**Number System**

* Base - unique symbols jinse numbers define honge.

**Decimal Number System**

* Base 10 has 10 possible unique symbols.
* In the decimal number system, a single number is a **digit**.

**Binary Number System**

* Similarly, we have a binary number system - 2 possible values 0 and 1.
* Computers ki bhasha bhi bol skte hai.
* Bit manipulation - **bits** ke sath khelna. (Single possible number ko **bit** bolte hai)
* Seedha aap bit ko manipulate krte ho isiliye faster approach hai yeh.

**Binary to Decimal**

* Right to left Multiply krte jaou power of 2 se power starting from 0.
* 101 - 1\***2^2** + 0\***2^1** + 1\***2^0**

**Decimal to Binary**

* Continuously divide the number by 2 jab tak 0 nhi aata hai and remainders ko niche se upar likho. Example given in [Intermediate\_Bit\_Manipulation\_1.pdf](https://drive.google.com/open?id=1BeBKOFZBBs5VRSxRzl8fZDfc5slDAmdr)

**Addition of any number system**

* Division goes - Division chala jayega carry main - result/base
* Modulo stays - bacha kuch bach hi jayega - result%base
* Eg:- 2 - can’t be stored in binary so, 2%2 = 0 - stays and 2/2 = 1 goes as carry.

**Bitwise Operators**

| AND | BOTH BITS ARE SET = SET |
| --- | --- |
| OR | EITHER OF BIT IS SET = SET |
| XOR | EXACTLY 1 BIT IS SET = SET |

**LSB and MSB**

* Last bit is the least significant bit.
* First set bit is the most significant bit.

**Even and Odd Using Bitwise Operators**

* 2 \* (something) + Last Bit
* Even + 1 - ODD number
* Even + 0 - Even number

[Intermediate\_Bit\_Manipulation\_1.pdf](https://drive.google.com/open?id=1BeBKOFZBBs5VRSxRzl8fZDfc5slDAmdr) refer for examples.

**How to check if the last bit is set or unset?**

* Bit & 1
* 0 & 1 = 0 - unset bit
* 1 & 1 = 1 → se bit. so the same bit is the answer - we will get if it is set or unset.

**How to check if a number is even or odd?**

* And with magical number …….00000000001 (Because I don’t want bits apart from last bit)
* N & 1 = 1 – Odd.
* N & 1 = 0 — Even.

**Properties**

* AND - When you want to nullify the effect do & with 0 to get 0. When you want the same bit as the answer do & with 1 to get the same bit as the answer.
  + A & 0 = 0
  + A & A = A
  + A & 1 = A
* OR -
  + A|0 = A
  + A|A = A
  + A|1 = 1
* XOR
  + A^0 = A
  + A^A = 0

**Follows** - **Commutative**, **Associative** **properties**.

**Left Shift Operator**

* <<
* We keep on discarding left most bit up to n times ( << n)
* A << n = A \* 2^n. (Holds true only if overflow does not happen)

Overflow happens - when the discarded bit is 1.

**Right Shift Operator**

* >>
* We keep on discarding right most bit up to n times ( >> n)
* A >> n = A / 2\*n

**Power of left shift operator**

* To check the last bit we did A & 1 ?
* So, to check ith bit we will do A & 2i

And, we know

A <<n = A \* 2 raised to n.

* so, A << i = A \* 2 raised to i
* If A is 1, 1 << i = 2 raised to i

So, to check ith bit we can simply do

* N & (1<<i) = 0 -> not set else set.

**Important**

1. AND - Check ith bit is set - N = N & (1<<i) = 0 SET else UNSET
2. OR - Set ith bit - N = N | (1<<i) -> ith bit will be set in N.
3. XOR - Toggle - N = N ^ (1<<i) -> ith bit will be toggled in N.

**Binary representation of negative numbers**

1. Take a compliment.
2. Add 1 to it.