Toy Instruction Set Architecture
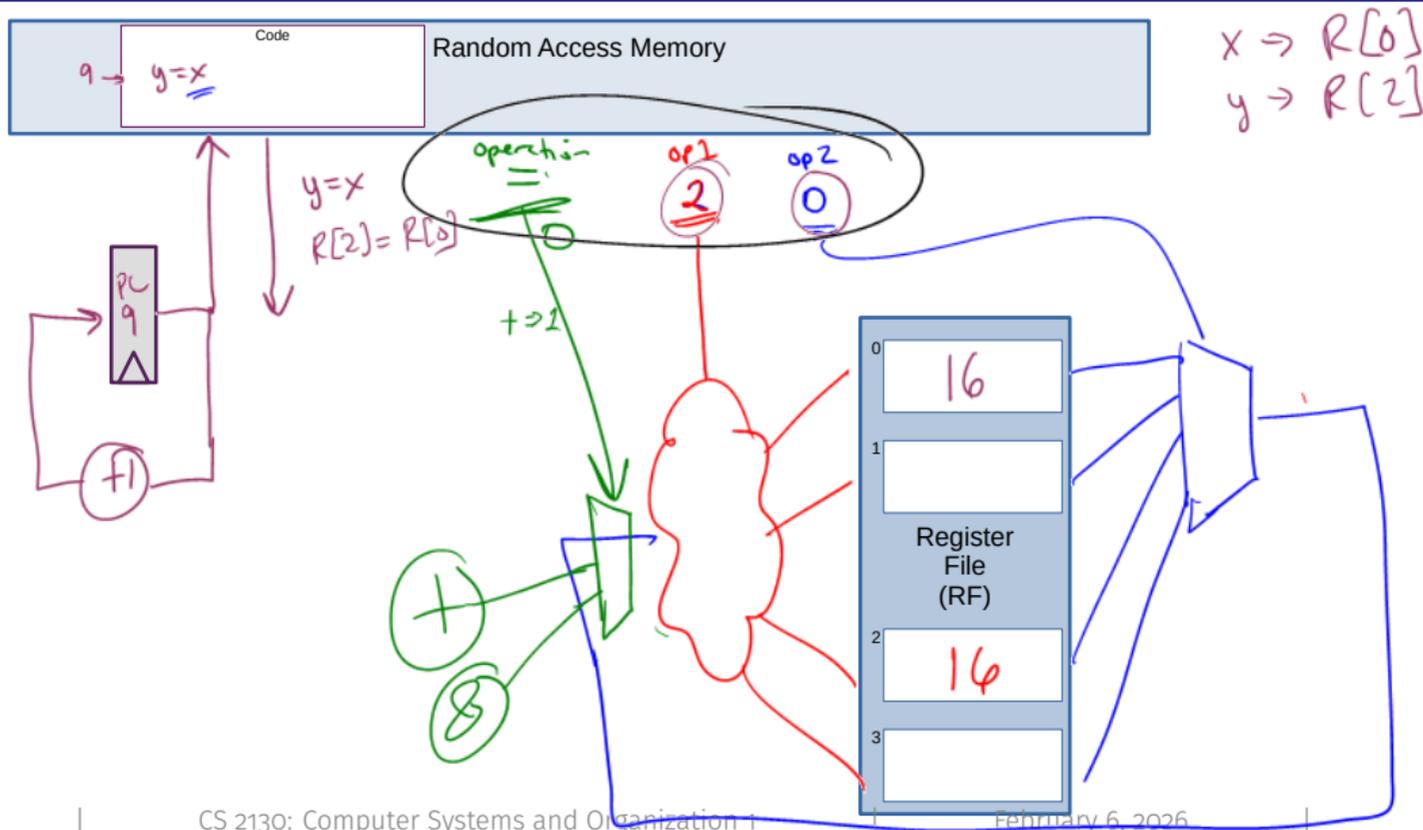
Building a computer today!

CS 2130: Computer Systems and Organization 1
February 9, 2026

Announcements

- Homework 2 due tonight at 11:59pm on Gradescope
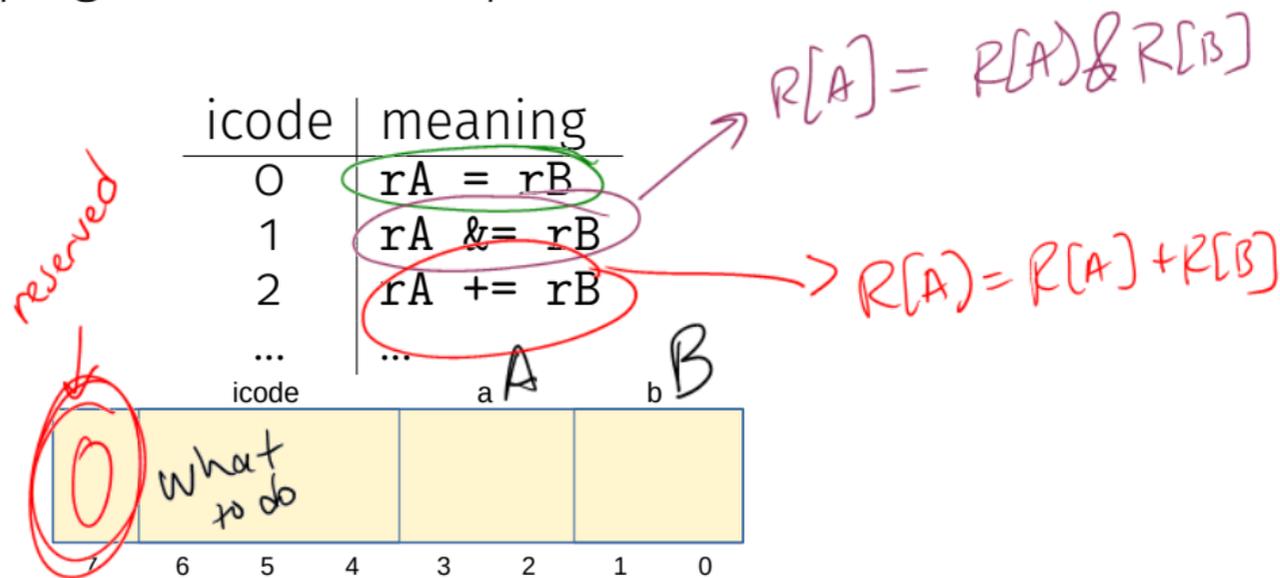- Homework 3 out tomorrow, due next Monday at 11:59pm on Gradescope

Building a Computer

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
...	...

R[A] = R[A] & R[B]

R[A] = R[A] + R[B]

reserved

what to do

icode	a	b
O		

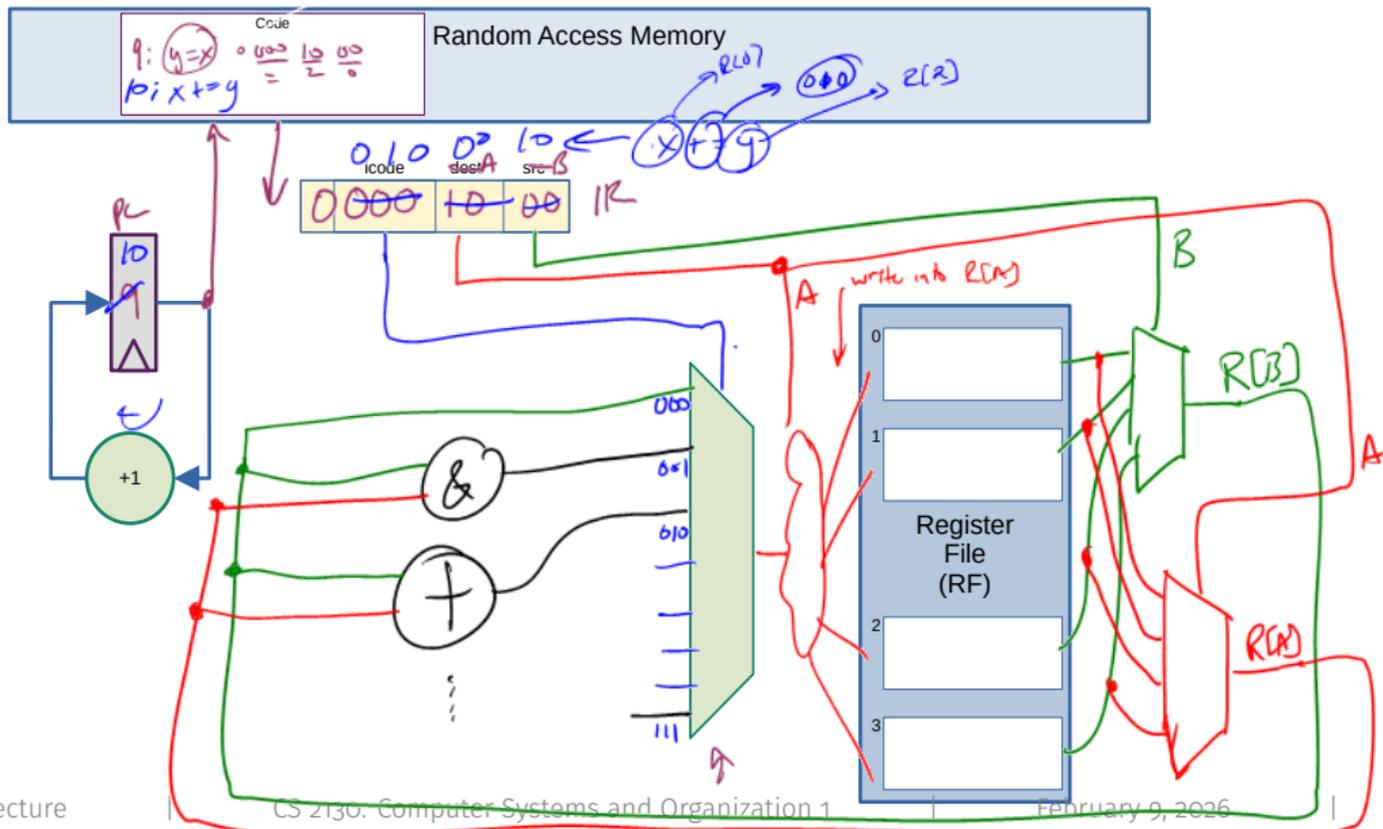| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |

Encoding Instructions

Encoding of Instructions

- 3-bit icode (which operation to perform)
    - Numeric mapping from icode to operation
- Which registers to use (2 bits each)
- Reserved bit for future expansion

Building a Computer

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

$$0000\ 0000$$
$$=\quad 0\quad 0\quad \longrightarrow\quad R[0] = \ell[0]$$

no op

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

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` ← M[p+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

X=16

M[p+1]

x= M[16]

M[M[pc+1]]

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