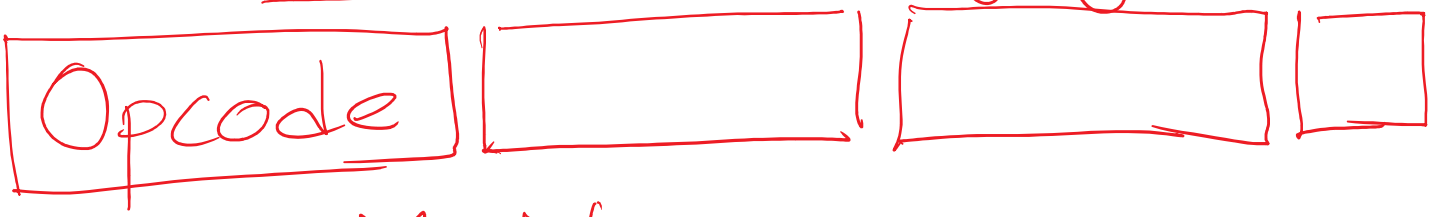
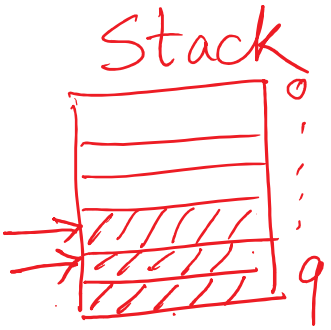
Let 12

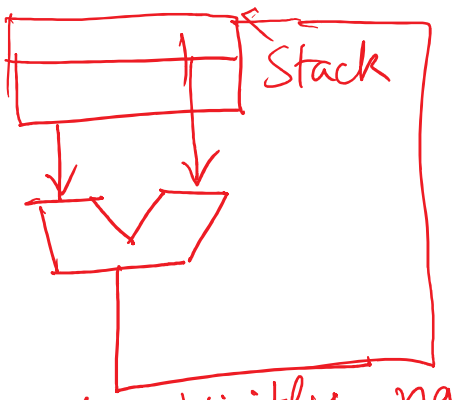
Thursday, June 2, 2022 12:00 PM

Assembly Language



Machine Models

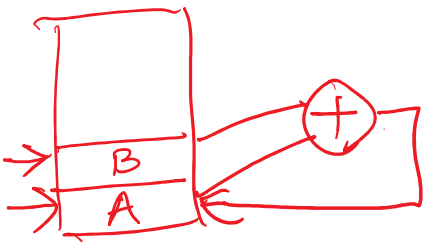




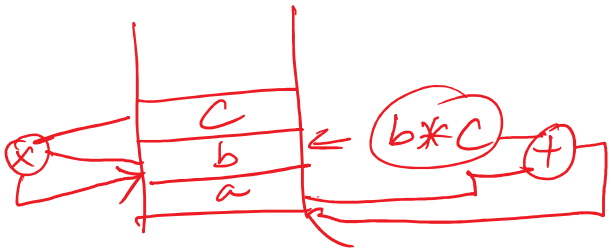
# of explicitly named operand '0'

C = A + B

Push A
Push B
Add
Pop C



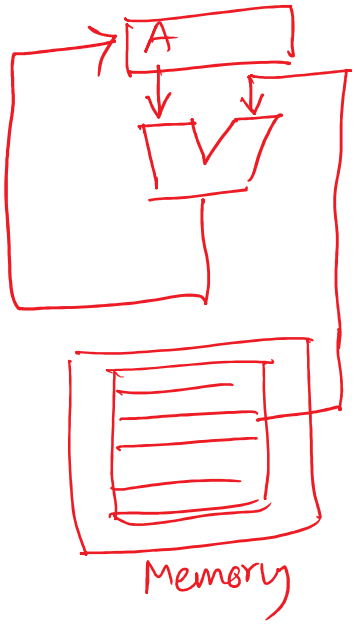
a + b * c



push a

push b
push C
Mul
Add
Pop

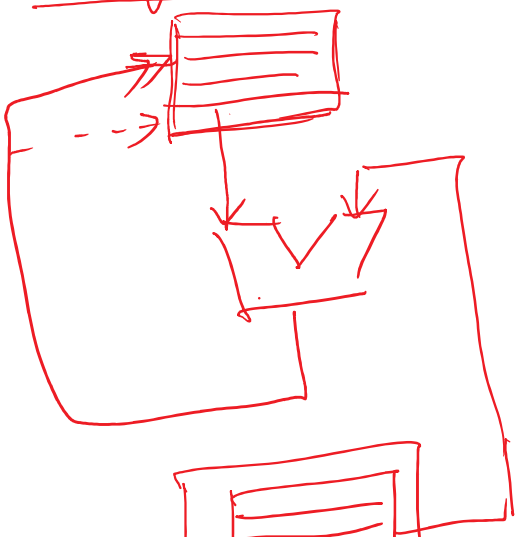
Accumulator



C = A + B
Load A
ADD B
Store C

# of explicitly named operand 1.

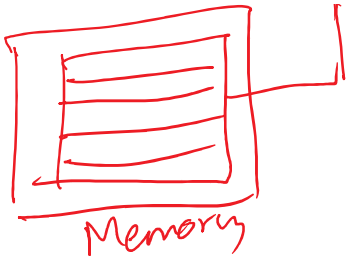
Register - Memory



C = A + B

Load R1, A
ADD R3, R1, B
Store R3, C

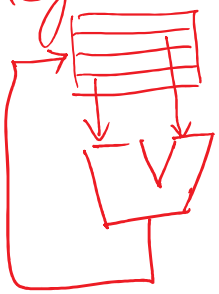
ADD R1, B



# of explicitly named operand 3 or 2

in x86 ISA, 1st operand is always destination as well.

Register - Register



Load R1, A
load R2, B
ADD R3, R1, R2

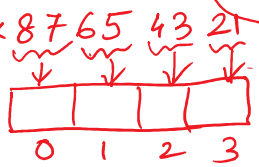
ADD R1, R2 # 2 or 3

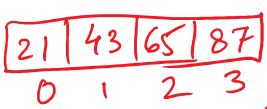
Registers: named storage locations.
General Purpose Registers
ARM: r0 .... r15

Stack Pointer
Program Counter
Return address
} Special registers.

PowerPC, Sparc

0x87654321 Big Endian

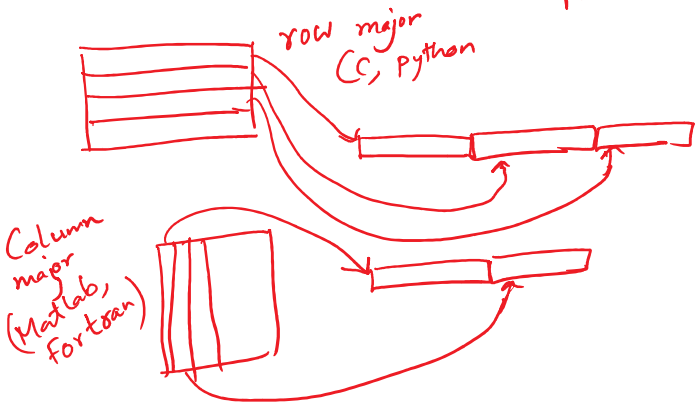




Little Endian
ARM, x86

row major ...thon

ARM,



Opcode Operand 1, ...

Operand:
- Constant (immediate)
- Register
- Memory

Types of Instructions
- Data Processing Instⁿ:
  add, sub, mul, div, com, logical operations
- Data transfer Instⁿ
  Load, store
- Branch Instr.
- Special Instr.

$r_1 \leftarrow r_2$

$r_1 \leftarrow r_2 + 4$

$r_1 \leftarrow [r_2]$