Understanding the Single Cycle Processor

Lab Tonight: Gradescope submission and debugging tricks!

Image Credit: https://americanhistory.si.edu/collections/object/nmah_334396

First “personal computer” sold for ~$500 and required assembly upon purchase

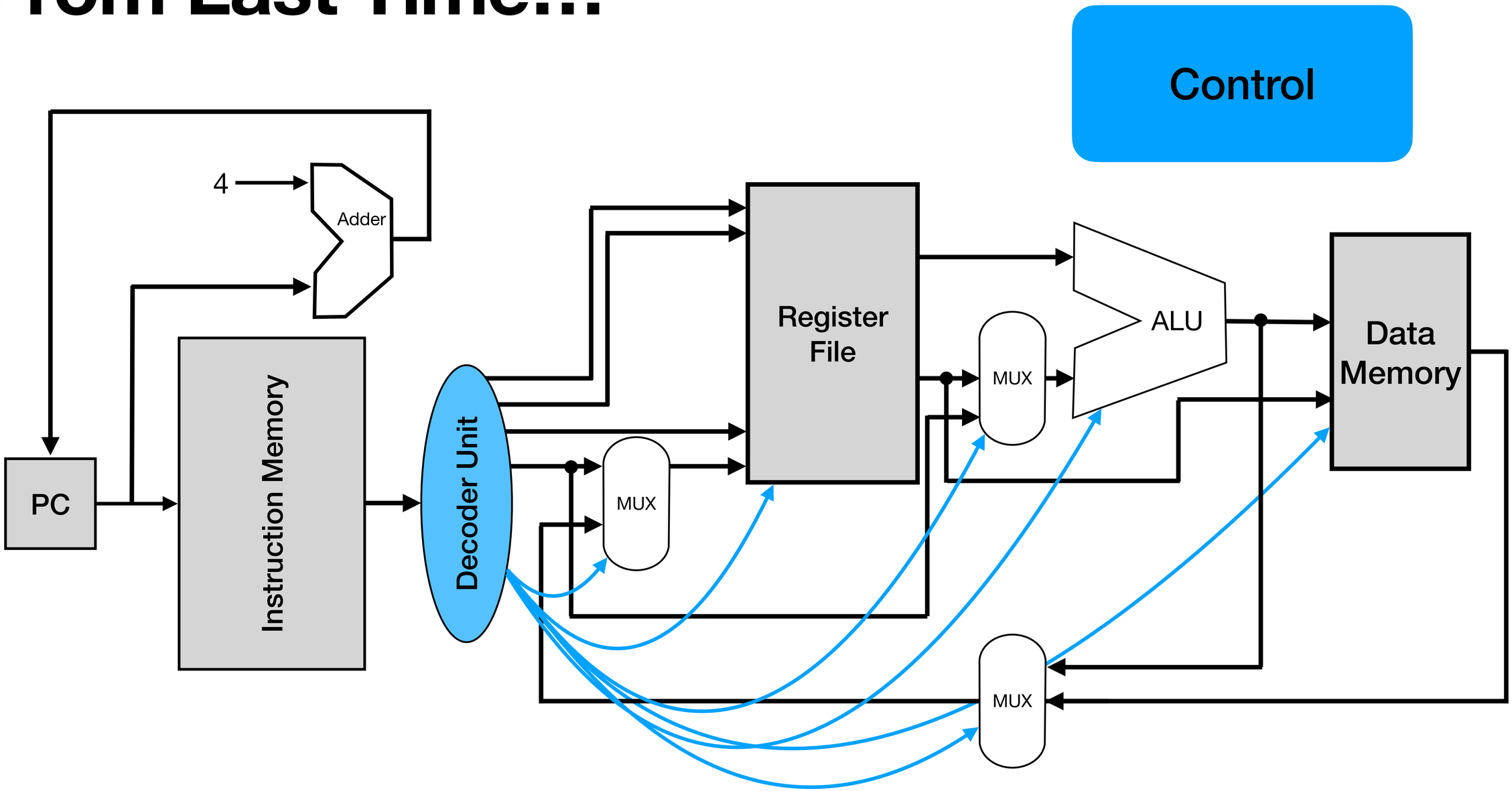
Built around the Intel 8080 chip

Clock speed of 2MHz!

Outline

- Understanding control signals
- Processor clock considerations
- Drawbacks of a single-stage processor

From Last Time…



Control Signals

- ALU Select: determines whether the second ALU data input is from the instruction (i.e., an immediate) or from a register source
- ALU Op: determines which operation the ALU will perform
- Memory Read/Write: determines whether the address should be read from or written to
- Memory to Register: determines whether the data to write to the register file should come from the execution units or from memory

All of this information can be inferred from the instruction opcode! The decode unit sends signals to the appropriate fields

Setting Control Signals

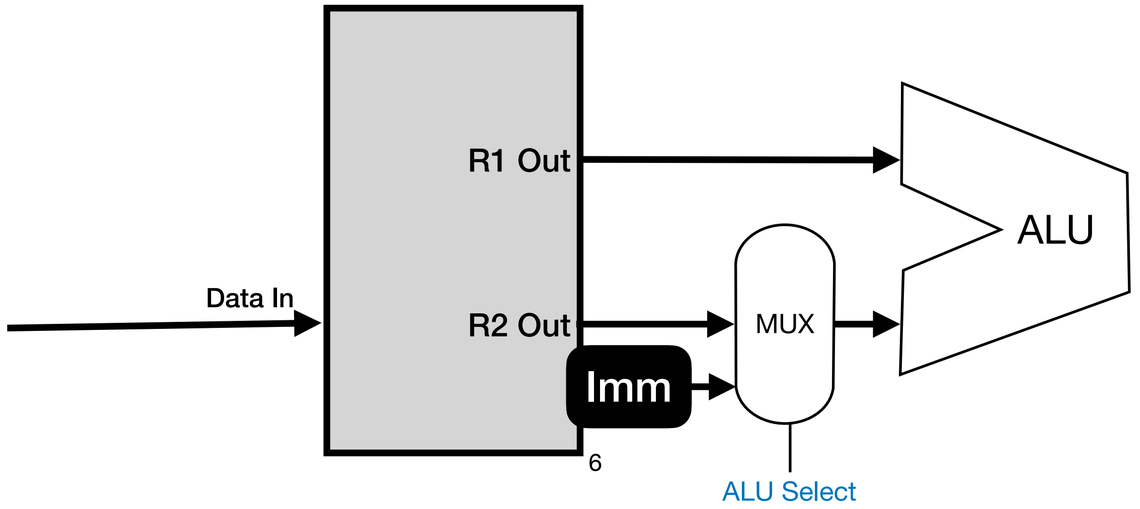
31 25	24 20	19 15	14 12	11 7	6 0	
funct7	rs2	rs1	funct3	rd	opcode	R-type
imm[11:0]		rs1	funct3	rd	opcode	I-type
imm[11:5]	rs2	rs1	funct3	imm[4:0]	opcode	S-type
imm[12|10:5]	rs2	rs1	funct3	imm[4:1|11]	opcode	B-type
imm[31:12]				rd	opcode	U-type
imm[20|10:1|11|19:12]				rd	opcode	J-type

Immediate Format

0xE5B18393

I-type

0b11100101101100011000001110010011



Wire from decoder set to true!

Setting Control Signals

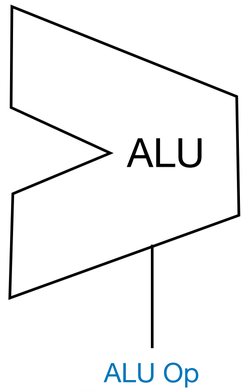
31 25	24 20	19 15	14 12	11 7	6 0	
funct7	rs2	rs1	funct3	rd	opcode	R-type
imm[11:0]		rs1	funct3	rd	opcode	I-type
imm[11:5]	rs2	rs1	funct3	imm[4:0]	opcode	S-type
imm[12|10:5]	rs2	rs1	funct3	imm[4:1|11]	opcode	B-type
imm[31:12]				rd	opcode	U-type
imm[20|10:1|11|19:12]				rd	opcode	J-type

Immediate Format

0xE5B18393

add I-type

0b11100101101100011000001110010011



Wire set to “add” encoding by decoder!

Setting Control Signals

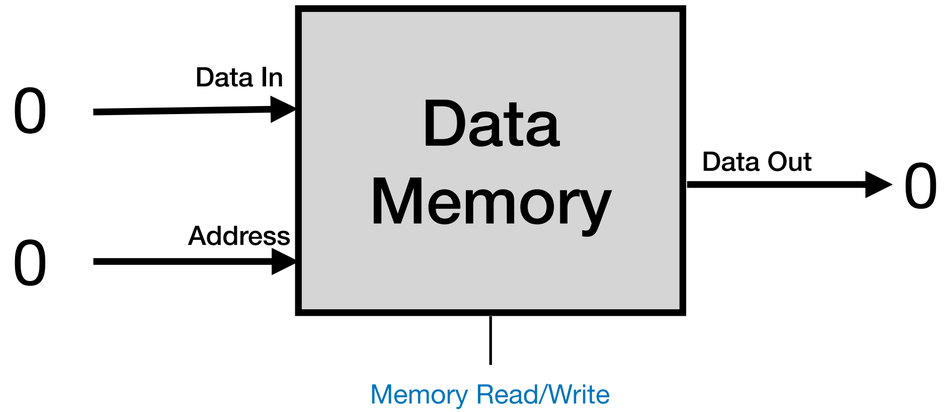
31 25	24 20	19 15	14 12	11 7	6 0	
funct7	rs2	rs1	funct3	rd	opcode	R-type
imm[11:0]		rs1	funct3	rd	opcode	I-type
imm[11:5]	rs2	rs1	funct3	imm[4:0]	opcode	S-type
imm[12|10:5]	rs2	rs1	funct3	imm[4:1|11]	opcode	B-type
imm[31:12]				rd	opcode	U-type
imm[20|10:1|11|19:12]				rd	opcode	J-type

Immediate Format

0xE5B18393

add I-type

0b11100101101100011000001110010011



Signals set arbitrarily, could even have unique signal for memory “no-op”

Setting Control Signals

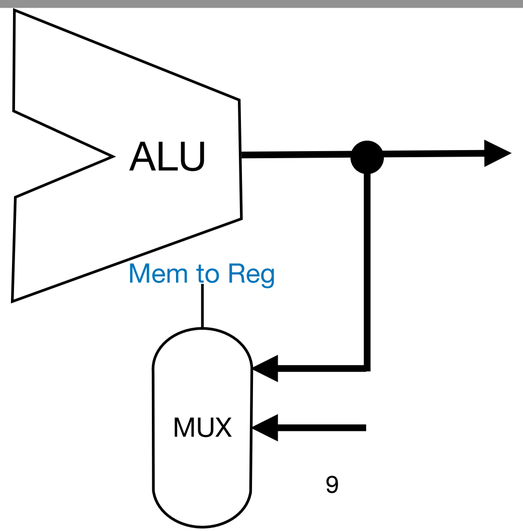
31 25	24 20	19 15	14 12	11 7	6 0	
funct7	rs2	rs1	funct3	rd	opcode	R-type
imm[11:0]		rs1	funct3	rd	opcode	I-type
imm[11:5]	rs2	rs1	funct3	imm[4:0]	opcode	S-type
imm[12|10:5]	rs2	rs1	funct3	imm[4:1|11]	opcode	B-type
imm[31:12]				rd	opcode	U-type
imm[20|10:1|11|19:12]				rd	opcode	J-type

Immediate Format

0xE5B18393

add I-type

0b11100101101100011000001110010011



The opcode indicates that our result will not come from memory. Set to 0 by decoder

Chat with your neighbor(s)!

Consider the instruction "`lw r2, r4`". What are the control signals sent by the decoder to the processor components? How does this change for "`sw r2, r4`"?

ALU Select

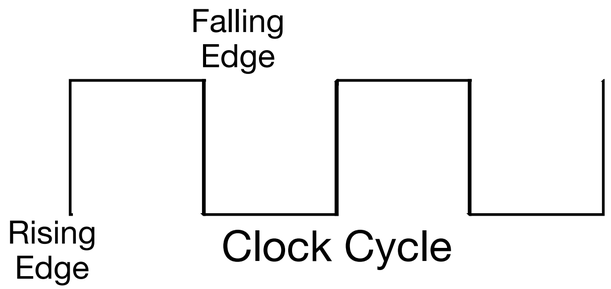
ALU Operation

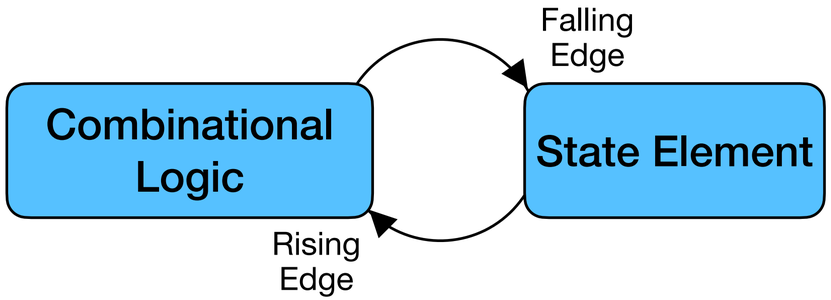
Memory Read/Write

Memory to Register

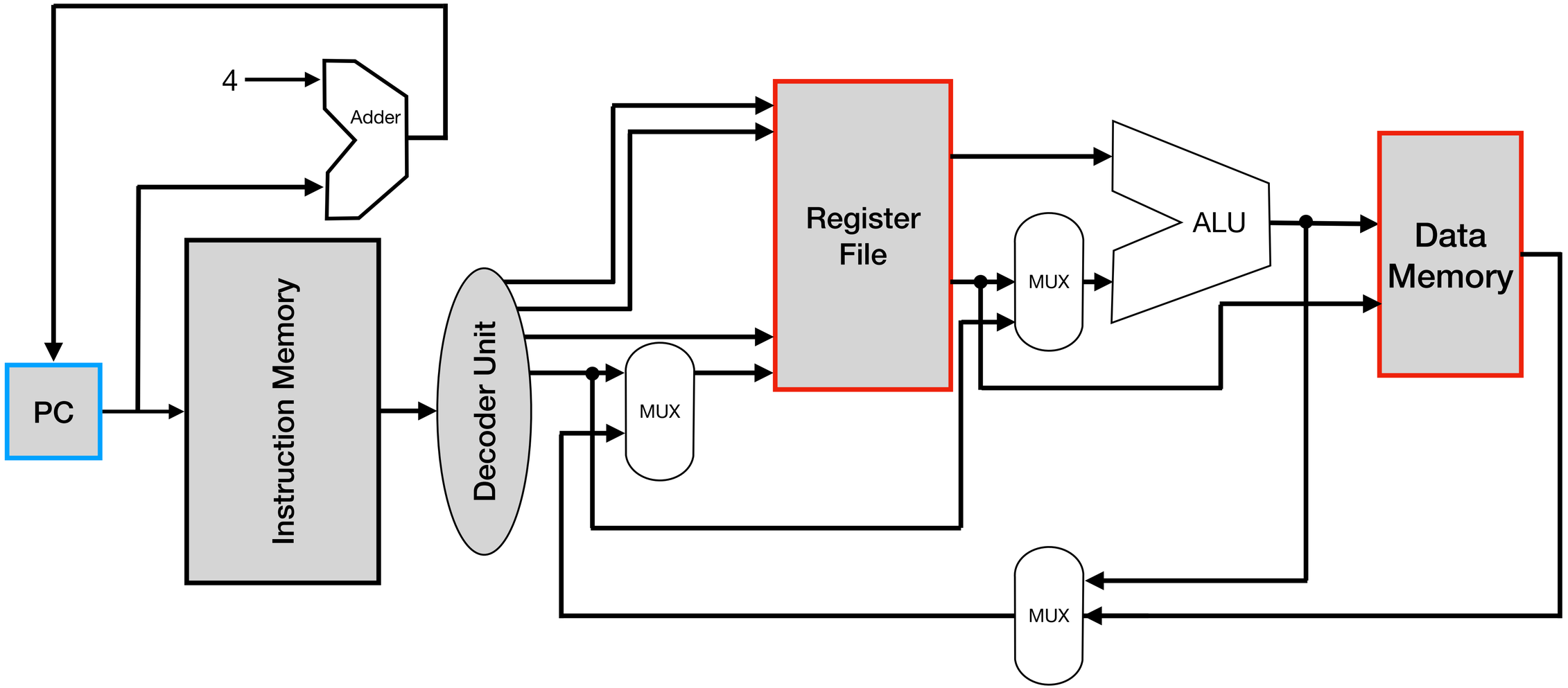
Processor Clock Revisited

- We need to move data along the data path so that it can start and end in a “state element” at each clock cycle
- We now know that “state elements” refer to memory and register-based components!





Processor Clock Revisited



An Instruction per Cycle

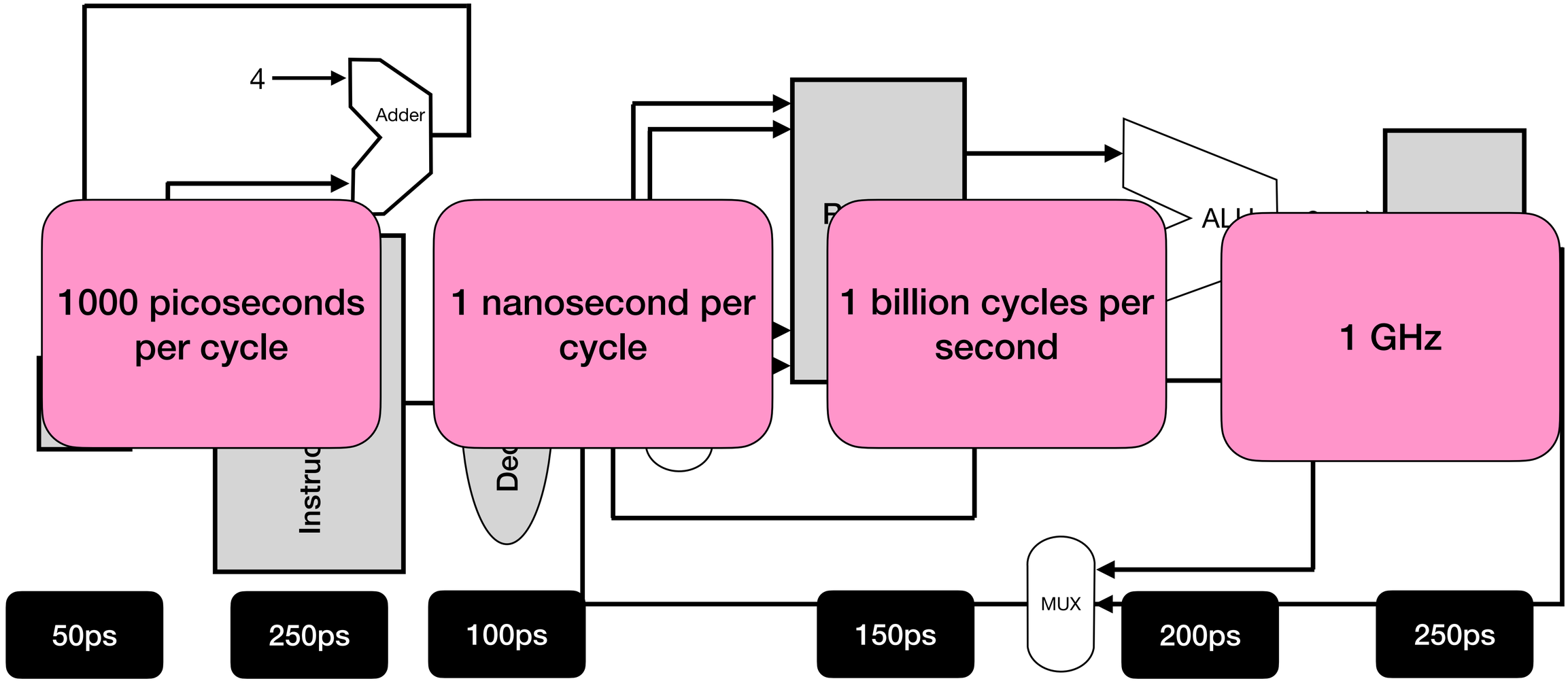
- If we want to execute an instruction every cycle, then we need to set our clock rate to account for the time it takes to move through each component in the data path

Clock Rate: the frequency of a clock cycle; measured in hertz (Hz)

Clock Speed: the time it takes to move through a clock cycle; measured in pico/nanoseconds (ps/ns)

- Clock cycle is determined by the rate/speed that each component takes to execute a step
- Overall clock cycle time: ∑ clock speed of each component

Example: Computing Clock Speed



Estimating Execution Times

- CPU execution time = Instruction Count * clock cycle time * cycles per instruction
- In a processor that executes “an instruction per clock cycle”, the CPI is 1 so CPU execution time = Instruction Count * clock cycle time
- All instructions take the same amount of time to execute!
- What happens if the majority of instructions are arithmetic? What happens if none depend on the ALU?

We will come back to this on Friday!

Takeaways

- Control signals can be inferred from decoded instructions
- The clock cycle dictates the flow of signals through the data path on a “step” granularity
- Clock cycle time is a function of the execution time of each component in the data path for an “instruction per cycle” processor