

Lecture 7: Datapath

COS / ELE 375

Computer Architecture and Organization

Princeton University
Fall 2015

Prof. David August

1

Datapath

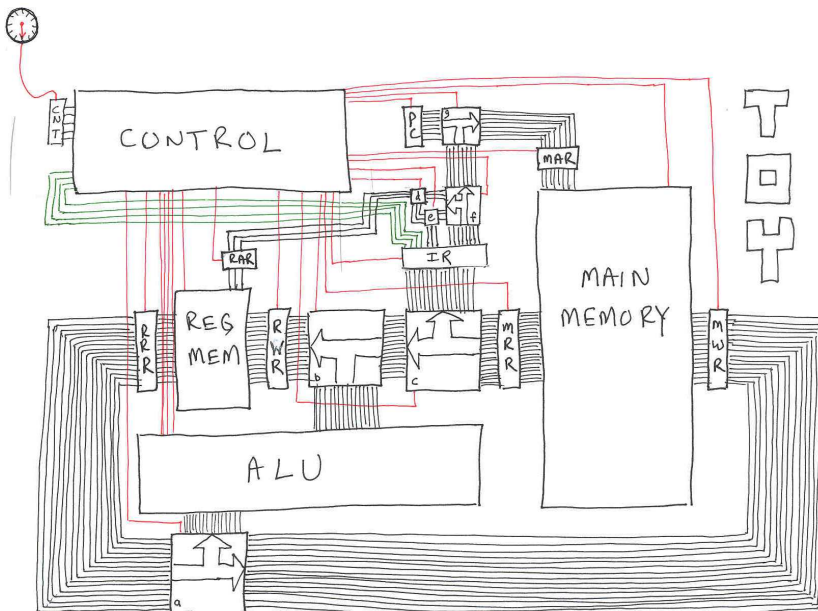
Datapath

“The component of the processor that performs arithmetic operations” – P&H

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

5



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 5th 1-bit ALU**

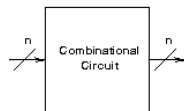
7

Datapath Elements

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

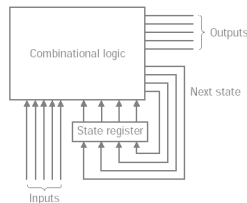
Computational Elements

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



State Elements

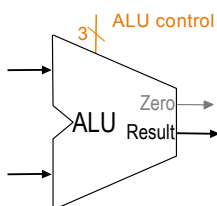
- Sequential Circuits
- Outputs change on clock edge
- Familiar Example: A Register



8

Computation Element: ALU

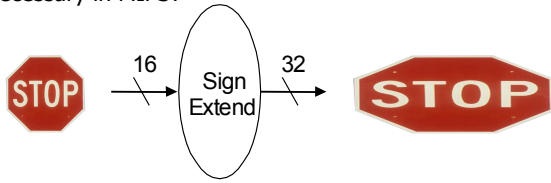
- Combinational - you had better know how to design it by now!!!
- Refine for MIPS
 - Zero equality test on all results - why?
 - Set on less than for `slt` instruction



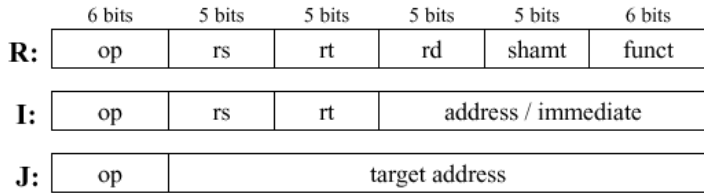
ALU Control	Function
000	AND
001	OR
010	add
110	subtract
111	set on less than

Computation Element: Sign Extender

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



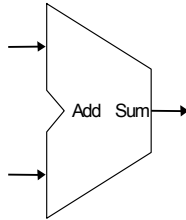
- Hint:



Implementation?

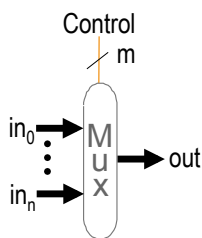
Computation Element: Adder

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



Computational Element: The Magical Mux

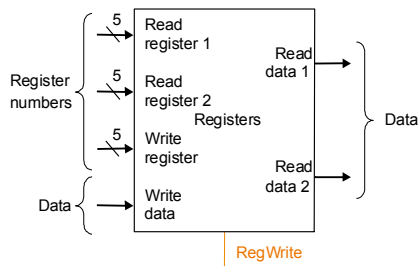
- Mux is short for Multiplexer (Think: selector)
- n input lines (of any common width)
- m control wires to select
- $n = 2^m$



Implementation?

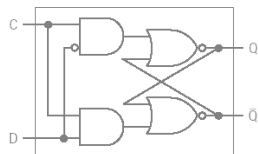
State Element: Register File

- 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

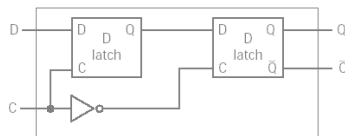


5-bits? 2 Reads? 1 Write?

State Element: Register File Register Implementation



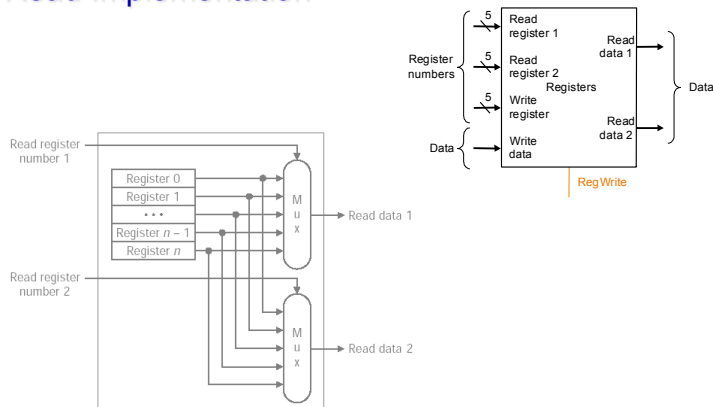
D latch



Falling edge triggered D flip-flop

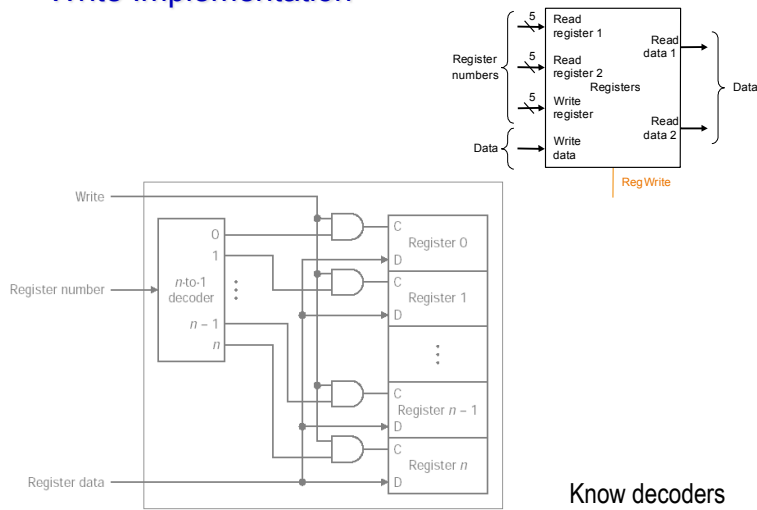
14

State Element: Register File Read Implementation



State Element: Register File

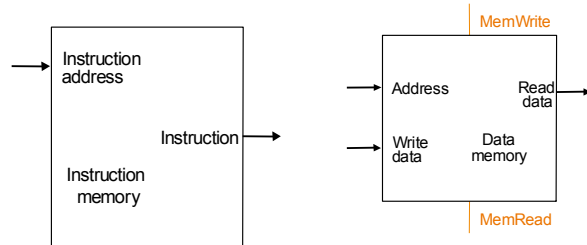
Write Implementation



Know decoders

State Element: Data and Instruction Memory

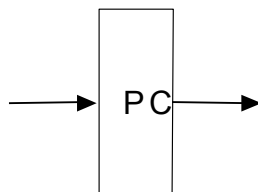
- Microarchitectural element to hold the architectural memory state
- See Appendix B for implementation details



17

State Element: The Program Counter

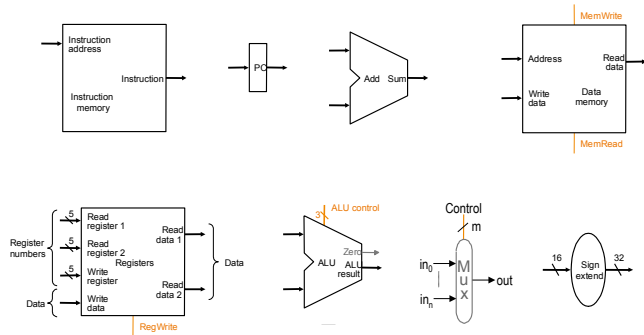
- To hold the architectural PC state
- Just like a single register



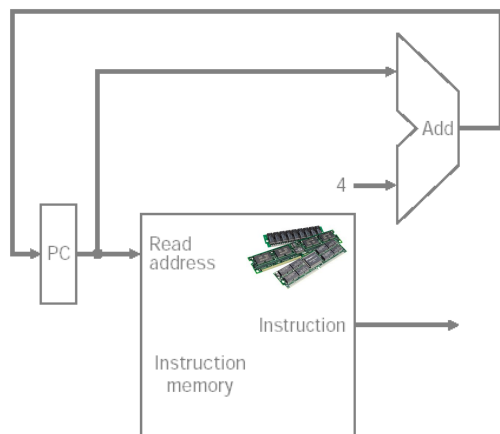
18

Our Complete Line of Products!

There may be others, but this is good for MIPS



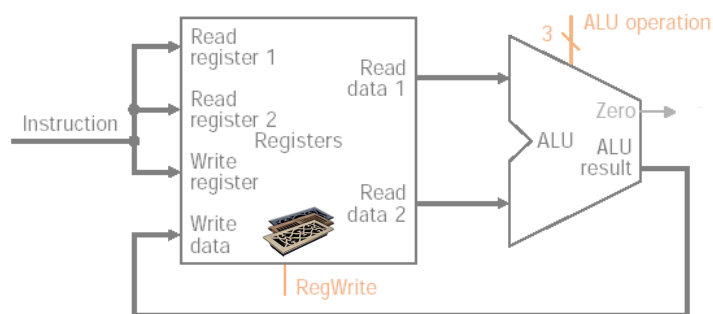
Fetching Instructions (no branching)



21

The ALU (R-Type) Instructions

	6 bits	5 bits	5 bits	5 bits	5 bits	6 bits
R:	op	rs	rt	rd	shamt	funct

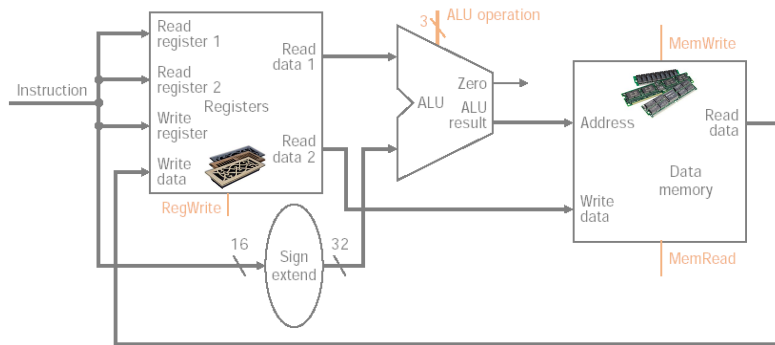


Consider: $r1 = r2 - r3$

22

Load and Store Instructions

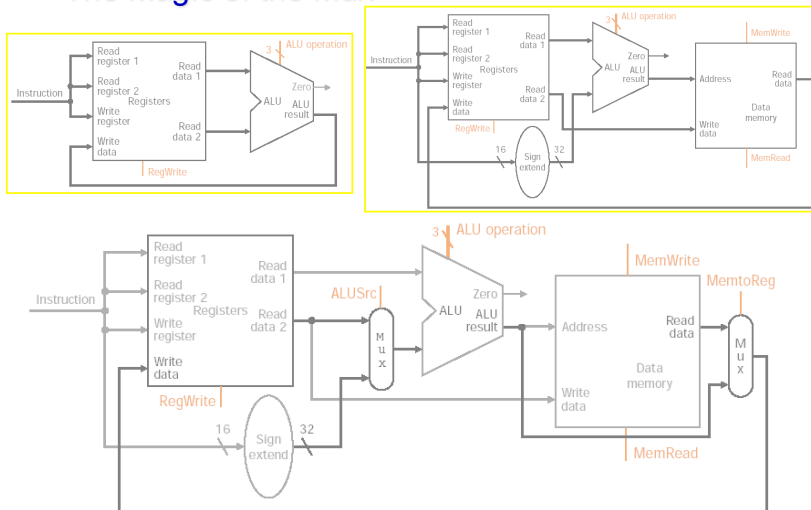
I:	op	rs	rt	address / immediate
----	----	----	----	---------------------



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

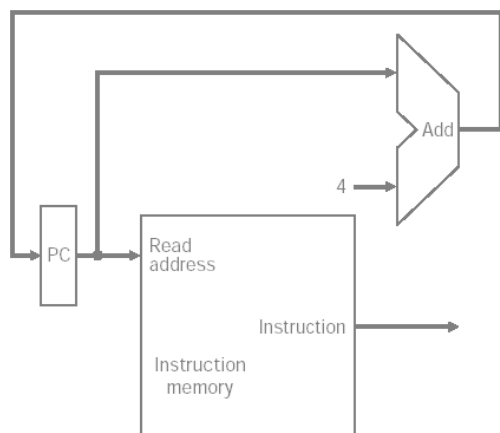
23

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



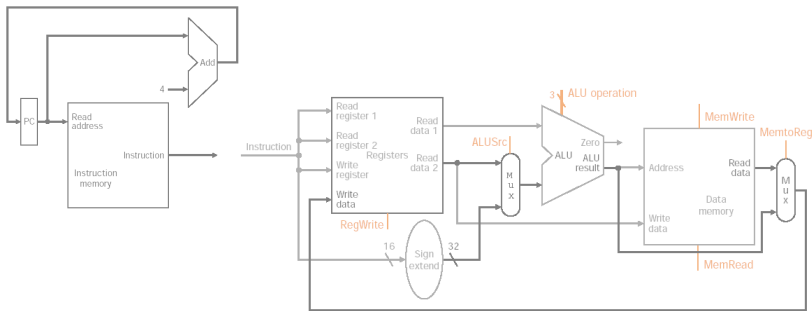
24

Recall Fetch



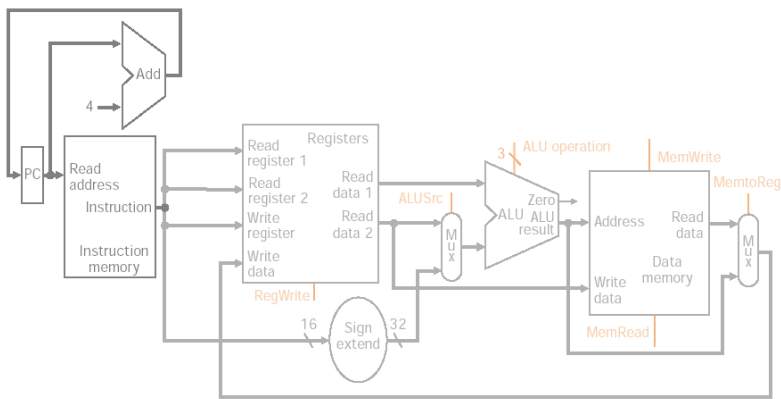
25

Now Add Instruction Fetch



26

Now Add Instruction Fetch (ALU + MEM + Fetch)

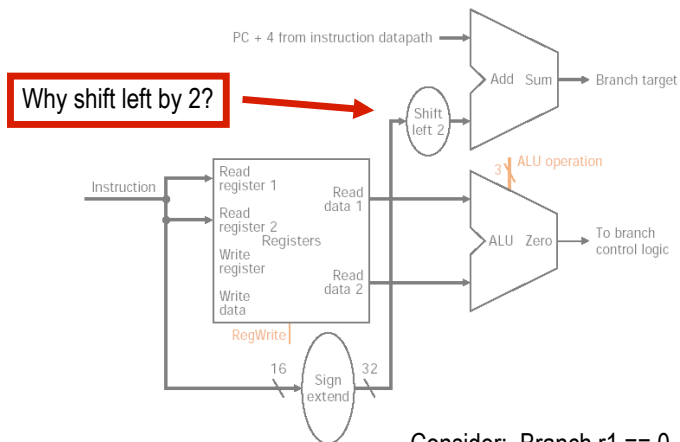


Data and Instruction memory?

27

Branch Instructions

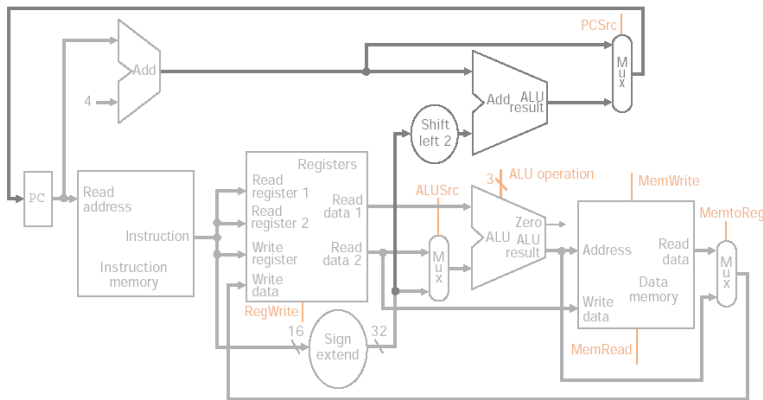
I:	op	rs	rt	address / immediate
----	----	----	----	---------------------



Consider: Branch r1 == 0, TARGET

28

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



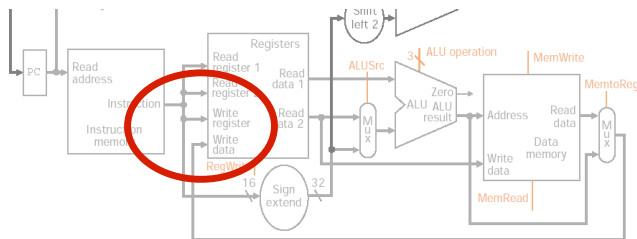
What will zero be connected to?

29

MIPS Instruction Quirk

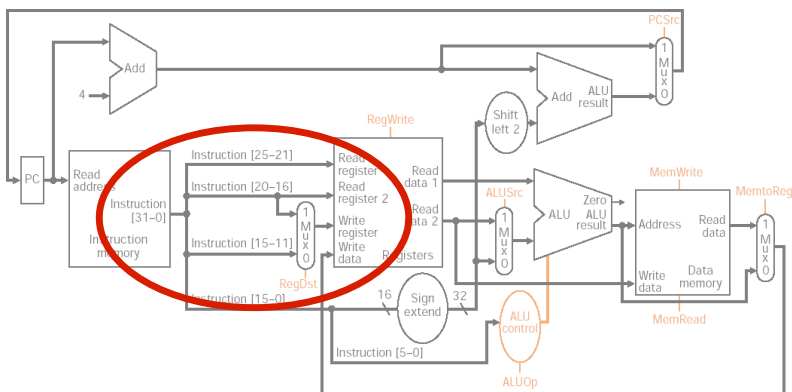
- The Destination Register may be in different locations
 - 11-15: Loads use rt
 - 16-20: All R-Types use rd

	6 bits	5 bits	5 bits	5 bits	5 bits	6 bits
R:	op	rs	rt	rd	shamt	funct
I:	op	rs	rt	address / immediate		



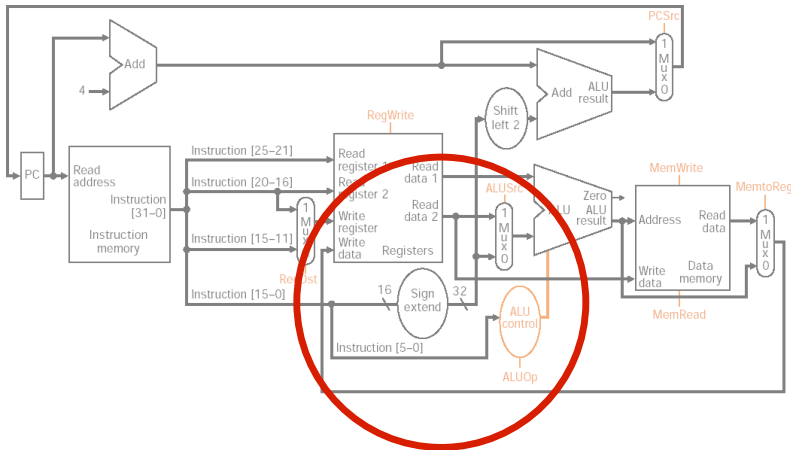
30

Again, The Magic of the Mux!



31

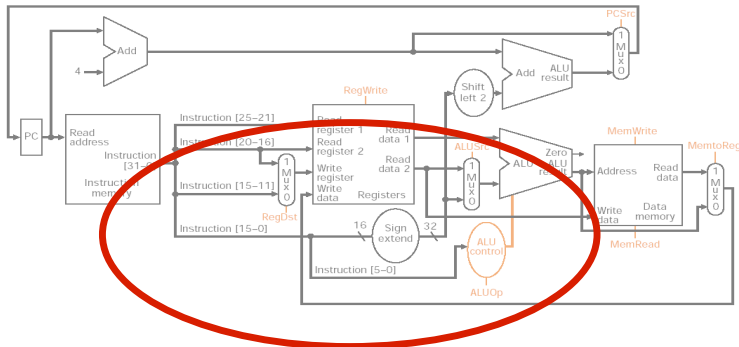
Ugh, what is going on here?!?



32

Control vs. Datapath (Blurring the Line)

	6 bits	5 bits	5 bits	5 bits	5 bits	6 bits
R:	op	rs	rt	rd	shamt	funct
I:	op	rs	rt	address / immediate		



33

What is Control?

Control

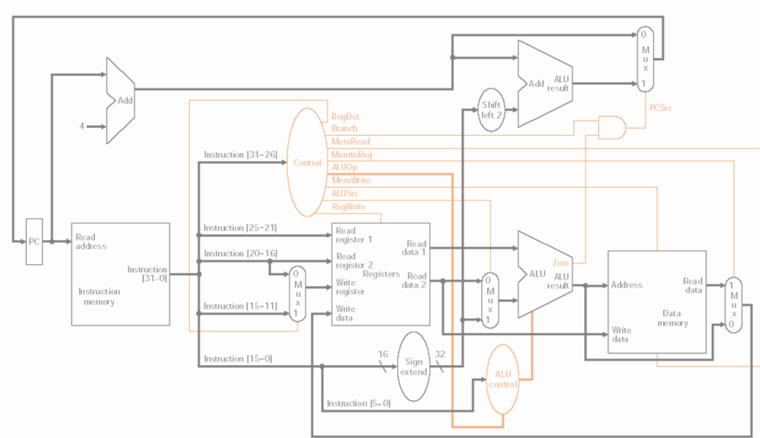
“The component of the processor that commands the datapath, memory, and I/O devices according to the instructions of the program.” – P&H

Control

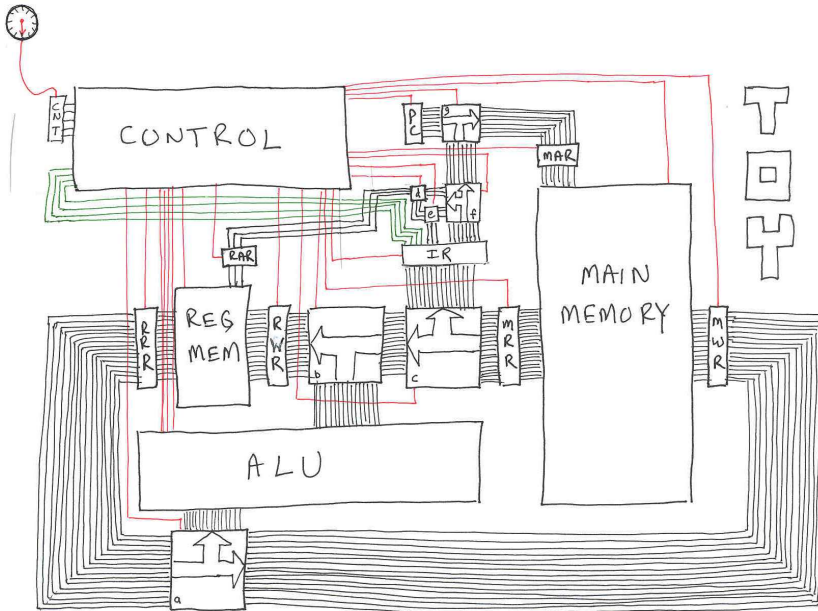
The component of the processor that commands the datapath, memory, and I/O devices according to the instructions of the program. - DIA

34

Full Datapath with Control



35



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

- Review finite state machines:

