Announcements

- Friday, there is a _ _ _ term _x_ _

Explaining What's Happening So It Makes Sense

Quick Return To … "Explaining Why Your Algorithm Works"

Lawrence Snyder
University of Washington, Seattle

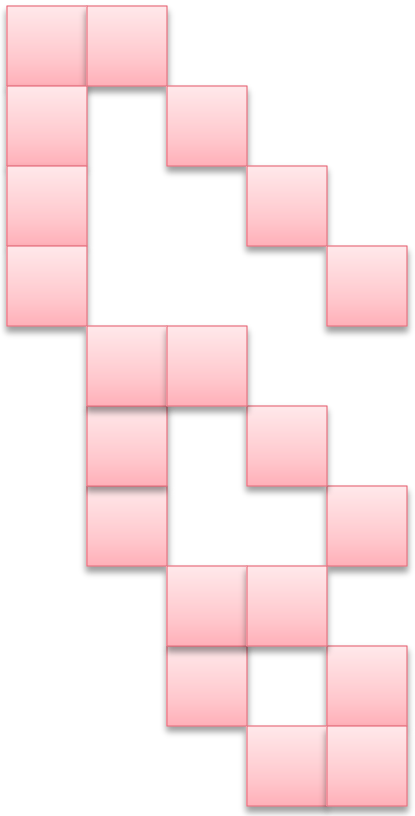
Know Why The Algorithm Works

- Last time we discussed sorting algorithms,
  - Algorithms that put words and numbers in order
  - We gave 3 algorithms
    - Exchange sort
    - Bubble sort
    - Merge sort
  - For all 3, we said **why the algorithm works** …

Give me a chance to try it one more time … to get it right!

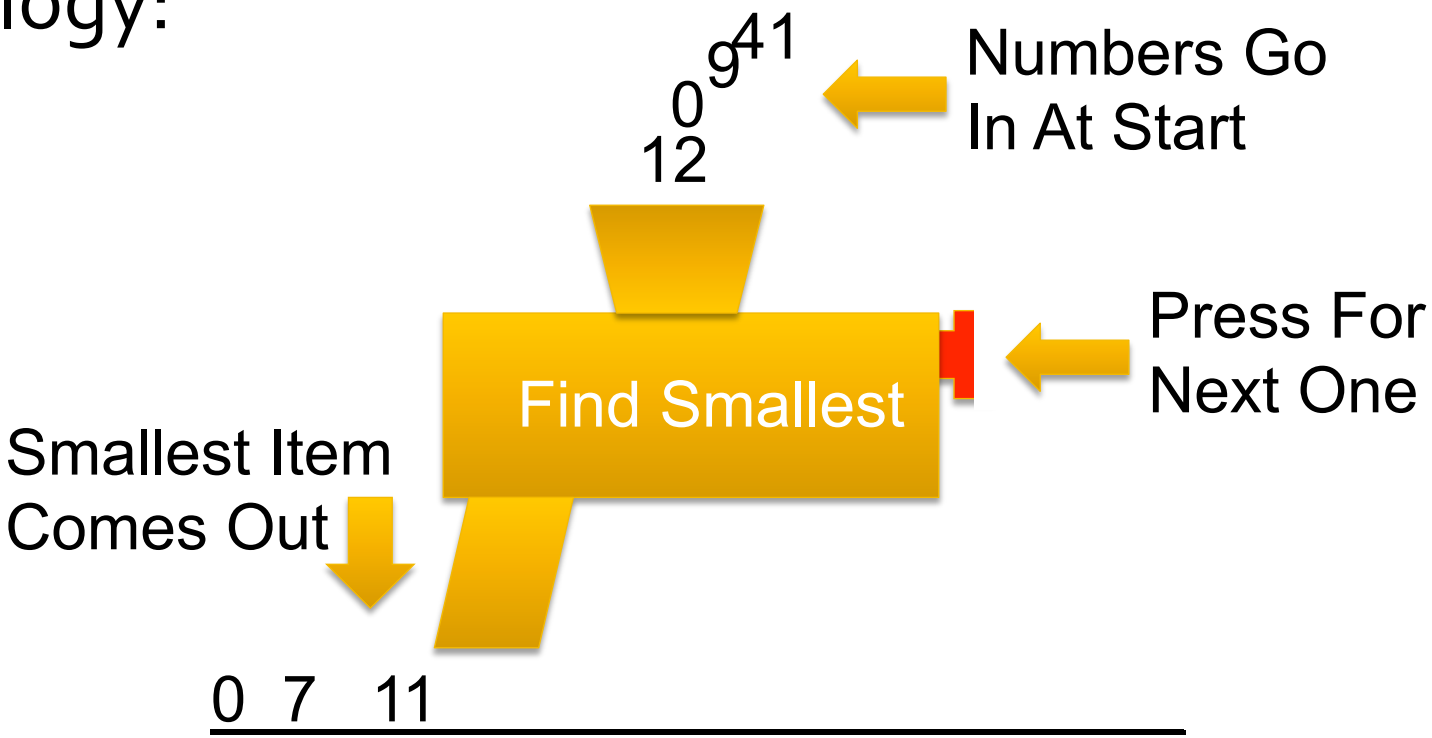
Explaining Why Algorithm Works

- Say What You're Claiming: "Exchange Sort Puts Numbers or Words in Ascending Order"
- It's not automatically obvious



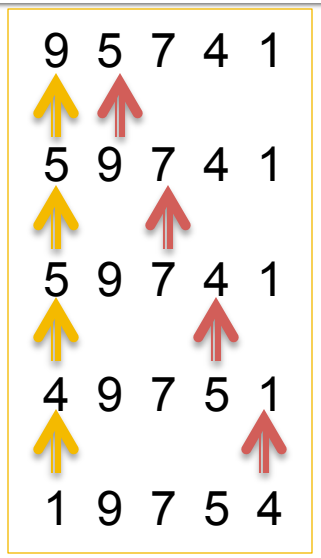
Explaining Why Algorithm Works

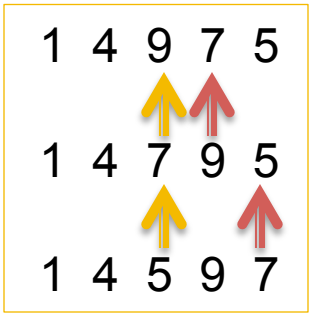
- Say What You're Claiming: "Exchange Sort Puts Numbers or Words in Ascending Order"
- Explain how to do it "big picture-wise" … use an analogy:

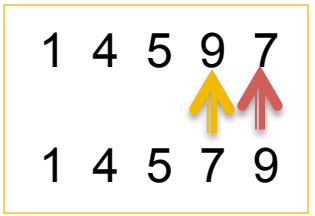


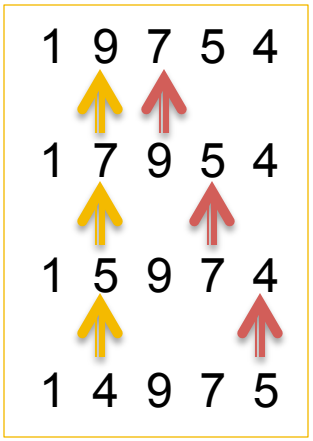
Explain Why Code Works That Way

- If Minimizer Sorts, So Does Exchange Sort
- The Operation of Exchange Sort Works Like The Minimizer … each pass "emits" the smallest item



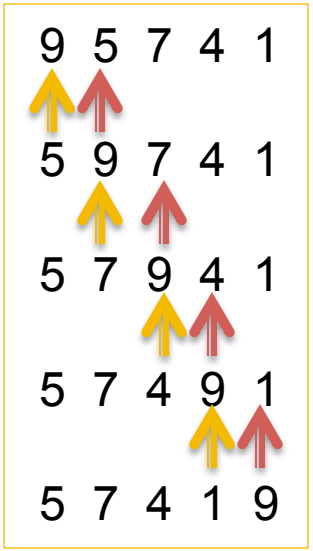


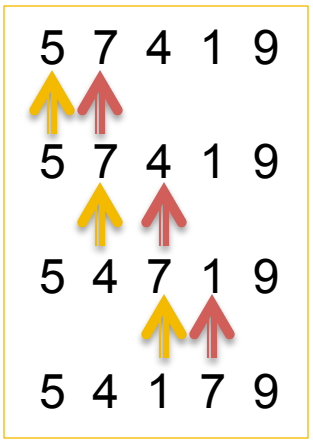


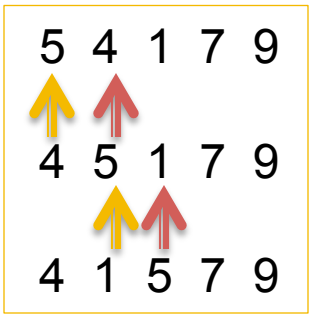


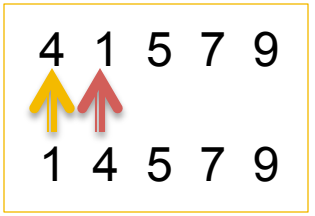
How About Bubble Sort?

- Recall, Bubble Sort "pushes" the largest as far as possible to the right



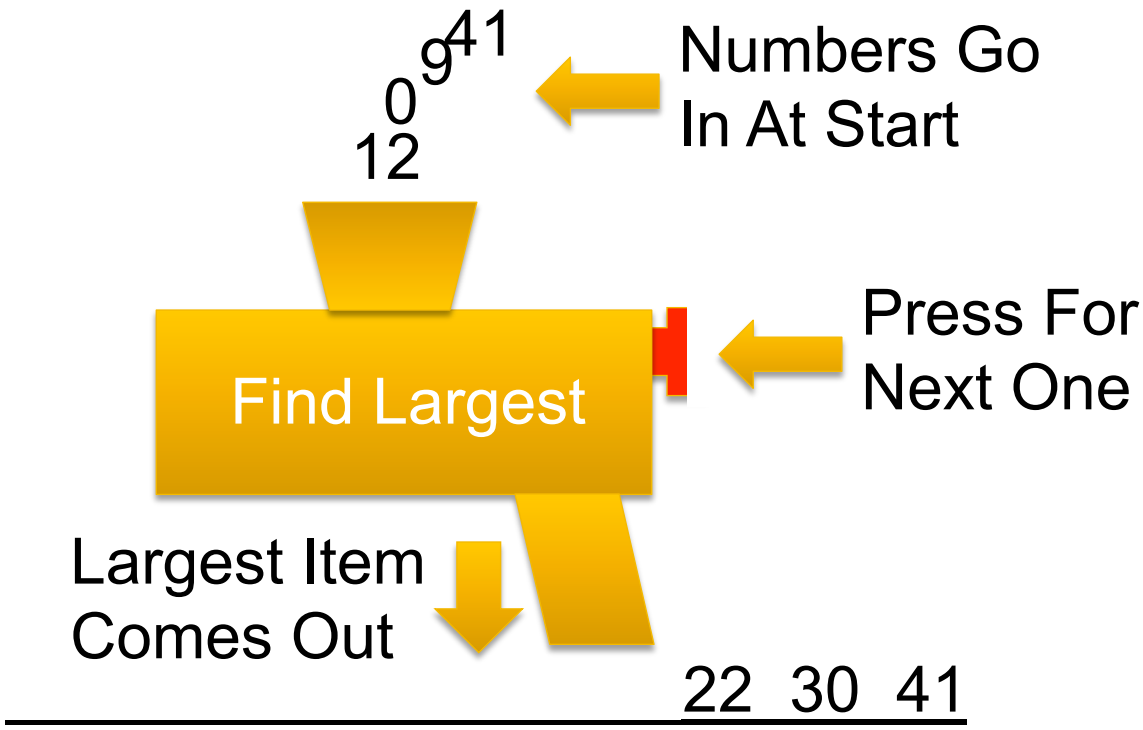






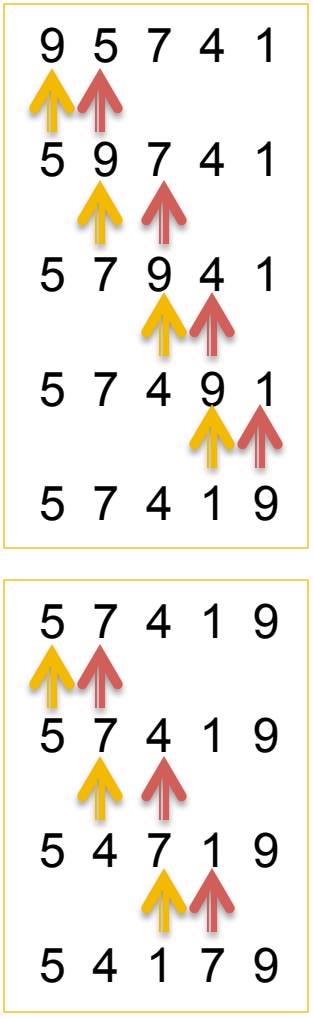
Explaining Why Algorithm Works

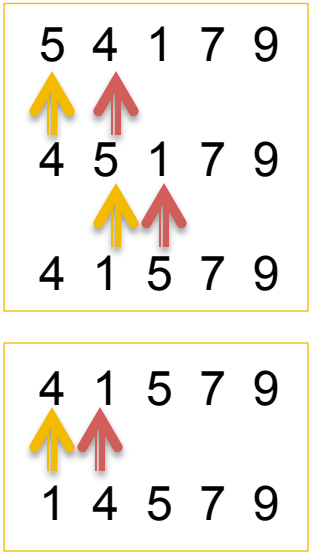
- Say What You're Claiming: "Bubble Sort Puts Numbers or Words in Ascending Order"
- Explain how to do it "big picture-wise" … use an analogy:



How About Bubble Sort?

- If The Maximizer Sorts then so does Bubble sort … the operation works like the maxi-mizer





Remember Back To The Lightbot

Instruction Execution is … So Simple Even A Computer Can Do It

Lawrence Snyder
University of Washington, Seattle

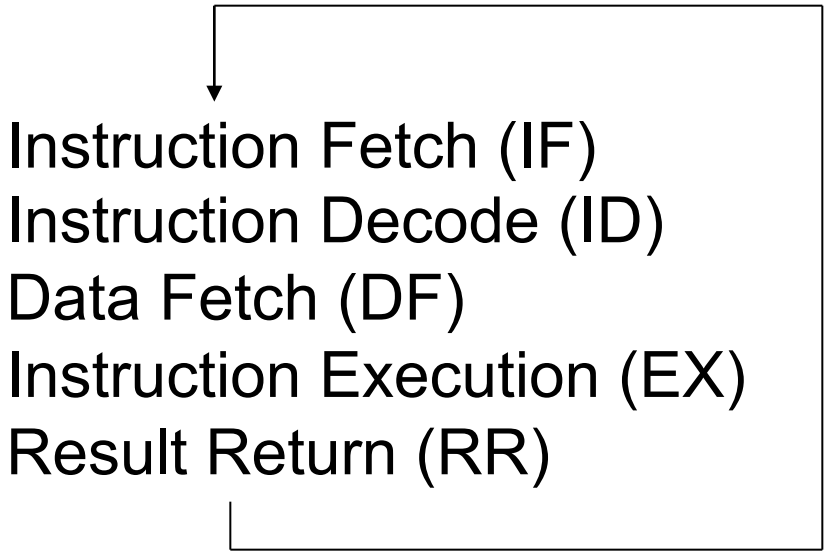
Computers …

- Deterministically execute instructions to process information
  - **"Deterministically" means that when a computer chooses the next instruction to perform it is required by its construction to execute a specific instruction based only on the program and input it is given**

Computers have no free will and they are not cruel

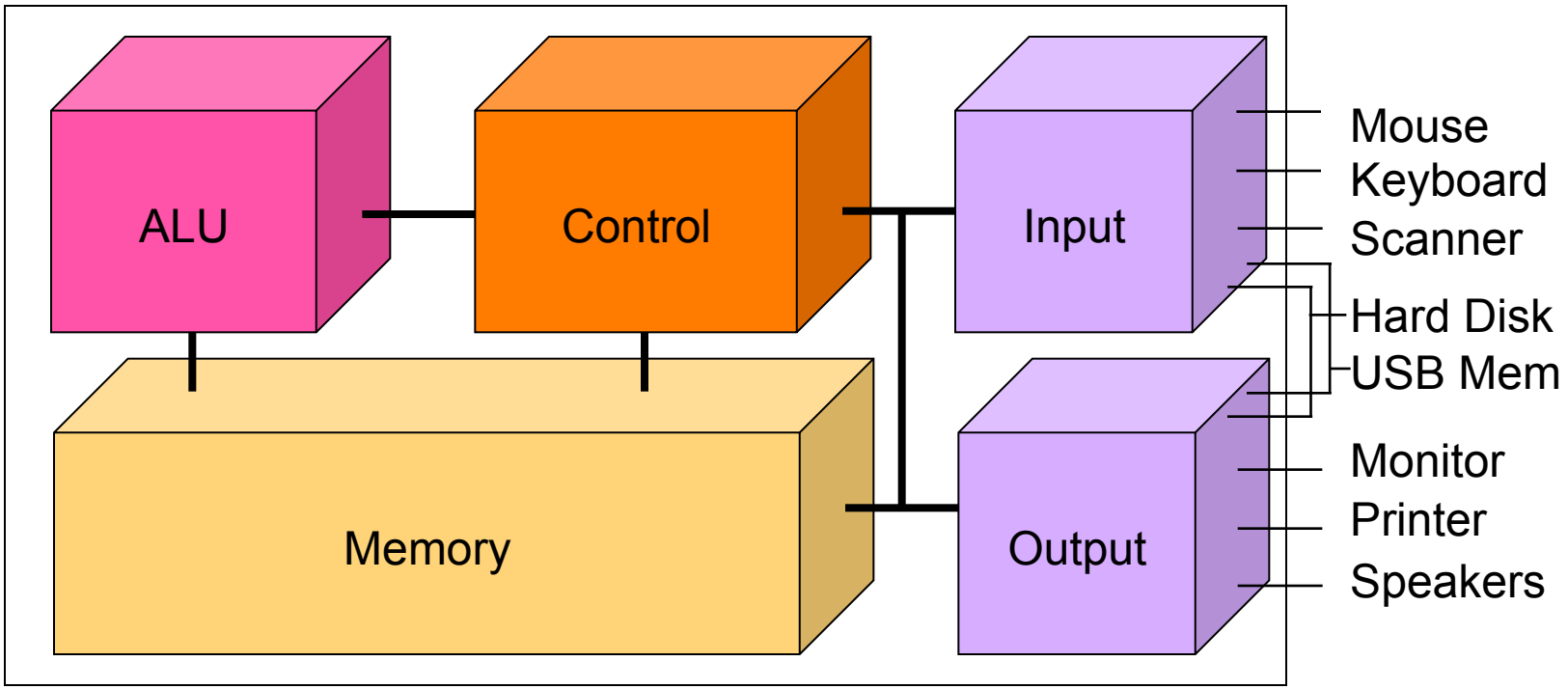
Fetch/Execute Cycle

- Computer = instruction execution engine
  - **The fetch/execute cycle is the process that executes instructions**



- The computer internal parts implement this cycle

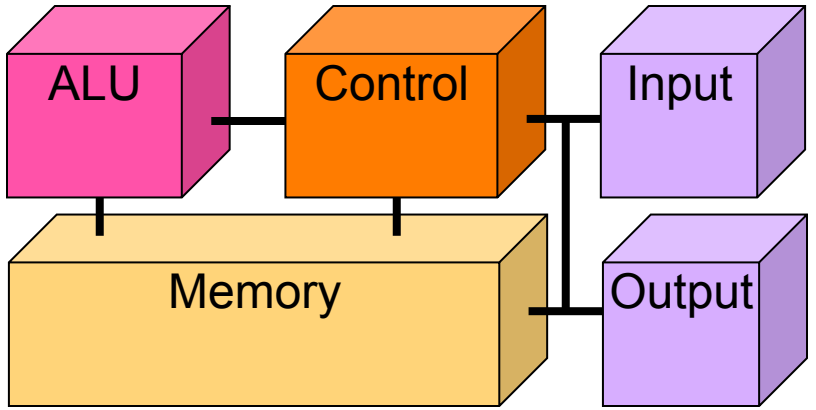
Anatomy of a Computer: The CPU

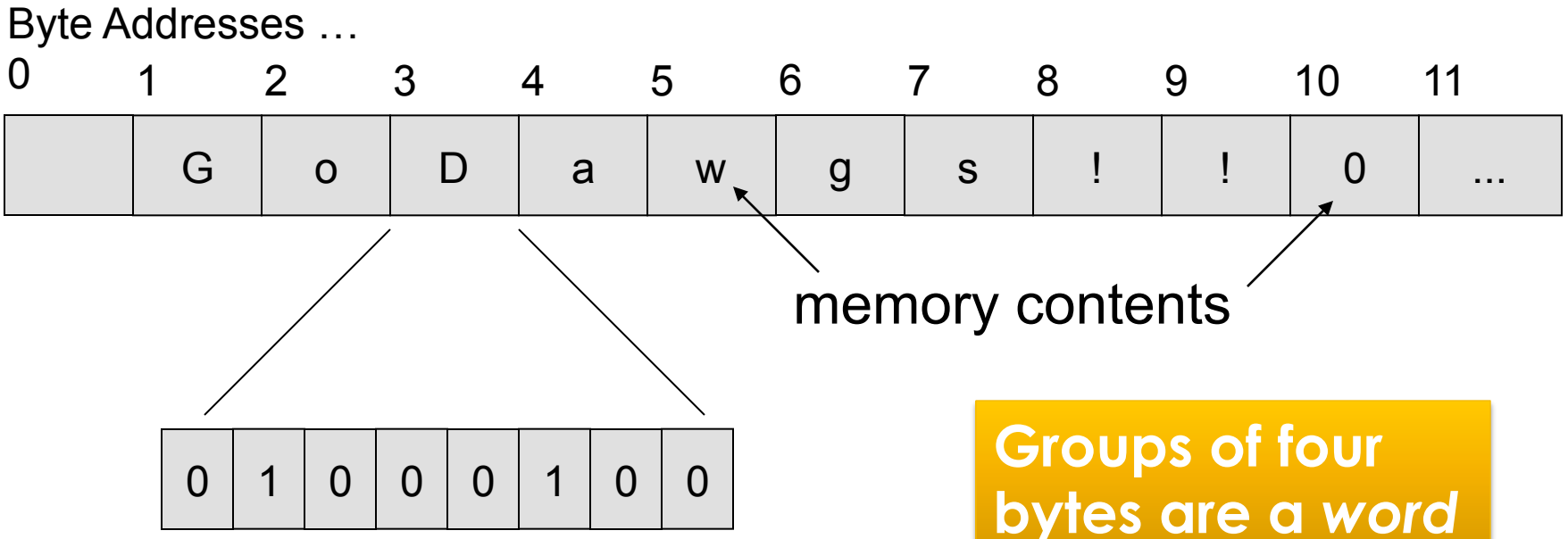


The Hard Disk is the α-device

Memory …

- Programs and their data must be in the memory while they are running

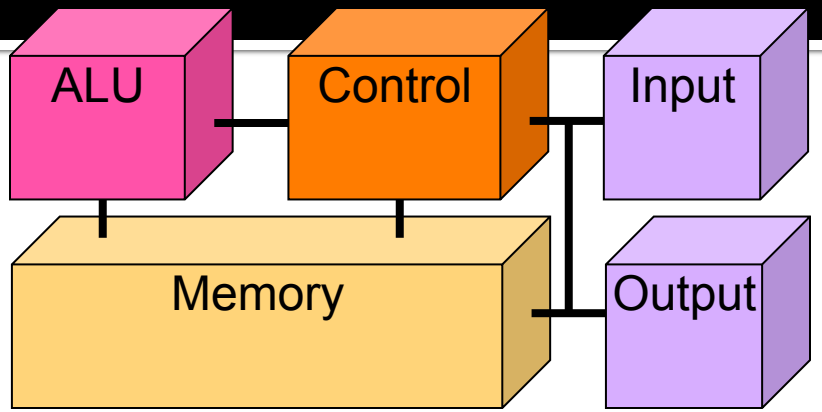




Groups of four bytes are a *word*

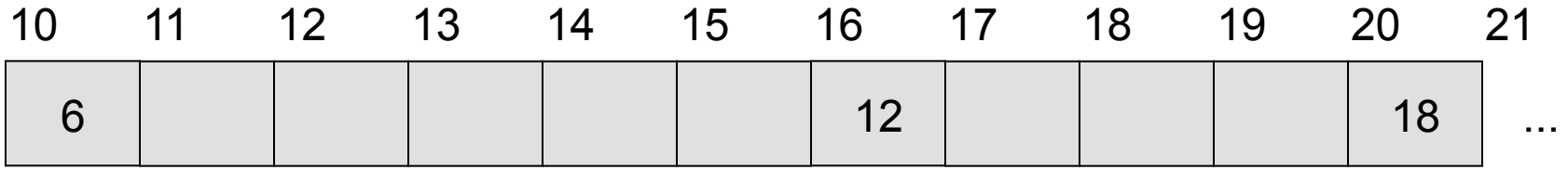
Control

- Fetch/Execute cycle is hardwired in computer's control; it's the "engine"



The instructions executed have the form
ADDB 20, 10, 16

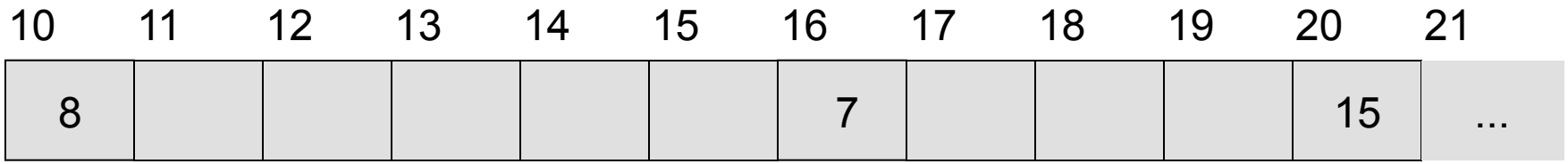
Put in memory location 20 the contents of memory location 10 + contents of memory location 16



Indirect Data Reference

- Instructions tell *where* the data is, not *what* the data is … contents change

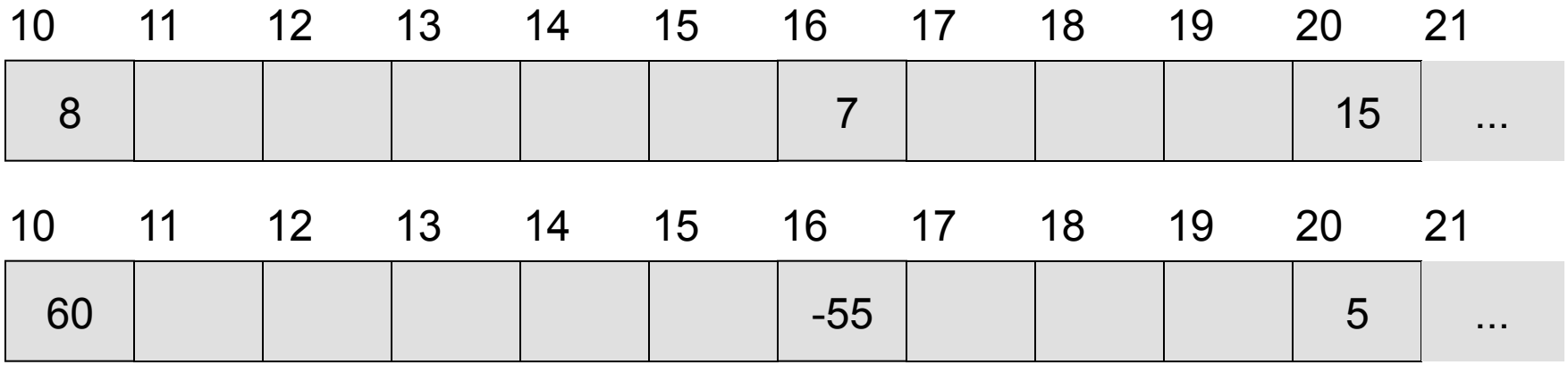
One instruction has many effects
ADDB 20, 10, 16



Indirect Data Reference

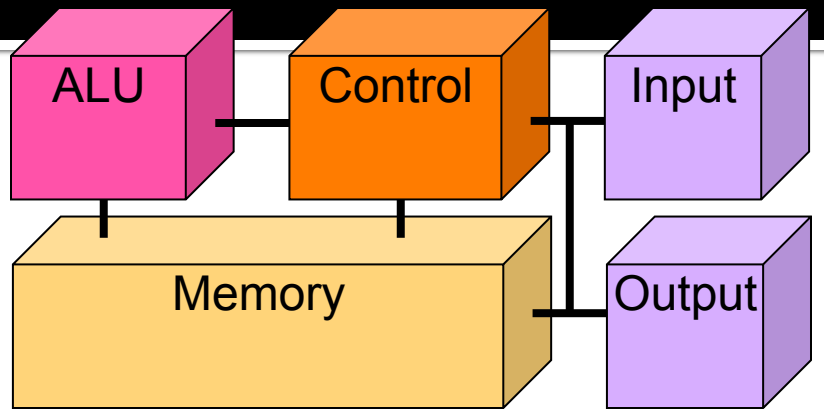
- Instructions tell *where* the data is, not *what* the data is … contents change

One instruction has many effects
ADDB 20, 10, 16



ALU

- Arithmetic/Logic Unit does the actual computing

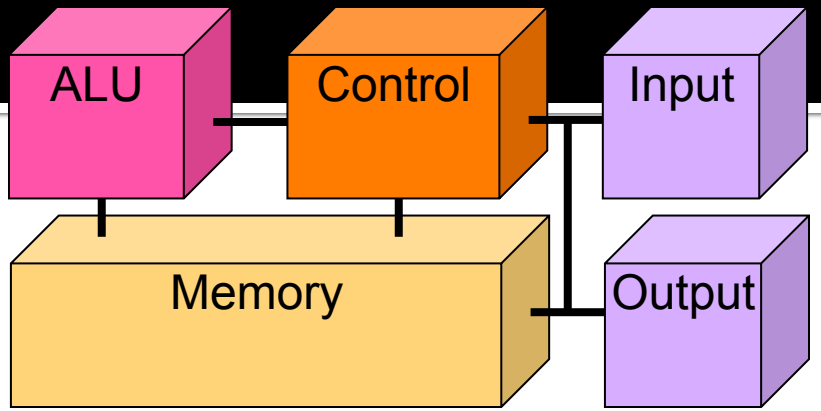


Each type of data has its own separate instructions

ADDB	: add bytes	ADDBU	: add bytes unsigned
ADDH	: add half words	ADDHU	: add halves unsigned
ADD	: add words	ADDU	: add words unsigned
ADDS	: add short decimal numbers		
ADDD	: add long decimal numbers		

Most computers have only about a 100-150 instructions hard wired

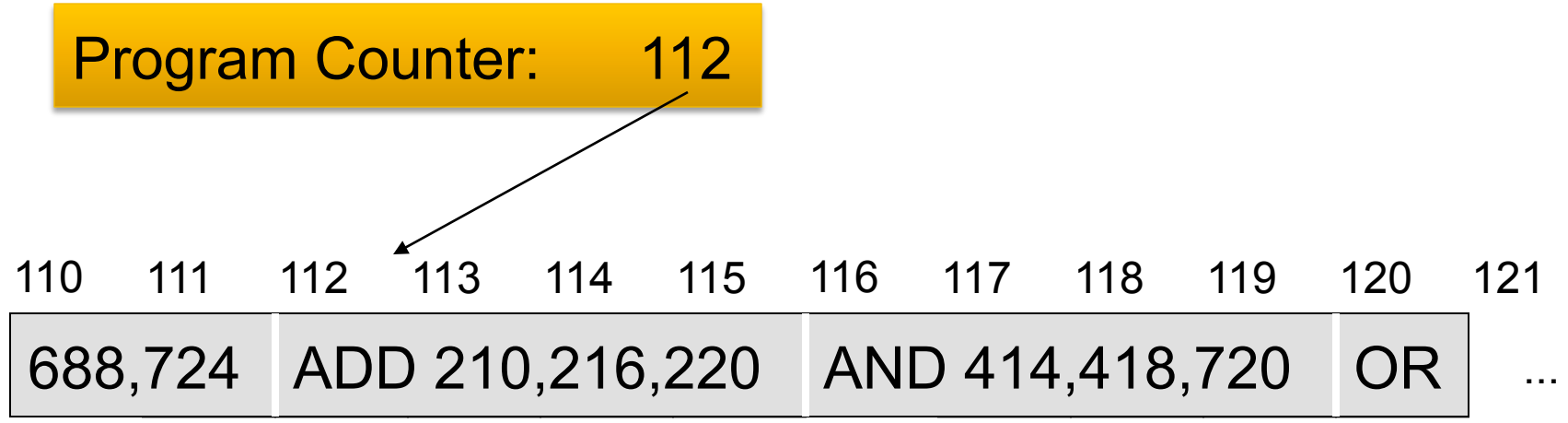
Input/Output



- Input units bring data to memory from outside world; output units send data to outside world from memory
  - Most peripheral devices are "dumb" meaning that the processor assists in their operation
  - Disks are *memory* devices because they can output information and input it back again

The PC's PC

- The program counter (PC) tells where the next instruction comes from
  - **Instructions are a word long, so add 4 to the PC to find the next instruction**

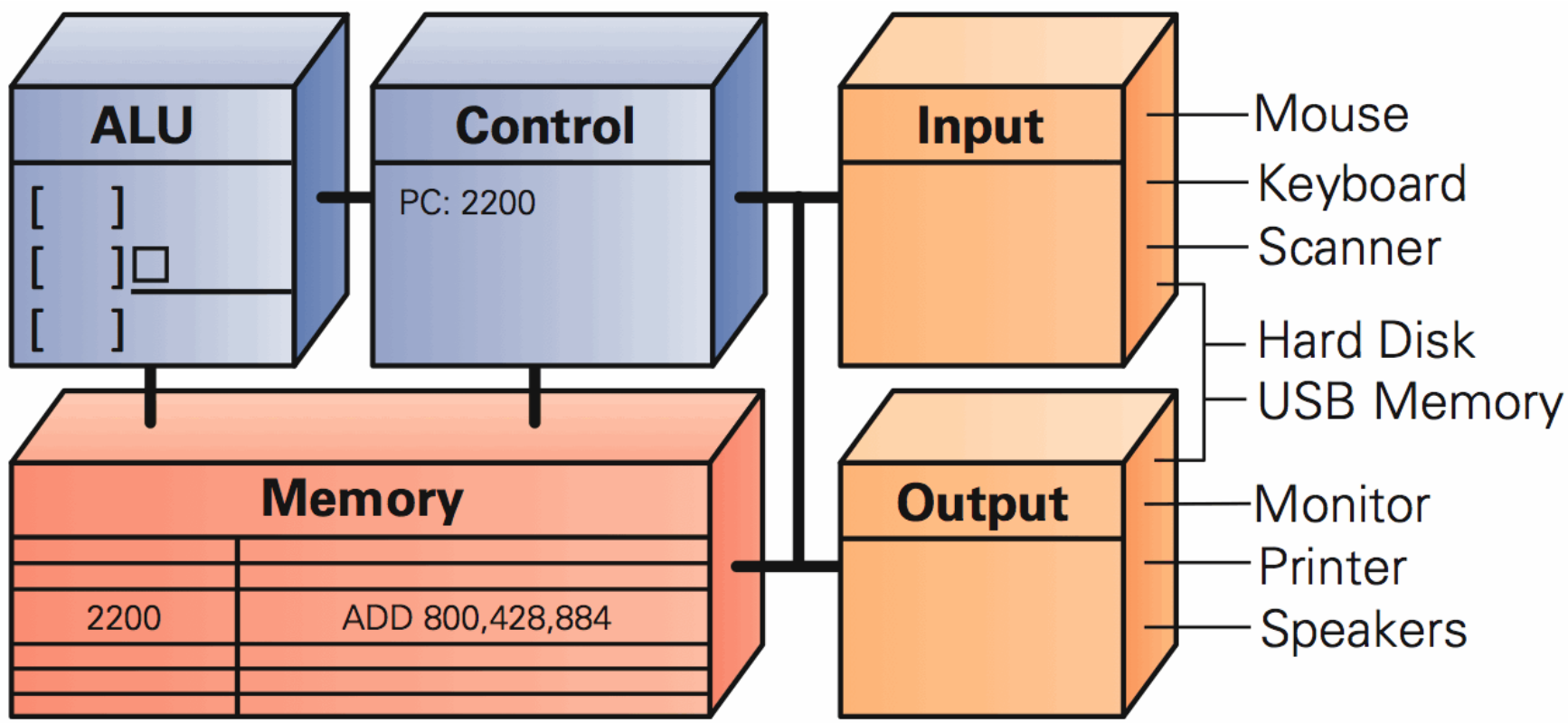


Suppose You Write a Program

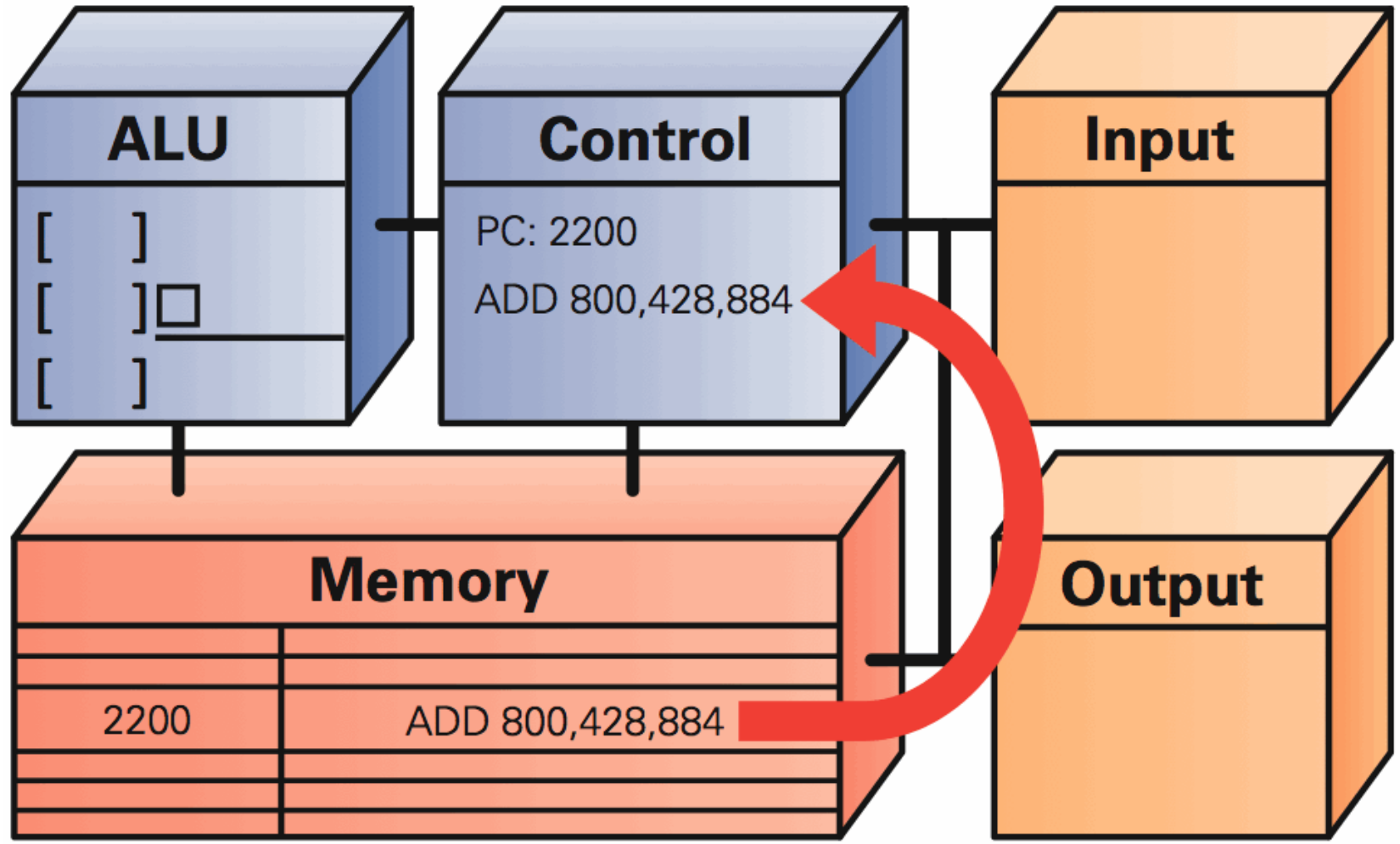
- In your program you write x = y + z;
- What happens?
  - A compiler translates the program written in the **high level language** (== Processing, Java, C, C#) into assembly language
  - **Assembly language** is a symbolic form of a computer's binary code
  - An **assembler translates** assembly code to binary
  - The OS places the **binary code** into the computer's memory, and begins performing the first instruction. Eventually, it gets to **ADD 800,428,884**

Instruction Execution: The Setup

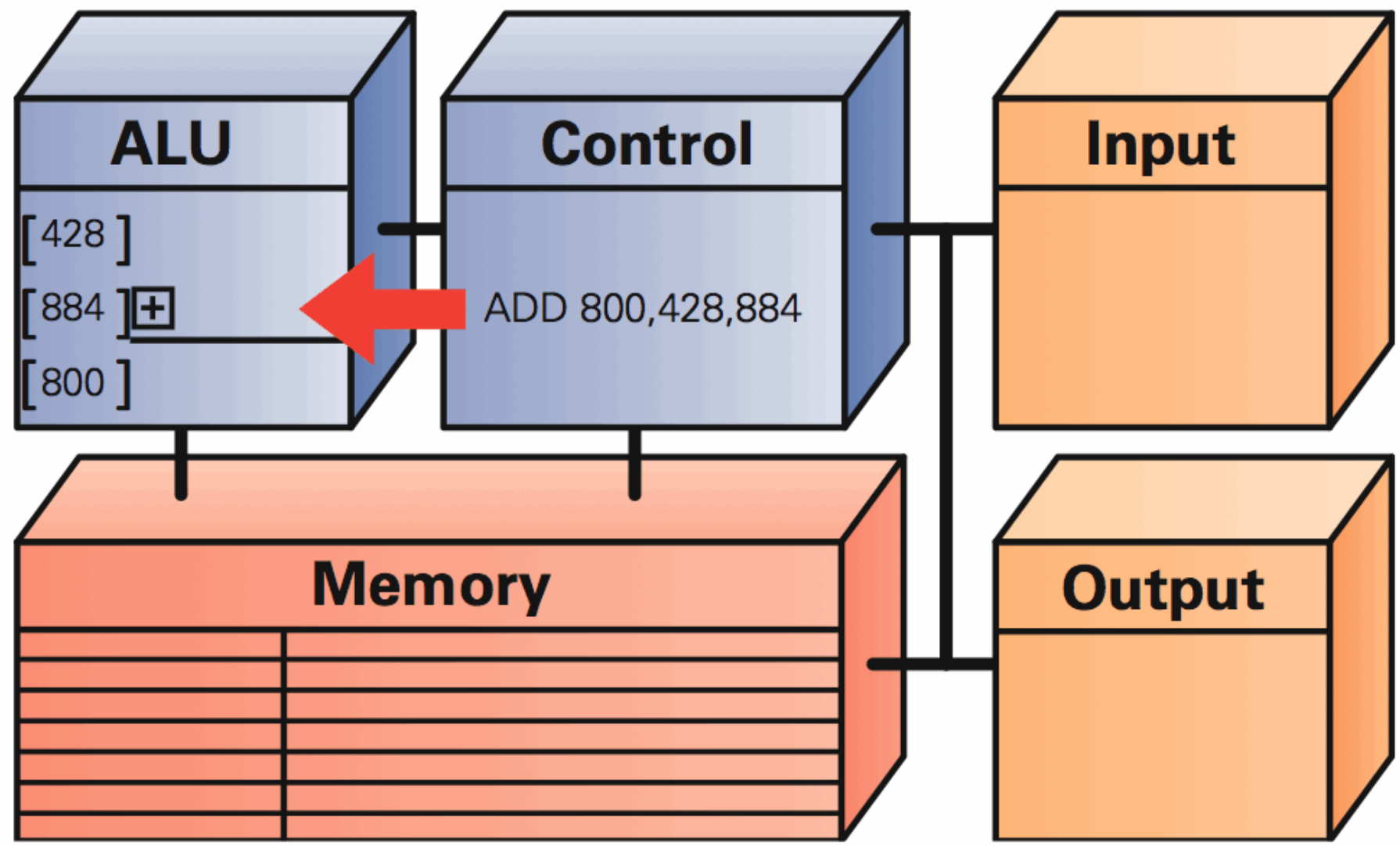
Run Instruction: 2200: Add 800, 428, 884



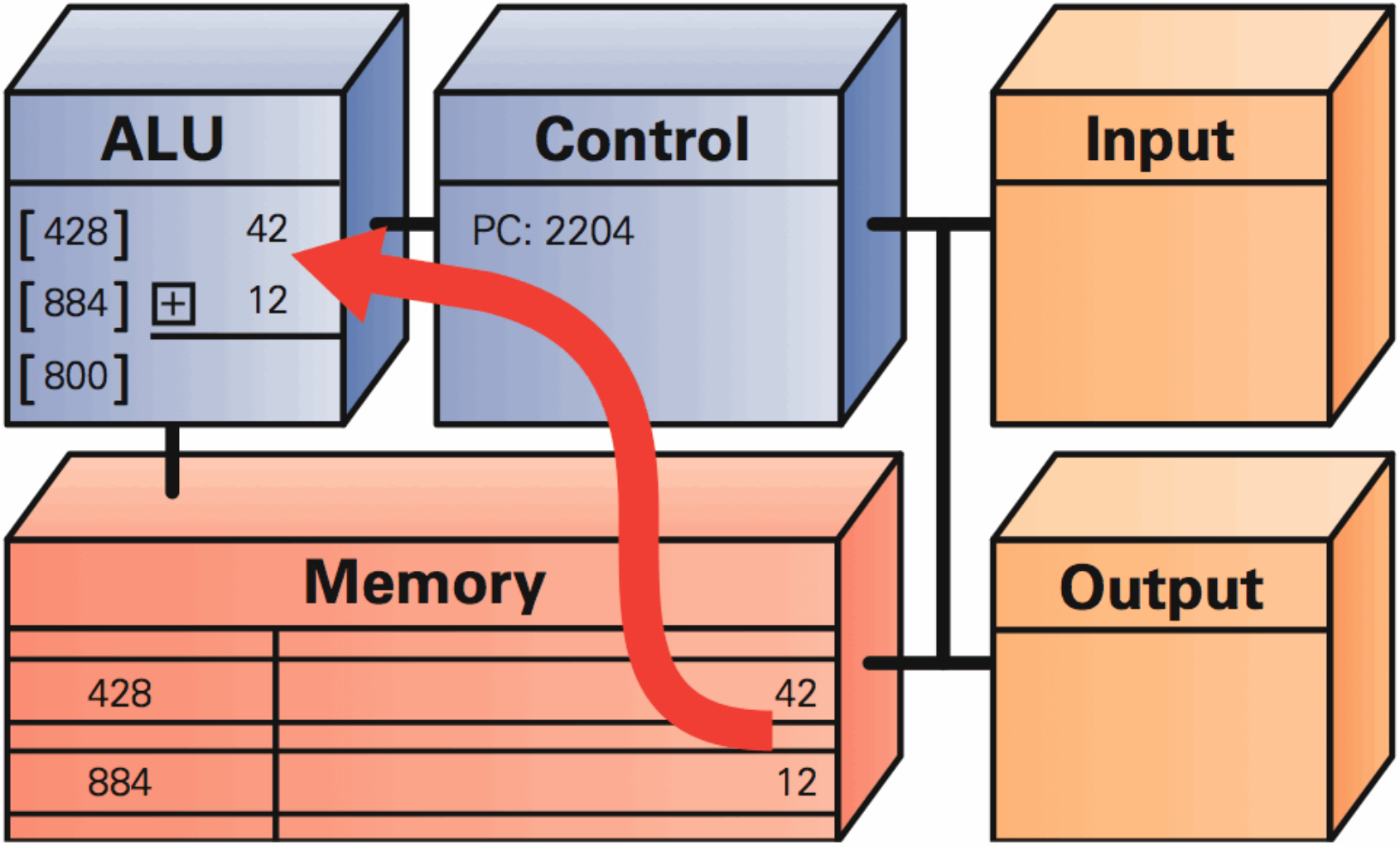
Instruction Fetch: Get Some Work



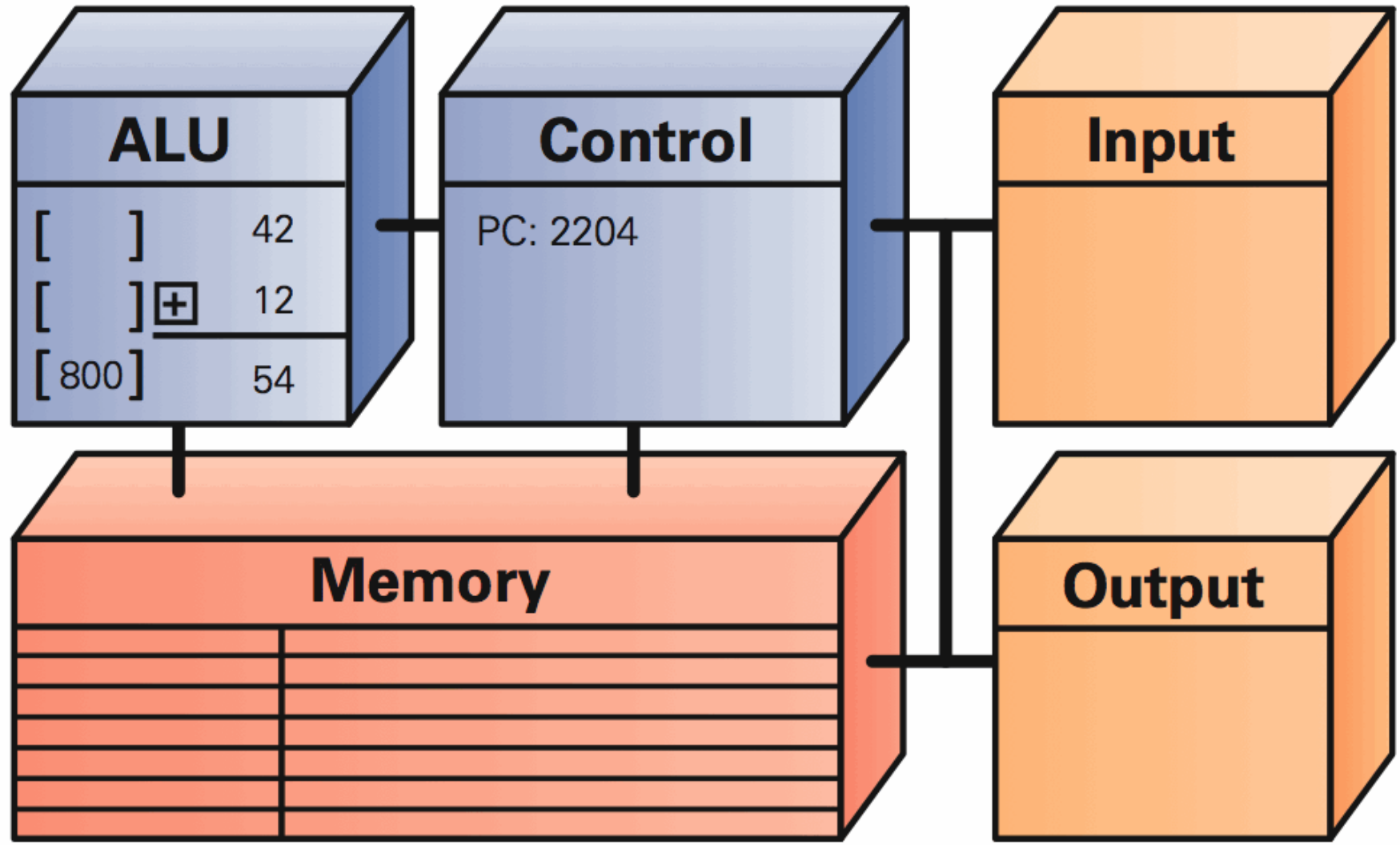
Instruction Decode: What To Do?



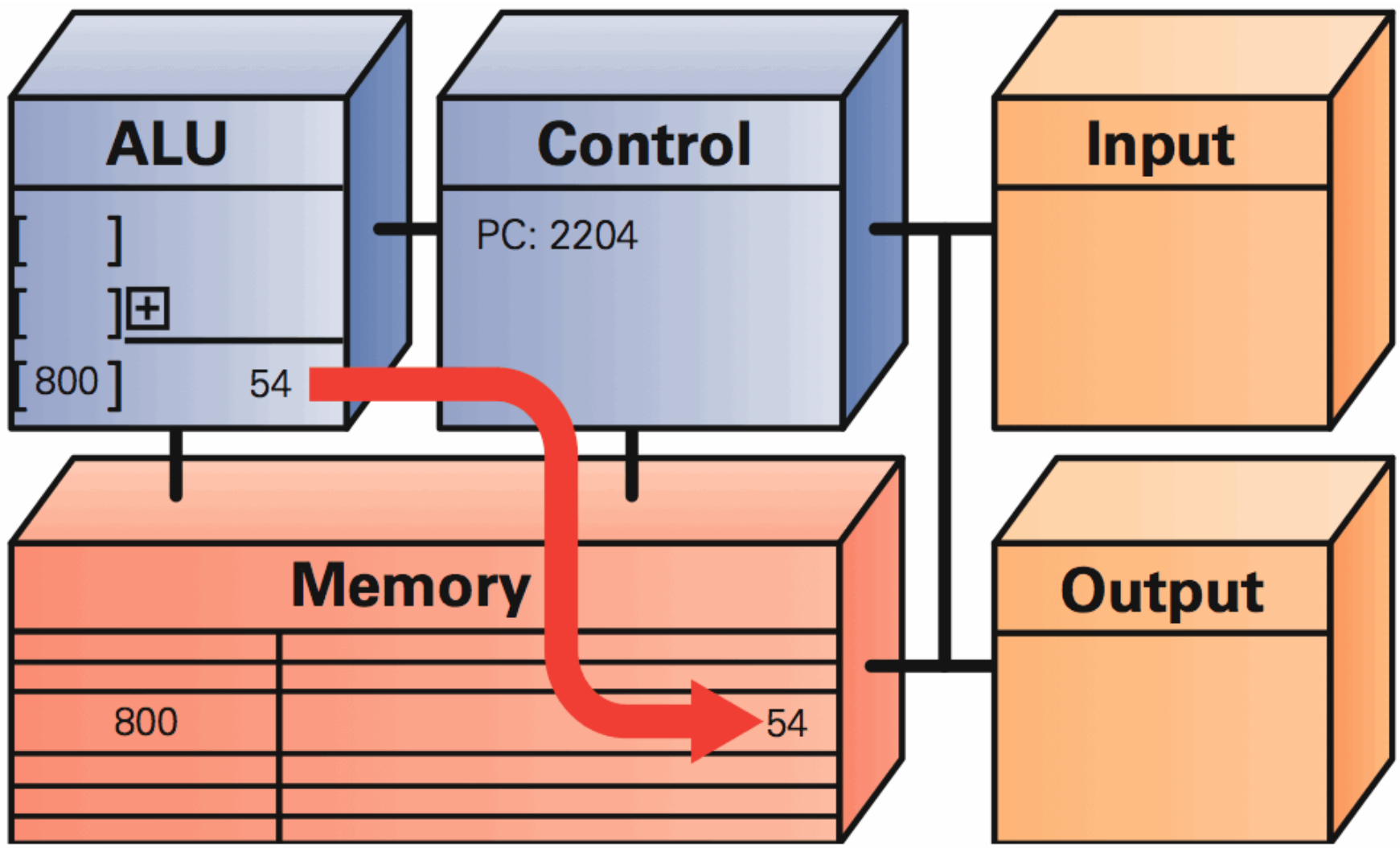
Data Fetch: What's The Input



Instruction Execution: Just Do It



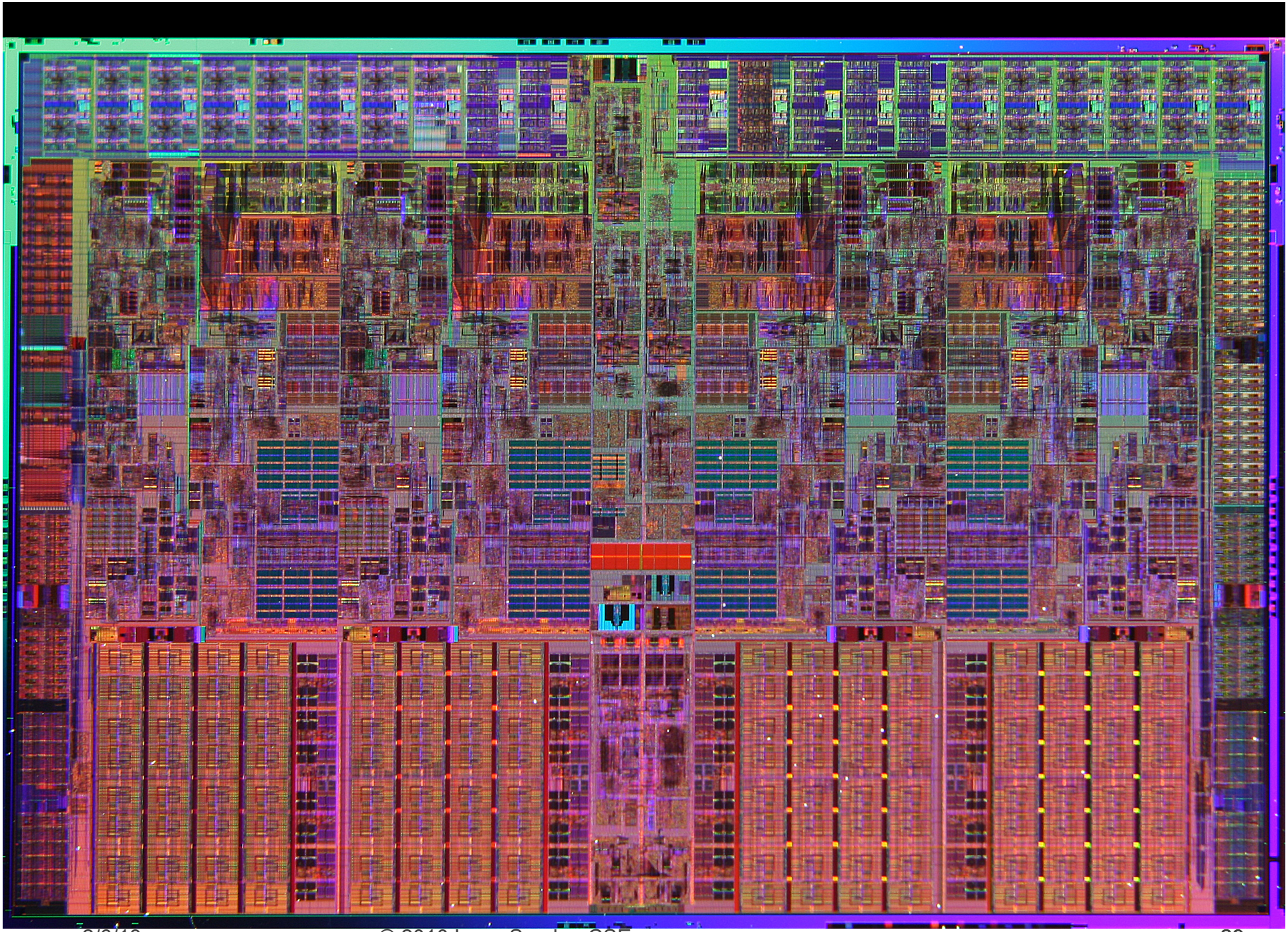
Result Return: Put It Away 4 Future



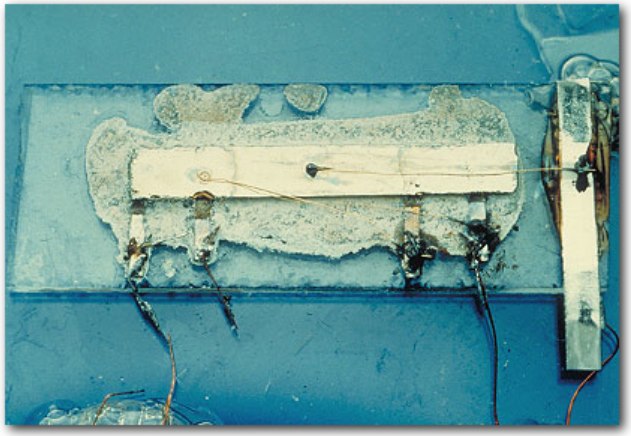
Clocks Run The Engine

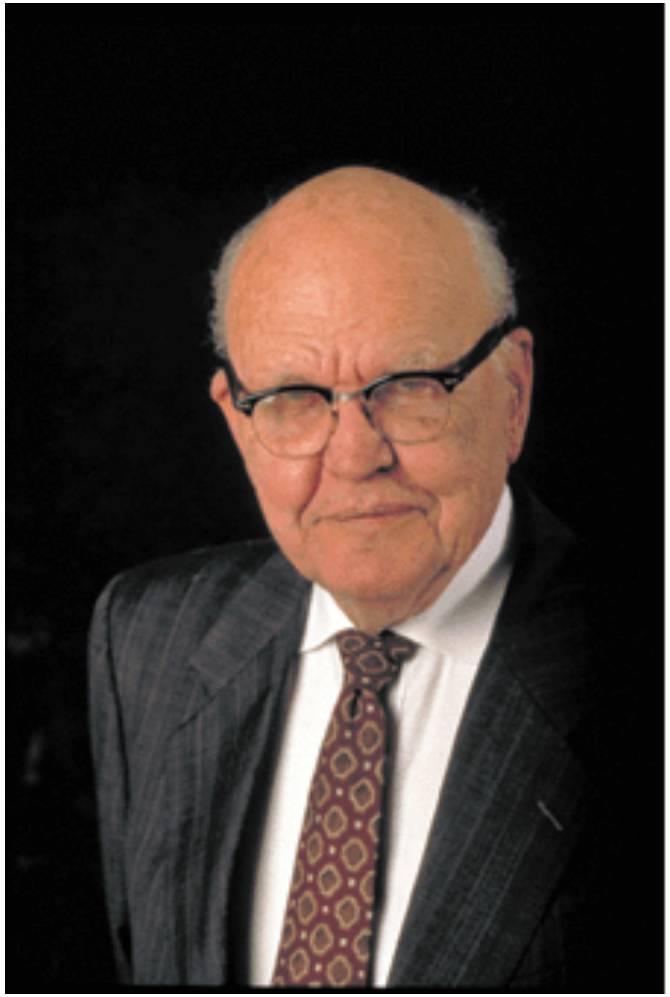
- The rate a computer "spins around" the Fetch/Execute cycle is controlled by it's clock
  - **Current clocks run 2-3 GHz**
  - **In principle, the computer should do one instruction per cycle, but often it fails to**
  - **Modern processors try to do more than one instruction per cycle, and often succeed**

Clock rate is not a good indicator of speed



Jack Kilby, Mr. Integrated Circuits



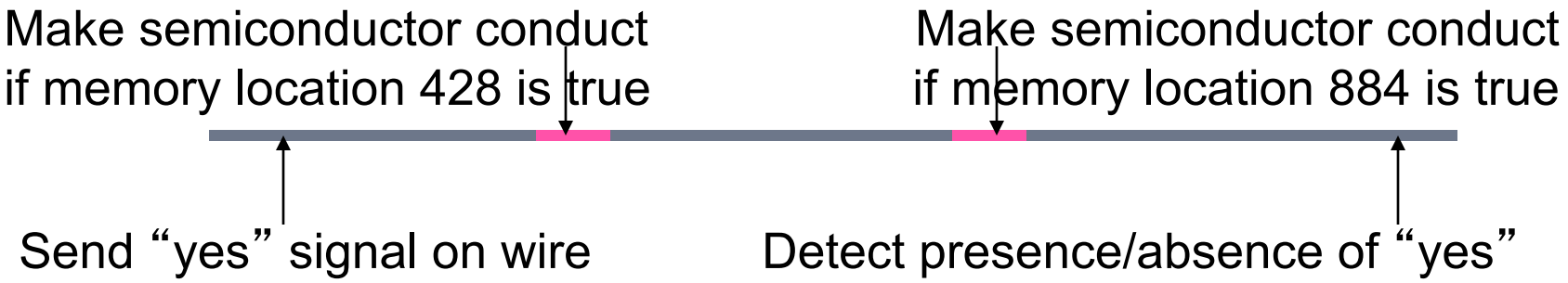


Semiconductors

- Silicon, a semiconductor -- sometimes it conducts and sometimes it doesn't
  - **It's possible to control when semiconductors do and don't conduct**

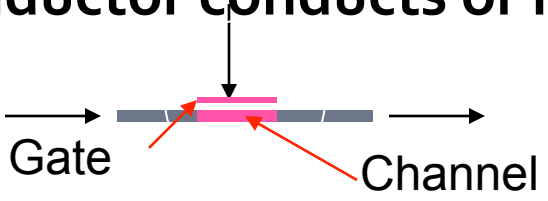
Compute by controlling conducting

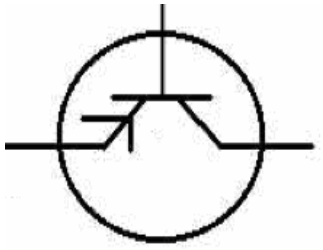
Ex.: AND 428, 884, 800



Field Effect

- Charged objects are familiar -- use a nylon comb on a dry day
  - **A charged field can control whether**
  - **a semiconductor conducts or not**





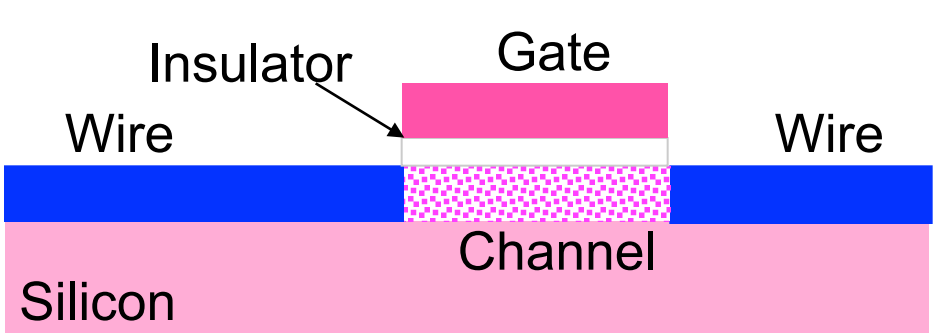
A transistor has 3 wires

The charge of the control wire (gate) is key

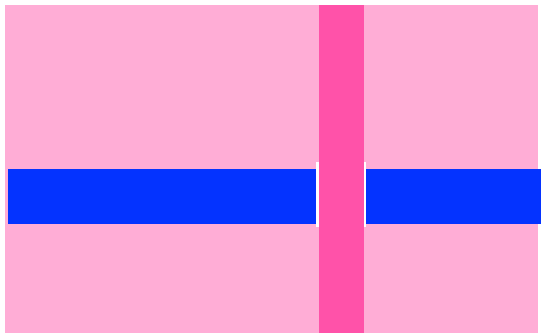
- **Neutral gate, channel doesn't conduct**
- **Charged gate, channel conducts**

MOS Transistors

- The field-effect idea is implemented in metal-oxide-semiconductor transistors
- Schematic in Si

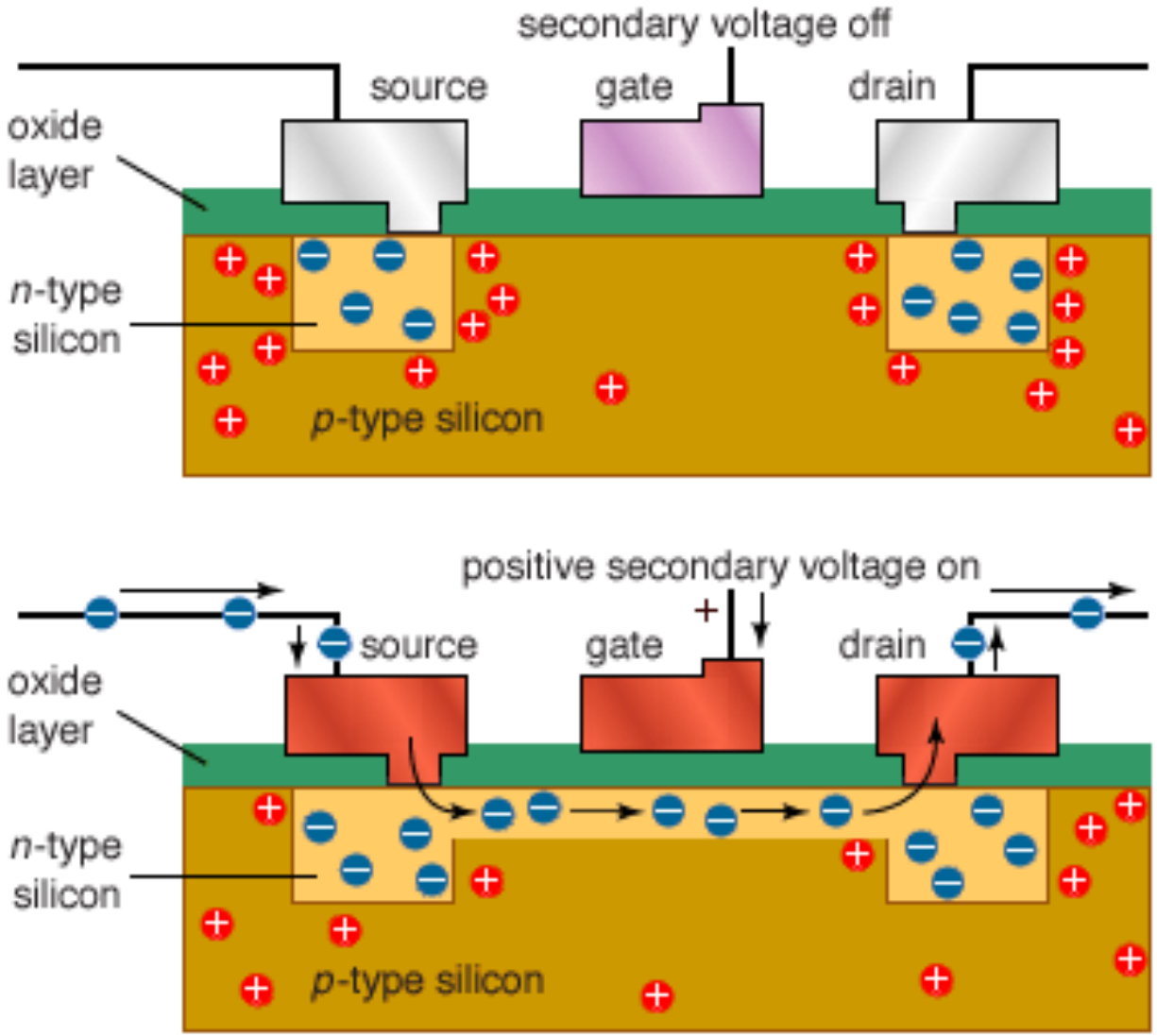


Slice across chip, look end on



From Above

nMOS Transistor



Fabrication …

- Check it out …
  - http://umumble.com/blogs/company_intel/385/

Summary

- Fetch/execute cycle runs instructions
  - 5 steps to interpret machine instructions
  - Programs must be in the memory
  - Data is moved in and out of memory

Instructions, data are represented in binary