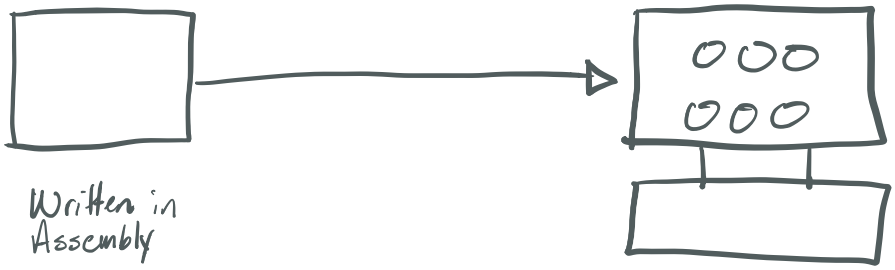
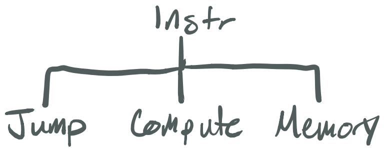


① Design an ISA



1st thing to decide is how many?

↳ N ↳ $\log_2 N$

Jump:

- Conditional
- Unconditional

Compute:

- Arithmetic
- Logical

Memory

- Read
- Write

Example: Add R1, R2

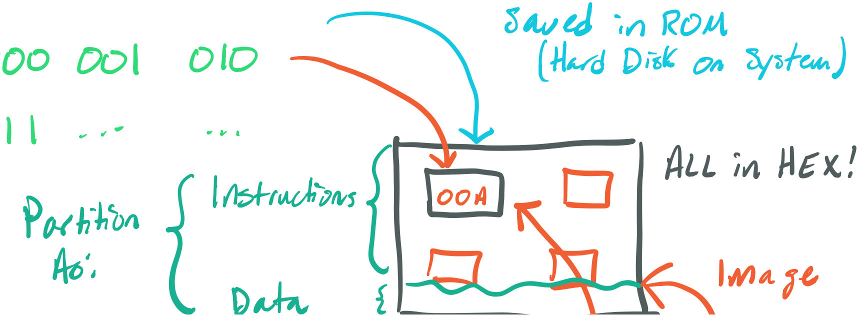
↑ Operation

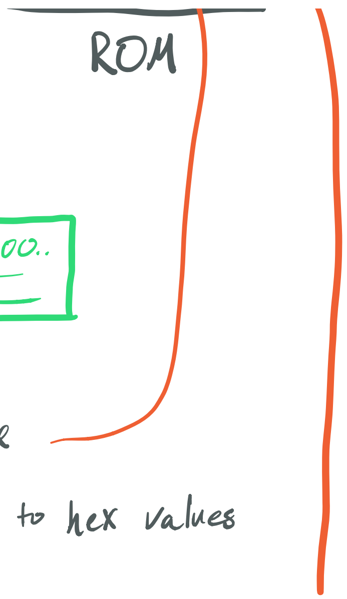
② How many registers?

Example: Add R1, R2

0000 001 010

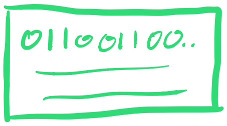
1111 ... ...





When compiling, you get a.out

↳ It is a binary executable



In the ROM, we convert to hex

- We need a script that converts code to hex values

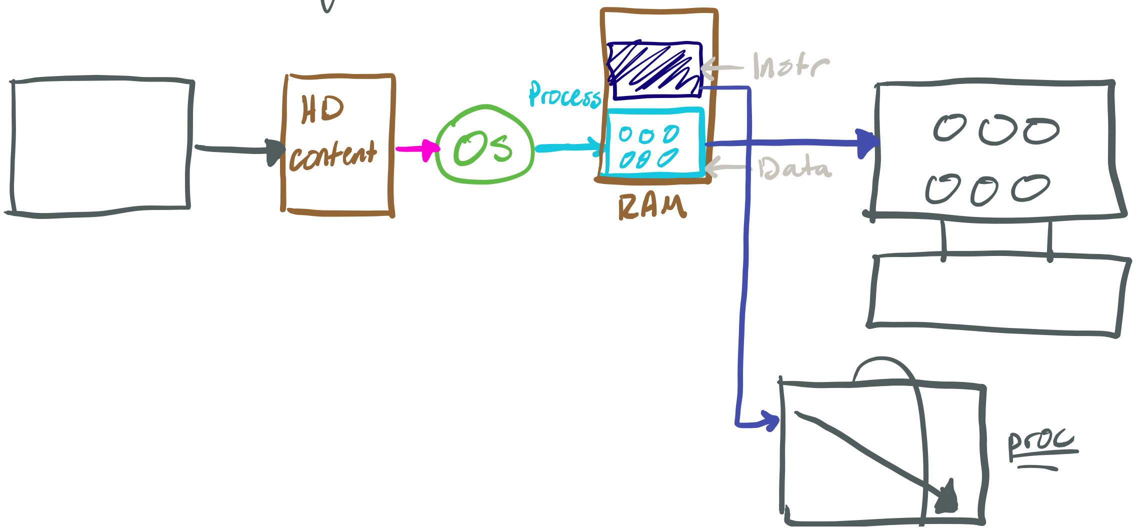
Contains your program as a series of Hex code. Stored in ROM before simulation

Need to implement ISA on hardware!

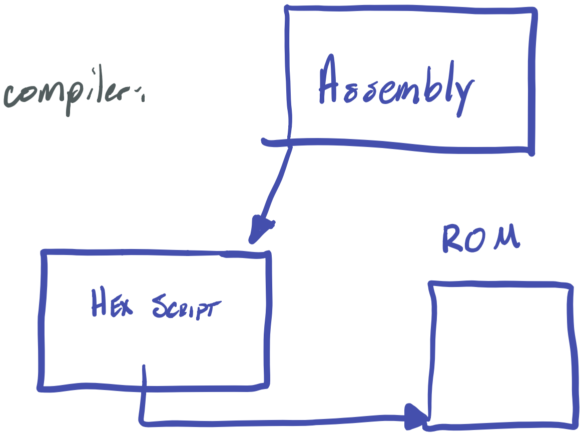
↳ Different ops need different modules

- Becomes OP code
- Tells series of steps needed for doing an operation

Back to Big Picture



At compiler:



We type our stuff in HEX into the ROM block and then we can run

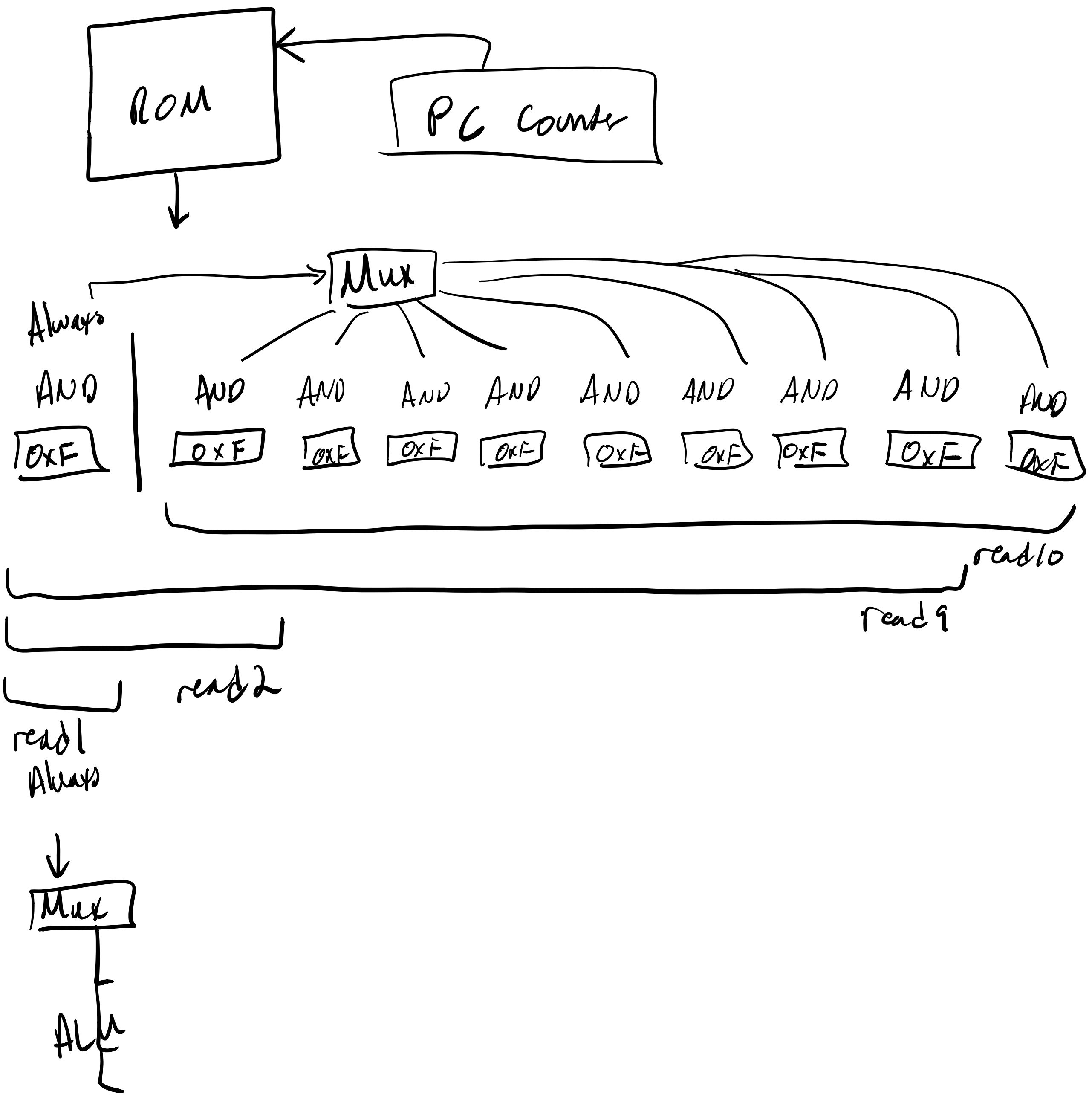
Must have:
ADD
SUB
MUL

AND
OR

Unconditional JMP
Conditional JMP

Also consider Matrix multiplication w/ JMP

Match Y86 opcodes



add	0 0
sub	1 0
mul	2 0
div	3 0
mrmov	4 0
rmmov	5 0
rrmov	6 0
irmov	7 0
immov	8 0
jmp	9 0
jle	9 1
jl	9 2
je	9 3
jne	9 4
jg	9 5
jge	9 6
halt	A 0