Announcements

- Midterm Friday
- Review in Lab tomorrow … bring your questions

Remember Back To The Lightbot

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

Lawrence Snyder
University of Washington, Seattle

Computers …

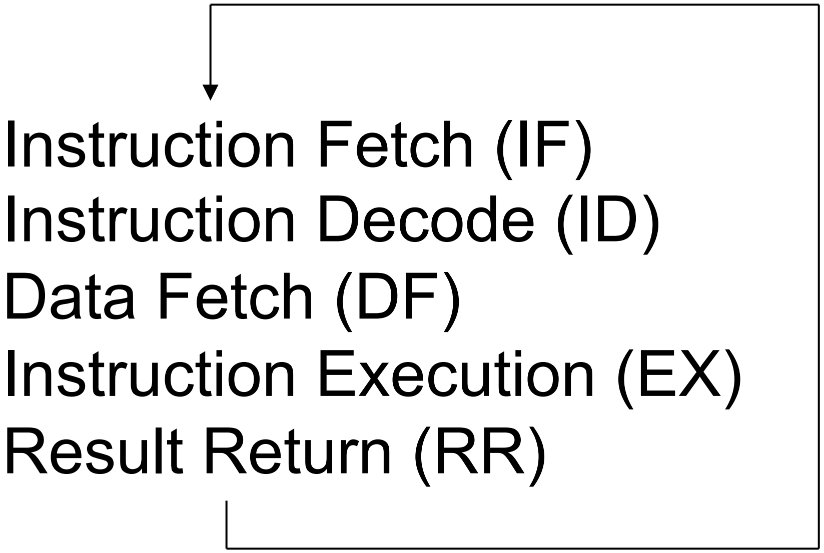
- Deterministically execute instructions to process information

  "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

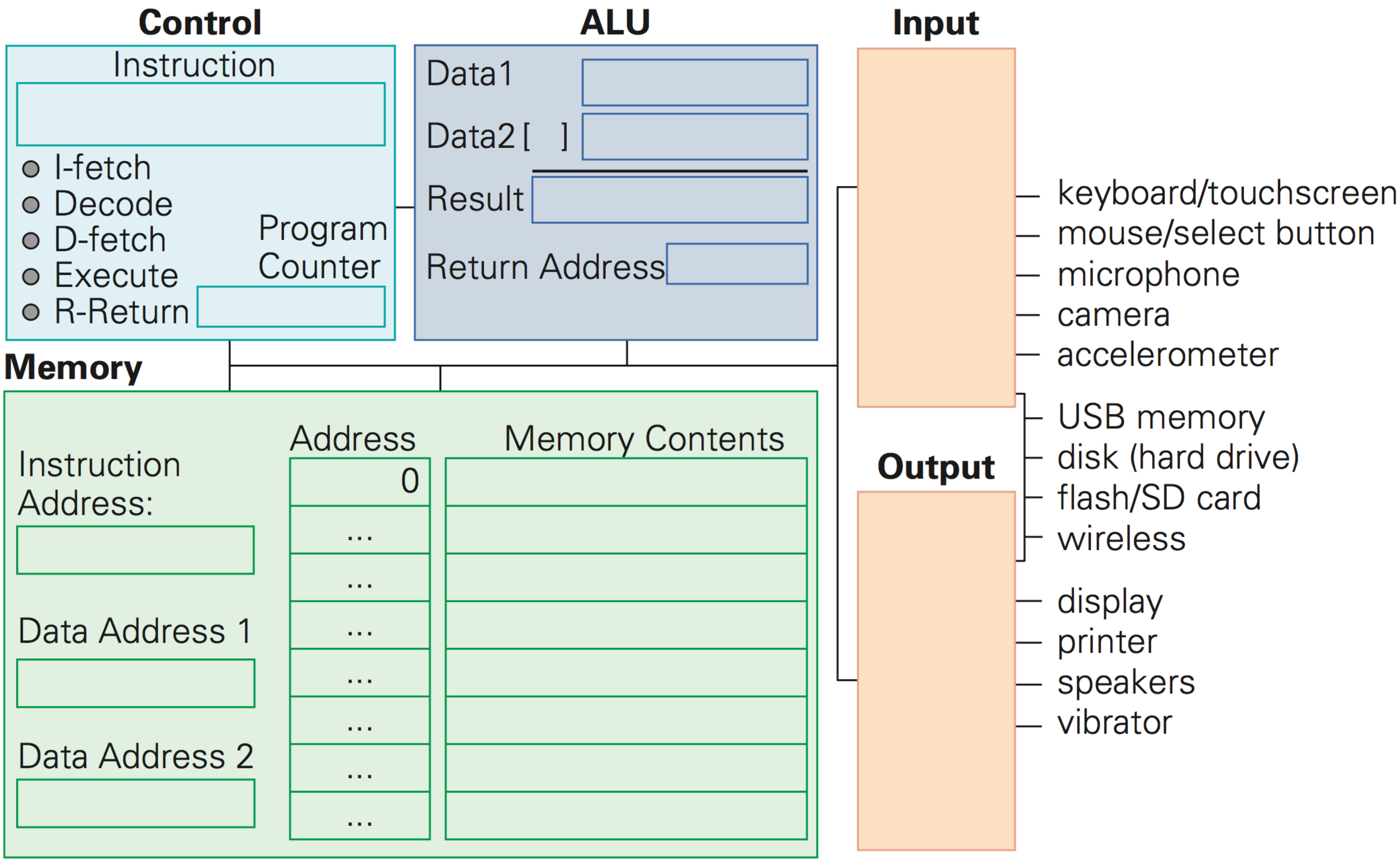
- Computer = instruction execution engine
  - **The fetch/execute cycle is the process that executes instructions**



Instruction Fetch (IF)
Instruction Decode (ID)
Data Fetch (DF)
Instruction Execution (EX)
Result Return (RR)

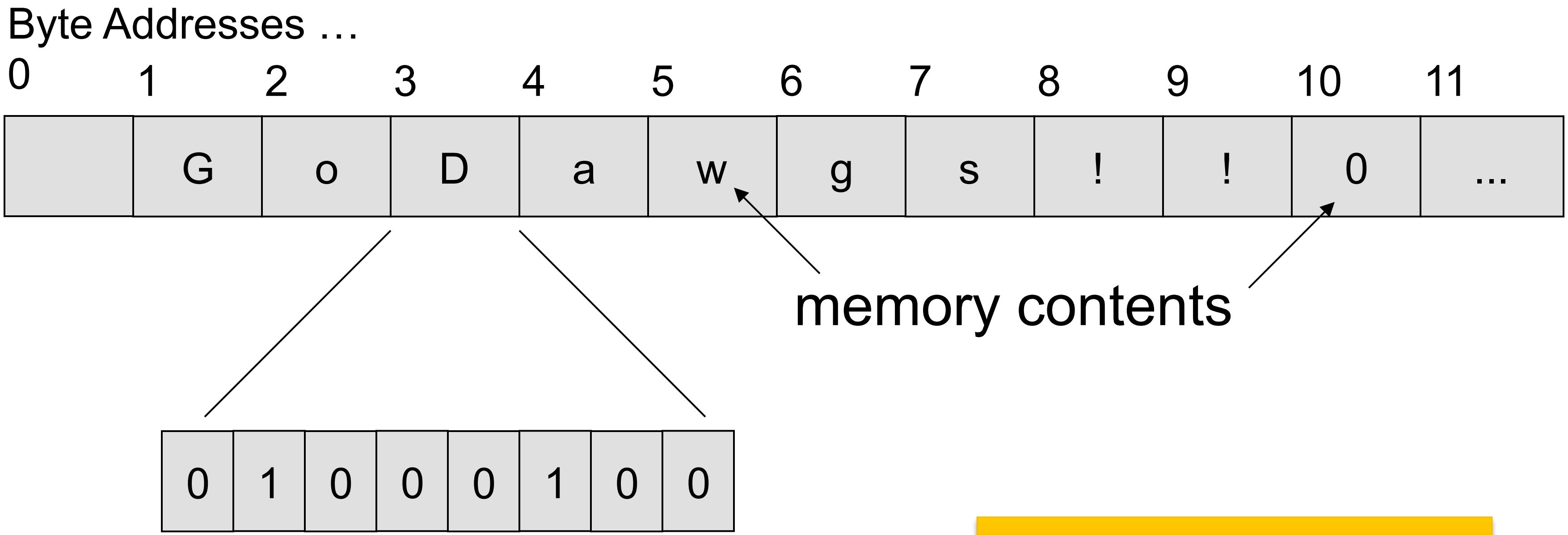
- The computer's internal parts implement this cycle

Anatomy of a Computer: The CPU



Memory …

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



Byte Addresses …

0	1	2	3	4	5	6	7	8	9	10	11
	G	o	D	a	w	g	s	!	!	0	...

memory contents

0	1	0	0	0	1	0	0

Groups of four bytes are a *word*

Control

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

The instructions have the form
ADDB 20, 10, 16
$\text{Mem}[20] = \text{Mem}[10] +_B \text{Mem}[16]$

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

10	11	12	13	14	15	16	17	18	19	20	21
6						12				18	...

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

10	11	12	13	14	15	16	17	18	19	20	21
8						7				15	...

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

10	11	12	13	14	15	16	17	18	19	20	21
8						7				15	...

10	11	12	13	14	15	16	17	18	19	20	21
60						-55				5	...

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		

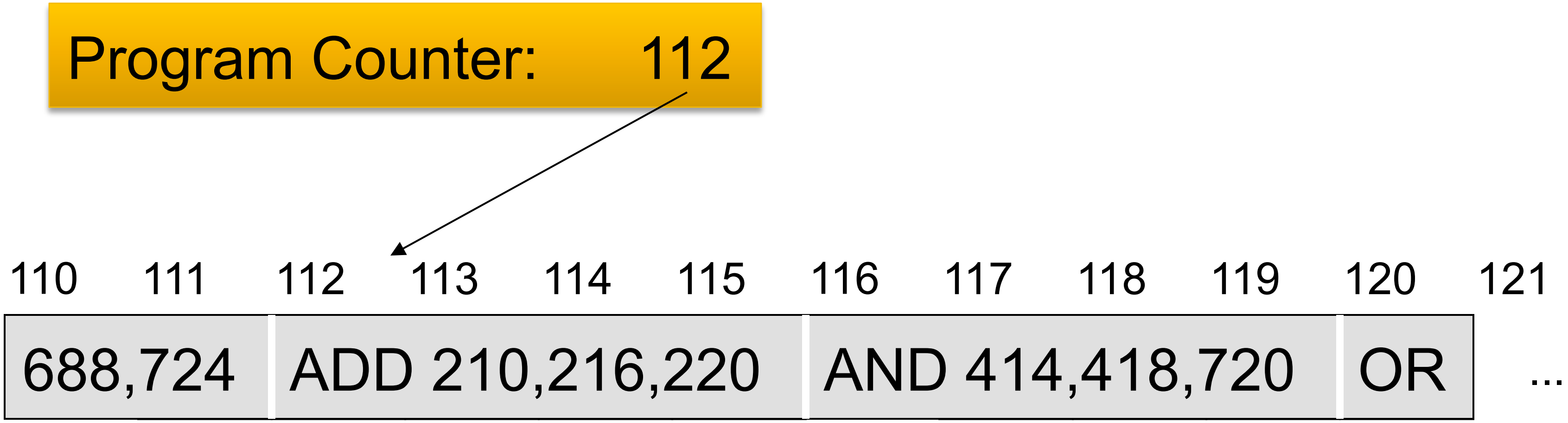
Most computers have only about a 100-150 instructions hard wired

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**



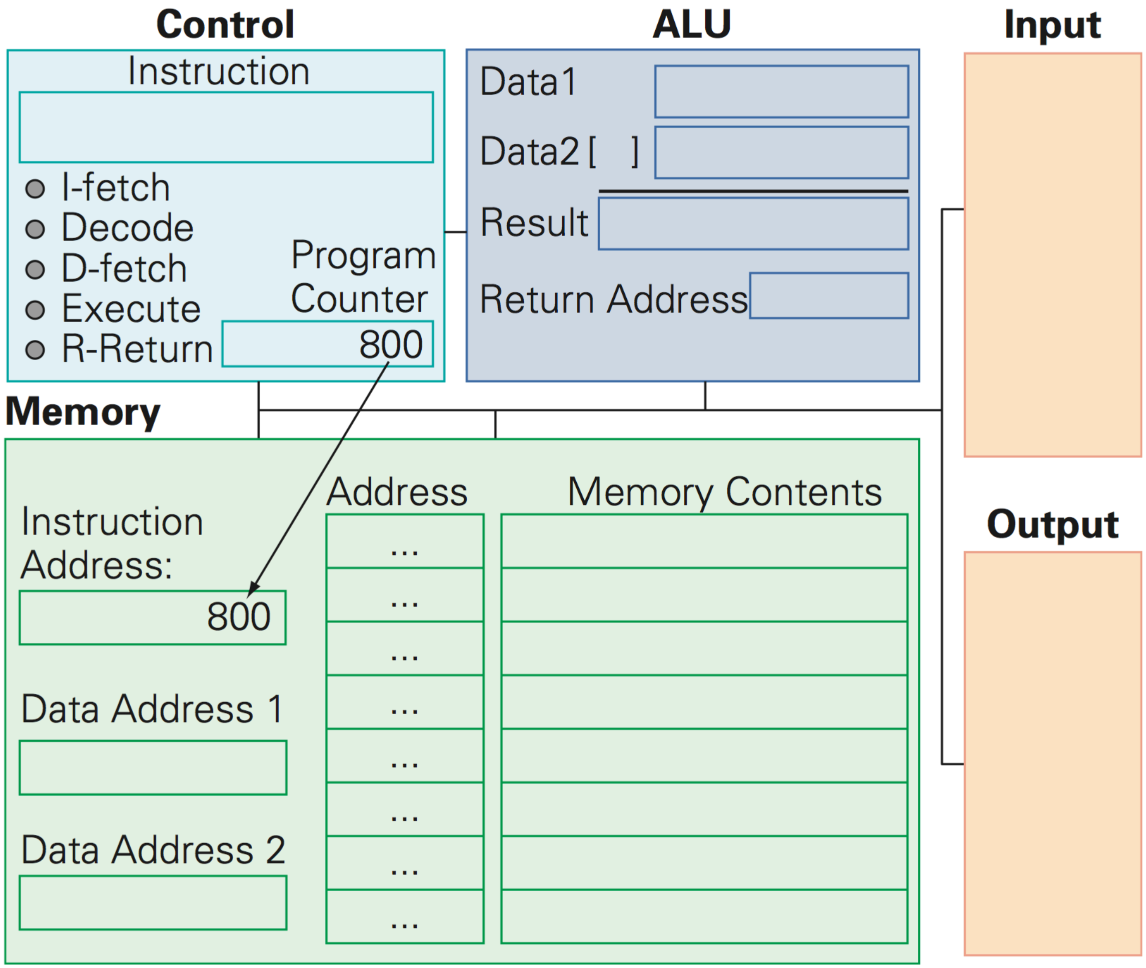
Program Counter: 112

110 111 112 113 114 115 116 117 118 119 120 121

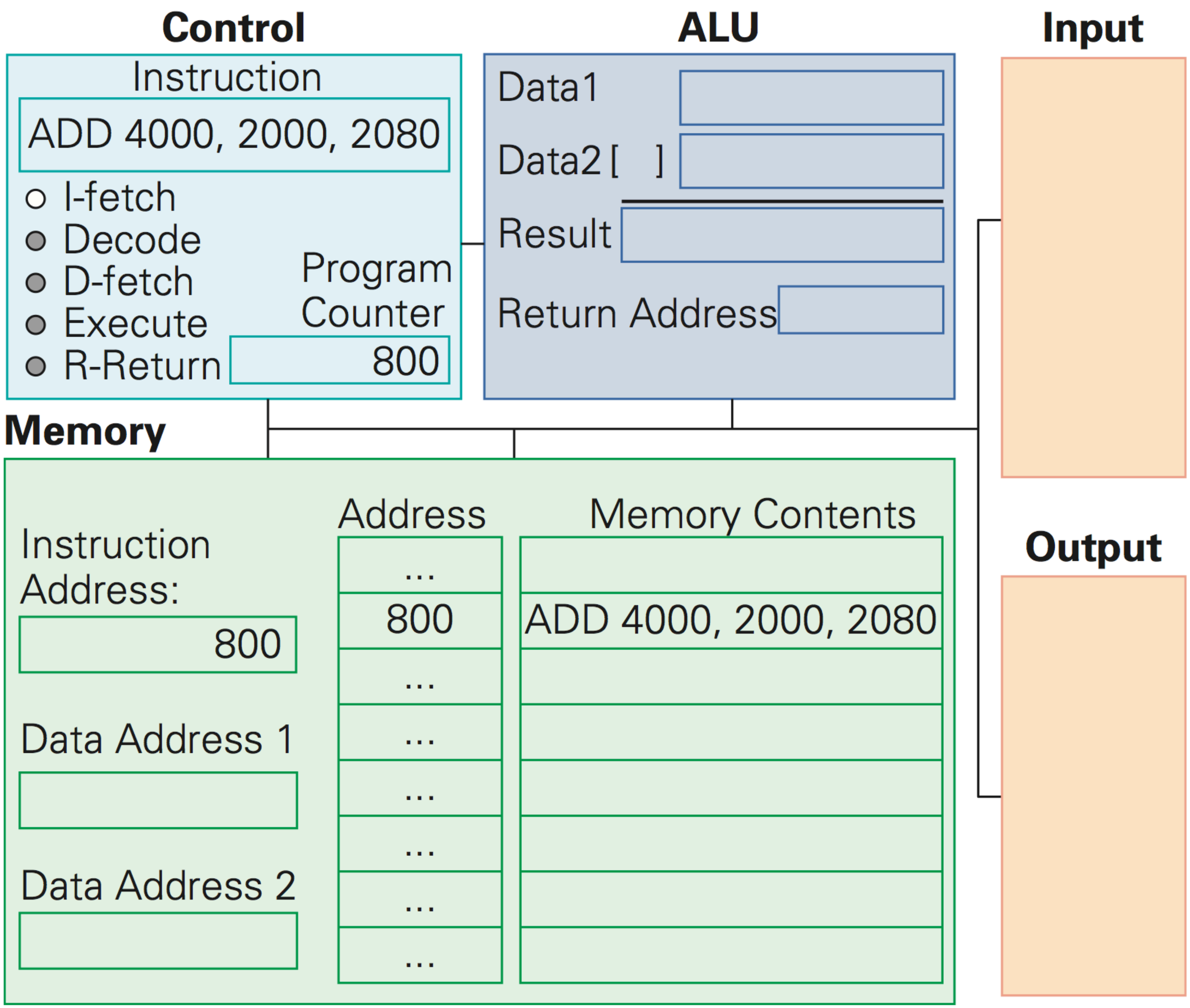
688,724 | ADD 210,216,220 | AND 414,418,720 | OR | …

Instruction Execution: The Setup

Run Instruction at 800: Add 4000, 2000, 2080



Instruction Fetch: Get Some Work



Control

Instruction

ADD 4000, 2000, 2080

- ○ I-fetch
- ● Decode
- ● D-fetch
- ● Execute
- ● R-Return

Program Counter

800

ALU

Data1

Data2 [ ]

Result

Return Address

Input

Output

Memory

Instruction Address:

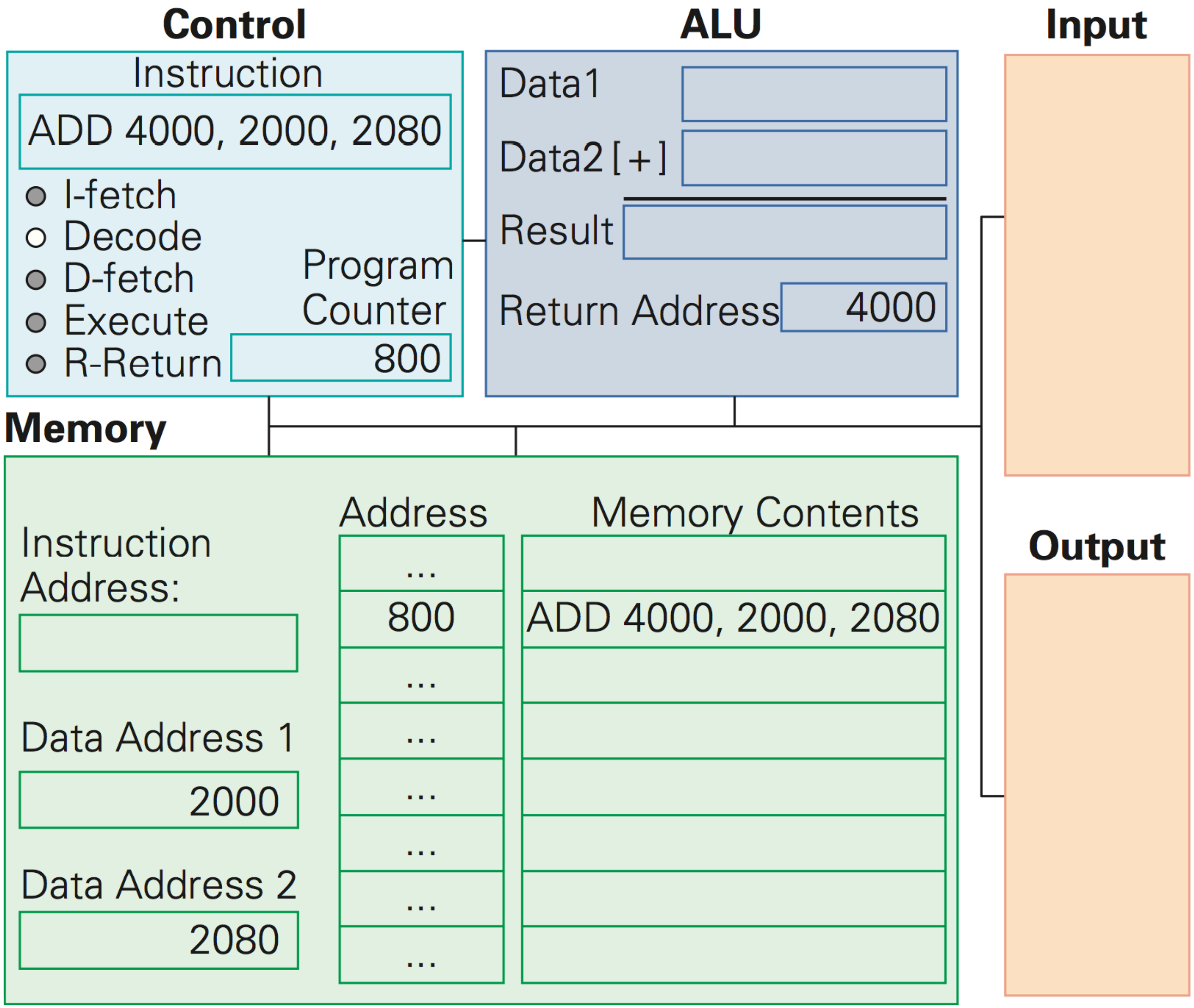
800

Data Address 1

Data Address 2

Address	Memory Contents
...	
800	ADD 4000, 2000, 2080
...	
...	
...	
...	
...	
...	

Instruction Decode: What To Do?



Control

Instruction

ADD 4000, 2000, 2080

- ● I-fetch
- ○ Decode
- ● D-fetch
- ● Execute
- ● R-Return

Program Counter

800

ALU

Data1

Data2 [+]

Result

Return Address 4000

Input

Output

Memory

Instruction Address:

Data Address 1

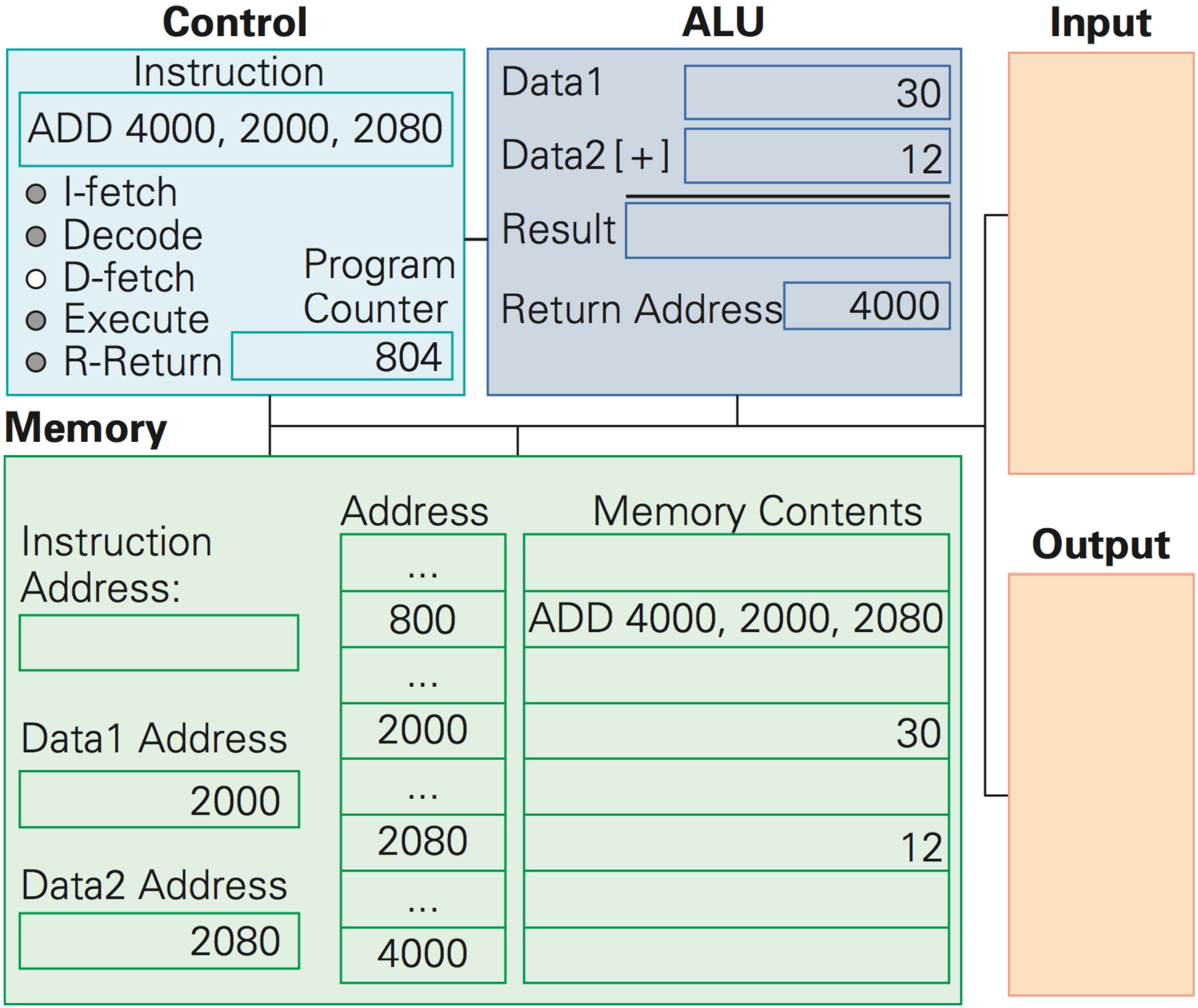
2000

Data Address 2

2080

Address	Memory Contents
...	
800	ADD 4000, 2000, 2080
...	
...	
...	
...	
...	
...	

Data Fetch: What's The Input



Control

Instruction

ADD 4000, 2000, 2080

● I-fetch
● Decode
○ D-fetch
● Execute
● R-Return

Program Counter

804

ALU

Data1	30
Data2 [+]	12
Result	
Return Address	4000

Input

Output

Memory

Instruction Address:

Data1 Address

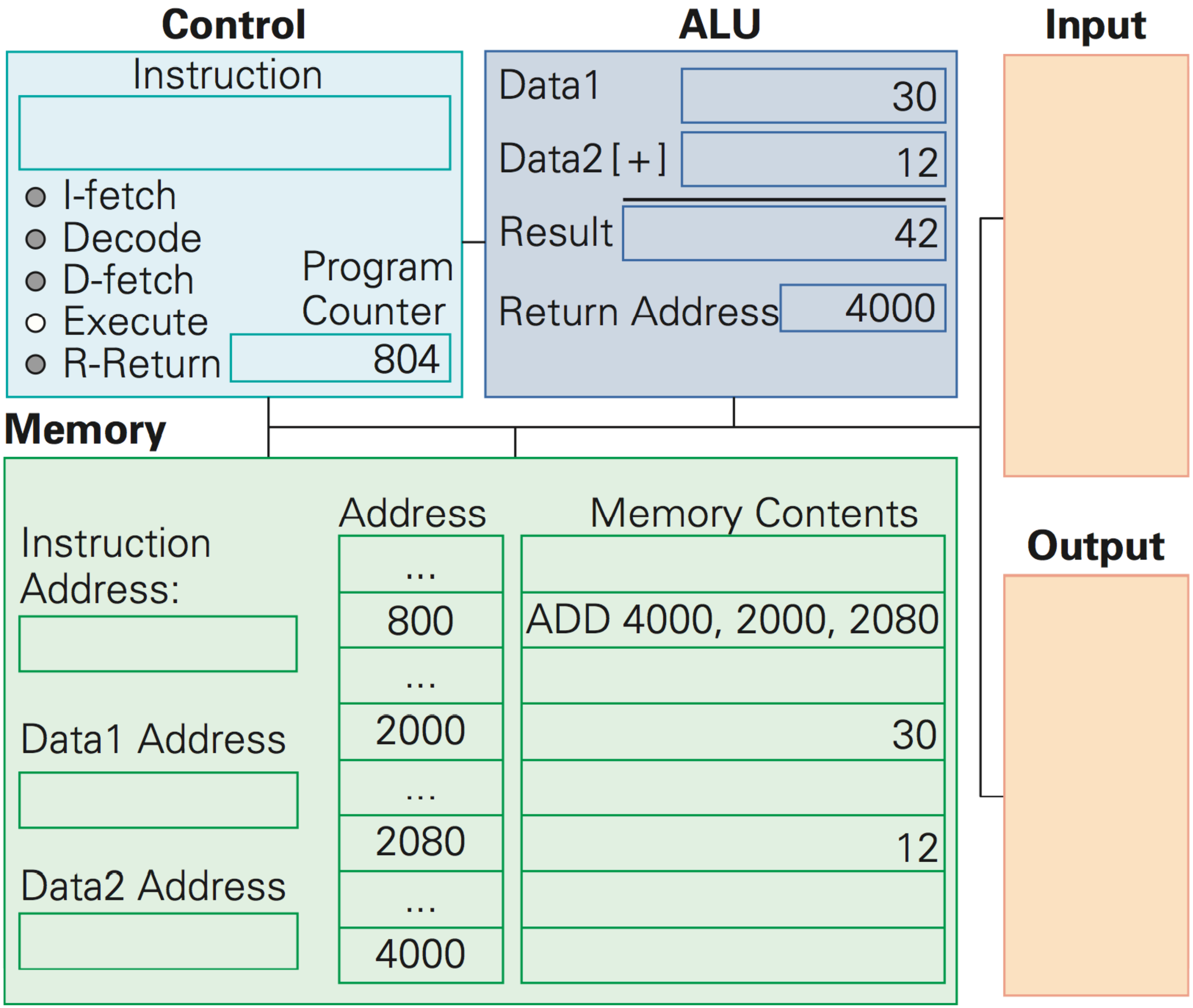
2000

Data2 Address

2080

Address	Memory Contents
...	
800	ADD 4000, 2000, 2080
...	
2000	30
...	
2080	12
...	
4000	

Instruction Execution: Just Do It



Control

Instruction

● I-fetch
● Decode
● D-fetch
○ Execute
● R-Return

Program Counter: 804

ALU

Data1: 30
Data2 [+]: 12
Result: 42
Return Address: 4000

Input

Output

Memory

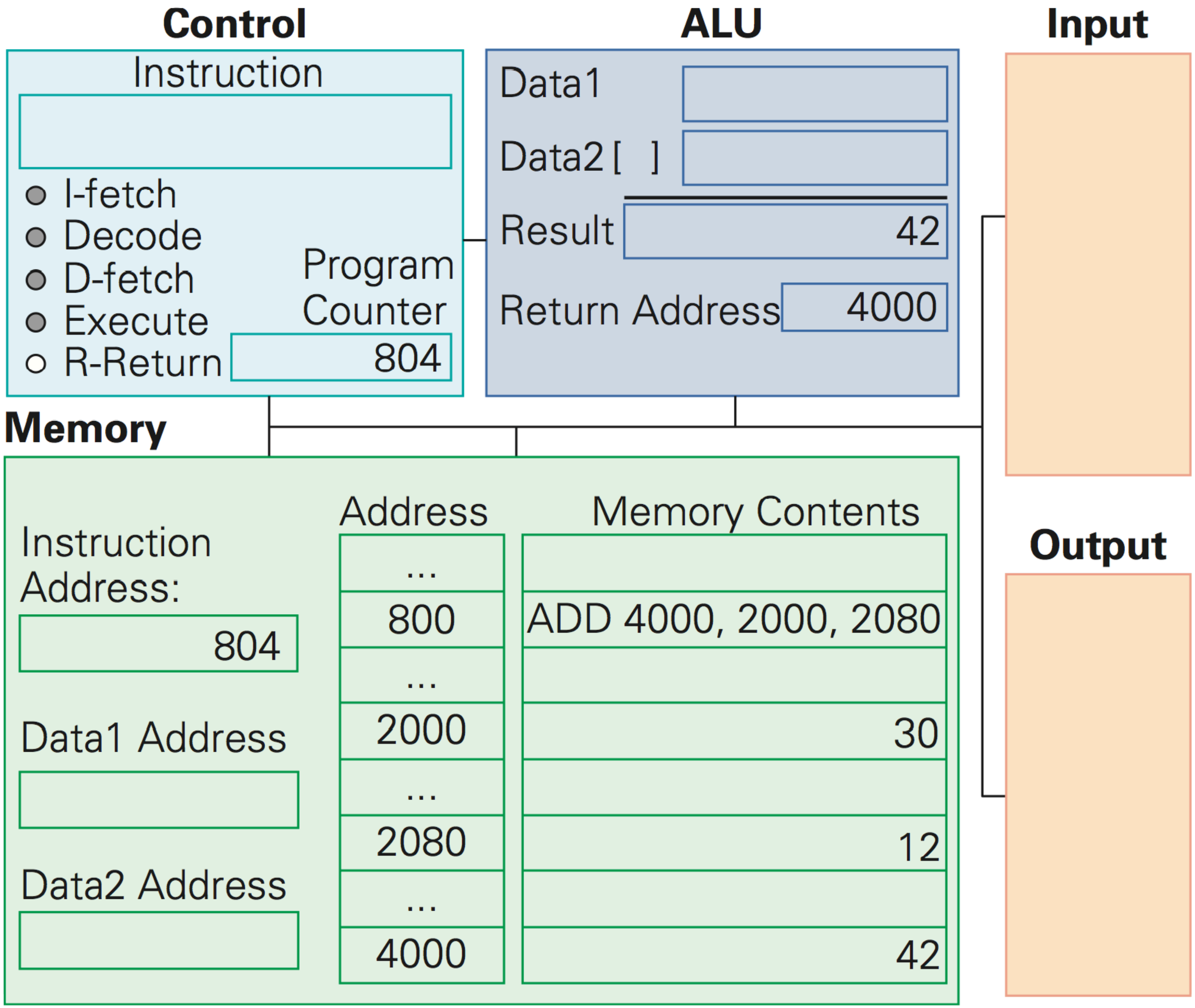
Instruction Address:

Data1 Address

Data2 Address

Address	Memory Contents
...	
800	ADD 4000, 2000, 2080
...	
2000	30
...	
2080	12
...	
4000	

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

Summary of F/E Cycle

- 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

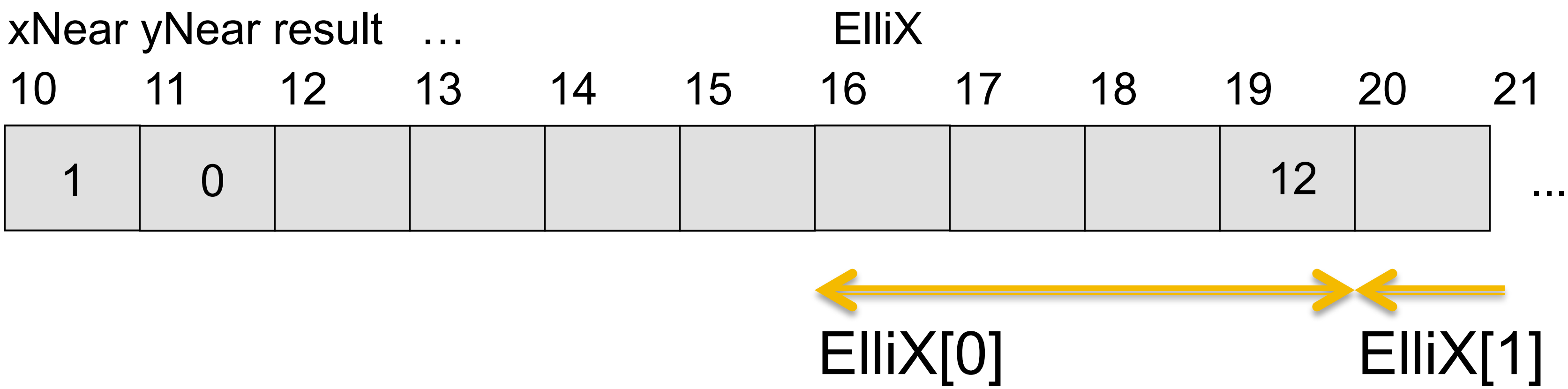
Execution: App → electrons

- Imagine an app written in Processing …

```
boolean xNear, yNear;
  ...
  xNear = abs(ElliX[i] - applX) < 25;
  yNear = abs(ElliY[i] - applY) < 25;
  ...
  if (xNear && yNear) {
    moveApple( );
  }
  ...
```

Processing Converts Code

- The Processing System “compiles” the code you write into machine language, the binary code the computer understands.
  - Step 1: Allocate Memory For Variables



xNear yNear result … ElliX

10 11 12 13 14 15 16 17 18 19 20 21

1 0 12 …

ElliX[0] ElliX[1]

Processing Converts Code

xNear	yNear	result	…			ElliX						
10	11	12	13	14	15	16	17	18	19	20	21	
1	0								12		...	

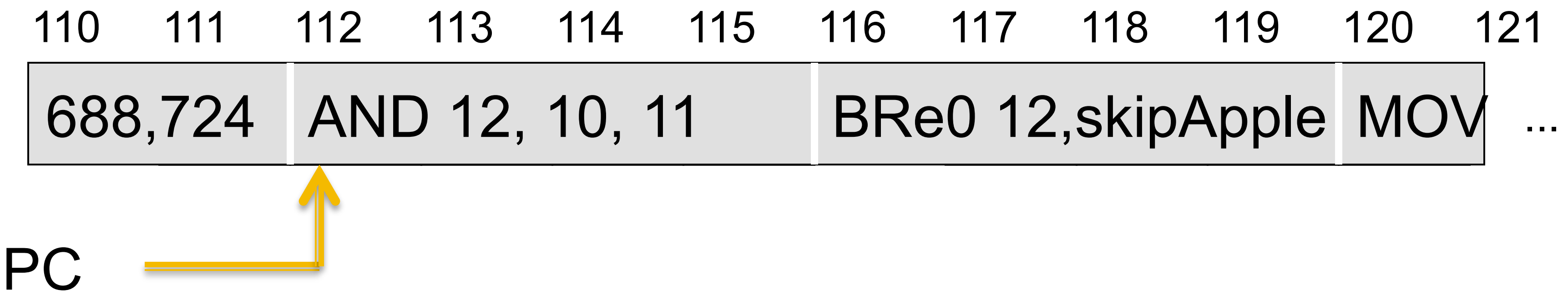
```
if (xNear && yNear) {
   moveApple( );
}
```

- Step 2: Translate Operations Into Machine Instr.s

	110	111	112	113	114	115	116	117	118	119	120	121
...	688,724		AND 12, 10, 11				BRe0 12,skipApple				MOV	...

Processing Executes Code

- Step 3: When PC == 112, Instruction Interpreted



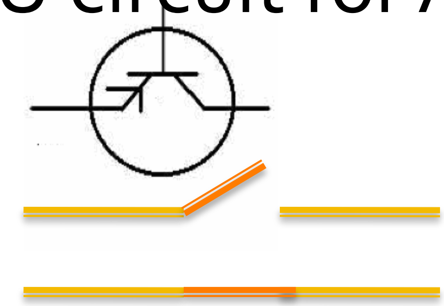
110	111	112	113	114	115	116	117	118	119	120	121	
688,724		AND 12, 10, 11				BRe0 12,skipApple				MOV		…

- Ifetch loads instruction at 112 into Control
- Decode sets up the ALU + Memory Registers
- Dfetch loads value in Mem[10], Mem[11] in ALU
- Execute performs the operation
- Result Return puts answer into Mem[12]

…

Processing Executes Code

- Structure of the ALU circuit for AND
  - Recall the transistor

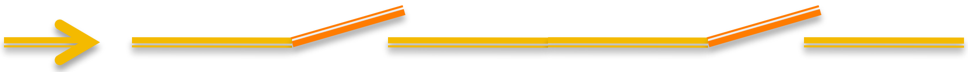


Not Conducting

Conducting

AND Logic

1



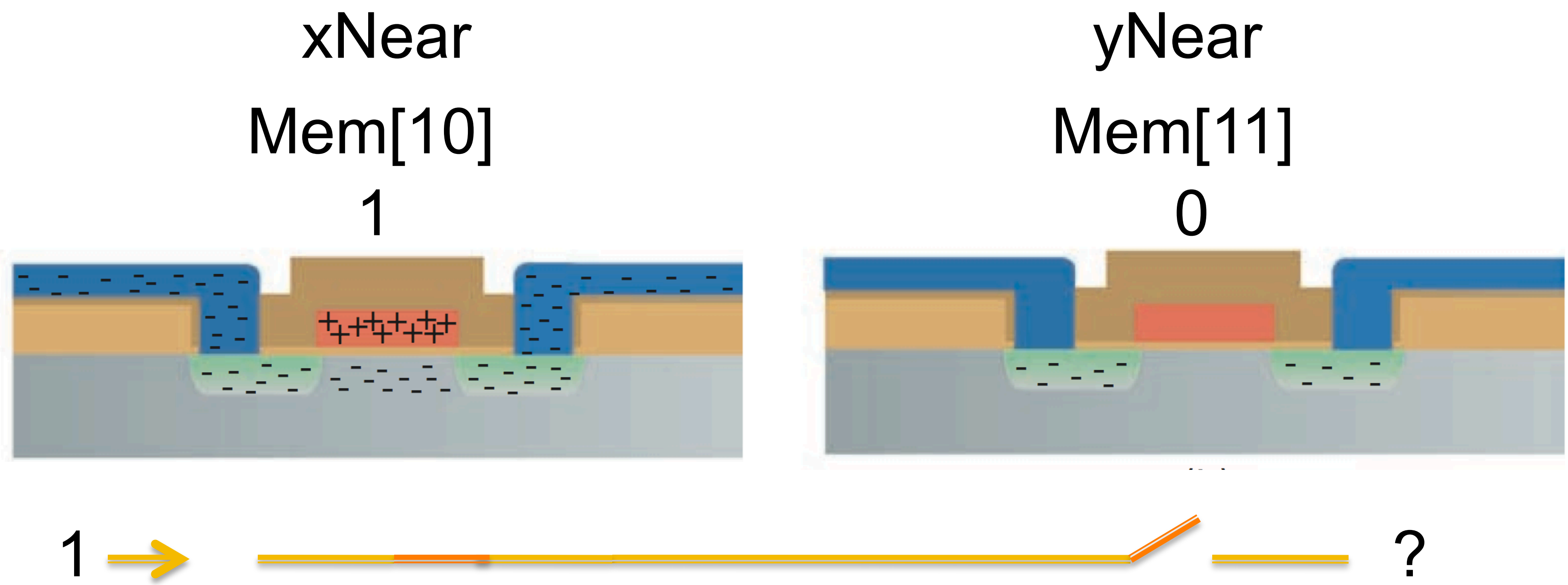
?

  - Control first gate with left input (xNear)
  - Control second gate with right input (yNear)
  - Place charge at left of wire
  - Detect presence/absence of charge at right end
  - Set result memory to 0 (absent) or 1 (present)

…

Processing Executes Code

- Transistors implementing the AND circuit



xNear

Mem[10]

1

yNear

Mem[11]

0

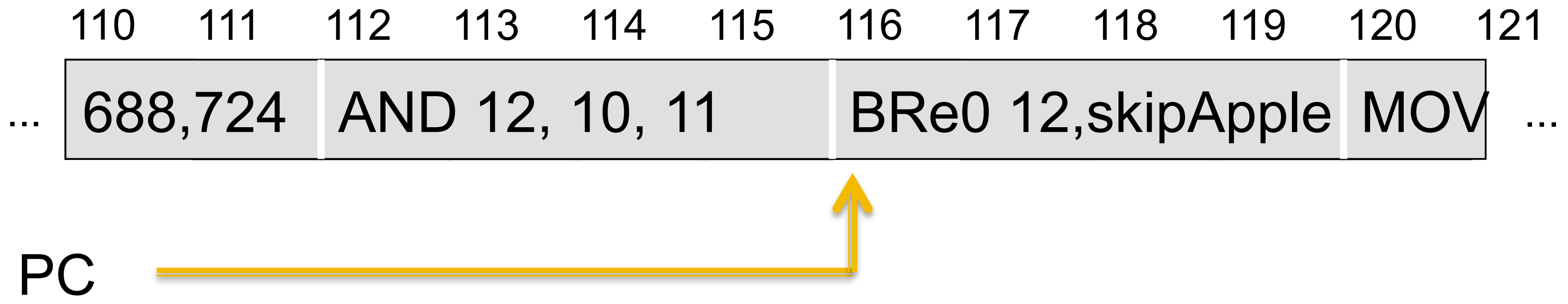
1 → ?

Processing Executes Code

- Data Memory after AND execution

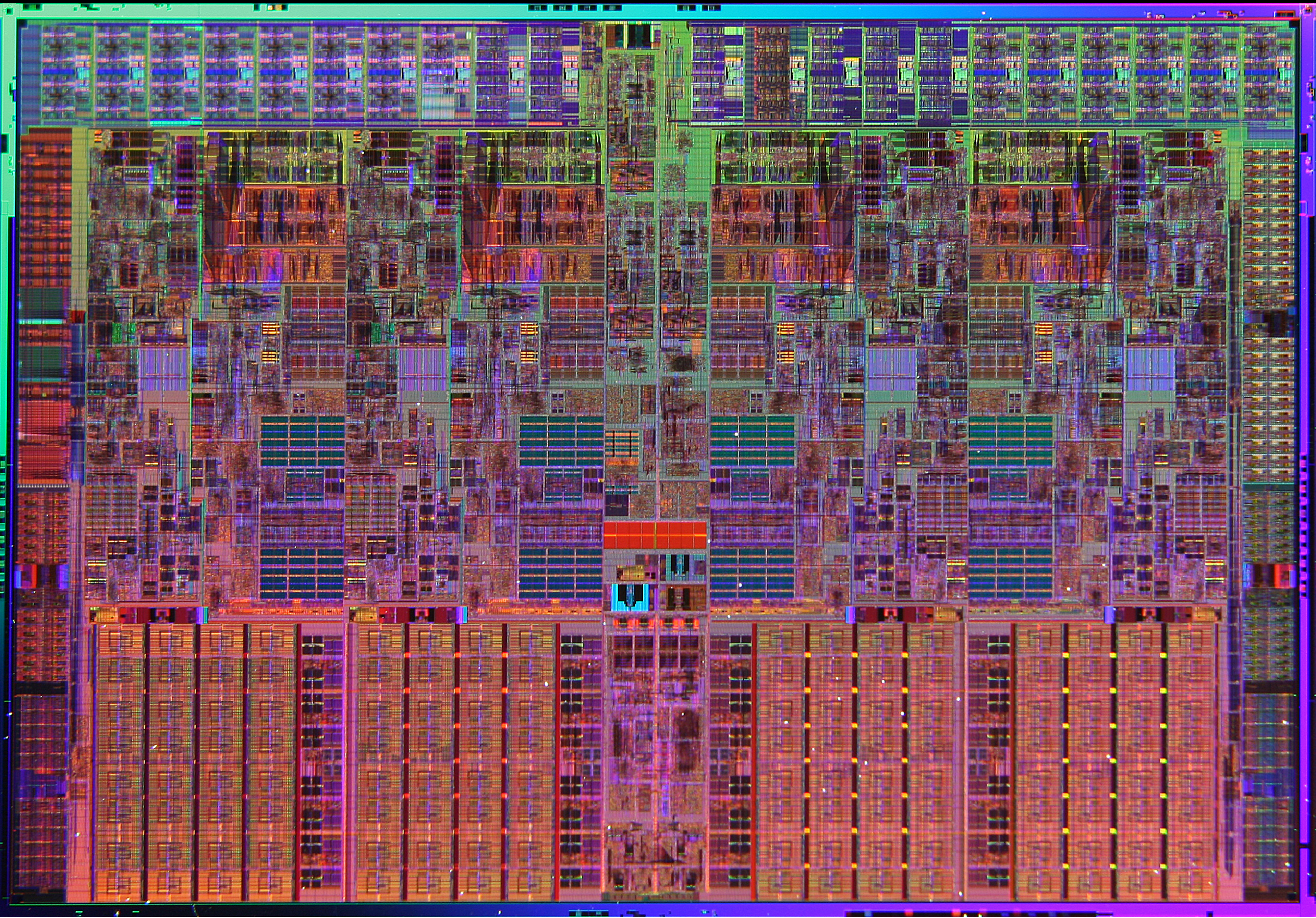
xNear	yNear	result	…			ElliX					
10	11	12	13	14	15	16	17	18	19	20	21
1	0	0							12		...

- Program Memory after AND execution



	110	111	112	113	114	115	116	117	118	119	120	121
...	688,724		AND 12, 10, 11				BRe0 12,skipApple				MOV	...

- Program Counter after AND execution



Summary

- It seems complicated … and it is
- But the electrons move at nearly the speed of light and they don't have to move far!
- The clock cycles 2-3 billion times a second
- And many technologies – from making pure silicon to photolithography to advanced software design – deliver huge amounts of computational power when we click on an App