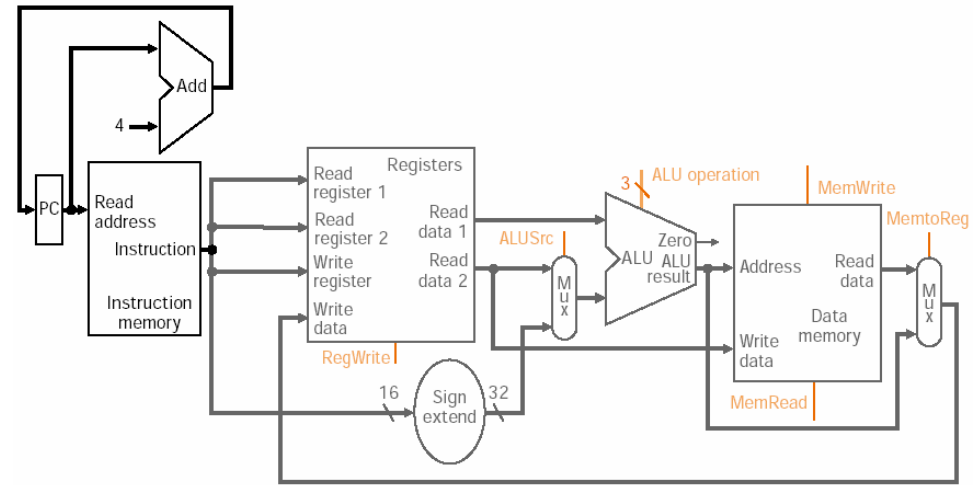
24

## Now Add Instruction Fetch



25

## Now Add Instruction Fetch (ALU + MEM + Fetch)



26

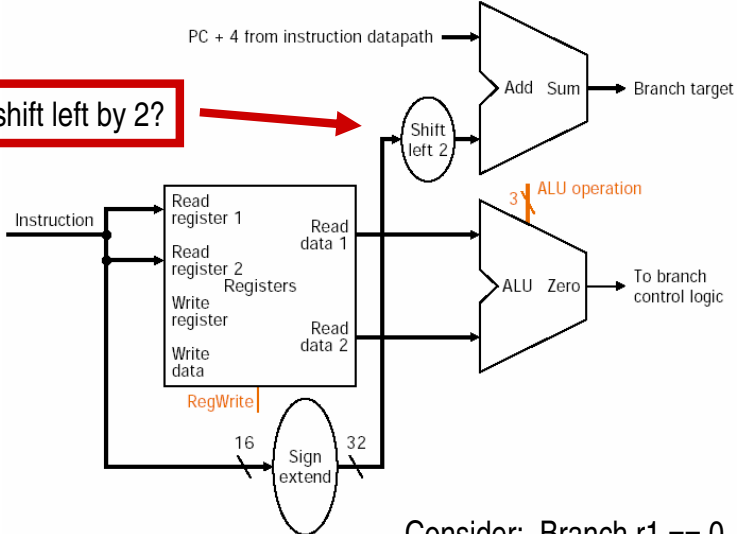
Data and Instruction memory?

## Branch Instructions

I: 

op	rs	rt	address / immediate
----	----	----	---------------------

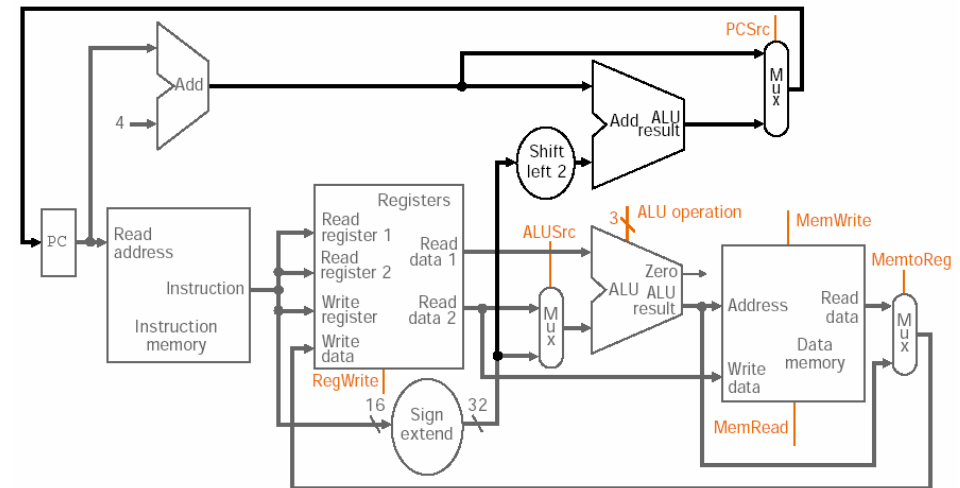
Why shift left by 2?



Consider: Branch r1 == 0, TARGET

27

## Add Branch to Datapath (ALU + MEM + Fetch + Branch)

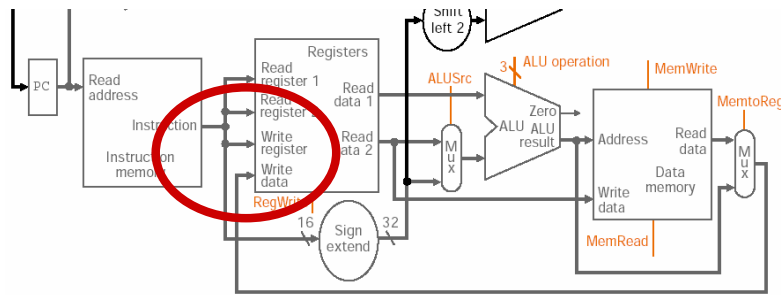
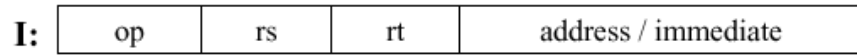
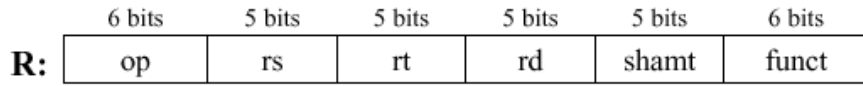


What will zero be connected to?

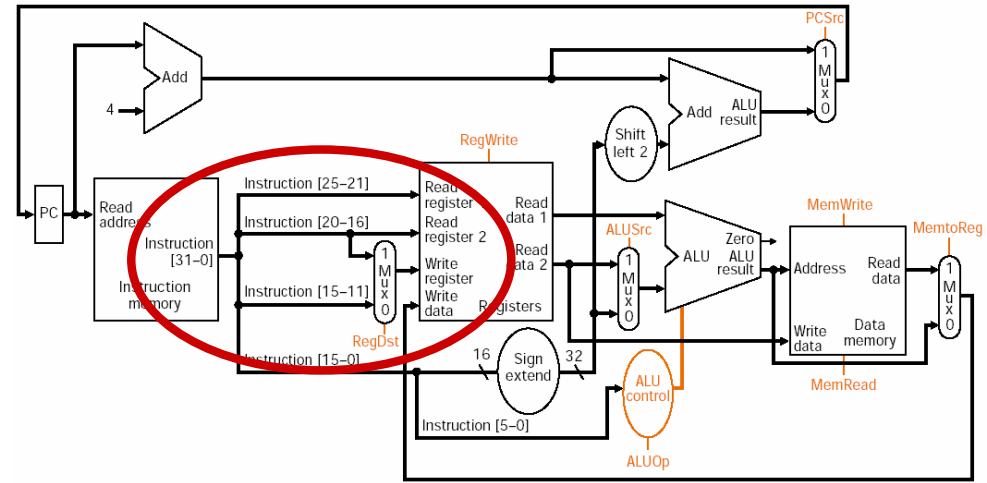
28

## MIPS Instruction Quirk

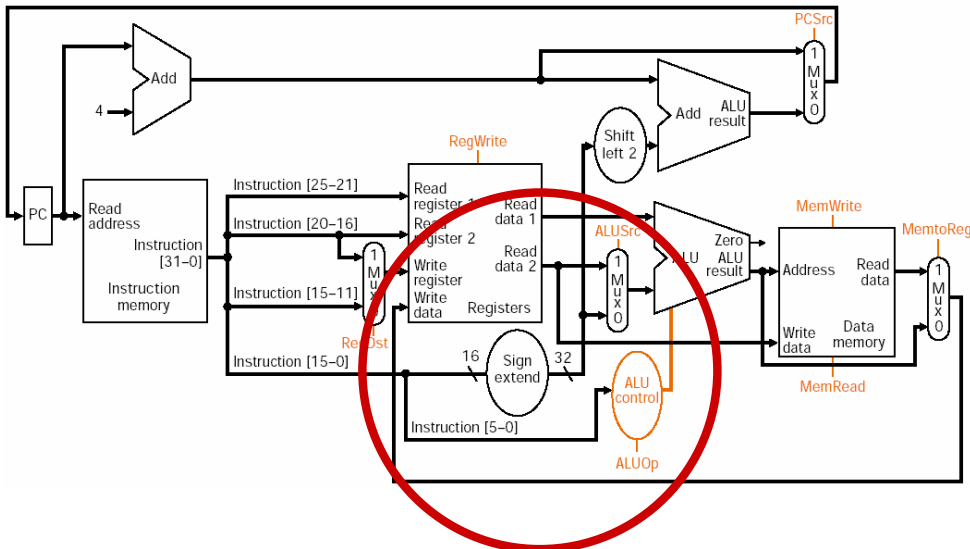
- The Destination Register may be in different locations
  - 11-15: Loads use rt
  - 16-20: All R-Types use rd



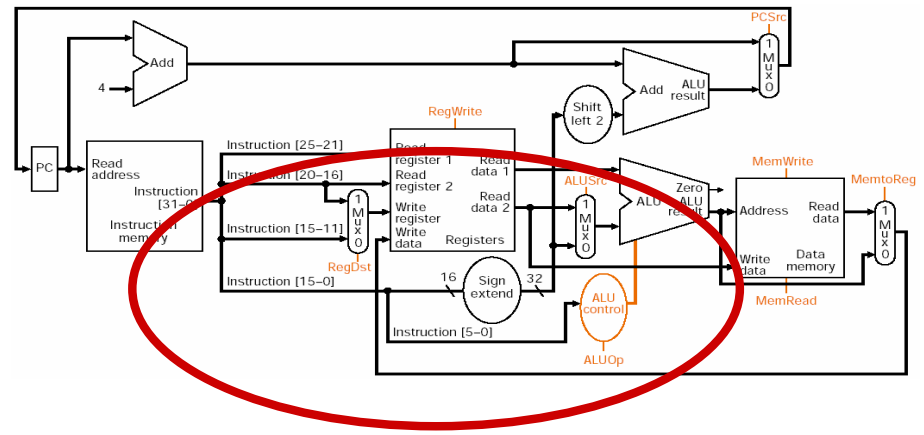
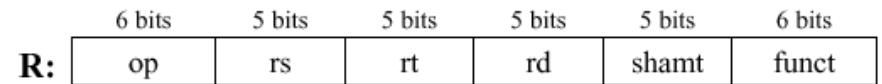
## Again, The Magic of the Mux!



## Ugh, what is going on here!?!



## Control vs. Datapath (Blurring the Line)



## What is Control?

### Control

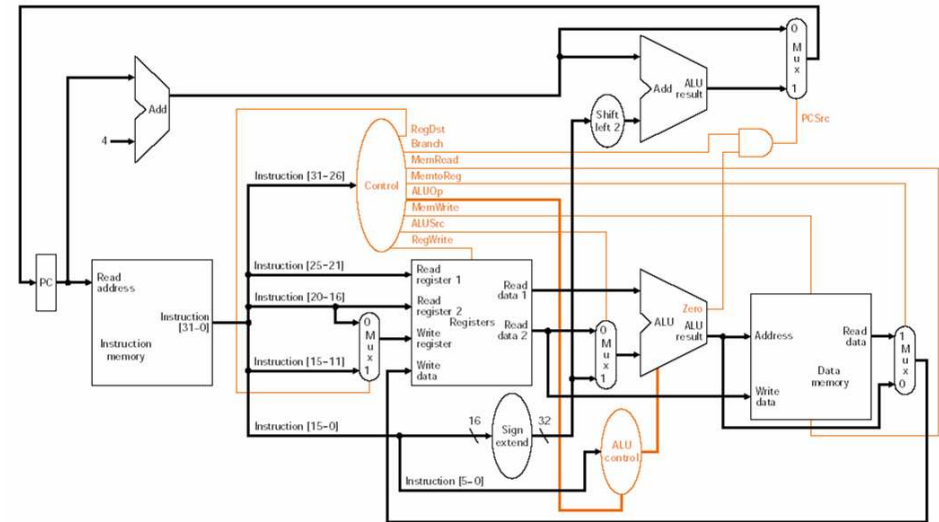
"The component of the processor that commands the datapath, memory, and I/O devices according to the instructions of the program." - H&P

### Control

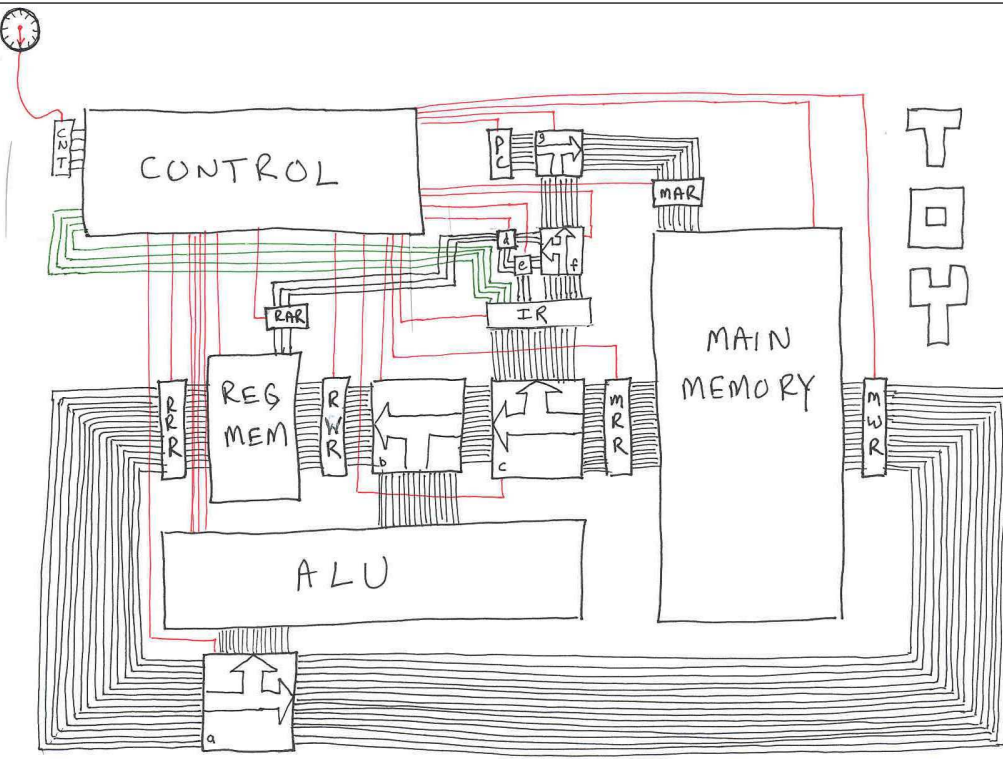
The component of the processor that commands the datapath, memory, and I/O devices according to the instructions of the program. - DIA

33

## Full Datapath with Control



34



## Summary and Next Steps

- The book doesn't define datapath well
- Computation and State elements compose datapath
- Look for reuse across instruction types
- Build minimal HW datapath with the magic of the mux

### Next Steps

- Need to define control
- Understand Timing
  - Single cycle
  - Multi-cycle
- Understand how to implement control

36

## For Next Time

- Be sure to know material in Appendix B
- Know how to make a finite state machine

