Building to a Computer
Fetch, Decode, Execute

CS 2130: Computer Systems and Organization 1

Xinyao Yi Ph.D.
Assistant Professor

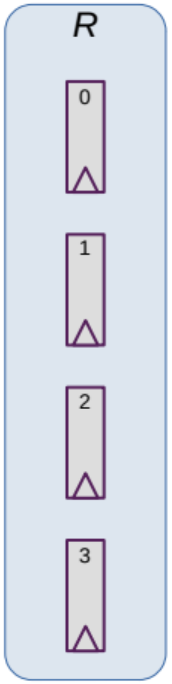
UNIVERSITY of VIRGINIA | ENGINEERING

Announcements

- Quiz 3 available today, due Sunday by 11:59pm
- Homework 2 due Monday

Writing

R[j] = y - connect *y* to input of registers based on index *j*



Code

How do we run code? What do we need?

Consider the following code:

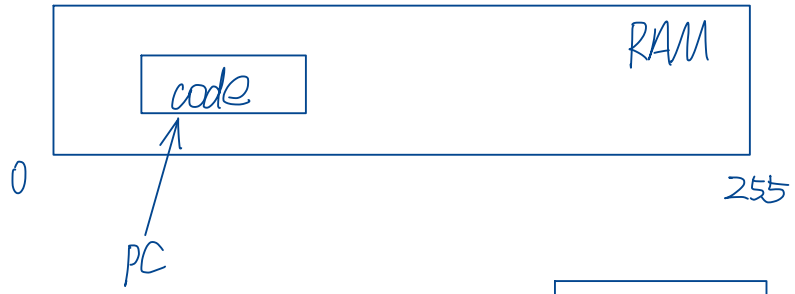
```
...
8:    x = 16
9:    y = x
10:   x += y
...
```

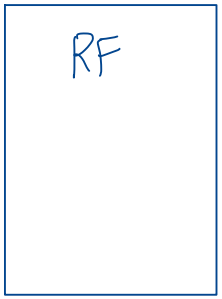
What is the value of *x* after line 10?

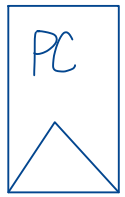
Bookkeeping

What do we need to keep track of?

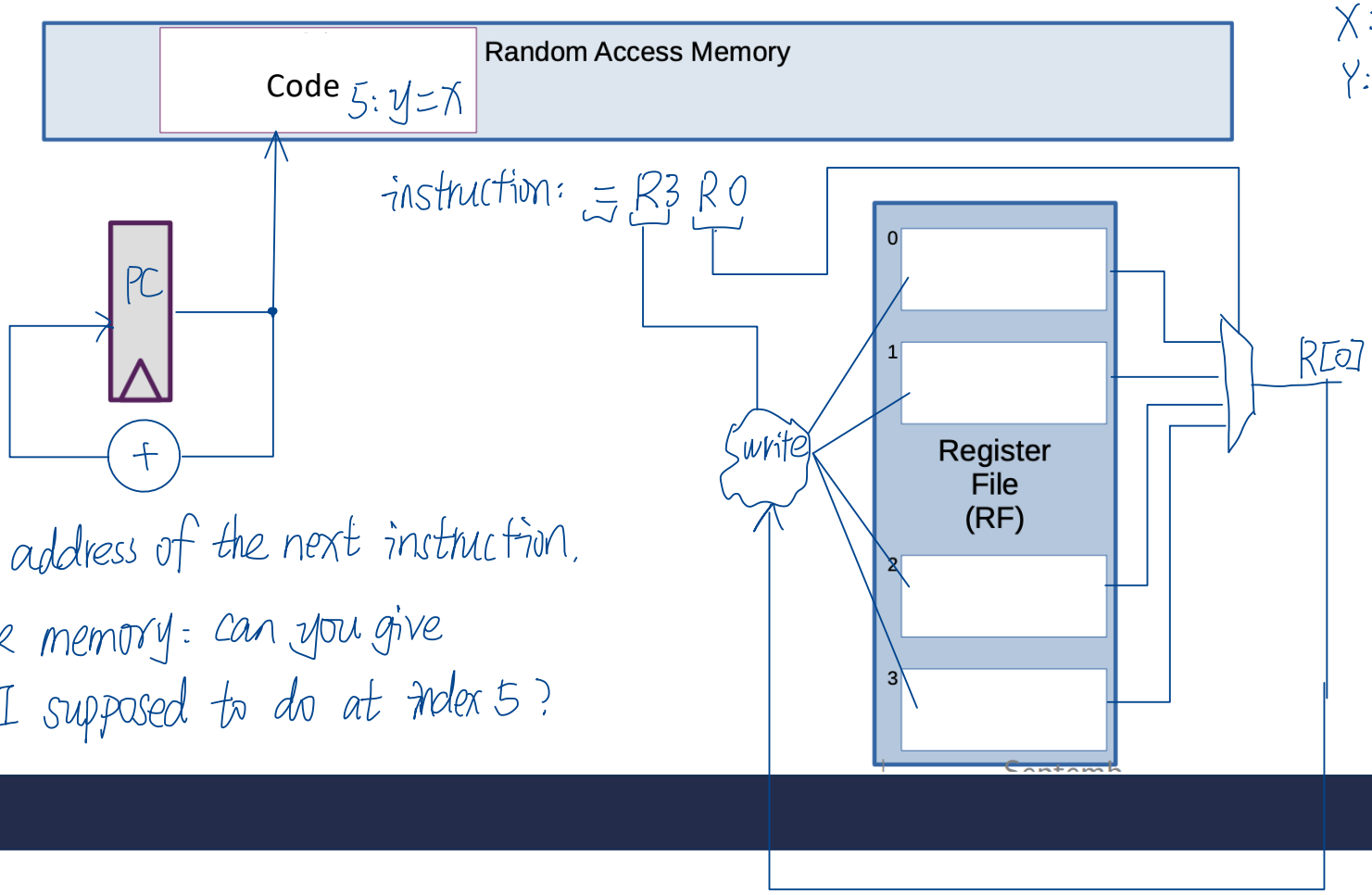
- **Code** - the program we are running
  - RAM (Random Access Memory)
- **State** - things that may change value (i.e., variables)
  - Register file - can read and write values each cycle
- **Program Counter (PC)** - where we are in our code
  - Single register - byte number in memory for next instruction







Building a Computer



PC has the address of the next instruction.
we can ask memory: can you give
me what I supposed to do at index 5?

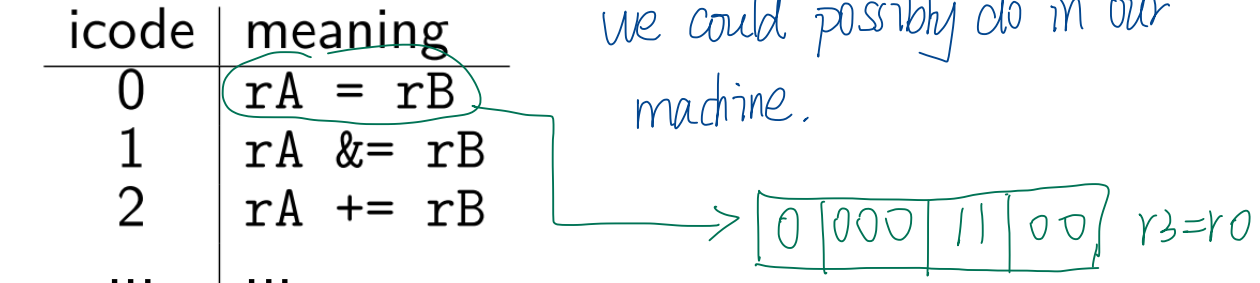
Encoding Instructions

Encoding of Instructions (**icode** or **opcode**)

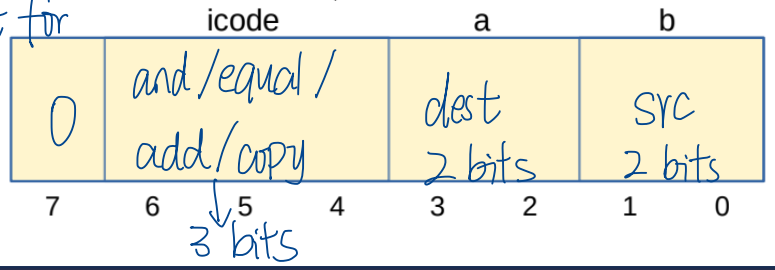
- Numeric mapping from icode to operation

icode	meaning
0	`rA = rB`
1	`rA &= rB`
2	`rA += rB`
...	...

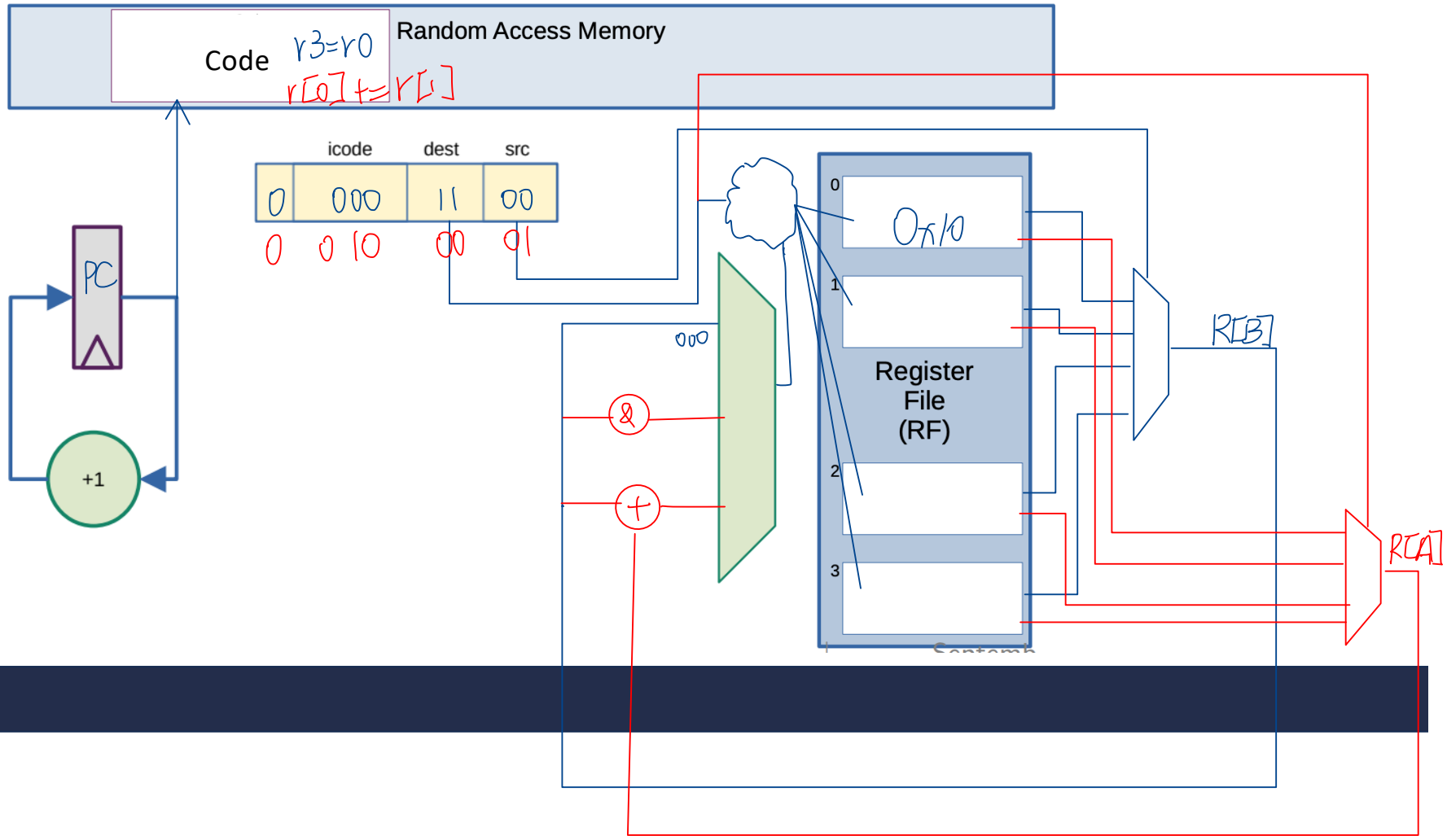
list out all the things that we could possibly do in our machine.



keep this bit for future newer things

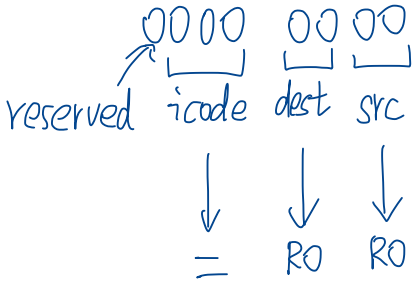


Building a Computer



Question

What happens if we get the 0-byte instruction? 00



r0=r0

This is a no operation (noop)

Doesn't do anything. ⇒ but useful

wait for cycles to pass while doing sth. else.

Our Computer's Instructions

Toy ISA 3-bit icode

icode	meaning
0	`rA = rB`
1	`rA &= rB`
2	`rA += rB`
...	...
4	`rA` = read from memory at address `rB`
5	write `rA` to memory at address `rB`
...	...
7	Compare `rA` as 8-bit 2's-complement to 0 if `rA <= 0` set `pc = rB` else increment `pc` as normal

M[rB]

M[rB] = rA

Update PC and say, I want to run the code at a different place.

jump

loop/if statement

Our Computer's Instructions

Toy ISA 3-bit icode

icode	b	action
3	0	`rA = ~rA`
	1	`rA = !rA`
	2	`rA = -rA`
	3	`rA = pc`
6	0	rA = read from memory at pc + 1
	1	rA &= read from memory at pc + 1
	2	rA += read from memory at pc + 1
	3	rA = read from memory at the address stored at pc + 1
		For icode 6, increase pc by 2 at end of instruction

Some operations that only need one register like flip, not.

Rather than sending B to the register file, we make B choose which one of these things we are going to do.

High-level Instructions

In general, 3 kinds of instructions

- **moves** - move values around without doing "work"
- **math** - broadly doing "work"
- **jumps** - jump to a new place in the code