Bit Manipulation - 2

Vinay Neekhra

Senior Instructor & Mentor

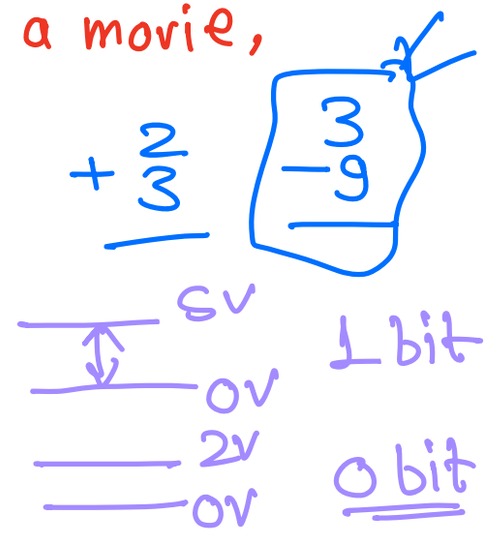
Reachable in Scaler Lounge 

"Brain is the hardware. Mindset is the operating system.
Thoughts are the programming instructions, and
emotions are the output"

* ChatGPT, Maps, video call, watching a movie, creating an animation.

AND, OR, XOR, NOT

Computers = fast + dumb



§ Application of left shift operator:

$(0\,0\,0\,0\,1\,0\,0)_2 \longrightarrow 2^2 = 4$ (1<<2)

$(0\,0\,1\,0\,0\,0\,0)_2 \longrightarrow 2^4 = 16$ (1<<4)

(1<<N) → 2^N

* Set ith bit in a number

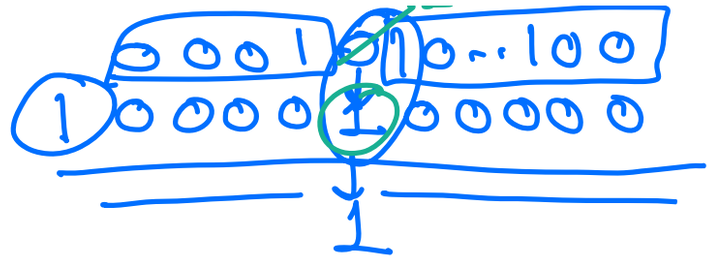
ex. N = 45, i = 4

5 4 3 2 1 0
1 0 1 1 0 1 ⇒ 1 1 1 1 0 1

ex. N = 45, i = 2 ⇒ 1 0 1 1 0 1

jth bit

$$N = N \mid (1 << i)$$



N=26
i = 4

	128	64	32	16	8	4	2	1
	0	0	0	1	1	0	1	0
	7	6	5	4	3	2	1	0
OR	0	0	0	1	0	0	0	0
N	0	0	0	1	1	0	1	0

$N \mid (1 << i)$

$(1 << 4)$

N with 4th bit is set

N=26
i = 2

	128	64	32	16	8	4	2	1
	0	0	0	1	1	0	1	0
	7	6	5	4	3	2	1	0
	0	0	0	0	0	1	0	0
N =	0	0	0	1	1	1	1	0

(*) Application of XOR(^) operator

(*) Flip (toggle) the i^{th} bit

XOR
same bits → 0
different bits → 1

ex. N = 45, i = 2

101101 → 101001

(5 4 3 2 1 0)

ex. N = 45, i = 4

101101 → 111101

(5 4 3 2 1 0)

XOR

1<<i

010000

111101

$N \wedge (1 << i)$

H.W. a → A → 97

b → B

65

32

* & operator.

* Check if i^{th} bit is set or not?

ex. N = 45, i = 2

101101 ⇒ true

(5 4 3 2 1 0)

101101

& 000100 (1<<2)

000100

Obsn: i^{th} bit is not set then $(N \& (1 << i) == 0)$

N = 45
i = 4

```
  101101
& 010000   (1<<4)
--------
  000000
```

```
if (N & (1<<i) == 0) {
    print ("not set")
else
    print ("bit is set)
```

0 → false
!0 → true

`N & (1<<i)` ⇒ if ith bit is set or Not

(II) Soln: using OR operation

N = 45
i = 1

```
  543210
  101101
| 000010
--------
  101111
```

OR

i = 2

```
  101101
| 000100
--------
  101101
```

OR

```
if (N == (N | (1<<i))) {
    print(ith bit is set)
```

Q Unset the i^{th} bit of the Number N if it is set.

ex. N = 6, i = 2 — 3 2 1 0 — 0 1 1 0 — ans = 0 0 1 0

ex. N = 6, i = 3 — 3 2 1 0 — 0 1 1 0 — ans = 0 1 1 0

idea: ① Check if i^{th} bit is set
② If yes, unset i^{th} bit

```
int unsetBit(N, i){
    X = N & (1<<i)
    if (X != 0){
        N = N ^ (1<<i)
```

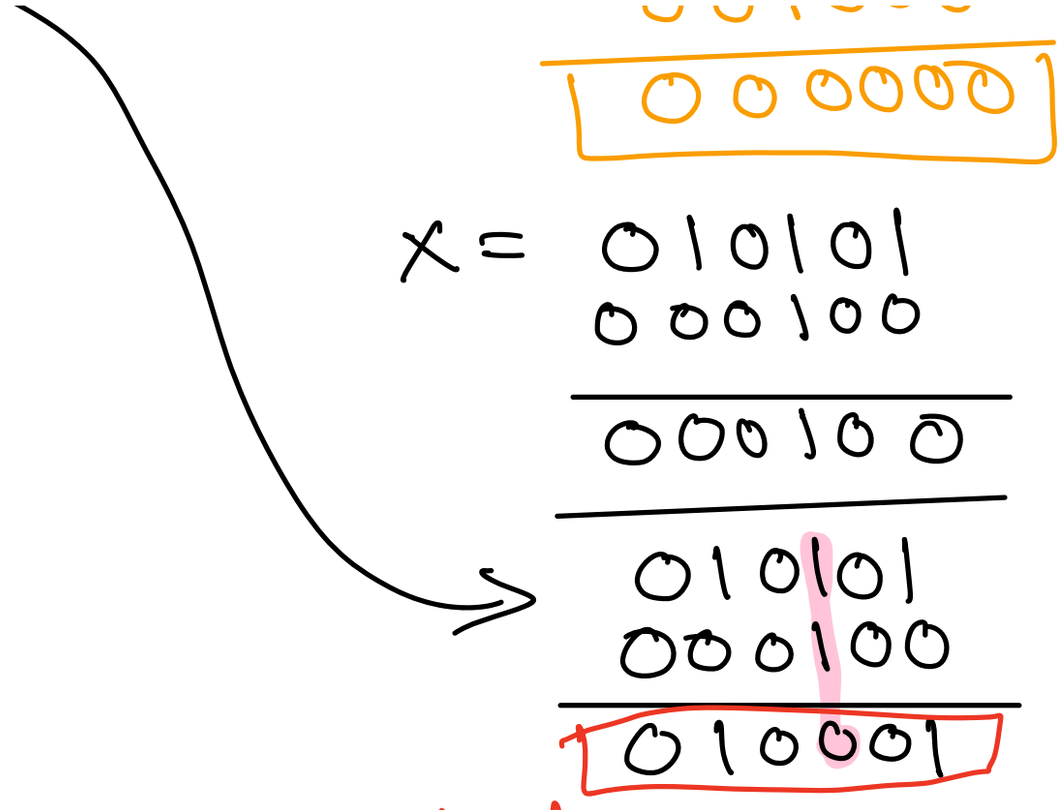
N = 21, i = 3, 2

0 1 0 1 0 1
3 2 1 0

ans = 0 1 0 1 0 1

X = 0 1 0 1 0 1
0 0 1 0 0 0

```
    }
    return N
}
```



Q: Given an integer N, count the total number of set bits.

sj
32
27

ex: N = 12, ...01100 (bits 3 2 1 0) ans = 2

N = 59, 111011 (bits 5 4 3 2 1 0) ans = 5

idea: ① we know how to check if i^{th} bit is set or not.

② If we do the ① for all the bits, we will get our answer.

```
bool checkbit (N, i) {
    if (N & (1<<i) != 0) {
        return true;
    }
    else
        return false;
}
int countBits (int N) {
    int setbits = 0;
    for (i=0; i<32; i++) {
        if (checkbit (N, i) {
            setbits++;
        }
    }
    return setbits;
}
```

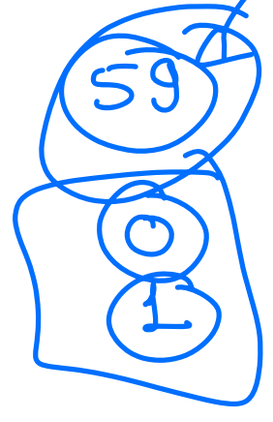
int → long

for (i=0; i<32; i++) → 64

int 4 Bytes
↓
32 bits

32 bits

long 8 bytes
64 bits



II Right shift $10101 \Rightarrow 01010$

$10101 >> 1$

idea: ① check the first bit (LSB)
② right shift the number.

N = 45 $101101 \Rightarrow 4$

Steps:

I 101101
$\& 000001$
$1 \; != 0$ count 1

II 010110 → right shift by 1
$\& 000001$
0 count won't change

III 001011
000001
$1 \; != 0$ count+1

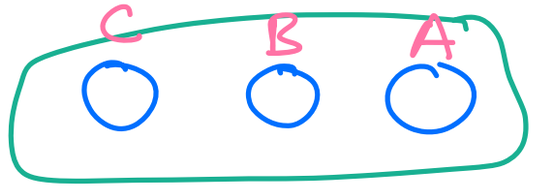
:

? If N == 0, set bits = 0

```
int countOfSetBits(N) {
    int count = 0
    while (N > 0) {
        if (N & 1) {
            count++;
        }
        N = N >> 1
    }
    return count;
}
```

§. Negative numbers:

3 cathode ray tubes



1 → on
0 → off

$2 \times 2 \times 2 = 8$

0	0	0	0
0	0	1	1
0	1	0	2
0	1	1	3
1	0	0	4 — 0
1	0	1	5 — 1
→ 1	1	0	6 — 2
1	1	1	7 — 3

C ✓
B ✗
A ✓
101

⇒ 5

punch cards

Addition

AND

~~XOR~~

How to denote negative numbers.

$-2+3 = 1$

1 1 0
0 1 1
1 0 0 1

$-3+1 = -2$

1 1 1
1 1 0 1
1 0 0 0

$0 \neq 2$

2's complement → to make subtraction easier

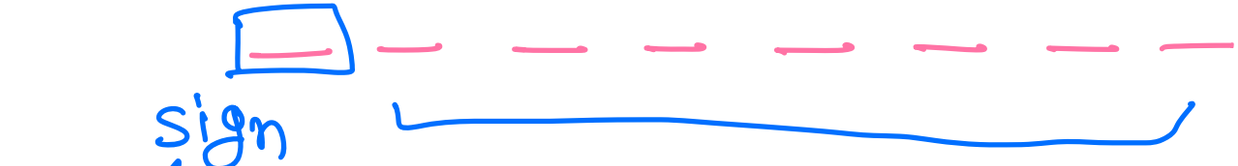
1. 1's complement
2. +1 to the number

```
  1 1 0
+   1
-------
  1 1 1
```

discarded	2's		1's complement			
(1)	0 0 0		1 1 1		0 0 0	→ 0
	1 1 1		1 1 0		0 0 1	→ 1
	1 1 0	←	1 0 1	←	0 1 0	→ 2
	1 0 1	←	1 0 0	←	0 1 1	→ 3
	1 0 0	←	0 1 1	←	1 0 0	
	0 1 1		0 1 0		1 0 1	
	0 1 0		0 0 1	←	1 1 0	
	0 0 1		0 0 0	←	1 1 1	

11:00

(*) 8 bit number



(1) ⇒ negative number.

⇒ take 2's complement to get the value

45 ⇒ 0 0 1 0 1 1 0 1 ⇒ +45

−45 ⇒ 1 1 0 1 0 0 1 0
+ 1

1 1 0 1 0 0 1 1 ⇒ −45

2's complement makes subtraction easier.

(45 − 45) ⇒ 45 + (−45)

1 1 1 1 1 1 1

⇒ 45 0 0 1 0 1 1 0 1
+ −45 1 1 0 1 0 0 1 1

0 (1) 0 0 0 0 0 0 0 0

⇒ 0 0 0 0 1 1 0 0 → +12

-12 → 1 1 1 1 0 1 0 0 → (2's complement)

45 - 12 ⇒ 45 + (-12)

⇒

1 1 1 1 1

0 0 1 0 1 1 0 1

\+ 1 1 1 1 0 1 0 0

1 0 0 1 0 0 0 0 1 → 45-12 = 33

128 64 32 16 8 4 2 1

⇒

12 → 0 0 0 0 1 1 0 0

-45 → 1 1 0 1 0 0 1 1

1 1 0 1 1 1 1 1

⇒ − 0 0 1 0 0 0 0 0
+ 1

⇒ − 0 0 1 0 0 0 0 1 ⇒ −33

can display ⇒ 8 number

Building a 3-bit computer.

			+ve	MSB is −ve	1's comple	2's comple
0	0	0	0	0	0	0
0	0	1	1	1	1	1
0	1	0	2	2	2	2
0	1	1	3	3	3	3
1	0	0	4	−0	−3	−4
1	0	1	5	−1	−2	−3
1	1	0	6	−2	−1	−2
1	1	1	7	−3	−0	−1

CAT

PIE

बिल्ली

1 0 0
− 1 1 0
+ 1

1 0 9

2's complement ⇒ subtraction is trivial

Quiz −3 in 2's complement

+3 ⇒ 0 0 0 0 0 0 1 1

1's ⇒ 1 1 1 1 1 1 0 0

+ 1

⇒ −3, 2's ⇒ 1 1 1 1 1 1 0 1

Quiz −10 in 2's complement?

+10 ⇒ 0 0 0 0 1 0 1 0

⇒ 1's ⇒ 1 1 1 1 0 1 0 1

+ 0 0 0 0 0 0 0 1

1 1 1 1 0 1 1 0

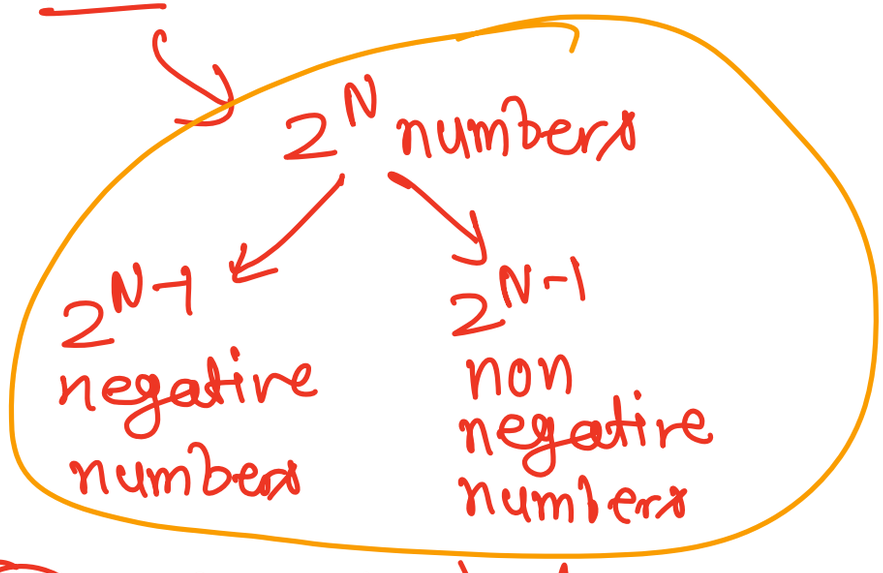
⊛ Range of N-Bit system: (2's complement)

3 bit system $\quad [-4, -3, -2, -1, 0, 1, 2, 3]$

2^3 -/+ $\Rightarrow$ $[\ -2^2 \longleftarrow 0 \longrightarrow 2^2-1$

$\Rightarrow [-2^2 \rightarrow 2^2-1]$

N bit system: $\Rightarrow [-2^{N-1},\ 2^{N-1}-1]$



⊛ Constraints:

int $\Rightarrow$ 32 bits

$[\ -2^{31} \quad 2^{31}\ ]$

$N \in [1, 10^6)$

$A(i) \in (1, 10^5)$

$[-2^{\_}, 2^{\_}-1]$

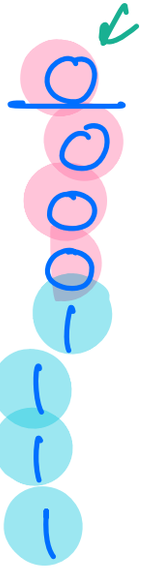
$2^{10} = 1024 \approx 10^3$

$(-(2^{10})^3 * 2, (2^{10})^3 * 2 - 1)$

$\Rightarrow (-(10^3)^3 * 2, (10^3)^3 * 2 - 1)$

$\Rightarrow (-2 \times 10^9, 2 \times 10^9 - 1)$

int range



2's
0
1
2
3
-4
-3
-2
-1

$\Rightarrow [-4, 3]$

$\Rightarrow$ $a = 2$, $b = 3$ $\quad a * b = 6$

a = 010
b = 011

$\Rightarrow$ 2*3 $\Rightarrow$ 2+2+2

$\Rightarrow$

0	1	0
0	1	0
0	1	0

$\Rightarrow$

	1	
0	1	0
0	1	0
1	0	0
0	1	0
1	1	0

$\Rightarrow$ a = 2
b = 3

$\Rightarrow$ a*b = ~~6~~ = -2 overflow

---

int a = 10^5
int b = 10^6

$\Rightarrow$ a*b = 10^{11}

int $[-2\times10^9, 2\times10^9-1]$
4 bytes

```
int c = a*b
      = 10^11
```

```
int a = 10^5
int b = 10^6
```

long c = a*b; ✗

(compiler two int, am int)

= long(a)*b ✓

= long(ab) ✓

= 1L*a*b ✓

long — 8 bytes (64 bits)

$[-2^{63}, 2^{63}-1]$

$\approx [-9\times10^{18}, 9*10^{18}]$

range of long

0 · 0 0 0 0 0 0 0 0

32 bits 32

Q: Given N elements, find the sum!

$1 \le N \le 10^5$

$1 \le A[i] \le 10^6$

long ~~int~~ sum = 0

```
for (int i=0; i<N; i++){
        sum += A[i]
}
return sum
```

$[10^6, 10^6 .. 10^6] \Rightarrow 10^{11}$

(10^5 terms)

out of int range

$(-2 \times 10^9, 2 \times 10^9)$

Doubt

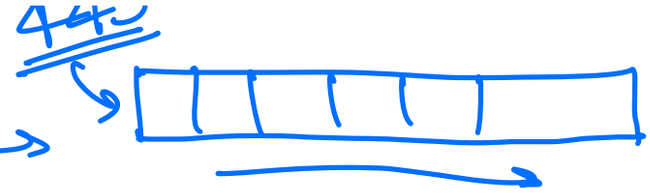
11 ⇒ is it eleven or three?

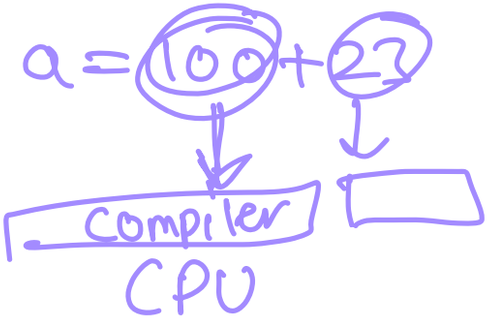
t-Shirt ⇒ there are 10 types of people
one who understand binary
and one who don't

11 ⇒ decimal if you are writing in the code

⇒ internally everything is binary

int a = 100;



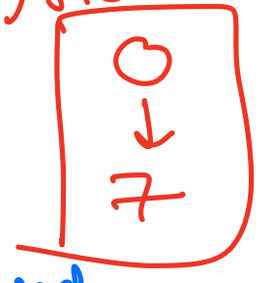


PM Modi → Mindi

Ram Puttin → Ramian

print(a) => 123

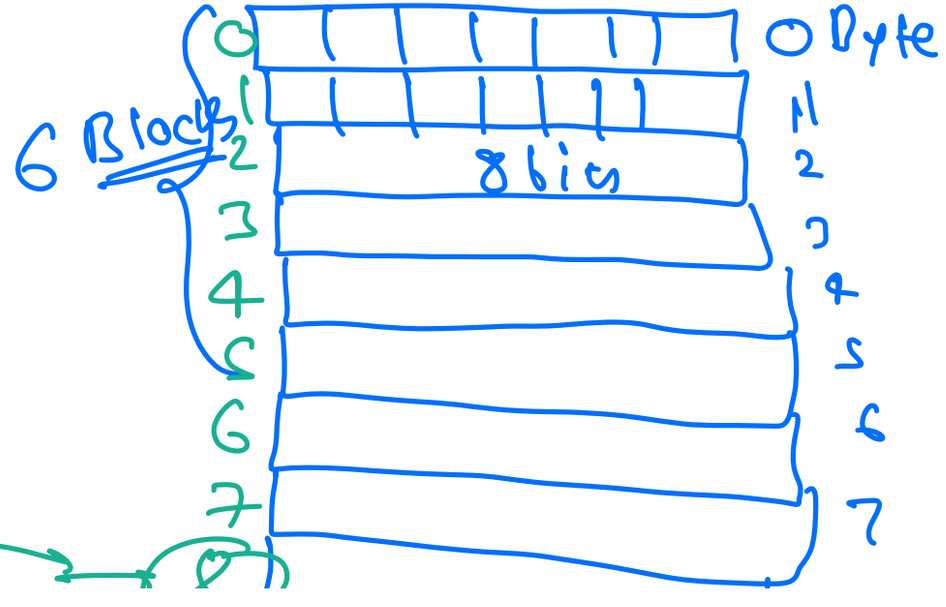
(*) 3 bit system

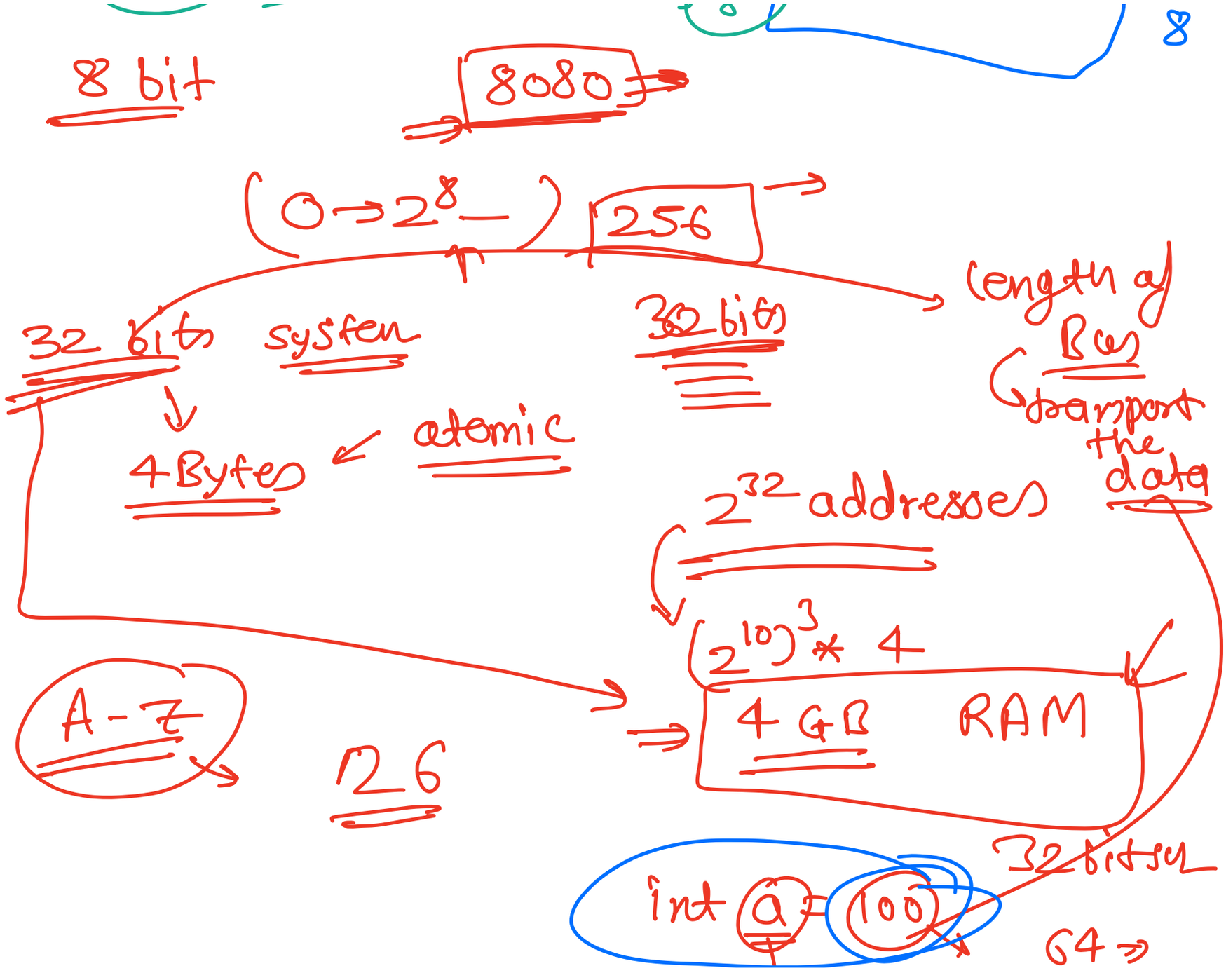


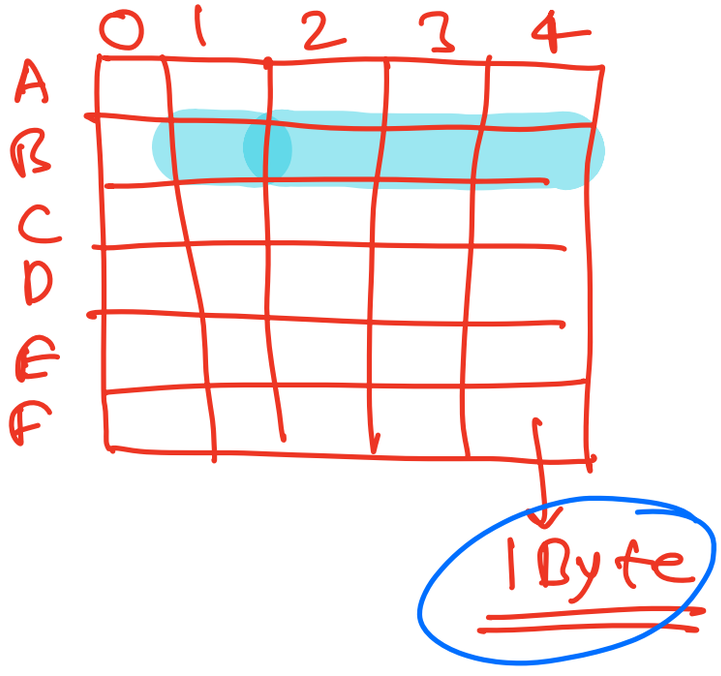
010 ⇒ 2nd memory block

110 → 6th block

Memory




8 bit
8080
8
(0 → 2^8 - 1)
256
32 bits system
32 bits
length of Bus
transport the data
4 Bytes
atomic
2^32 addresses
(2^10)^3 * 4
4 GB RAM
A - Z
26
int a = 100
32 bits
64 →



32 bit

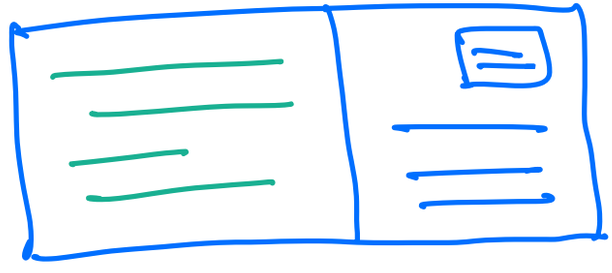
2^{64} addr

OS

give me memory

B1 - B4 to store your int

⇒ Postcards



0

$2^{32}-1$

$2^{15}+1$

Hexadecimal

compact