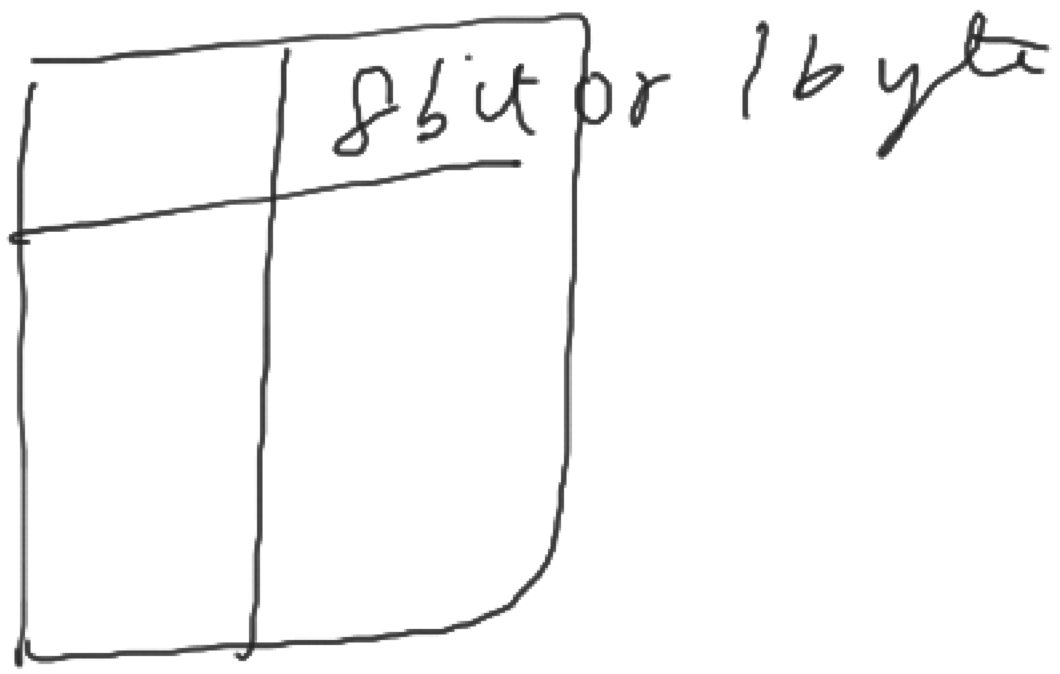
Big Endian Vs little Endian

→ Byte ordeing

Byte = 8 bit

32 bit

64 bit



V data $\Rightarrow$ A

Addr	content
A	V_1
B	V_2
C	
D	

$V_1 \rightarrow$ 1 byte

$V_2 \rightarrow$ 1 byte

$V \rightarrow$ 8 bit 16 bit

$\rightarrow$ 1 byte

V_1 8	8 V_2
1 byte	1 byte

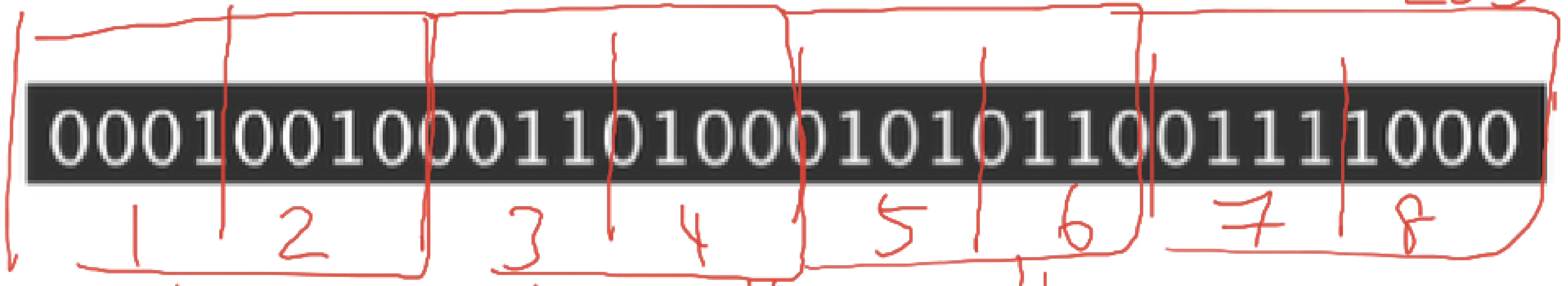
32 bit ⇒ 100

$\boxed{8 \text{ bit}}$

1 byte ⇒ 8 bit

MSB

LSB


00010010001101000101011001111000

1 byte　1b　1b　1b

12　34　56　78

MSB　LSB

Big Endian

Add.	Content
100	12
101	34
102	56
103	$\bigcirc$78
104	

$12 \rightarrow$ MSB 00010010

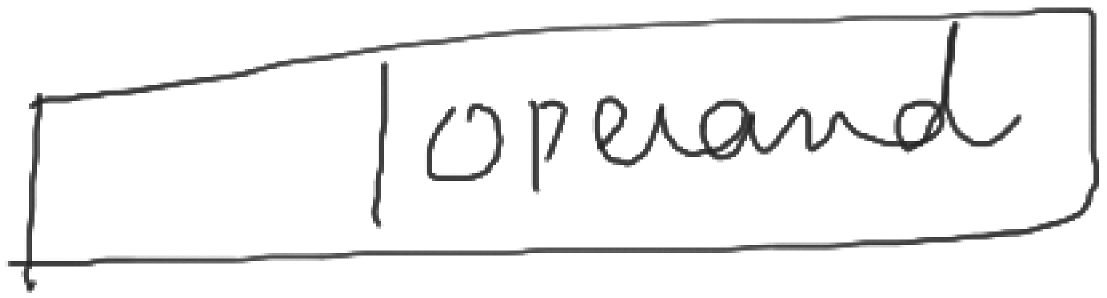
little End

Add.	Content
100	$\bigcirc$78
101	56
102	34
103	12

Instruction set : Addressing Set ⑦

① Immediate

$(1)0111$
↳ signed bit

↳ 

2's comp

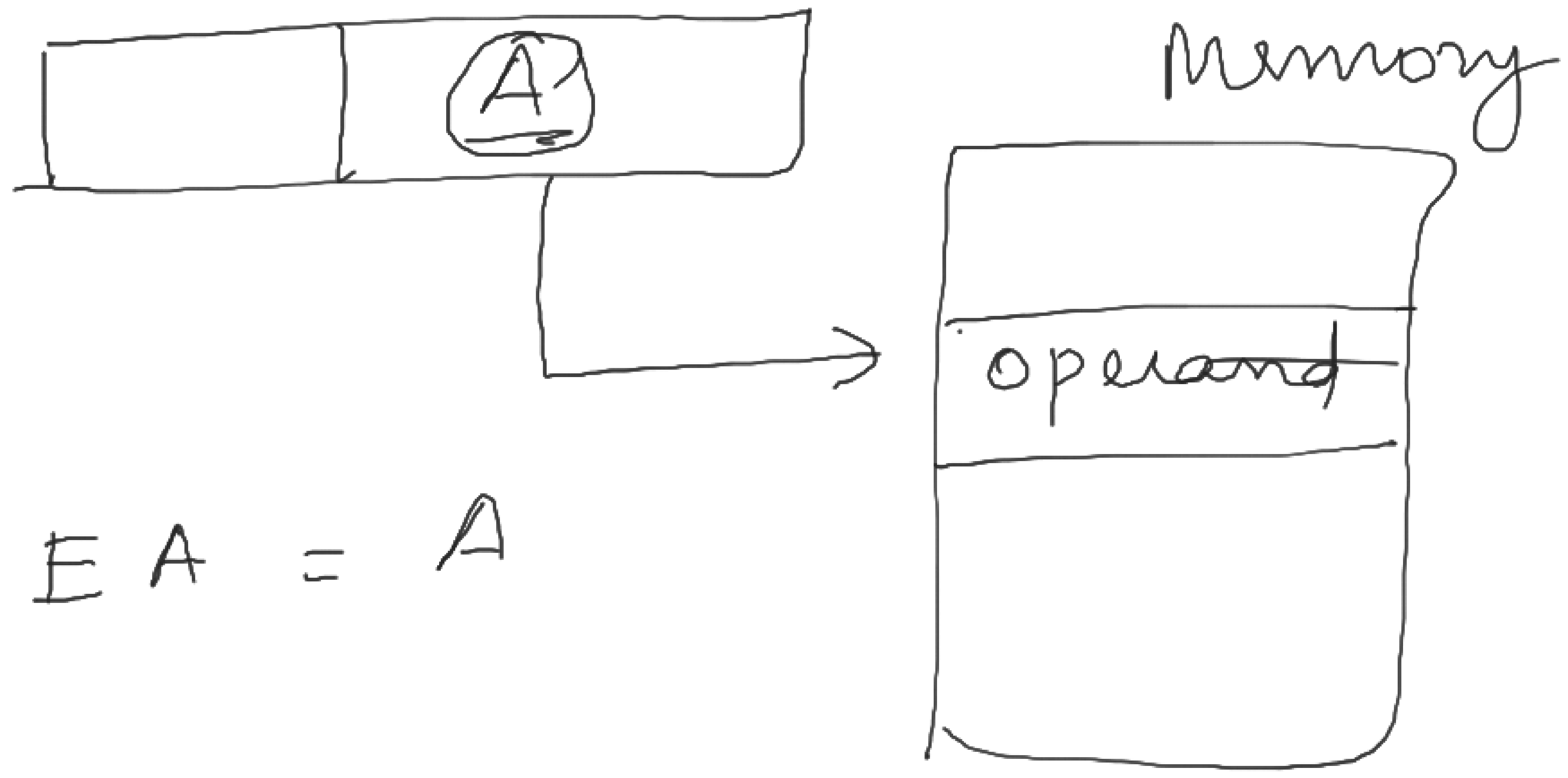
↳ Algorithm Operand = $\boxed{A} \rightarrow$ 2's comp

↳ simplest of Addressing

Ad → No memory reference

dis Ad → : - limited operand magnitude

② Direct Addressing mode

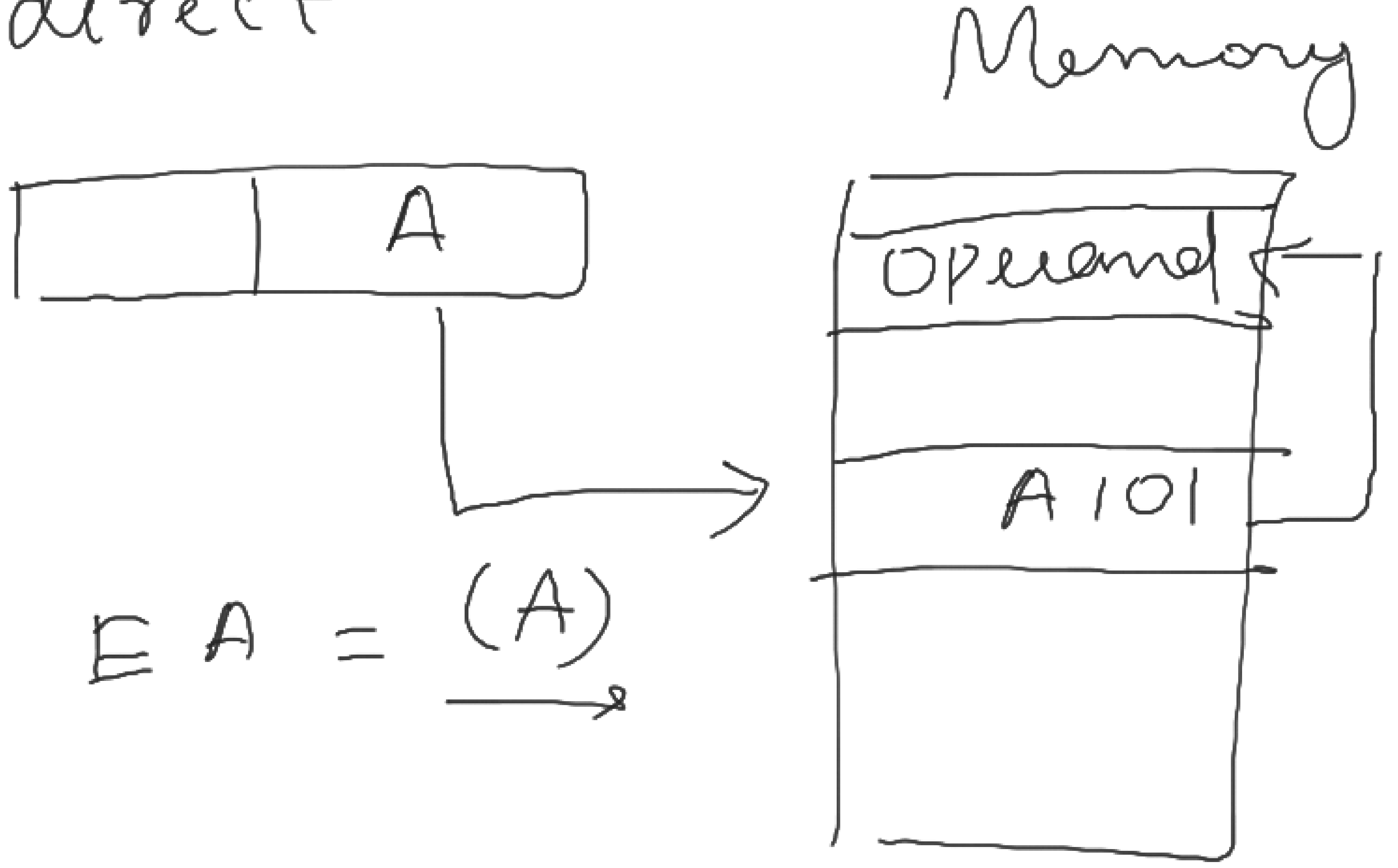


→ Algo $EA = A$

Ad :- req only one memory reference

disad :- limited address space

3 Indirect

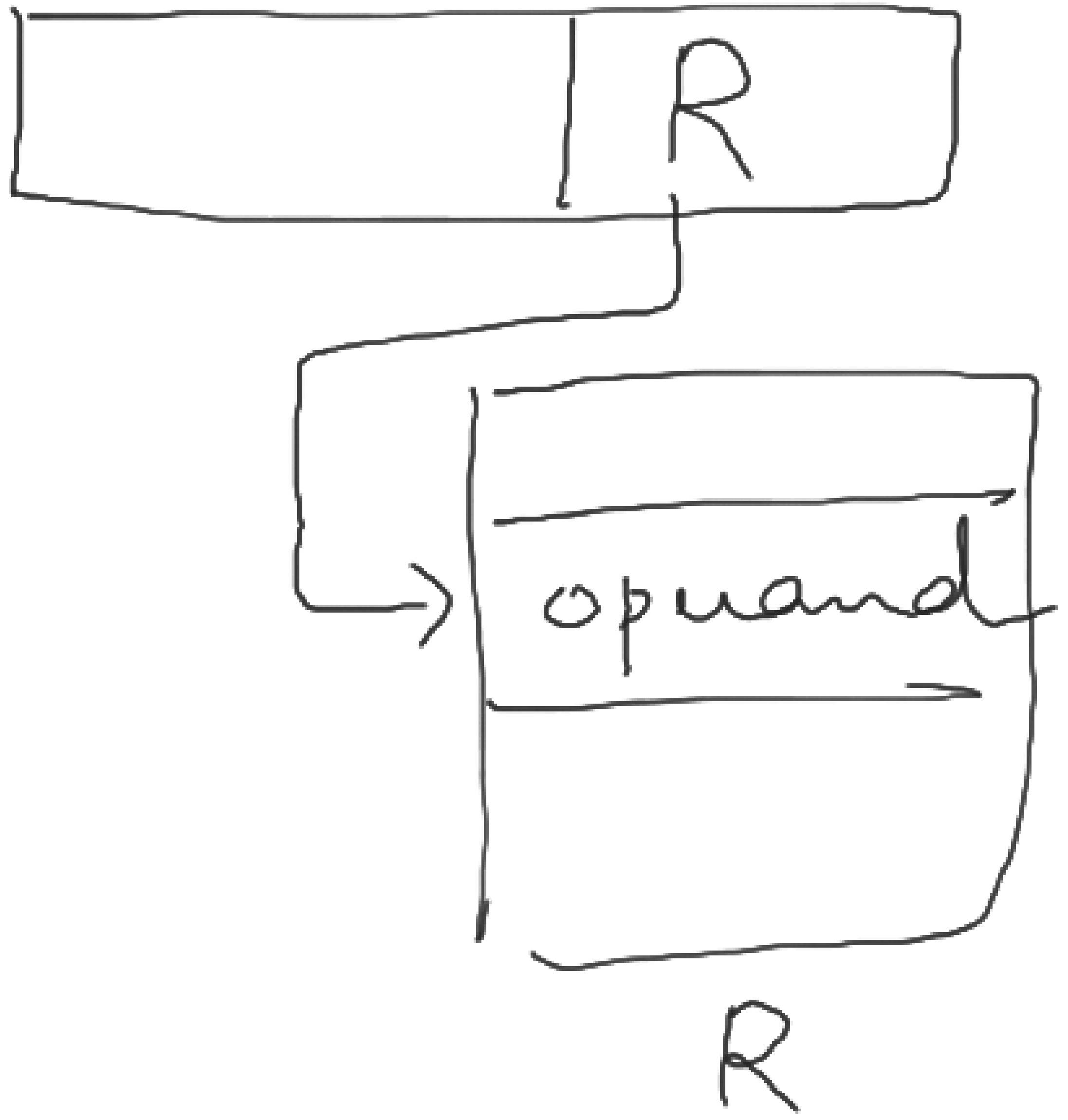


→ Algo $EA = \underline{(A)}$ →

Ad → word length N has $\underline{\underline{2^N}}$ address space
Large Address space

disad → multiple memory ref.

(4) Register Addressing



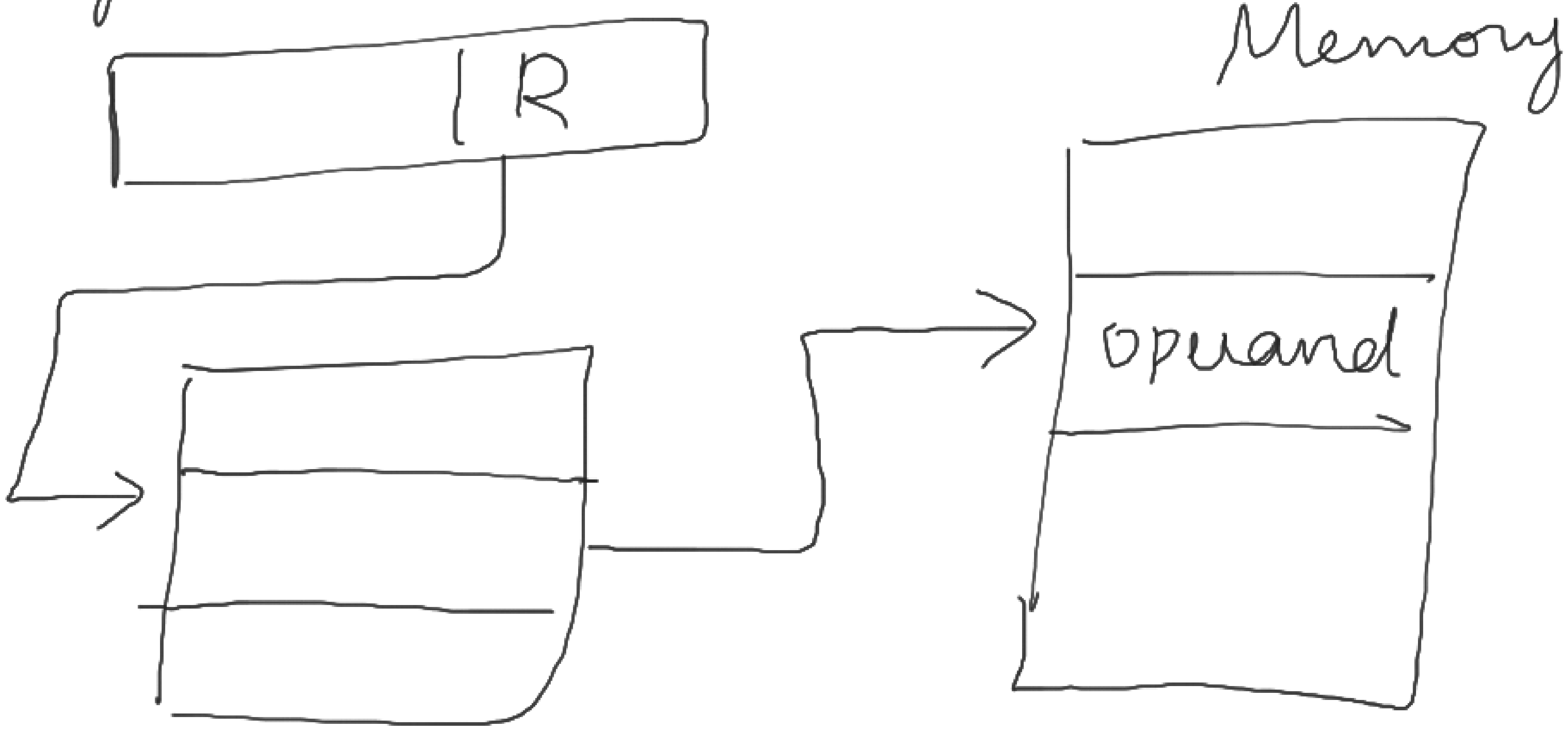
→ Address field refers to the Reg. rather than memory

Algo → EA = R

Ad → no need for time - consuming memory ref.

disad → limited address space

(5) Register Indirect Addressing



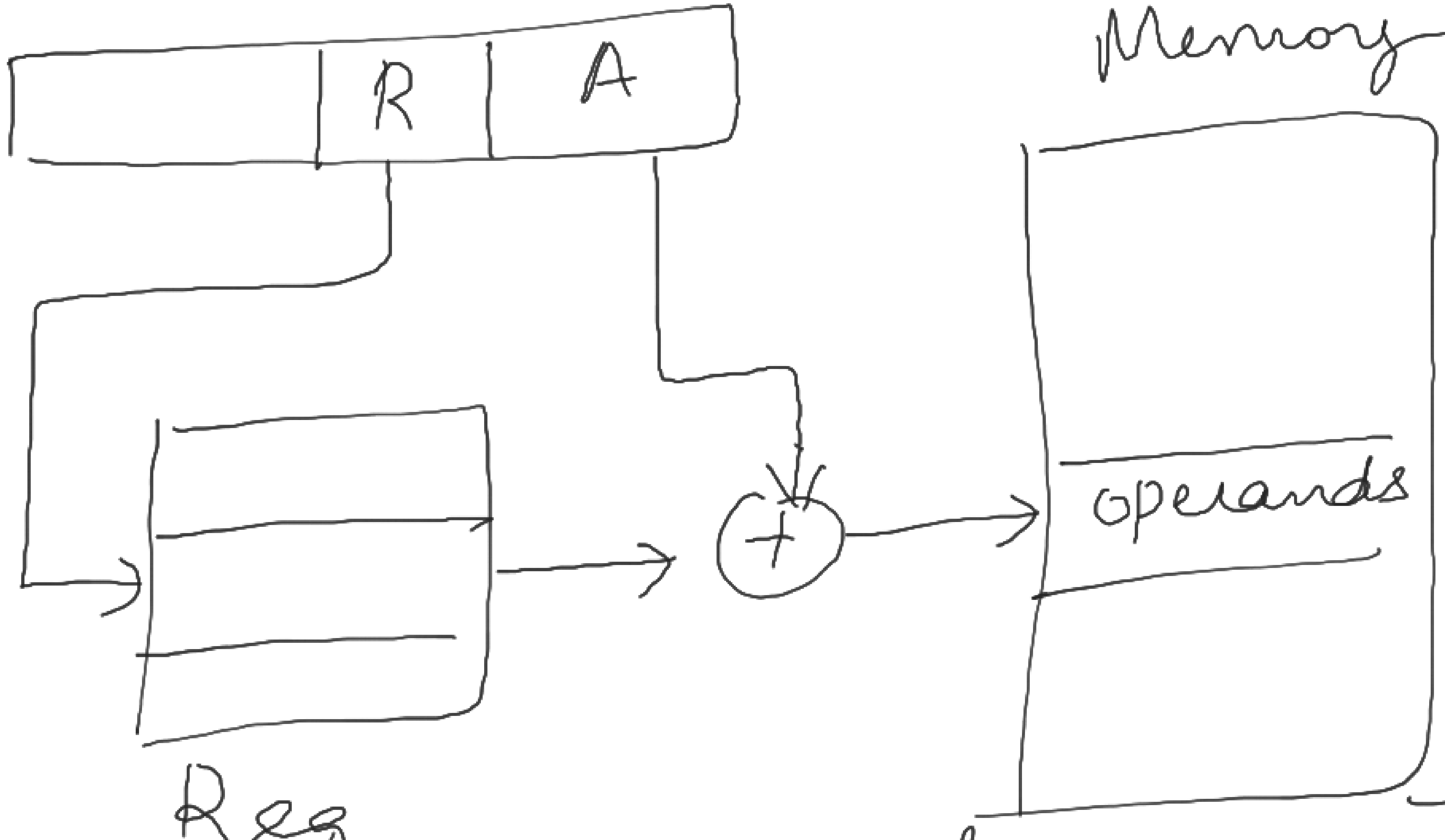
Reg

Algo $\rightarrow$ $EA = (R)$

Ad $\rightarrow$ Large address space

disad $\rightarrow$ Extra memory ref

⑥ Displacement Addressing mode



→ Two add

① direct add
A

② register
R

Ad → flexibity

→ direct ~~Add~~ + reg indirect

disad → complexity

Algo → $EA = A + (R)$

⑥ a) Relative Addressing

↳ reference Reg is PC

↳ Locality ✗

⑥ b Base- Reg Add

↳ ref reg contains → main memory Add + displacement

↳ may be explicit or implicit

(6) c) Indexing

↳ contains positive displacement

(i) Autoindexing :- $EA = \underline{A} + \underline{(R)}$

$(R) \leftarrow (R) + 1$

(ii) post indexing $EA = (A) + (R)$

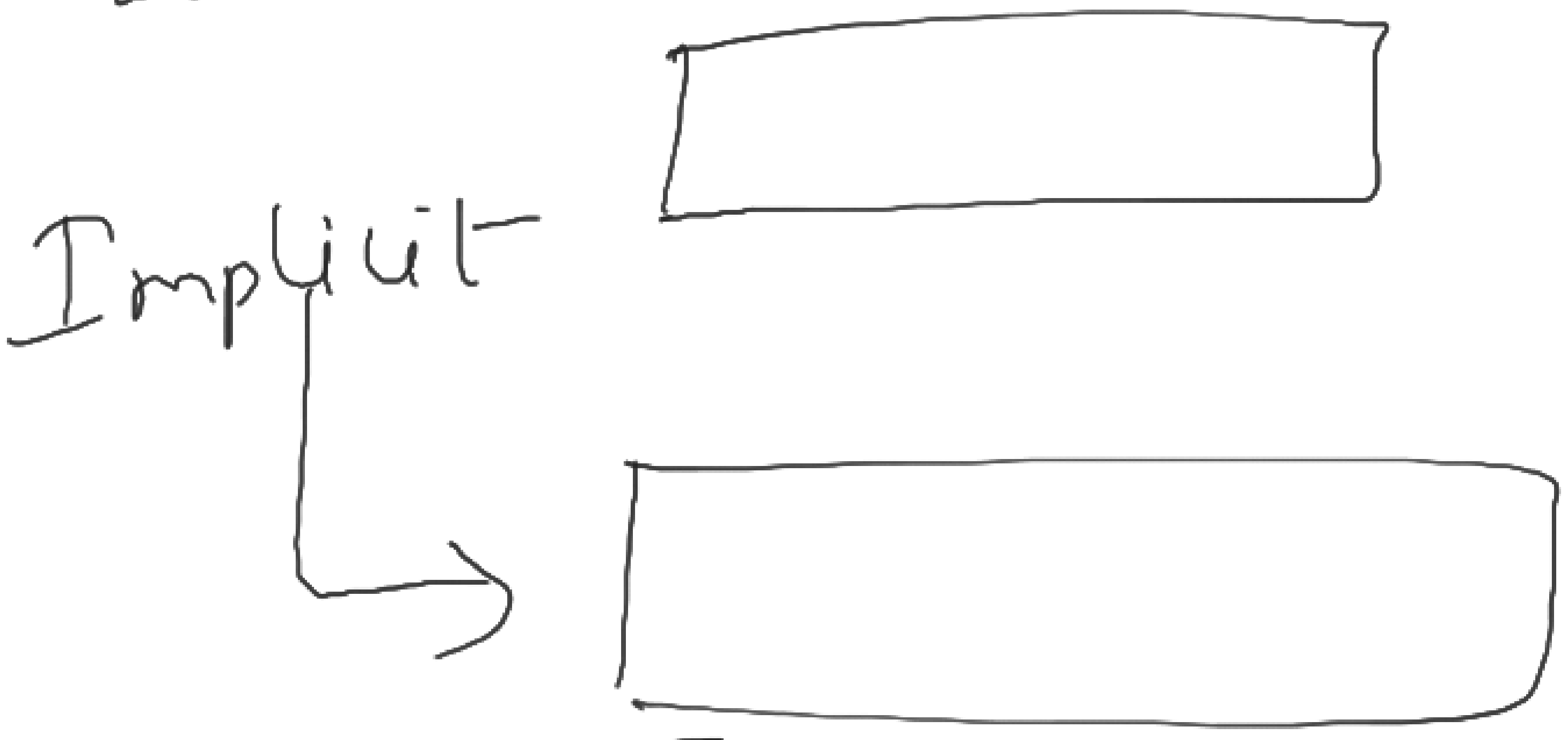
(iii) pre indexing $EA = (A + (R))$

(7) Stack Addressing

LIFO

$\rightarrow$ stack LIFO

$\hookrightarrow$ linear array of location



Top of Stack

Algo

EA = top of stack

Ad $\rightarrow$ no memory ref

disad :-

limited Applicability

Addressing mode	Algo	Ad	dis ad
Immediate	operand = A	No mem ref	limited operand mag
direct	EA = A	only one mem ref	limited addr space
Indirect	EA = (A)	Large add space	more than one mem ref
Reg	EA = R	time consuming	limited add space
Reg Indirect	EA = (R)	large add space	Extra memory ref
displacement	EA = A + (R)	flexibility	complexity
Stack	EA = Top of stack	No mem ref	limited applicability