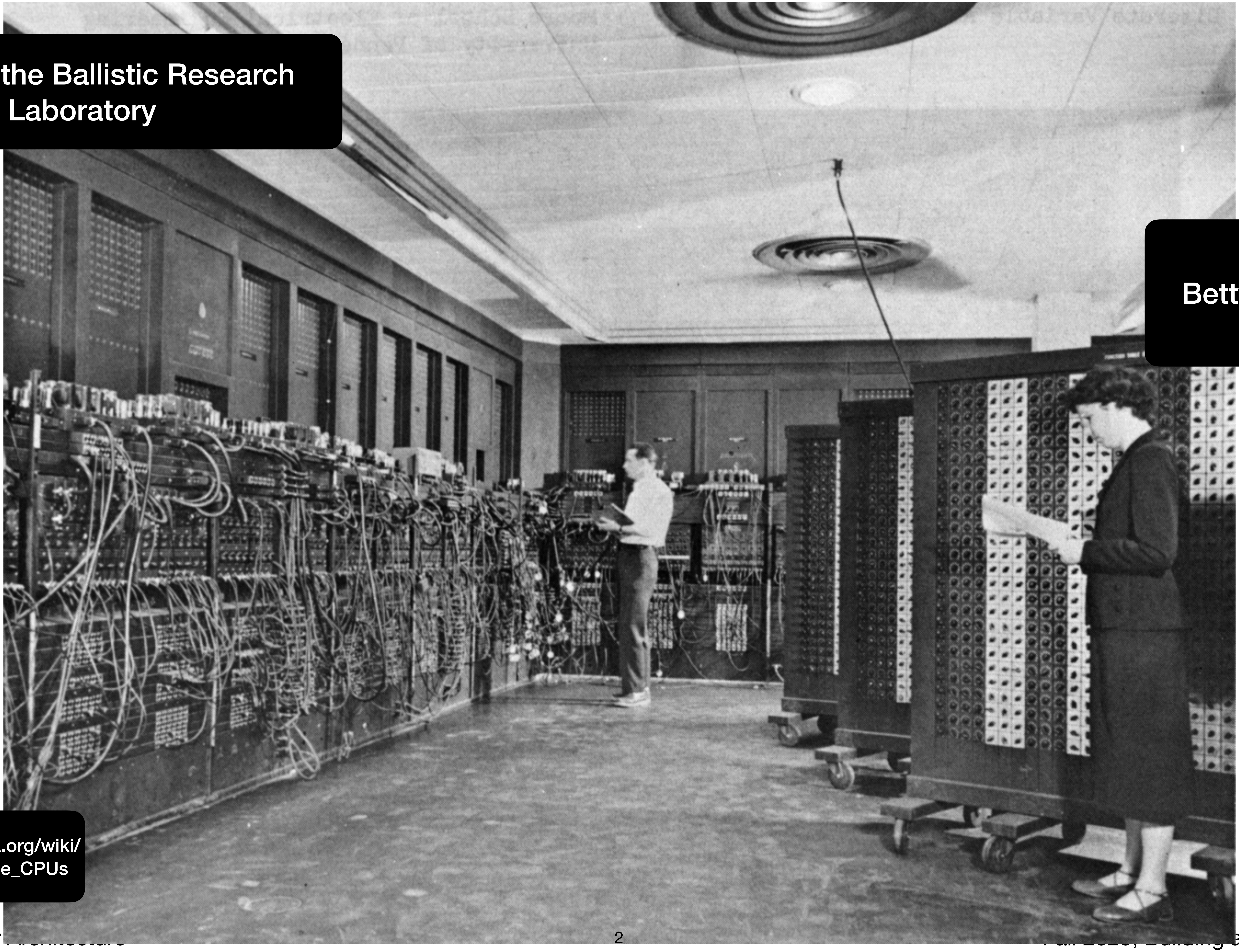
Building a Basic Processor: The Components



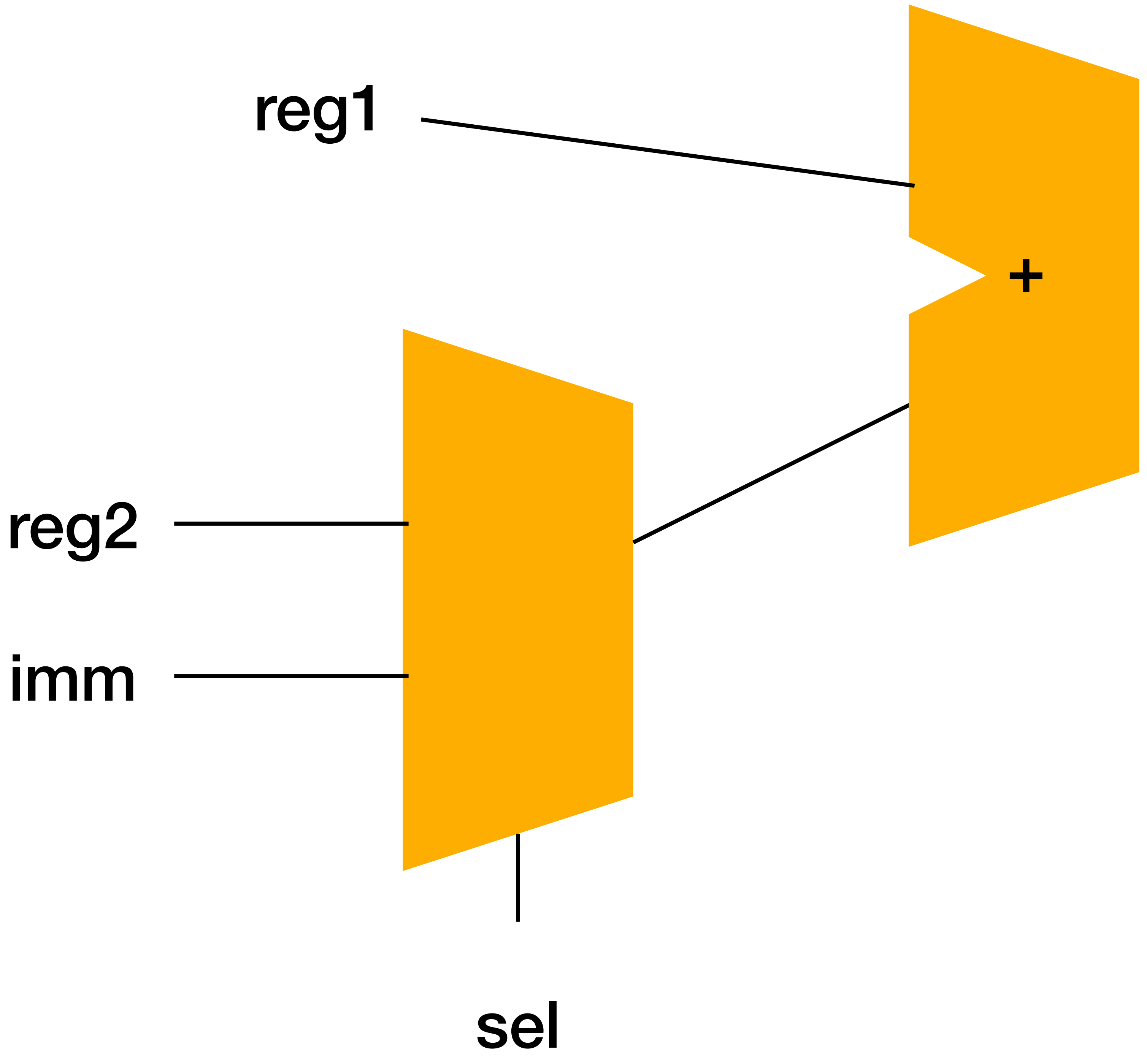
ENIAC at the Ballistic Research Laboratory

Betty Snyder

Source: https://en.wikipedia.org/wiki/History_of_general-purpose_CPUs

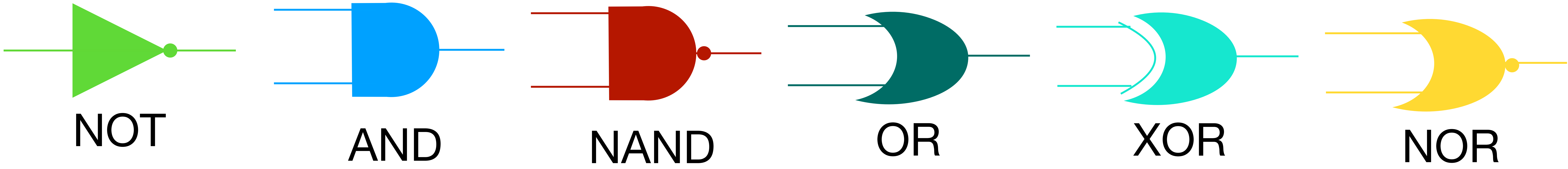
From Wednesday… Multiplexers

- “Hardware if-statements”
- Select between multiple inputs
- Example: choose the second operand for `add` and `addi`

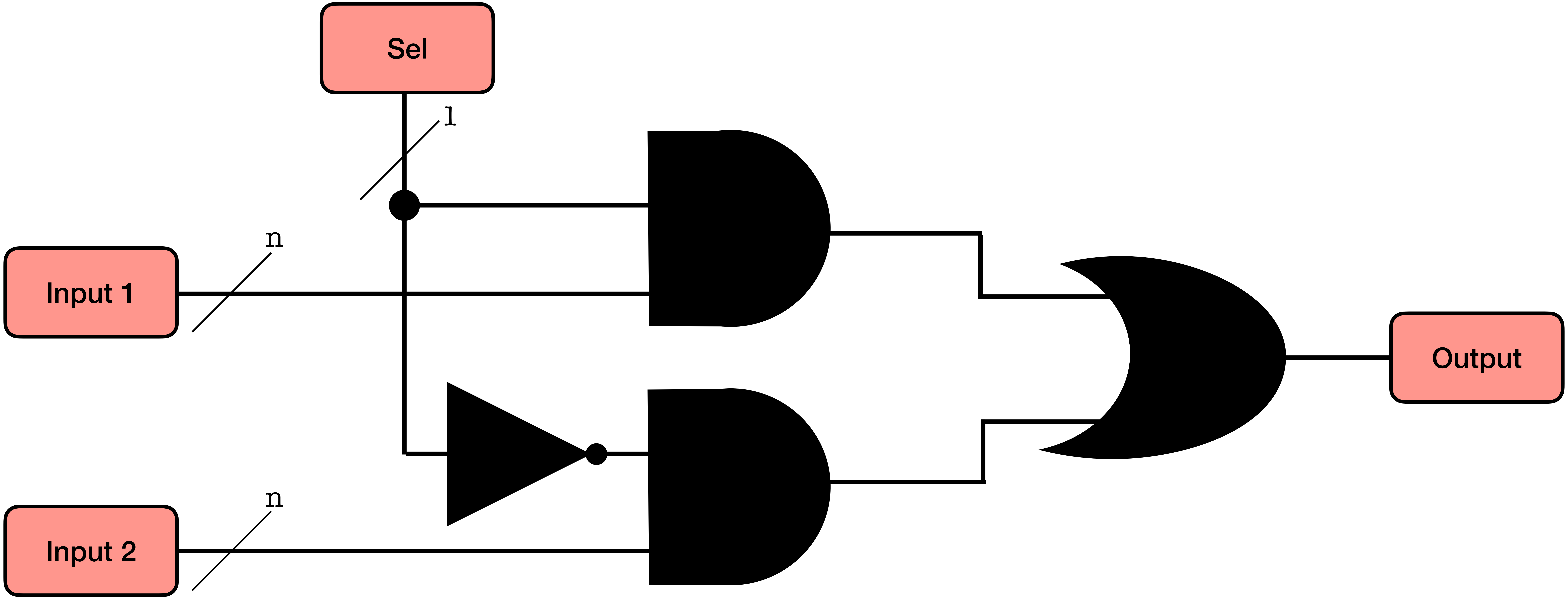


Chat with your neighbor(s)!

Build a two-input (1-bit selector) MUX out of logic gates!



Build a Two-Input (1-bit selector) MUX!



Outline

- Reintroducing the goals of a processor
- Overview of data path elements
- Constructing a data path

Processor Goals

- Track the location of what to execute
- Fetch instructions from memory
- Interpret instructions
- Execute instructions
- Maintain and update state

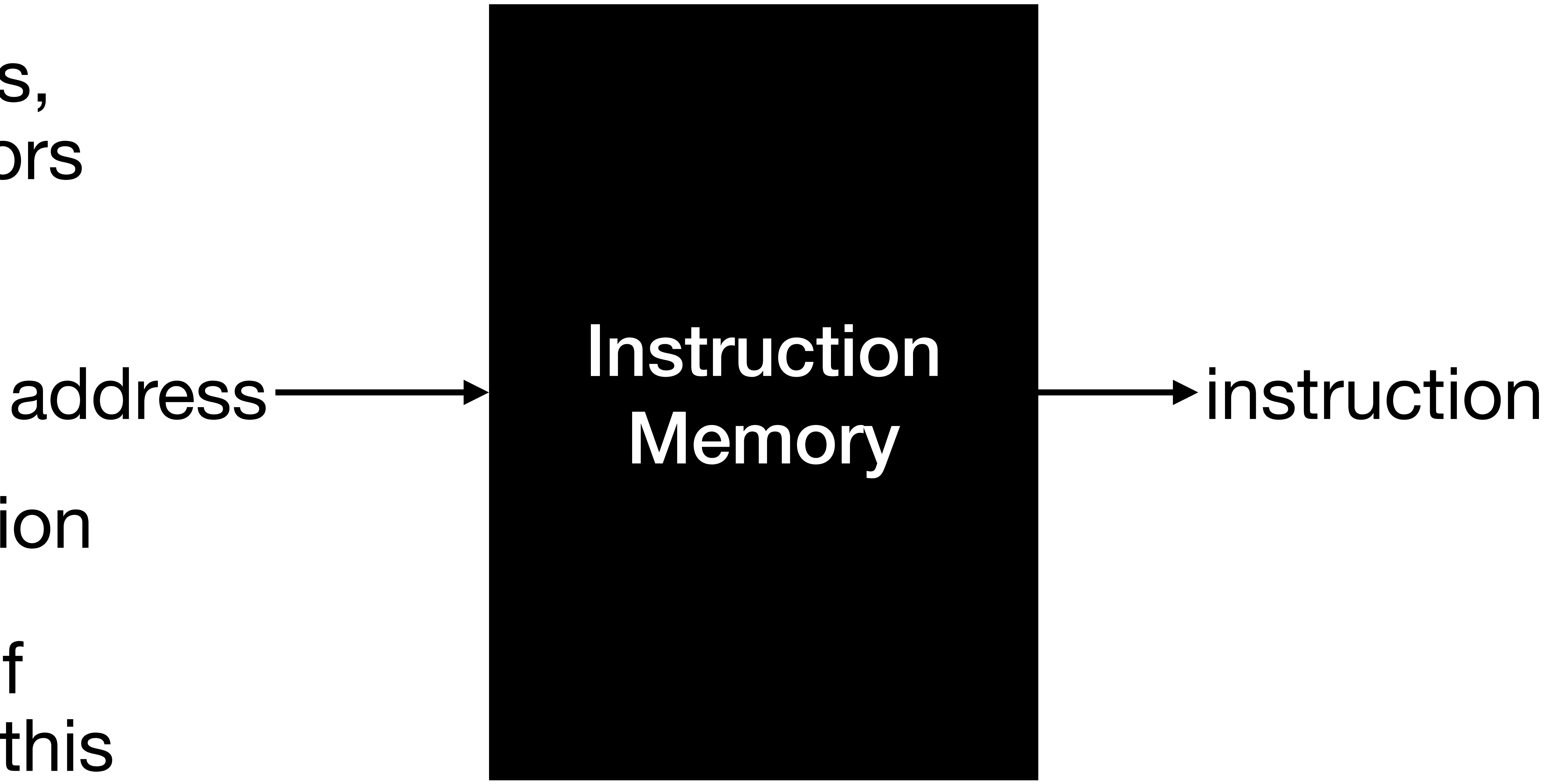
We now have instructions and methodology to do these operations! Referred to as a “decoder” unit

Program Counter (PC)

- After each instruction, we need to know what to fetch next
- The *program counter* is a special purpose register (maintains state) to tell the processor which instruction to fetch next
- How does the processor know what the next PC state should be?
- Demo!

Instruction Memory

- Just like combinational logic units, memory is composed of transistors (and capacitors)!
- Takes address as input
- Returns raw bytes of the instruction
- Over simplification for the sake of basic processor! We will explore this in more detail next module



Processor Goals: Fetching Instructions

Fetch *n* bytes from instruction memory!

0xfec30313

Increment the program counter by *n*!

Both of the PC and Instruction Memory are storage components!

13	03	c3	fe	00	00	00	ef	be	ad	de	ee	ff	c0
0xff00	0xff01	0xff02	0xff03	0xff04	0xff05	0xff06	0xff07	0xff08	0xff09	0xff0a	0xff0b	0xff0c	0xff0d

Instruction Memory

Processor Goals: Execute Instructions

- What components do we need to execute instructions? Remember how our instructions can be defined!
- Instructions refer to register/memory locations where data should be stored
- Instructions refer to some functionality to manipulate data

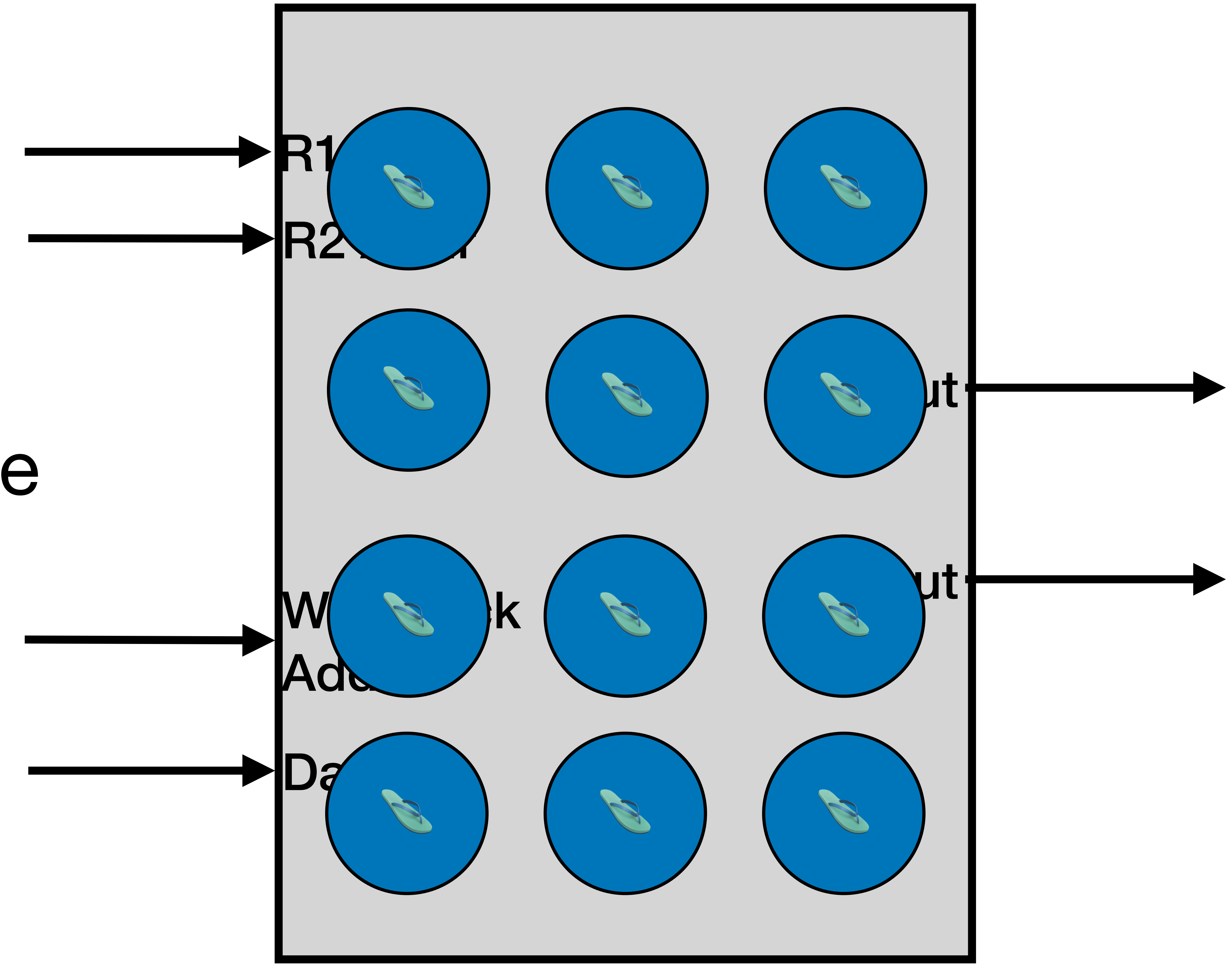
Data Transfers

Computations

Control Logic

Register File

- Takes in four inputs: register addresses to “get” (R1 and R2), any data to write, and the destination it should be written to
- Registers composed of flip flops to store the data
- Reading state happens on the “rising edge” of the clock cycle whereas updates to state happen on the “falling edge” of the clock cycle



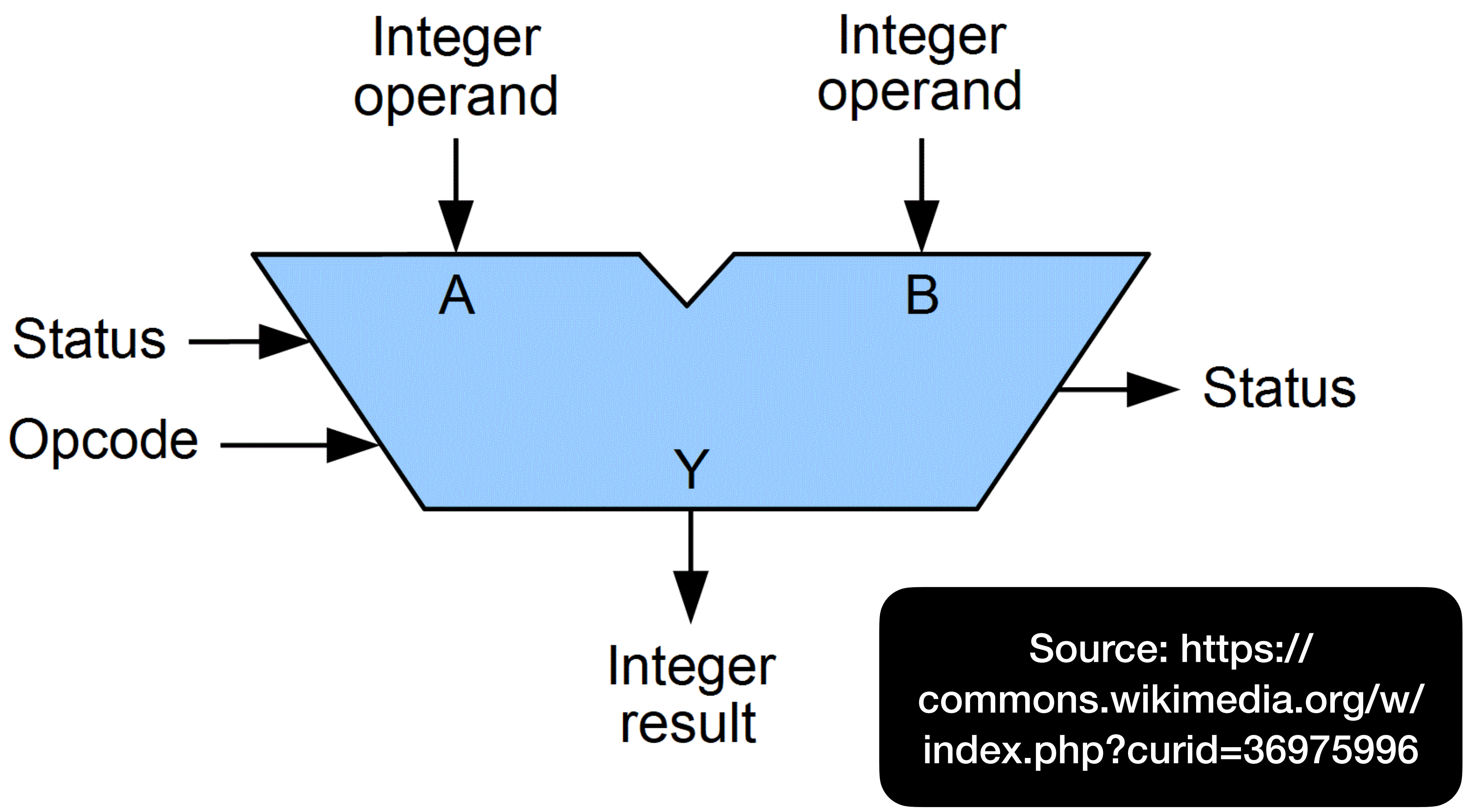
Register File

Chat with your neighbor(s)!

Suppose a processor is executing the instruction `add x1, x1, x1`. How will this be interpreted by the register file?

Computational Units

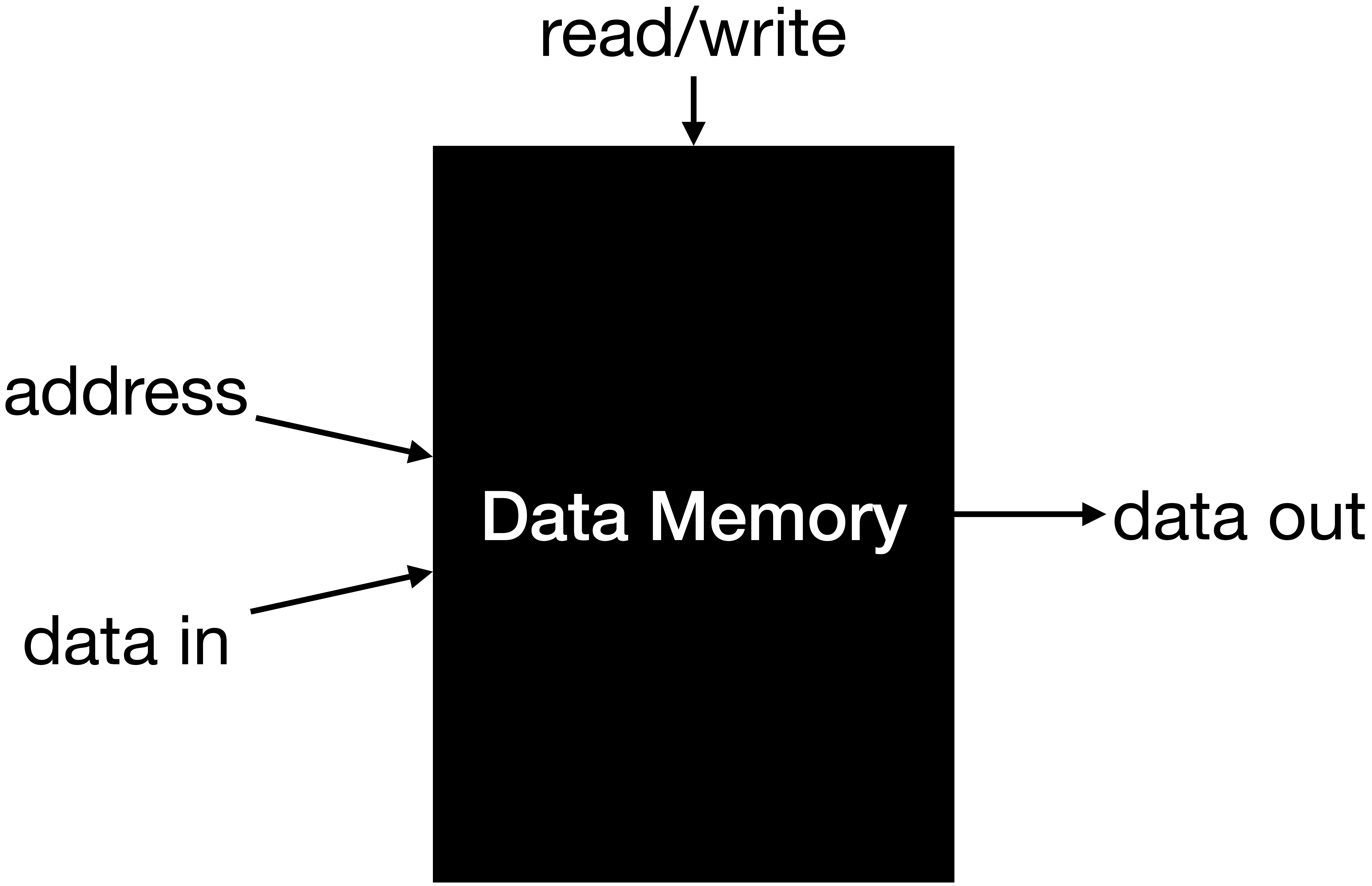
- Arithmetic Logic Unit (ALU): a large combinational logic unit for arithmetic operations
- Integer operands (inputs) passed to the ALU with values to operate on
- Takes “opcode” as input to determine which functionality to use where functionality is implemented as a “pure function” from combinational logic
- Produces integer result; status output used to detect overflows, exceptions, etc.



Source: https://commons.wikimedia.org/w/index.php?curid=36975996

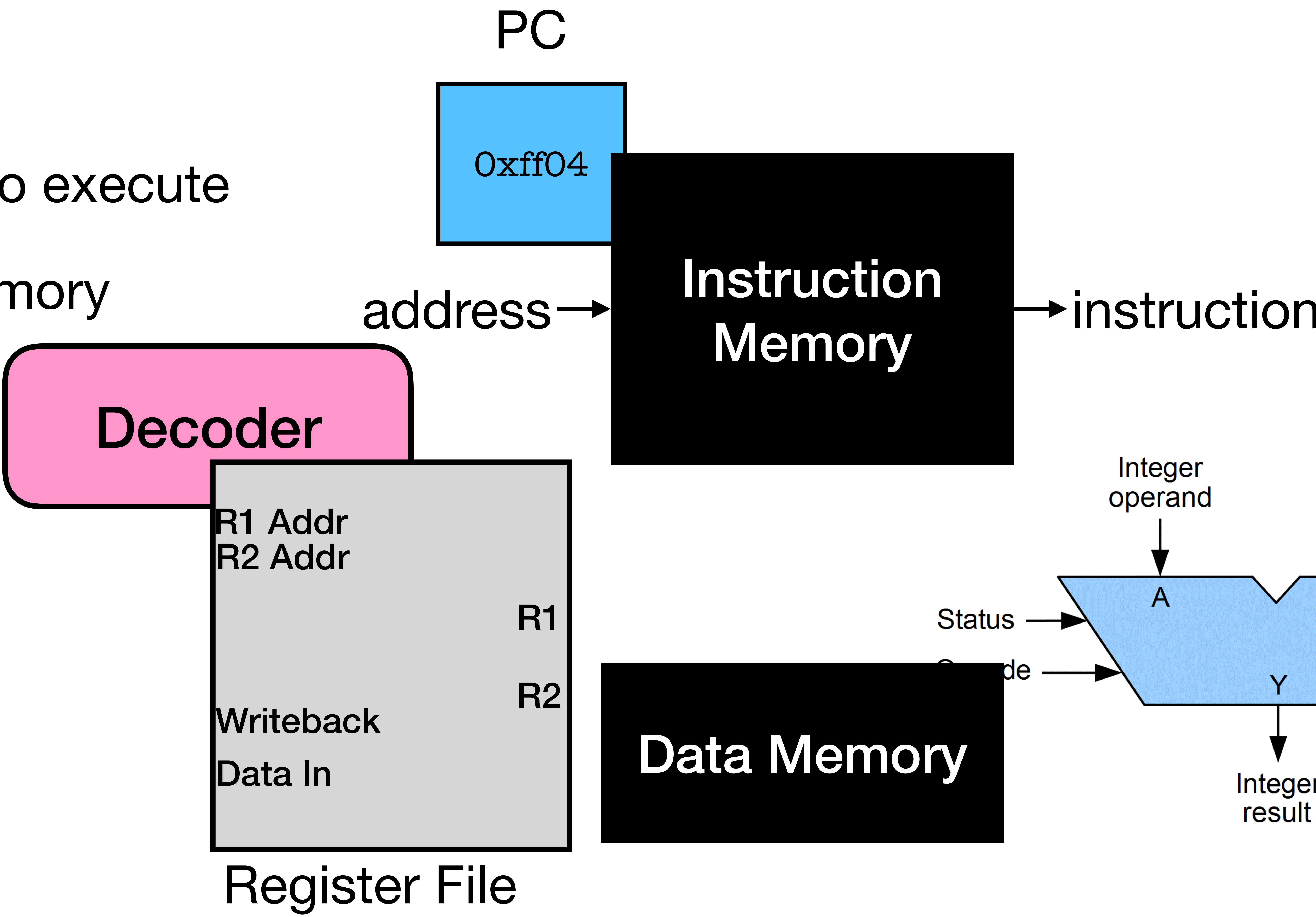
Data Memory

- Much like instruction memory, data memory is composed of transistors/capacitors
- Data may be read or written, so there is an additional control signal that this component uses as input
- Data out only set if input signal is set to read
- Data in only used if input signal is set to write



Takeaways

- Track the location of what to execute
- Fetch instructions from memory
- Interpret instructions
- Execute instructions
- Maintain and update state



Exit Ticket!

https://forms.cloud.microsoft/r/jZpZnu9Mas