

Number of Addresses in Instruction

- 3 addresses
 - Operand 1, Operand 2, Result
 - ADD a,b,c ($a = b + c$;))

Example: $Y = (A - B) / (C + DE)$

Instruction	Comment
SUB Y, A, B	$Y \leftarrow A - B$
MPY T, D, E	$T \leftarrow D \times E$
ADD T, T, C	$T \leftarrow T + C$
DIV Y, Y, T	$Y \leftarrow Y / T$

Number of Addresses

- 2 addresses
 - One address doubles as operand and result
 - ADD a,c (a = a + b)
 - Reduces length of instruction
 - Requires some extra work
 - Temporary storage to hold some results

<u>Instruction</u>	<u>Comment</u>
MOVE Y, A	$Y \leftarrow A$
SUB Y, B	$Y \leftarrow Y - B$
MOVE T, D	$T \leftarrow D$
MPY T, E	$T \leftarrow T \times E$
ADD T, C	$T \leftarrow T + C$
DIV Y, T	$Y \leftarrow Y \div T$

Number of Addresses

- 1 address
 - Implicit second address
 - Usually a register (accumulator)
 - ADD B (AC = AC + B)
 - Common on early machines

<u>Instruction</u>	<u>Comment</u>
LOAD D	AC ← D
MPY E	AC ← AC × E
ADD C	AC ← AC + C
STOR Y	Y ← AC
LOAD A	AC ← A
SUB B	AC ← AC - B
DIV Y	AC ← AC ÷ Y
STOR Y	Y ← AC

Number of Addresses

- 0 (zero) addresses
 - Applicable to a special memory organization called **Stack**
 - Stack is known location
 - Often at least the top two stack elements are in processor registers
- ADD
- All addresses implicit

Push A	Add
Push B	Div
Sub	Pop Y
Push D	
Push E	
Mult	
Push C	

Number of Addresses

- 4 addresses
 - Operand 1, Operand 2, Result, and next instruction
 - Not common
 - Needs very long words to hold everything

Number of Addresses

Number of Addresses	Symbolic Representation	Interpretation
3	OP A, B, C	$A \leftarrow B \text{ OP } C$
2	OP A, B	$A \leftarrow A \text{ OP } B$
1	OP A	$AC \leftarrow AC \text{ OP } A$
0	OP	$T \leftarrow (T - 1) \text{ OP } T$