Introduction to Number Systems

Overview of Different Number Systems

1. Decimal (Base-10)
    - Digits Used: 0-9
    - Common Use: Everyday counting and arithmetic.
2. Binary (Base-2)
    - Digits Used: 0, 1
    - Common Use: Digital electronics and computing.
3. Octal (Base-8)
    - Digits Used: 0-7
    - Common Use: Shorter representation of binary numbers.
4. Hexadecimal (Base-16)
    - Digits Used: 0-9, A-F
    - Common Use: Memory addresses and color codes in computing.

Importance of Number Systems in Computing

Number systems are fundamental in computing because they provide a way to represent and manipulate data . like:

- Binary System: The foundation of all modern computing systems. Computers use binary (0s and 1s) to process and store data.
- Hexadecimal System: Often used in programming and debugging because it is more compact and easier to read than binary.
- Octal System: Sometimes used in computing as a more human-friendly representation of binary numbers.
- Decimal System: While not directly used in computing, it is essential for human interaction with computers, such as input and output operations.

Binary Number System

Definition and Characteristics of Binary Numbers

The binary number system is a base-2 numeral system that uses only two symbols: 0 and 1. Each digit in a binary number is called a bit. Here are some key characteristics:

- Base: 2
- Digits Used: 0 and 1
- Representation: Each position in a binary number represents a power of 2, starting from the rightmost bit (which represents (2^0)).

Comparison Between Binary and Decimal Systems

- Decimal System (Base-10):
  - Uses ten digits: 0-9.
  - Each position represents a power of 10.
  - Commonly used in everyday arithmetic and counting.
- Binary System (Base-2):
  - Uses two digits: 0 and 1.
  - Each position represents a power of 2.
  - Fundamental to digital electronics and computing.

Binary Digits (Bits)

Definition of a Bit

A bit (short for binary digit) is the smallest unit of data in computing and digital communications. It can have one of two values: 0 or 1.

Understanding Bits and Bytes

- Bit: A single binary digit (0 or 1).
- Byte: A group of 8 bits.

Concept of Binary Place Values

In the binary system, each position represents a power of 2, starting from the rightmost bit (which represents (2^0)):

- 1st position: (2^0 = 1)
- 2nd position: (2^1 = 2)
- 3rd position: (2^2 = 4)
- 4th position: (2^3 = 8)

Binary Arithmetic

Binary arithmetic involves performing mathematical operations on binary numbers, which are numbers expressed in the base-2 numeral system. This system uses only two symbols: 0 and 1. Here are the basic operations:

Addition of Binary Numbers

Binary addition is similar to decimal addition but simpler because it only involves two digits. Here are the basic rules:

- 0 + 0 = 0
- 0 + 1 = 1
- 1 + 0 = 1
- 1 + 1 = 10 (which is 0 with a carry of 1)

Subtraction of Binary Numbers

Binary subtraction uses the concept of borrowing, similar to decimal subtraction. Here are the basic rules:

- 0 - 0 = 0
- 1 - 0 = 1
- 1 - 1 = 0
- 0 - 1 = 1 (with a borrow of 1)

Multiplication of Binary Numbers

Binary multiplication is straightforward, similar to decimal multiplication but simpler. Here are the basic rules:

- 0 * 0 = 0
- 0 * 1 = 0
- 1 * 0 = 0
- 1 * 1 = 1

Division of Binary Numbers

Binary division is similar to long division in the decimal system. It involves dividing the binary number (dividend) by another binary number (divisor) to get a quotient and a remainder.
Example:
1101 ÷ 101 = 1 (quotient), 100 (remainder)

Signed Binary Numbers

Understanding Positive and Negative Binary Numbers

In binary systems, the most significant bit (MSB) is often used as the sign bit. If the MSB is 0, the number is positive; if it is 1, the number is negative. This is known as sign-magnitude representation.

One's' Complement Representation

One's complement is a method of representing negative binary numbers. To find the ones' complement of a binary number, invert all the bits (change 0 to 1 and 1 to 0).

Two's Complement Representation

Two's complement is another method for representing negative binary numbers. It is widely used because it simplifies the design of arithmetic circuits. To find the two's complement:

1. Invert all the bits (find the ones' complement).
2. Add 1 to the least significant bit (LSB).

Arithmetic Operations with Signed Binary Numbers

Addition

When adding two binary numbers in two's complement, you add them as if they were unsigned, and discard any carry beyond the most significant bit.

Subtraction

Subtraction can be performed by adding the two's complement of the number to be subtracted.

Multiplication

Multiplication of signed binary numbers can be done using the same method as unsigned, but taking care of the sign.

Division

Division of signed binary numbers can be performed using long division, similar to unsigned binary division.

Binary Coded Decimal (BCD)

Definition and Uses of BCD

Binary Coded Decimal (BCD) is a method of encoding decimal numbers in which each digit is represented by its own binary sequence. Typically, each decimal digit (0-9) is represented by a 4-bit binary number. This encoding is useful in digital systems where numerical data needs to be displayed, such as in calculators and digital clocks, because it simplifies the conversion between binary and decimal representations.

Conversion Between Binary and BCD

Decimal to BCD Conversion

To convert a decimal number to BCD, replace each decimal digit with its 4-bit binary equivalent.
Example:

- Decimal: 123
- BCD: 0001 0010 0011

BCD to Decimal Conversion

To convert a BCD number to decimal, split the BCD into 4-bit groups and convert each group to its decimal equivalent.
Example:

- BCD: 0001 0010 0011
- Decimal: 123

Applications of Binary System

Binary in Computer Systems

Data Representation: In computer systems, binary is used to represent all types of data. Each bit (binary digit) can be either 0 or 1, and combinations of bits can represent numbers, characters, and other data types. For example, the ASCII code for the letter ‘A’ is 01000001 in binary.
Memory Addressing: Memory addresses in computers are also represented in binary. Each address points to a specific location in memory where data is stored. For example, a 32-bit address can represent (2^{32}) different memory locations.

Binary in Programming (Bitwise Operations)

Bitwise operations are used in programming to manipulate individual bits within a binary number. These operations are fundamental in low-level programming, such as systems programming and hardware interfacing.

Bitwise AND (&)

The bitwise AND operation compares each bit of two binary numbers and returns 1 if both bits are 1, otherwise it returns 0.

Bitwise OR (|)

The bitwise OR operation compares each bit of two binary numbers and returns 1 if at least one of the bits is 1, otherwise it returns 0.

Bitwise XOR (^)

The bitwise XOR operation compares each bit of two binary numbers and returns 1 if the bits are different, otherwise it returns 0.

Bitwise NOT (~)

The bitwise NOT operation inverts all the bits of a binary number, turning 0s into 1s and 1s into 0s.

Bitwise Left Shift (<<)

The bitwise left shift operation shifts all the bits of a binary number to the left by a specified number of positions, filling the rightmost bits with 0s.

Bitwise Right Shift (>>)

The bitwise right shift operation shifts all the bits of a binary number to the right by a specified number of positions, filling the leftmost bits with 0s.

Tutorial for child

Today, we're going to learn about the binary system. It's like a special language that computers use to understand and talk to each other.

What is Binary? Binary is a way of counting using only two numbers: 0 and 1. Imagine you have a light switch that can be either off (0) or on (1). That's how binary works!

Counting in Binary Just like we count in decimal (1, 2, 3, …), we can count in binary too:

0 – 1 – 10 (2 in decimal) – 11 (3 in decimal) – 100 (4 in decimal)

Converting Decimal to Binary To change a decimal number to binary, we keep dividing the number by 2 and writing down the remainders. Let's try converting 5 to binary:

5 ÷ 2 = 2 remainder 1

2 ÷ 2 = 1 remainder 0

1 ÷ 2 = 0 remainder 1 So, **5 in binary is 101.**

Binary Arithmetic Computers use binary to do math too! Here are some simple operations:

Addition: 101 + 110 = 1011

Subtraction: 1101 - 1010 = 11

Multiplication: 1010 * 1101 = 111110

Division: 1101 ÷ 101 = 10 remainder 11

Signed Binary Numbers Sometimes, we need to show negative numbers. We use something called two's complement. To find the two's complement, we flip all the bits and add 1. For example, the two's complement of 5 (0101) is 1011.

Binary Coded Decimal (BCD) BCD is a way to show decimal numbers in binary. Each decimal digit gets its own 4-bit binary code. For example, 123 in BCD is 0001 0010 0011.

Bitwise Operations Bitwise operations are like magic tricks with binary numbers:

AND (&): 1100 & 1010 = 1000

OR (|): 1100 | 1010 = 1110

XOR (^): 1100 ^ 1010 = 0110

NOT (~): ~1100 = 0011

Left Shift (<<): 1100 << 2 = 110000

Right Shift (>>): 1100 >> 2 = 11

Binary is important for computers. It helps them understand numbers.

Resources

- BCD or Binary Coded Decimal
- Binary-coded decimal
- Binary-Coded Decimal (BCD)
- Introduction to Number Systems and Binary
- Number System (Definition, Types, Conversion & Examples)
- Number System in Maths
- Computer Number Systems 101: Binary & Hexadecimal Conversions
- Binary Number System - Definition, Conversion, Examples
- Binary Number System – Definition, Chart, Table, Examples, and Diagram
- Difference Between Decimal and Binary Number System
- What Is Binary? (Definition, vs. Decimal, Importance)
- Electronics Tutorials: Signed Binary Numbers
- Two’s Complement vs Sign-Magnitude
- Rochester Institute of Technology: Two’s Complement
- Bits (binary digits)
- Bits, Bytes, and Binary
- Bit
- Bit and Byte Explained in 6 Minutes
- Binary & Data
- Binary Number System
- Number System in Computer
- Binary arithmetic

* Convert The following binary numbers to decimal :- 1011 , 11001

$1011_2 \Rightarrow 1*2^0 + 1*2^1 + 0*2^2 + 1*2^3$

$1 + 2 + 8 = 11$

$\boxed{11 \text{ in decimal}}$

$11001 \Rightarrow 1*2^0 + 0*2^1 + 0*2^2 + 1*2^3 + 1*2^4$

$1 + 0 + 0 + 8 + 16 = 25$

$\boxed{25 \text{ in decimal}}$

* Convert The following decimal number to binary, 25 , 47 , 123

2	25	
2	12	1
2	6	0
2	3	0
2	1	1
	0	1

$\boxed{11001}$

2	47	
2	23	1
2	11	1
2	5	1
2	2	1
2	1	0
	0	1

$\boxed{101111}$

2	123	
2	61	1
2	30	1
2	15	0
2	7	1
2	3	1
2	1	1
	0	1

$\boxed{1111011}$

* Convert The following Hexadecimal to decimal. 1A3, 4F2, B72

$1A3 \Rightarrow 3*16^0 + 10*16^1 + 1*16^2$

$3 + 160 + 256 \boxed{= 419}$

$4F2 \Rightarrow 2*16^0 + 15*16^1 + 4*16^2$

$2 + 240 + 1024 \boxed{= 1266}$

$B72 \Rightarrow 2*16^0 + 7*16^1 + 11*16^2$

$2 + 112 + 2816 \boxed{= 2930}$

Convert The following decimal numbers to hexadecimal (255, 1024, 4096

16	255	15 F
16	15	15 F
	0	

$\boxed{FF}$

16	1024	
16	64	0
16	4	0
	0	4

$\boxed{400}$

16	4096	
16	256	0
16	16	0
16	1	0
	0	1

1000

Binary arithmetic problem:-

```
  1011        1001        1110
  1101        1010        0111
 -----       -----       -----
 11000       10011       10101
```

binary addition

```
  1101        1001
  1010        0110
 -----       -----
  0011        0011
```

subtractions

```
   101         110         111
    11         101         100
 -----       -----       -----
   101         110       11100
  1010        0000
 -----       11000
  1111       -----
             11110
```

* Convert The following Signed binary numbers to decimal (Using Two's)

$$\begin{array}{r} 1101 \rightarrow 0010 \\ +\ \ \ 1 \\ \hline 0011 \end{array} \rightarrow 1*2^0 + 1*2^1 \rightarrow 1 + 2 = 3$$

$$\begin{array}{r} 1010 \rightarrow 0101 \\ +\ \ \ 1 \\ \hline 0110 \end{array} \rightarrow 0*2^0 + 1*2^1 + 1*2^2 + \cancel{0*2^3}$$

$$\cancel{0} + 2 + 4 = 6$$

$$\begin{array}{r} 1111 \rightarrow 0000 \\ +\ \ \ 1 \\ \hline 0001 \end{array} \quad 1*2^0 \rightarrow 1$$

---

* perform The following operations with signed binary numbers.

~~1011~~
~~1101~~

1011 + 1101 → 1011 → $\begin{array}{r} 0100 \\ 1 \\ \hline 0101 \end{array}$ | 1101 → $\begin{array}{r} 0010 \\ 1 \\ \hline 0011 \end{array}$

$$\begin{array}{r} 0101 \\ 0011 \\ \hline 0111 \end{array}$$

1001 + 1010 → 1001 → $\begin{array}{r} 0110 \\ 1 \\ \hline 0111 \end{array}$ | 1010 → $\begin{array}{r} 0101 \\ 1 \\ \hline 0110 \end{array}$

$$\begin{array}{r} 1\ \ 1\ \ \ \ \ \ \\ 0111 \\ 0110 \\ \hline 1100 \end{array}$$

1110 + 0111 → 1110 → $\begin{array}{r} 0001 \\ 1 \\ \hline 0010 \end{array}$ | 0111 → $\begin{array}{r} 1000 \\ 1 \\ \hline 1001 \end{array}$

$$\begin{array}{r} 1\ 1\ \ \ \ \ \ \\ 0010 \\ 1001 \\ \hline 1011 \end{array}$$

* Convert a set of decimal numbers to binary and vice versa.

* Decimal to Binary :

→ Convert 13 to binary

2	13	
2	6	1
2	3	0
2	1	1
	0	1

1101

→ Convert ~~12~~ 25 to binary

2	25	
2	12	1
2	6	0
2	3	0
2	1	1
	0	1

11001

* Binary to decimal

→ Convert 1010 to decimal

$\cancel{0 * 2^0} + 1 * 2^1 + \cancel{0 * 2^2} + 1 * 2^3$

$0 + 2 + 8 = 10$

* Convert 1111 to decimal

$1 * 2^0 + 1 * 2^1 + 1 * 2^2 + 1 * 2^3 \Rightarrow 8 + 4 + 2 + 1 \Rightarrow 15$

Convert Binary numbers to Octal and hexadecimal

→ Binary to Octal → Bin → dec → oct

$110101 \Rightarrow 1*2^0 + \cancel{0*2^1} + 1*2^2 \cancel{+0*2^3} + 1*2^4 + 1*2^5 \Rightarrow 32+16+4+1 \Rightarrow 53$

8	53	
8	48	5
8	6	0
	0	6

603

---

5 4 3 2 1 0

$101010 \Rightarrow \cancel{0*2^0} + 1*2^1 + \cancel{0*2^2} + 1*2^3 + \cancel{0*2^4} + 1*2^5 = 32+8+2 \Rightarrow 42$

8	42	
8	5	2
	0	5

52

$110110 \Rightarrow \cancel{0*2^0} + 1*2^1 + 1*2^2 + \cancel{0*2^3} + 1*2^4 + 1*2^5 = 32+16+4+2=54$

5 4 3 2 1 0

8	54	
8	6	6
	0	6

66

---

5 4 3 2 1 0

$111000 \Rightarrow 1*2^3 + 1*2^4 + 1*2^5 \Rightarrow 32+16+8 = 56$

8	56	
8	7	0
	0	7

70

* Convert the following binary numbers to hexadecimal

bin → dec → hex

10 1010 → 42

16	42	
16	2	10
	0	2

2A

110110 → 54

16	54	
16	3	6
	0	3

36

111000 → 56

16	56	
16	3	8
	0	3

38

Binary arithmetic problems:-

Addition → 1010 + 1101 ; 111 + 1010

```
  1010
  1101
 -----
 10111
```

```
   111
  1010
 -----
 10001
```

Subtraction :- 1101 - 1010 | 1010 - 111

```
 1101
 1010
-----
 0011
```

```
 1010
  111
-----
  011
```

multiplication :- 101 * 11 | 110 * 101

```
  101
   11
-----
  101
 1010
-----
 1111
```

```
   110
   101
------
   110
  0000
 11000
------
 11110
```

Division :- 1100 / 10 | 1010 \ 10

```
      110
10 | 1100
     100
    -----
     0100
```

1100
84 = 12
2

$\frac{12}{2} = 6 \Rightarrow 110$

```
 10
 10
----
 00
 00
----
100
```