Processor Performance

Check In 2 today!

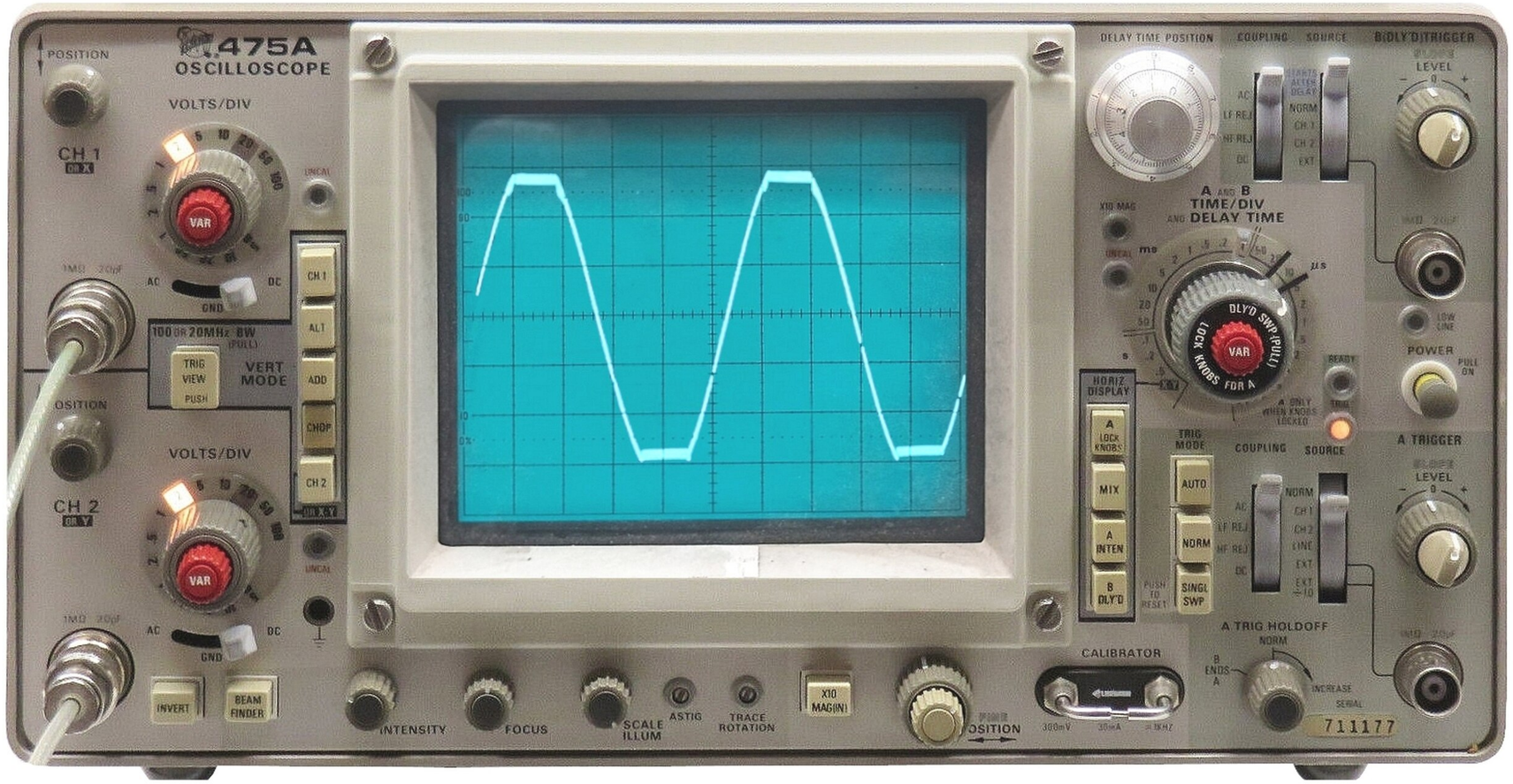


Image Credit: https://en.wikipedia.org/wiki/Oscilloscope

Outline

- Reviewing processor performance
- How to improve performance
- Pipeline motivation

Calculating Execution Times

- CPU execution time = Instruction Count * Clock Cycle Time * Cycles per Instruction
- Goal: improve execution time!

Reduce instruction count!

Reduce clock cycle time!

Reduce cycles per instruction!

Deconstructing Our Data Path

Overall Clock Speed: 550ps

Instruction Class	PC (50ps)	Instruction Memory (150ps)	Register File (50ps)	ALU (100ps)	Data Memory (200ps)
`ldi`	✅	✅	✅	❌	❌
`add`	✅	✅	✅	✅	❌
`lw`	✅	✅	✅	❌	✅
`sw`	✅	✅	✅	✅	✅

250ps

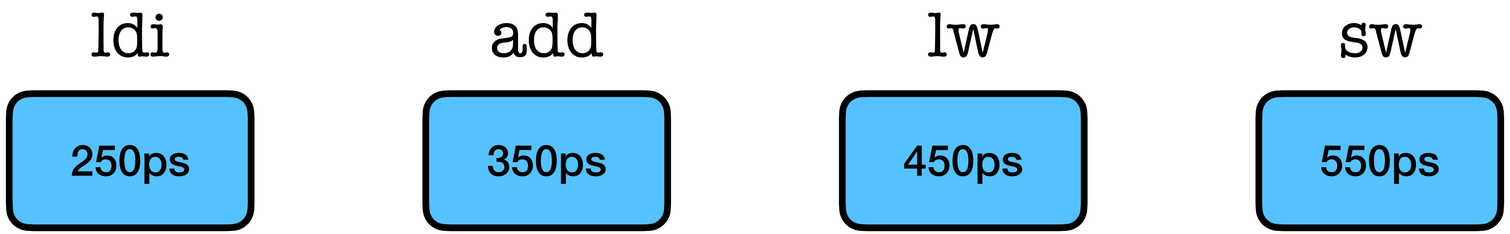
350ps

450ps

550ps

Chat with your neighbor(s)!

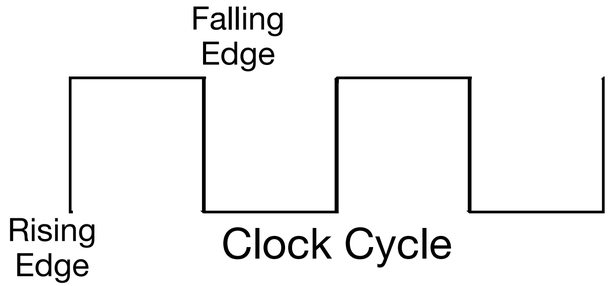
What's the best case runtime for a 100 instruction program with 65% `add` instructions, 20% `ldi` instructions, and 15% `sw` instructions? What's the actual runtime in a "instruction per cycle" processor?

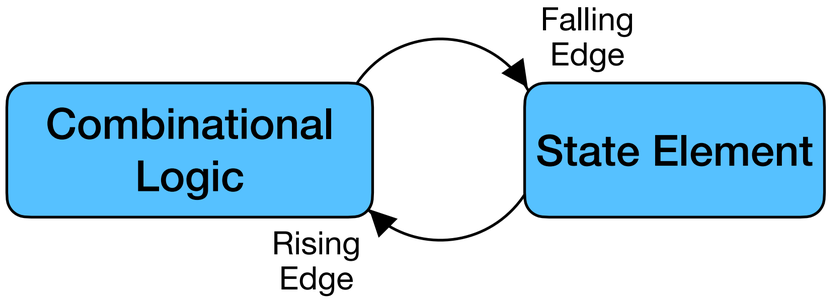


Improving Cycle Time

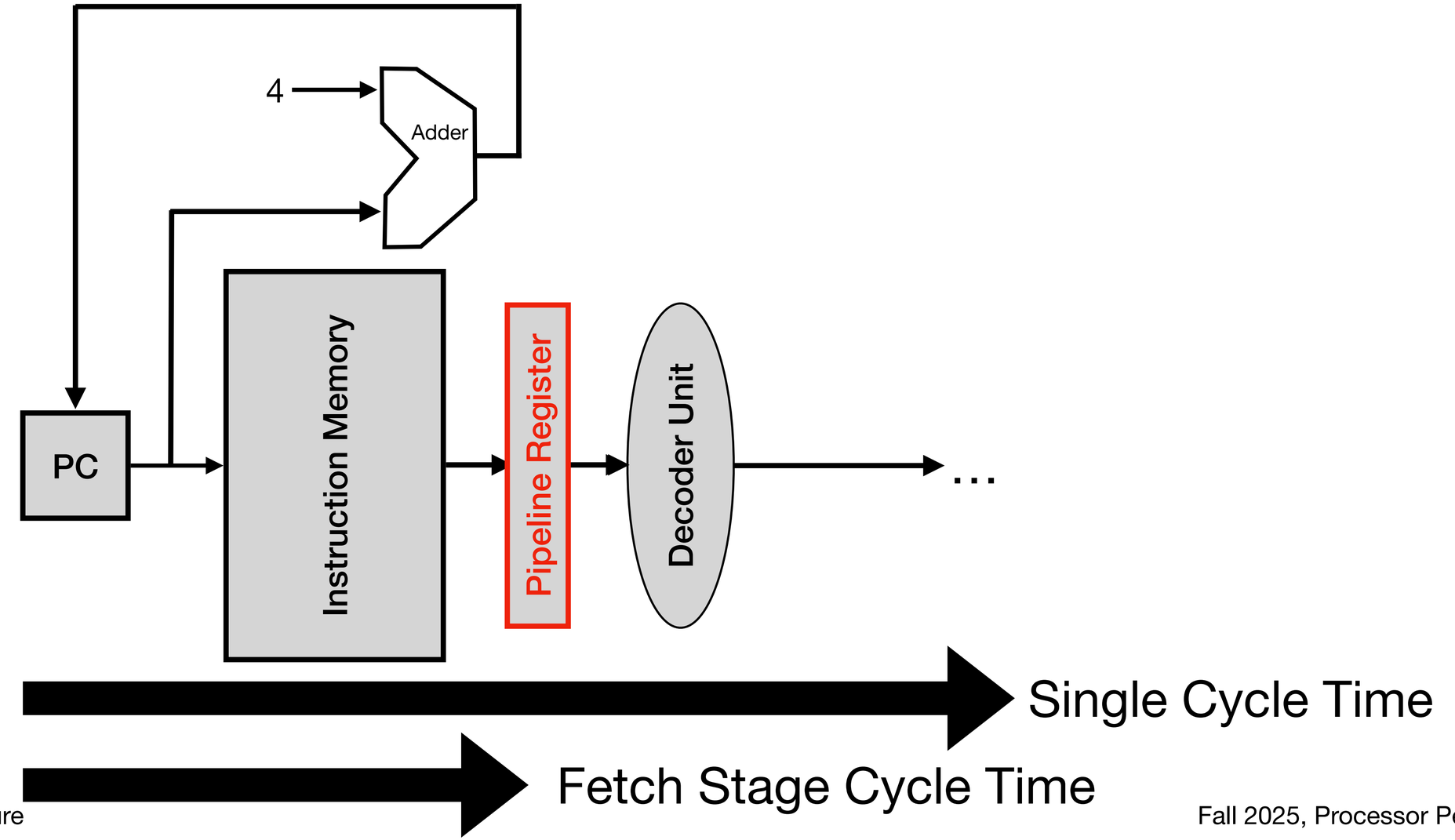
Reduce clock cycle time!

- In a “one instruction per cycle” processor, the clock cycle latency is the sum of the latencies for all components in the data path
- We can reduce the clock cycle by building a processor that uses more cycles per instruction to do the same work
- We need to add a state element after each combinational logic element for information to get set!

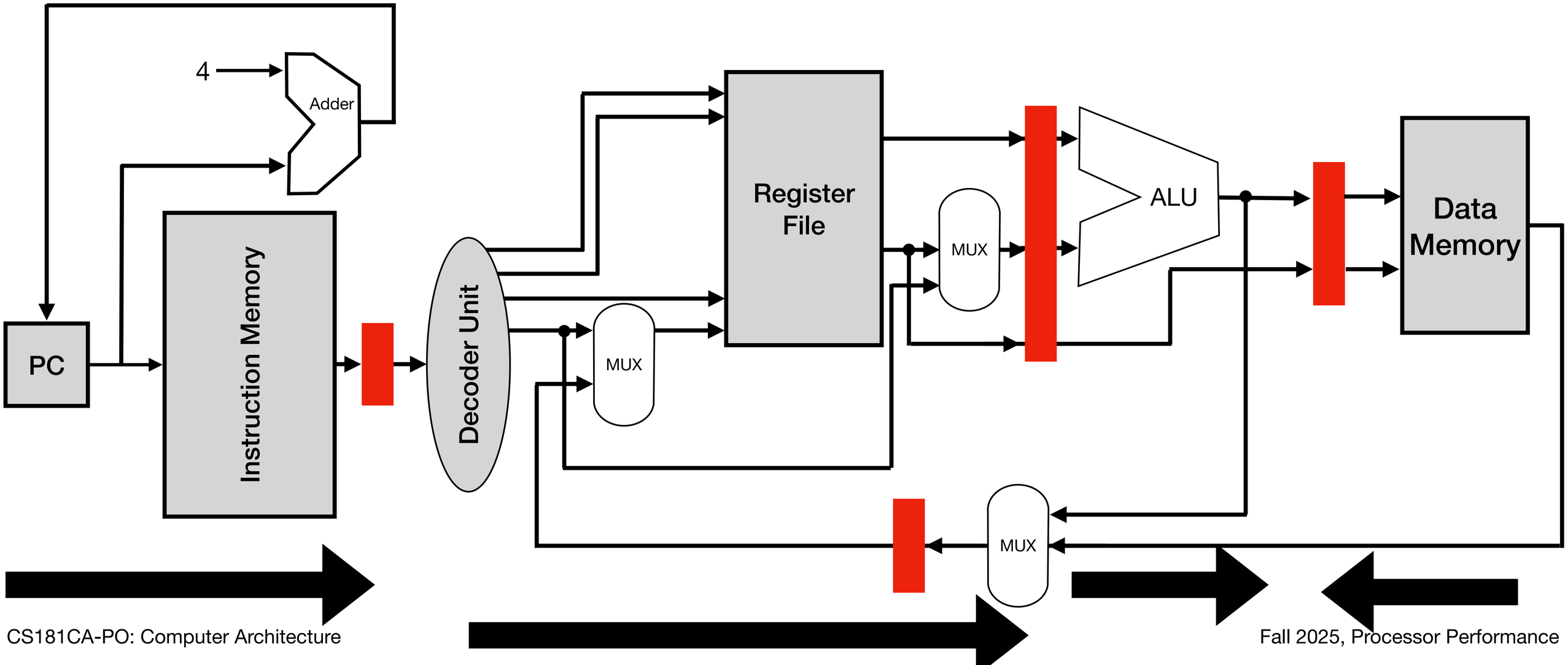




Adding State Elements to Our Data Path



Adding State Elements to Our Data Path



Takeaways

- To improve processor performance, we can try to reduce the clock cycle time!
- Shorter cycle times may be achieved by adding storage elements (registers) to the data path
- Adding elements to the data path adds latency… what to do about this?