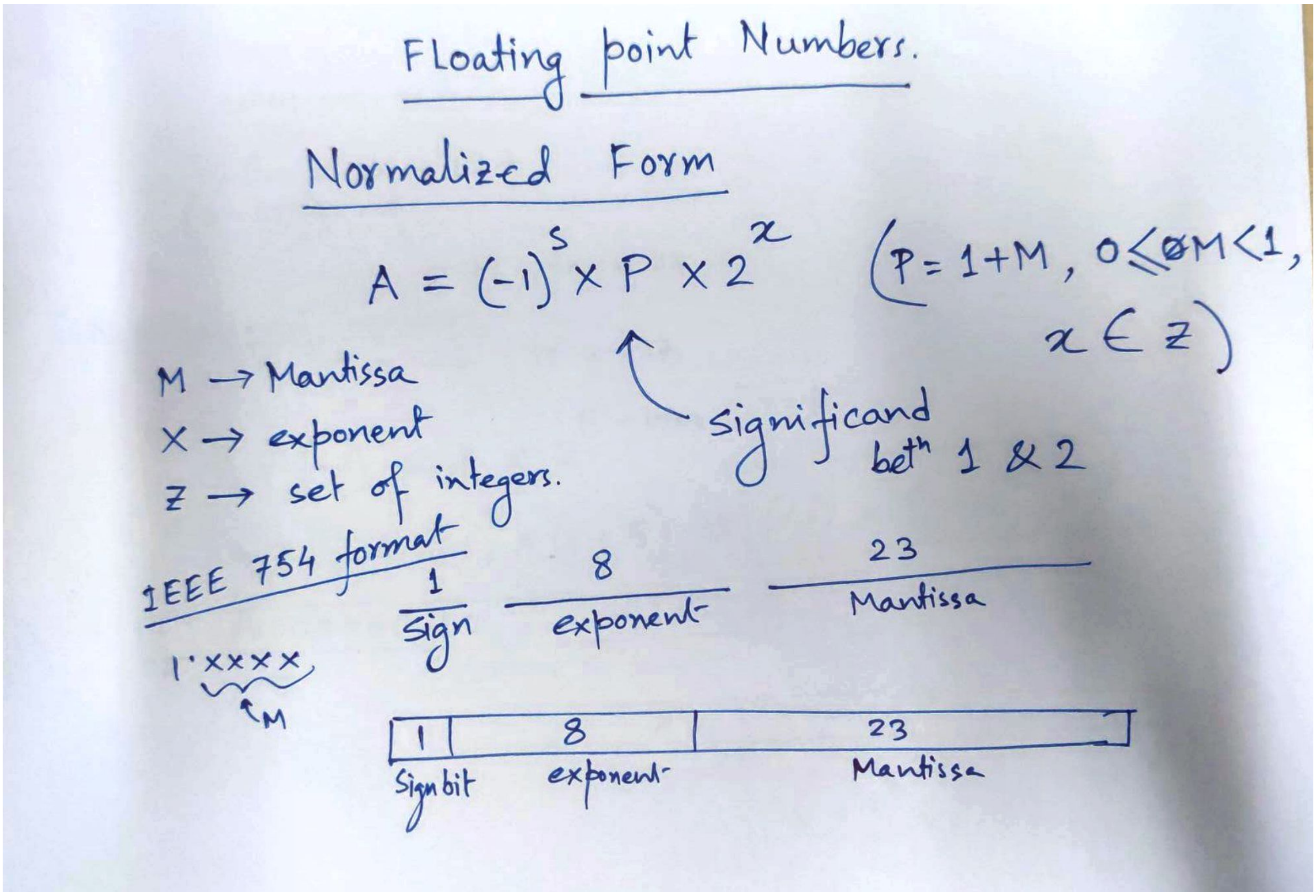


Floating point Numbers.

Normalized Form

$$A = (-1)^{s} \times P \times 2^{x} \quad \left(P = 1 + M,\ 0 \leq \cancel{0} M < 1,\ x \in Z\right)$$

significand bet^{n} 1 & 2

M → Mantissa

X → exponent

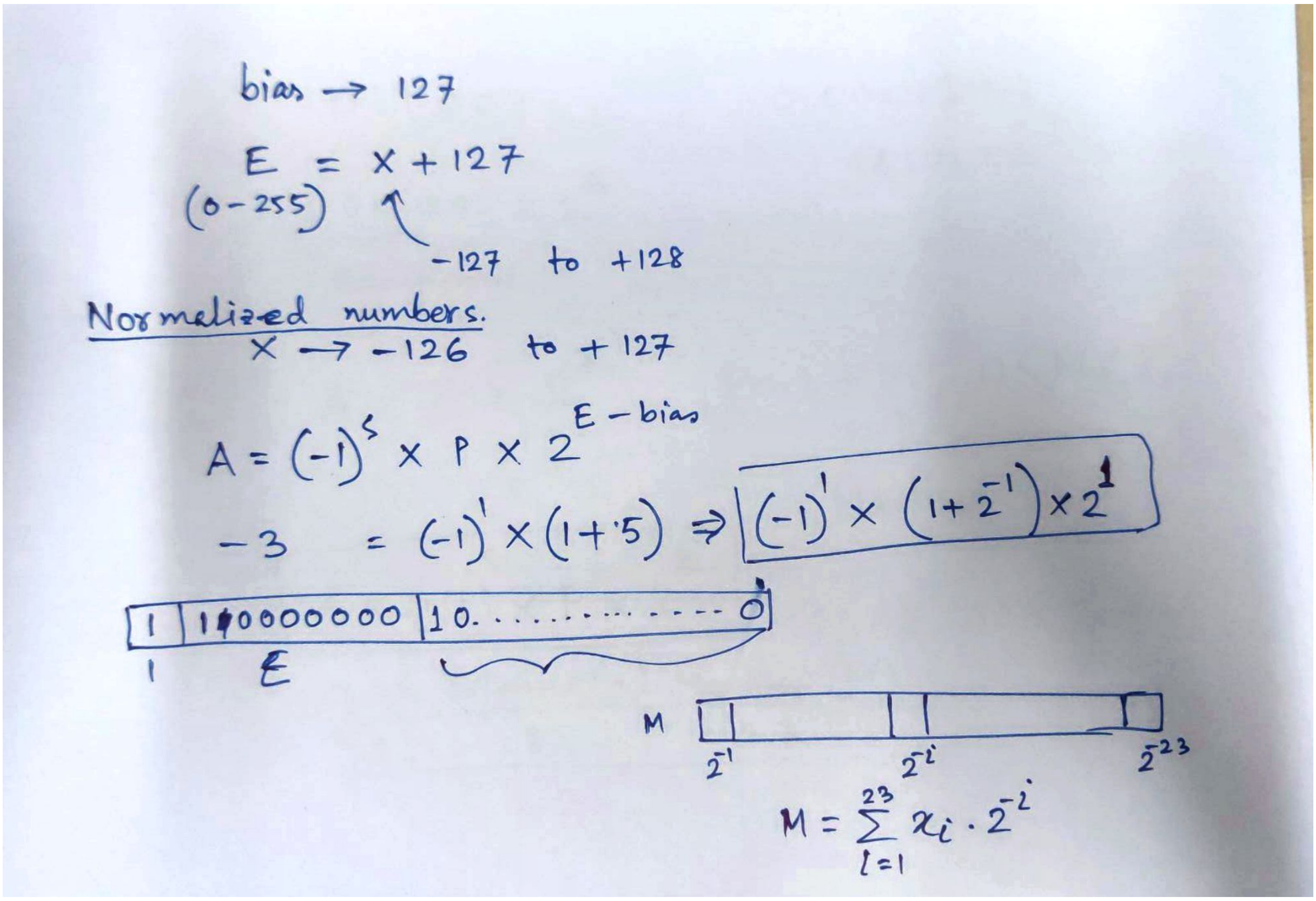
Z → set of integers.

IEEE 754 format

1	8	23
Sign	exponent	Mantissa

$1\cdot\underbrace{xxxx}_{M}$

1	8	23
Sign bit	exponent	Mantissa



bias → 127

$E = X + 127$

$(0-255)$ ↑ -127 to $+128$

Normalized numbers.

$X \rightarrow -126$ to $+127$

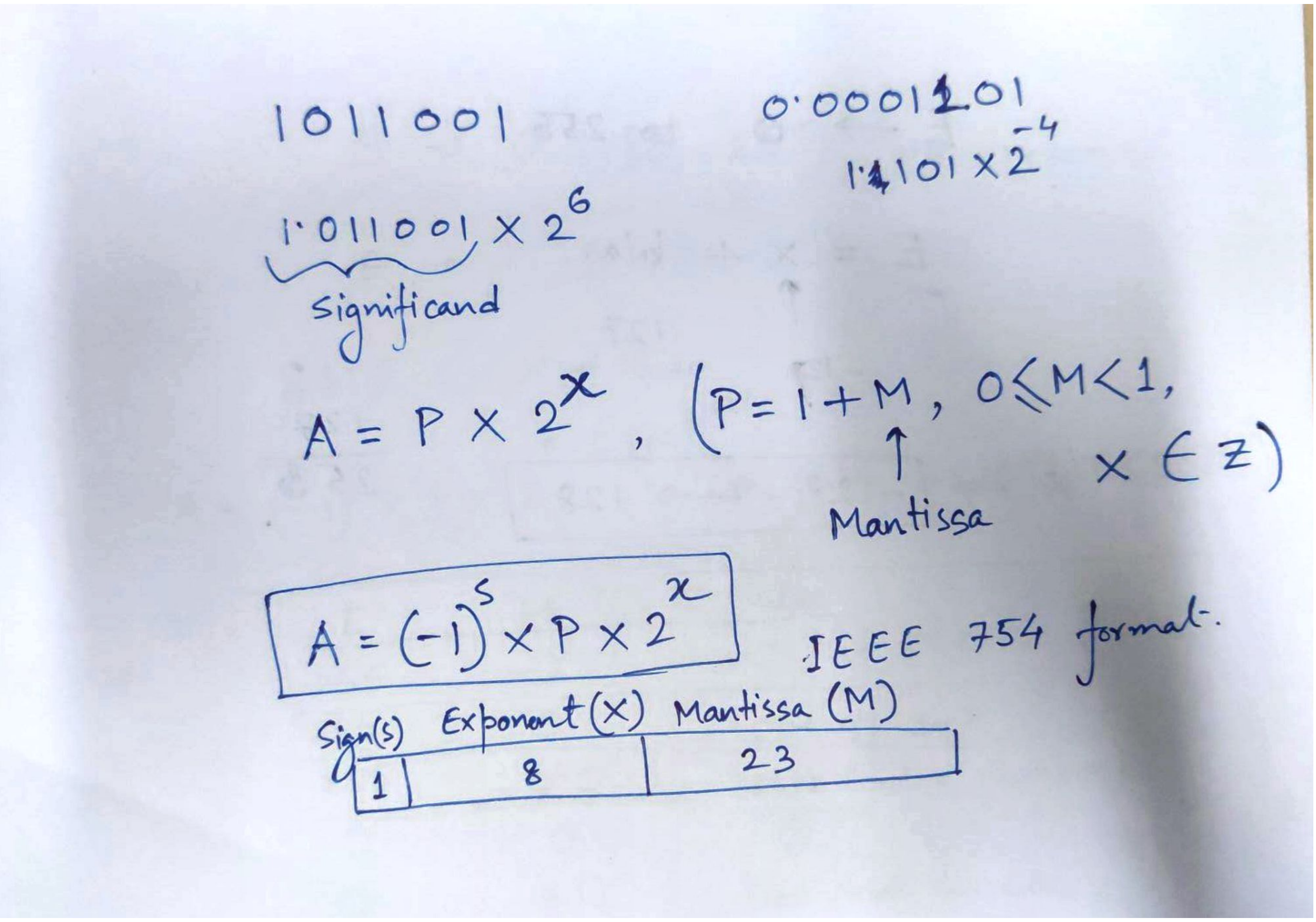
$A = (-1)^s \times P \times 2^{E-\text{bias}}$

$-3 = (-1)^1 \times (1 + .5) \Rightarrow \boxed{(-1)^1 \times (1 + 2^{-1}) \times 2^1}$

1	100000000	10. 0
1	E	

M

$$M = \sum_{i=1}^{23} x_i \cdot 2^{-i}$$



1011001

$$\underbrace{1\cdot011001}_{\text{significand}} \times 2^{6}$$

0·00011101

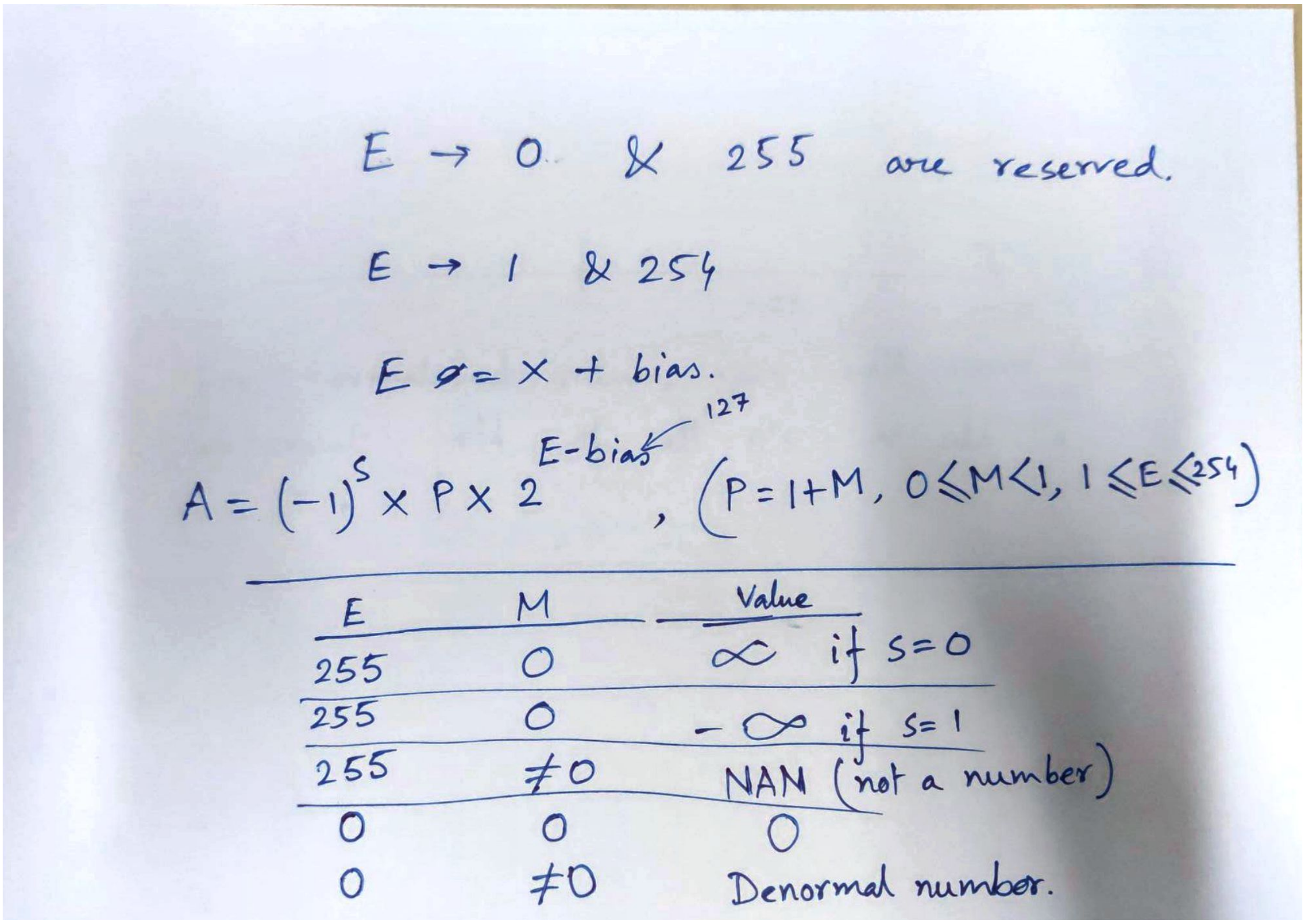
$1\cdot1101 \times 2^{-4}$

$$A = P \times 2^{x}, \quad \left(P = 1 + \underset{\substack{\uparrow \\ \text{Mantissa}}}{M}, \; 0 \leq M < 1, \; x \in \mathbb{Z}\right)$$

$$\boxed{A = (-1)^{s} \times P \times 2^{x}}$$

IEEE 754 format.

Sign(s)	Exponent(X)	Mantissa (M)
1	8	23



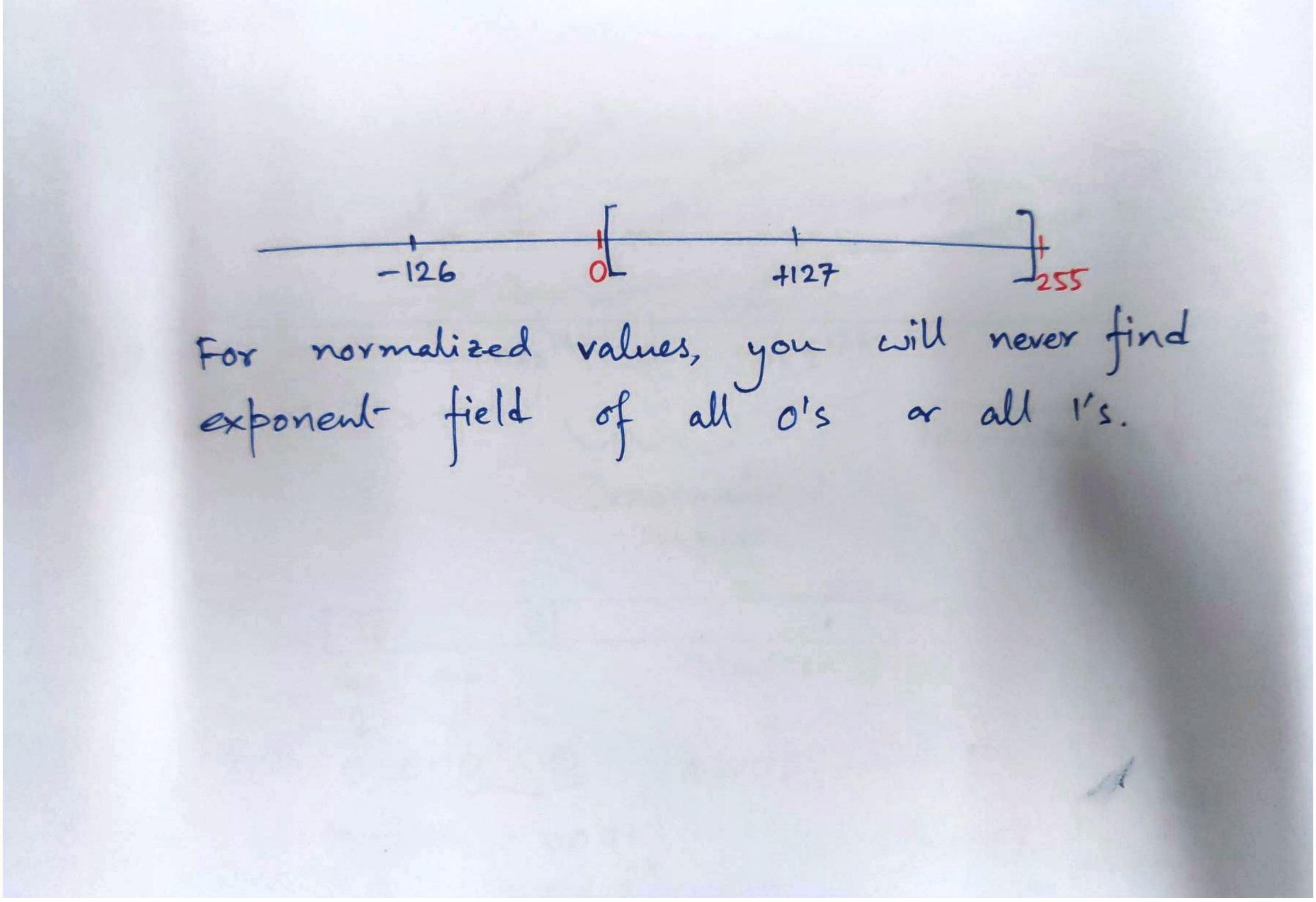
E → 0 & 255 are reserved.

E → 1 & 254

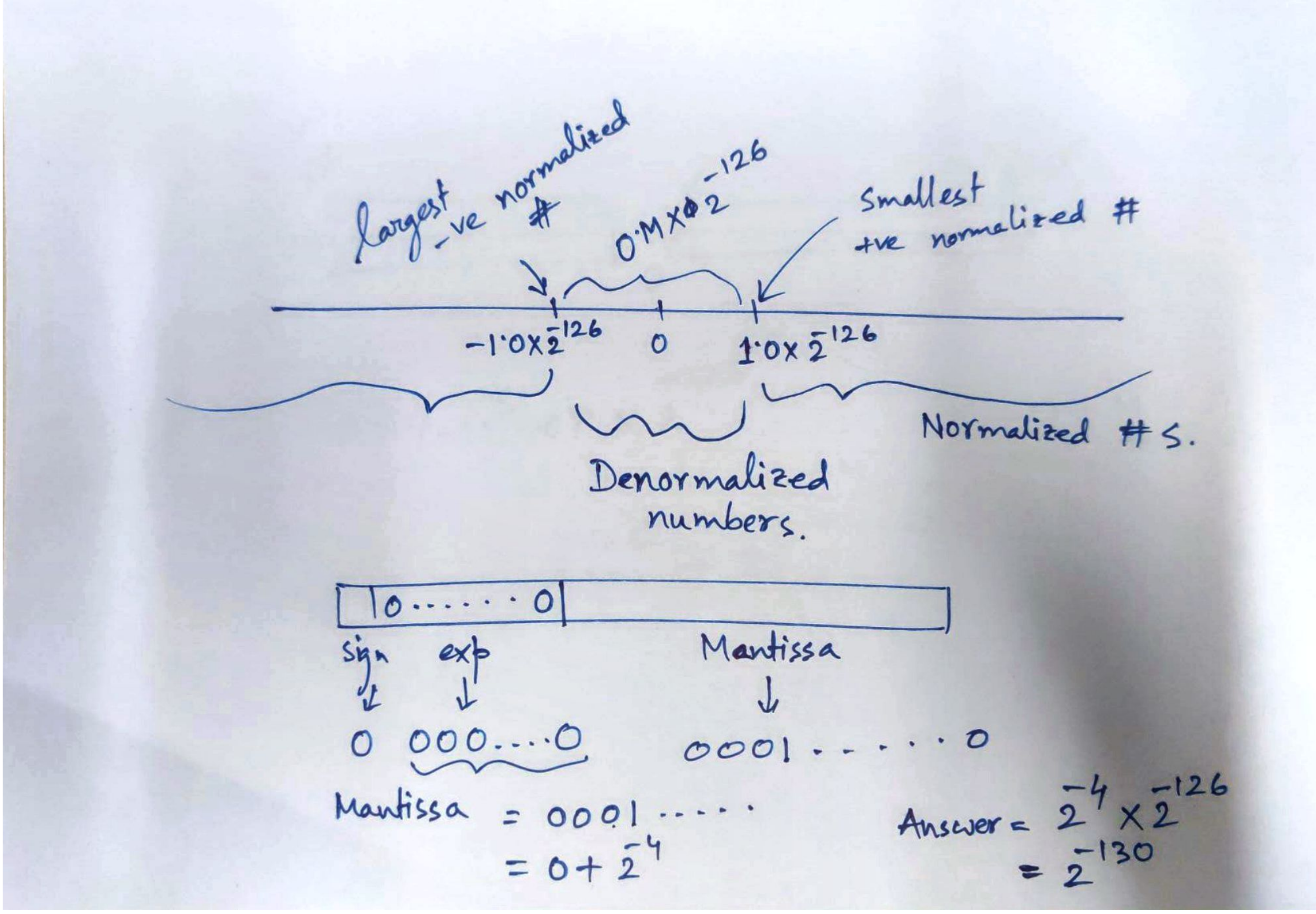
E ~~0~~ = x + bias.

$$A = (-1)^S \times P \times 2^{E-\overset{127}{\text{bias}}} , \left(P = 1+M,\ 0 \leqslant M < 1,\ 1 \leqslant E \leqslant 254\right)$$

E	M	Value
255	0	∞ if s=0
255	0	−∞ if s=1
255	≠0	NAN (not a number)
0	0	0
0	≠0	Denormal number.


−126
0
+127
255

For normalized values, you will never find exponent field of all 0's or all 1's.


largest -ve normalized #
$0 \cdot M \times 2^{-126}$
Smallest +ve normalized #
$-1 \cdot 0 \times 2^{-126}$
0
$1 \cdot 0 \times 2^{-126}$
Normalized #s.
Denormalized numbers.
| 0 . . . . . . 0 |
sign
exp
Mantissa

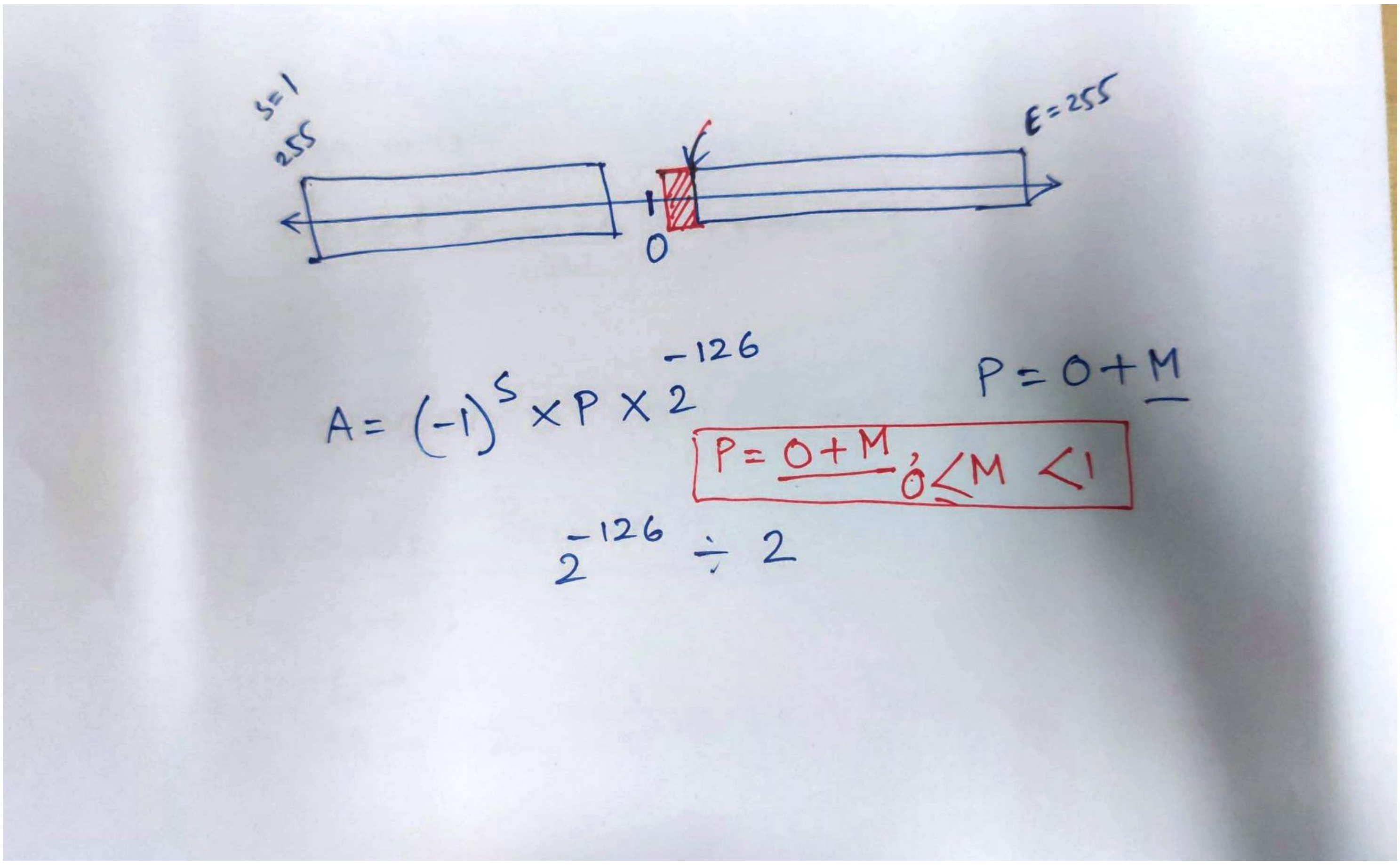
0 000....0 0001......0

Mantissa $= 0001 \ldots\ldots$

$= 0 + 2^{-4}$

Answer $= 2^{-4} \times 2^{-126}$

$= 2^{-130}$

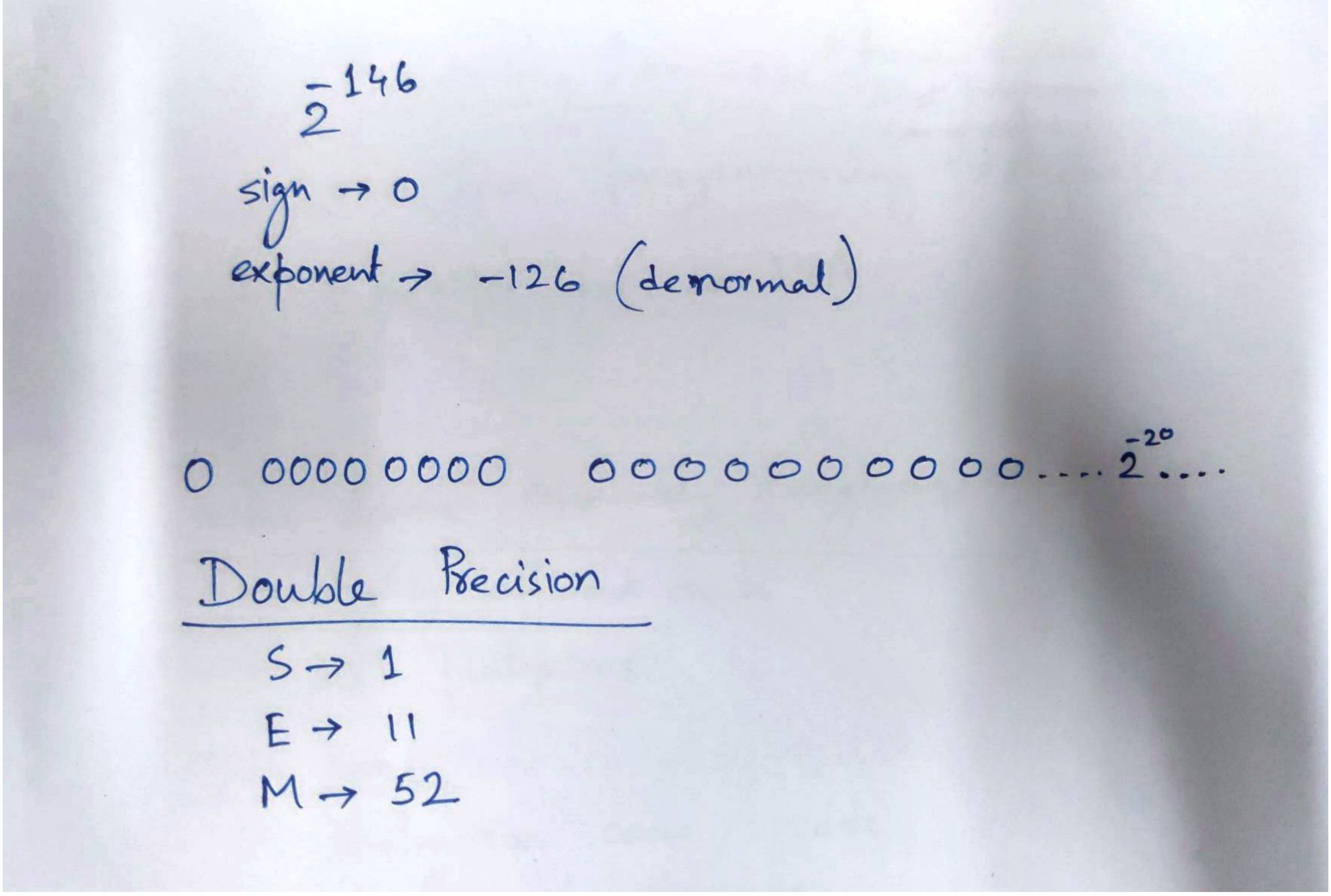


$$A = (-1)^S \times P \times 2^{-126}$$

$$P = 0 + \underline{M}$$

$$\boxed{P = \underline{0+M},\ 0 \leq M < 1}$$

$$2^{-126} \div 2$$



2^{-146}

sign → 0

exponent → −126 (denormal)

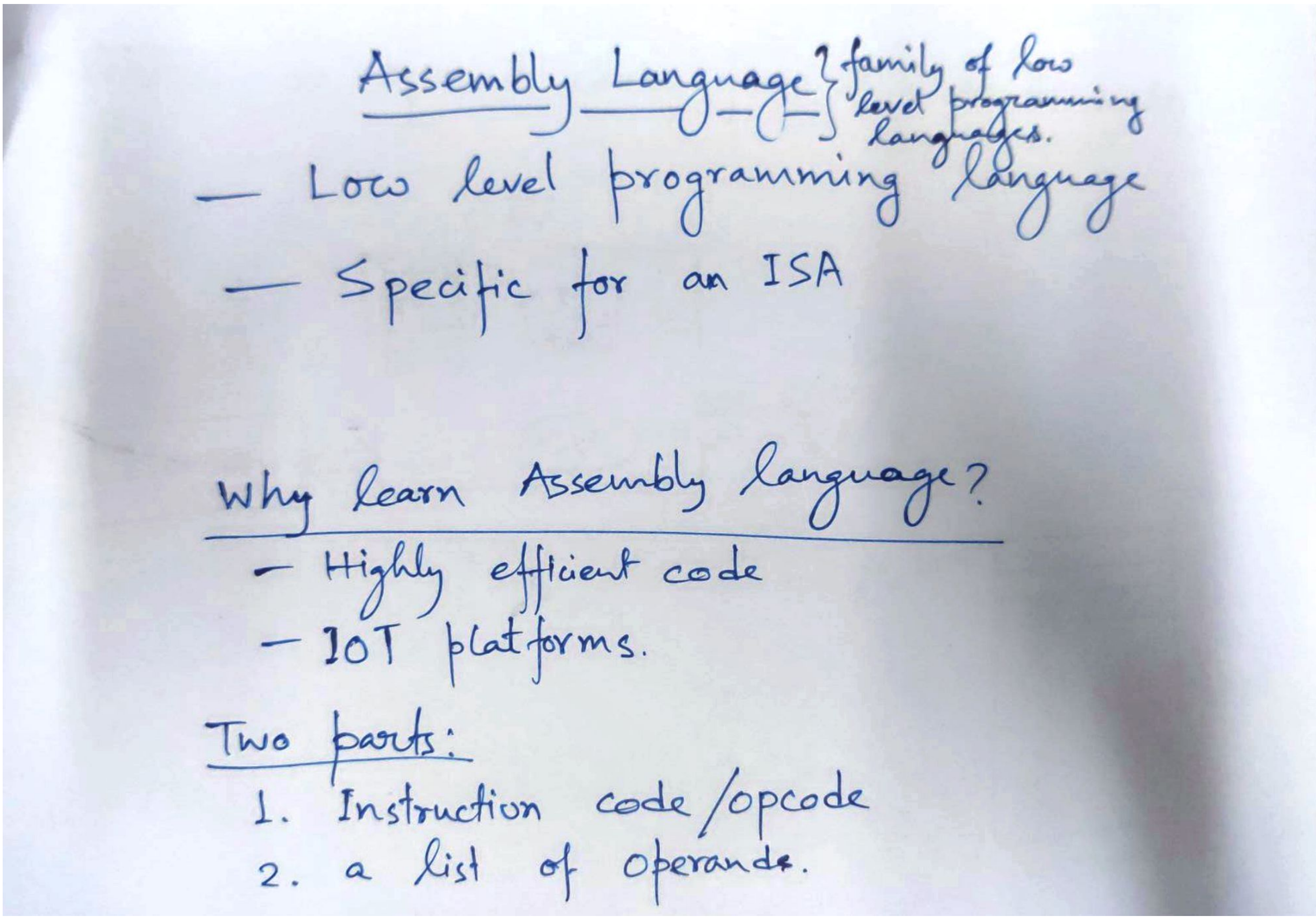
0 0000 0000 0000000000....2^{-20}....

Double Precision

S → 1

E → 11

M → 52



Assembly Language } family of low level programming languages.

— Low level programming language

— Specific for an ISA

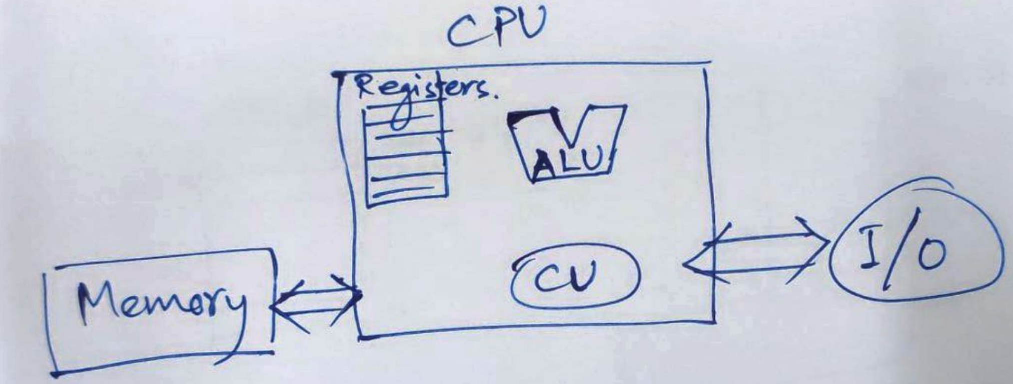
Why learn Assembly language?

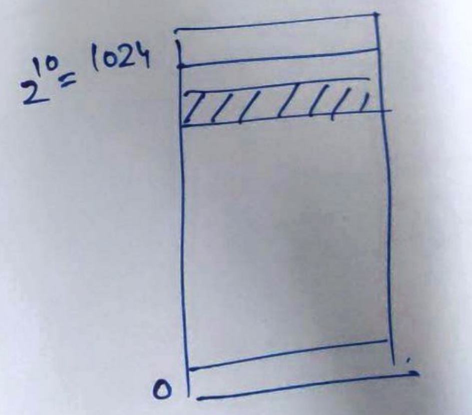
- Highly efficient code
- IOT platforms.

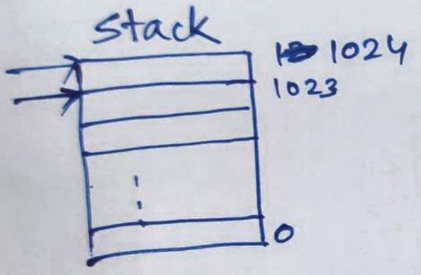
Two parts:

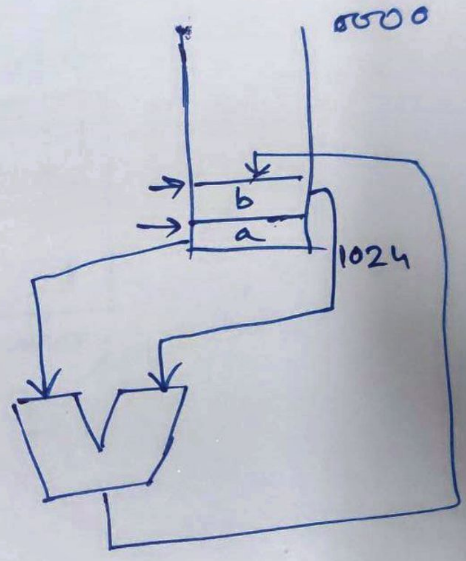
1. Instruction code/opcode
2. a list of operands.

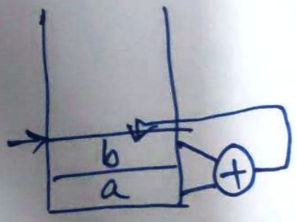
* Where do operands come from & where do results go?











C = A+B.

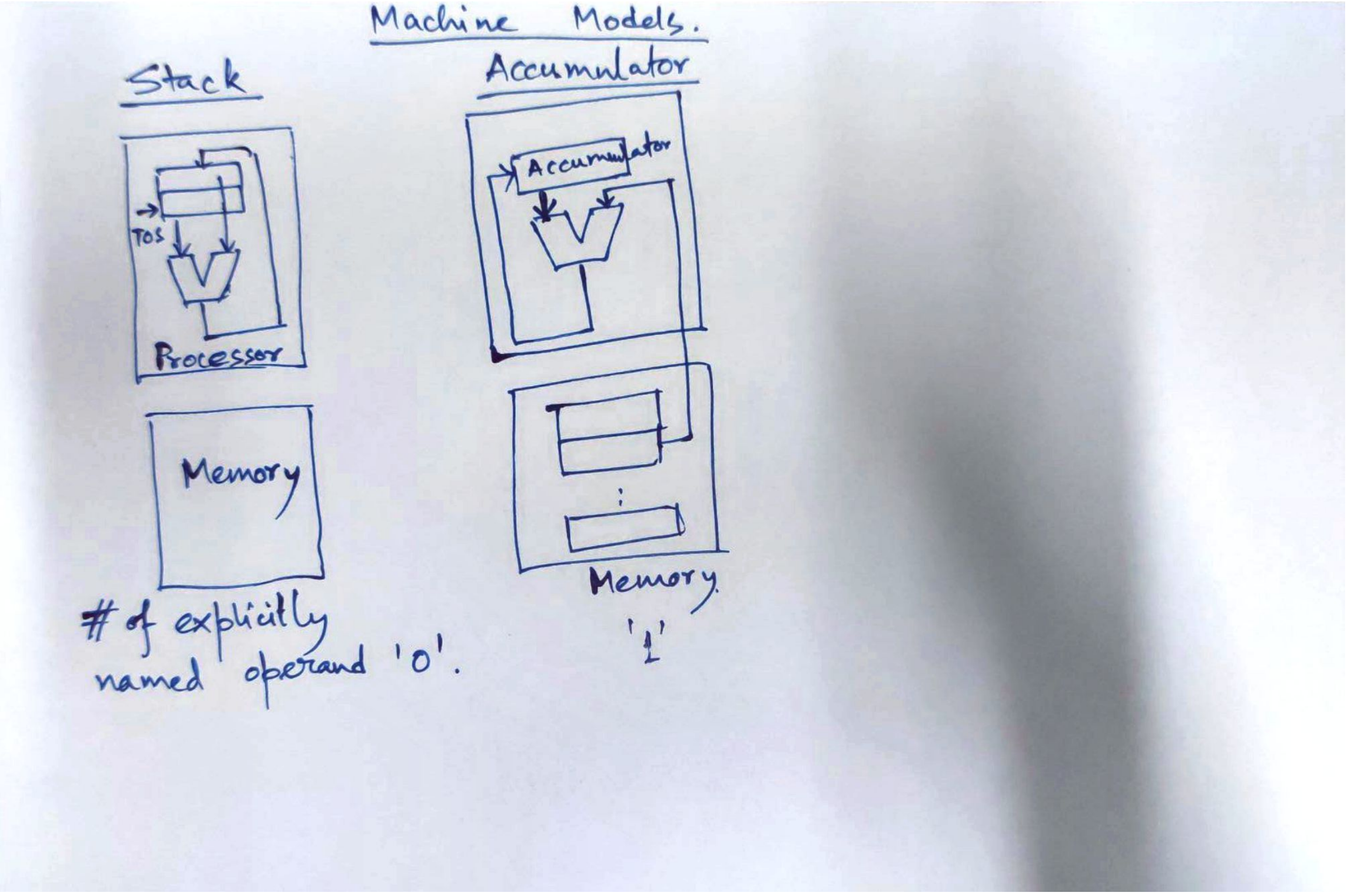
Push A

Push B.

Add.

~~Push C.~~

Pop C.



Machine Models.

Stack

# of explicitly named operand '0'.

Accumulator

'1'