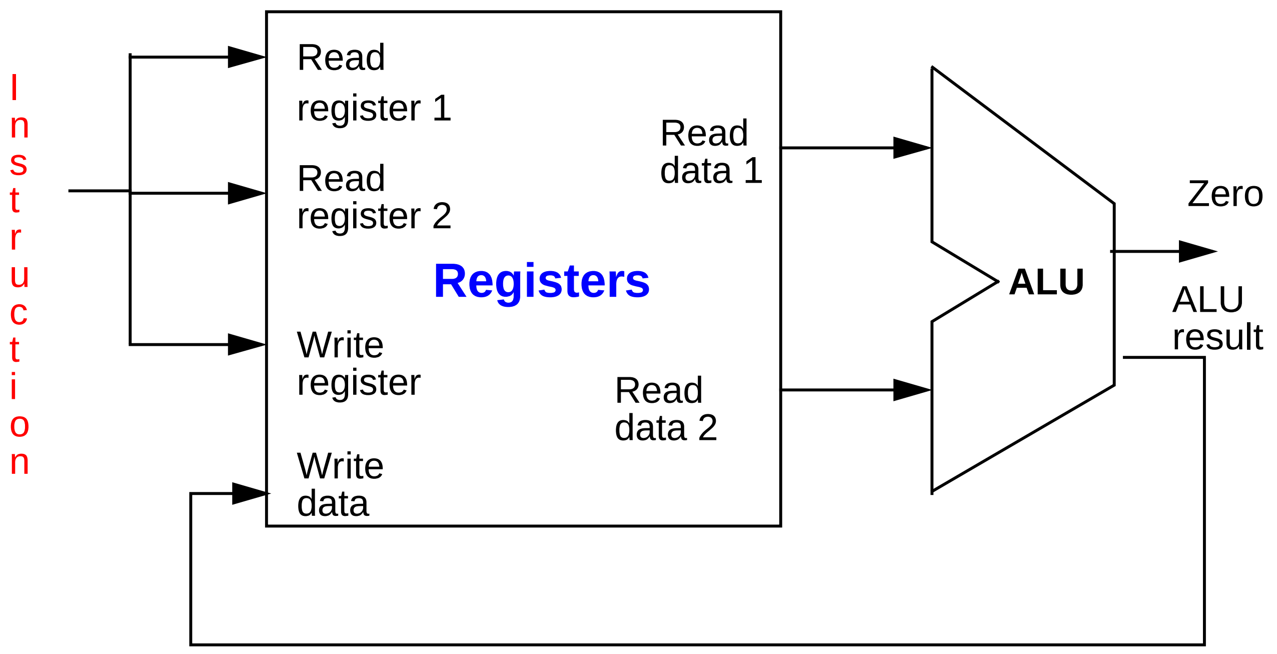
Outline of Lecture

- **Building Datapath for Various Operations**

The Data Path for R-type Instructions

- **The Data path for R-type instructions is as follows:**



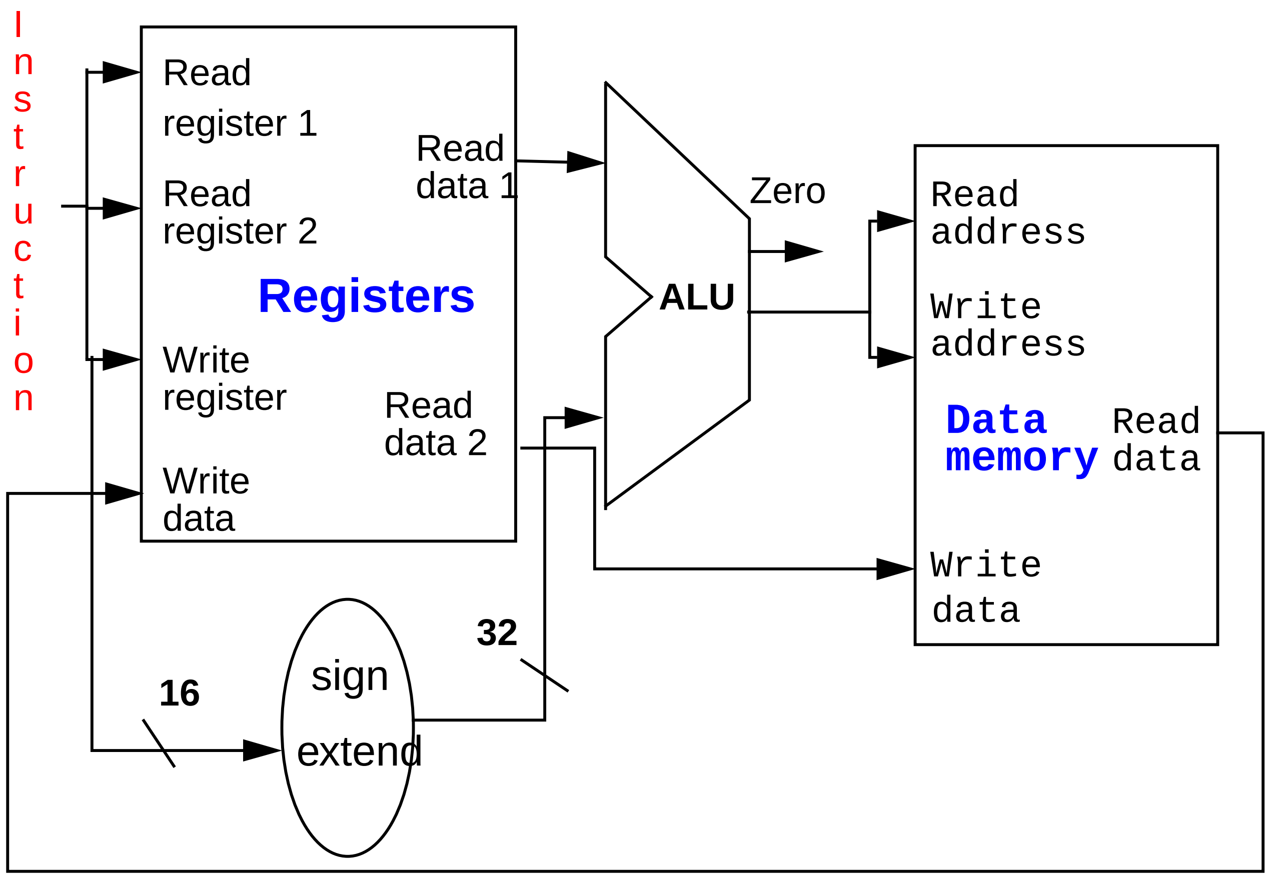
Load and Store Instructions

These are instructions (*I-type instructions*): they compute a memory address by adding a base register to a 16-bit signed offset field.

op	rs	rt	address
6 bits	5 bits	5 bits	16 bits

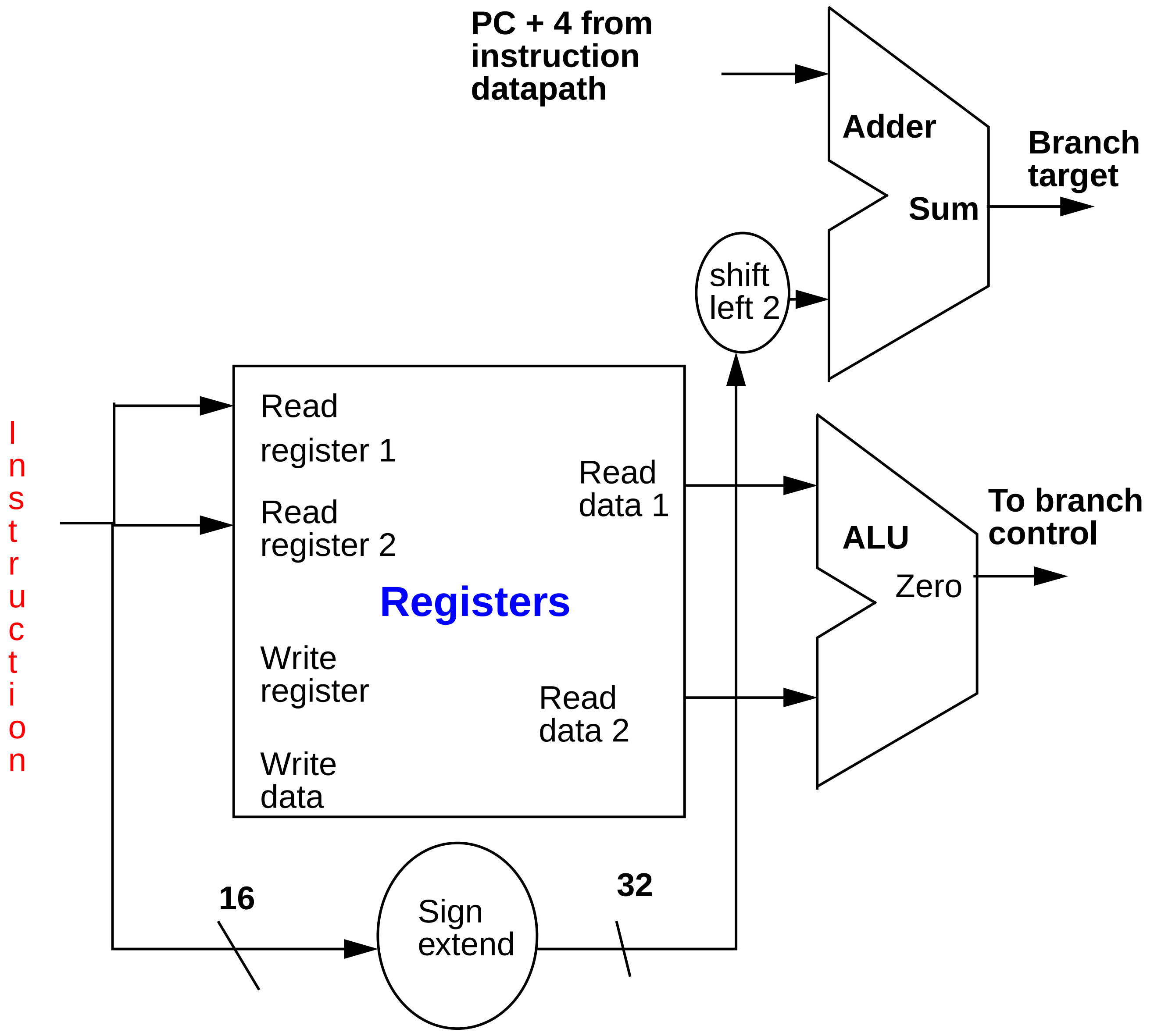
- → We need the same register file and same ALU as in R-type instructions.
- → We need a *data memory unit* to get the data (`load`) and store the data (`store`).
- → We also need to *sign-extend* the 16-bit offset into a 32-bit offset to be input into our 32-bit ALU.

- The data path for the `load` and `store` operations is as follows:



Branching Instructions

- The **beq** instruction: it has 3 operands, 2 registers that are compared for equality, and a 16-bit offset used to compute the branch target address relative to the branch instruction address.



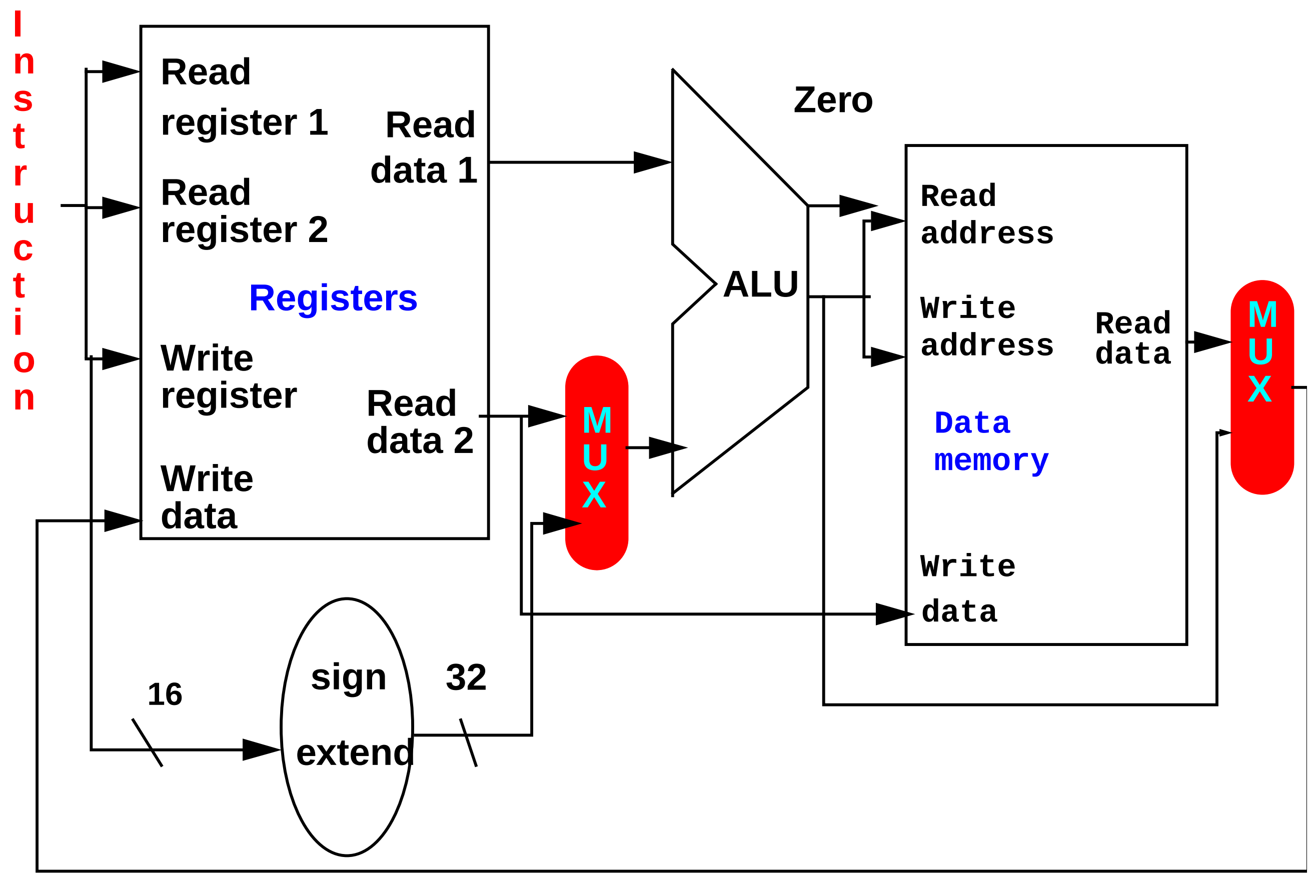
Simple Implementation of a Single Datapath

- We will design a very simple implementation of our MIPS datapath - it executes a partial subset of the MIPS instructions: **`lw`**, **`sw`**, **`beq`**, **`add`**, **`sub`**, **`and`**, **`or`**, and **`slt`**.

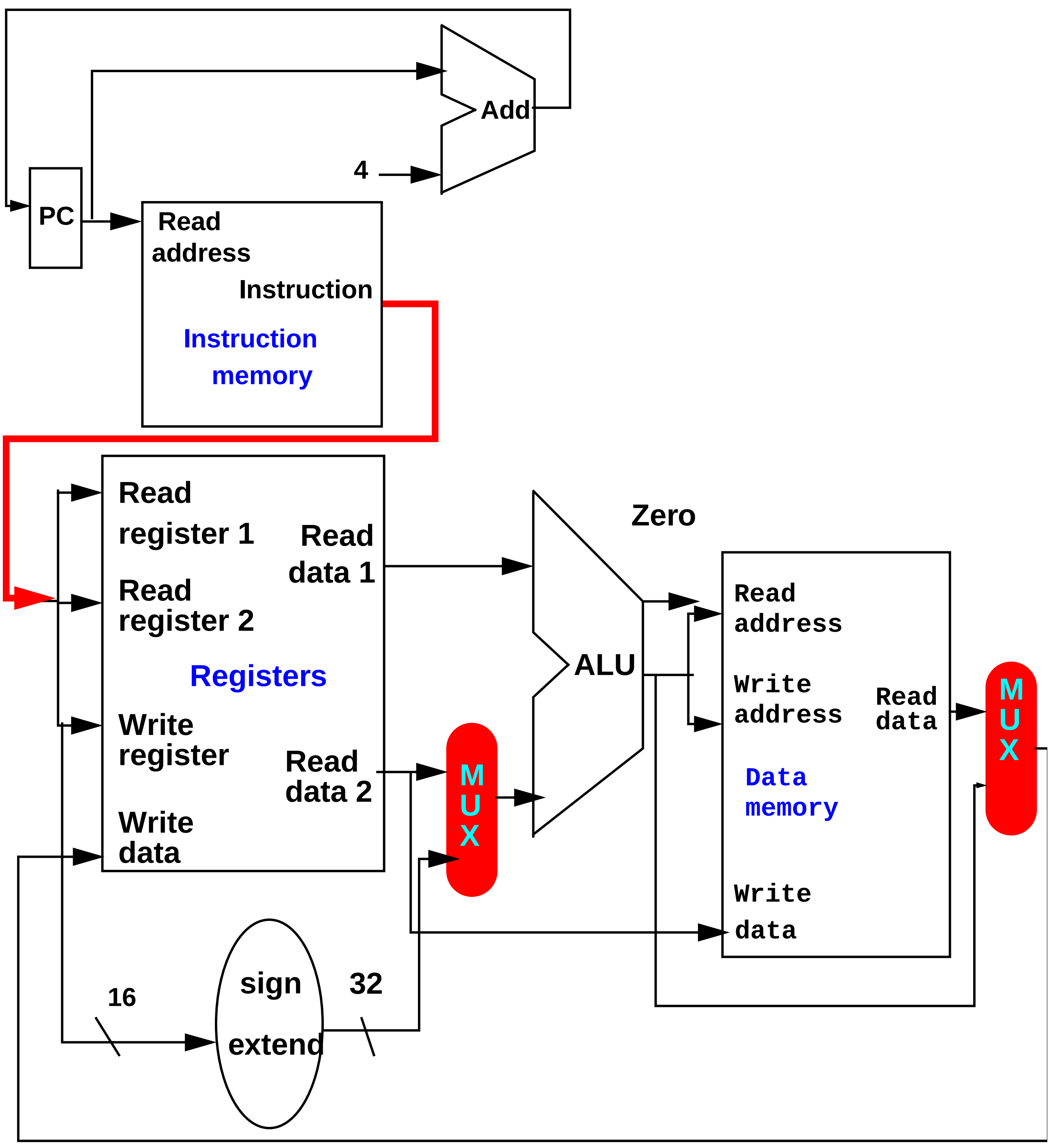
- We already built the datapath for each of these instructions ***separately***. Now, we need to combine them into a single datapath. Thus, we need to devise ways of ***sharing*** some of the resources (e.g., ALU) between the different instructions.

We should avoid duplicating resources.

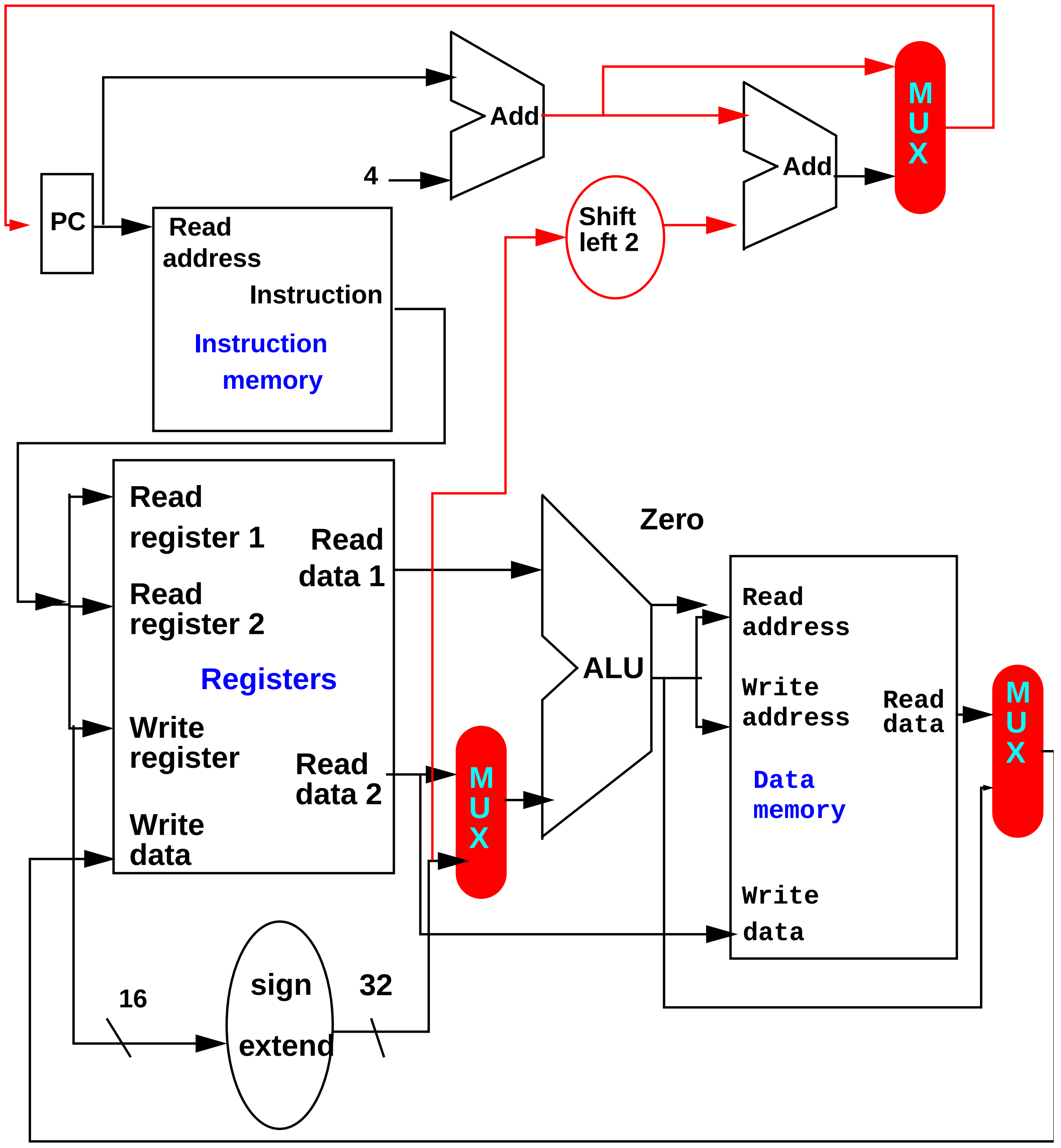
- We can easily combine the datapath for the R-type instructions (e.g., `add`, `sub`) with the datapath for the I-type instructions (e.g., `lw`, `sw`) (by using multiplexers to allow sharing of resources).



- The instruction ***fetch*** part of the datapath can also be easily added.



- Next, we add the datapath for ***branches*** - so that we get a complete single datapath.



ALU Control

- Now that we finished the design of this partial single MIPS datapath, we need to add the ***control unit*** that controls the whole operation of the datapath.

- From the previous chapter - when we designed the ALU - these are the control signals that we came up with:

ALU Control lines	Function
0 00	AND
0 01	OR
0 10	ADD
1 10	SUB
1 11	SLT

- By using the ***function field*** (6 bits for MIPS), the table above can be expanded to cover a wider range of instructions.

Instruction opcode	ALUOp	Instruction operation	Function code	Desired ALU action	ALU control unit
LW	00	load word	XXXXXX	add	010
SW	00	store word	XXXXXX	add	010
Branch equal	01	branch equal	XXXXXX	subtract	110
R-type	10	add	100000	add	010
R-type	10	subtract	100010	subtract	110
R-type	10	AND	100100	and	000
R-type	10	OR	100101	or	001
R-type	10	SLT	101010	slt	111

Further Reading

Chapter 5: **David A. Patterson and John L. Hennessy. *Computer Organization & Design: The Hardware / Software Interface*. Morgan Kaufman Publishers, 1998. (348-355).**