Table Of Contents

Positional Notation: Base 10 (Decimal System)

Positional notation refers to how the position of a digit in a number determines its value based on the base (or radix) of the numeral system. In the case of **base 10**, also called the **decimal system**, each digit in a number is multiplied by a power of 10, depending on its position.

Let's break down the example:

1. Example: 527

In base 10, this means each digit of the number represents a power of 10 based on its position:

- **527** can be broken down as: 527 = 5 x 10^2 + 2 x 10^1 + 7 x 10^0

Place	10^2 100's Place	10^1 10's Place	10^0 1's Place
Value	100	10	1
Evaluate	5 x 100	2 x 10	7 x 1
Sum	500	20	7

2. Step-By-Step Breakdown

- **Digit 5** is in the hundreds place, so:

  5 x 10^2 = 5 x 100 = 500

- **Digit 2** is in the tens place, so:

  2 x 10^1 = 2 x 10 = 20

- **Digit 7** is in the ones place, so:

  7 x 10^0 = 7 x 1 = 7

- **Final result:** 527 = 500 + 20 + 7 = 527
  - This shows how the value of the number 527 is determined based on the position of each digit in relation to powers of 10.

3. Further Example

Let's take a larger number for clarity: **1,234**

- $1234 = 1 \times 10^3 + 2 \times 10^2 + 3 \times 10^1 + 4 \times 10^0$

Breaking it down:

- $1 \times 10^3 = 1 \times 1000 = 1000$
- $2 \times 10^2 = 2 \times 100 = 200$
- $3 \times 10^1 = 3 \times 10 = 30$
- $4 \times 10^0 = 4 \times 1 = 4$

Adding them up:

1,234 = 1000 + 200 + 30 + 4 = 1,234

4. Key Takeaways

- **Base 10 (Decimal)**: Uses digits 0 through 9.
- **Positional Notation**: The value of a digit is determined by its position relative to the base (in this case, powers of 10).
- The leftmost digits represent higher powers of 10, and as you move right, the power decreases.

Positional Notation: Base 8 (Octal System)

Positional notation in the **octal system** is similar to the decimal system but with **base 8**. In base 8, the digits range from **0 to 7**, and each digit is multiplied by a power of 8 depending on its position.

Let's break down the example:

1. Example: 624_8 (Octal) Converted To Decimal

The octal number **624_8** can be expressed in decimal (base 10) by expanding it in terms of powers of 8. Each digit is multiplied by a power of 8 according to its position.

- $624_8 = 6 \times 8^2 + 2 \times 8^1 + 4 \times 8^0$

Place	8^2 64's Place	8^1 8's Place	8^0 1's Place
Value	64	8	1
Evaluate	6 x 64	2 x 8	4 x 1
Sum for Base 10	384	16	4

2. Step-By-Step Breakdown

- **Digit 6** is in the "hundreds" place in octal, which corresponds to **8^2** (64 in decimal):

  6 x 8^2 = 6 x 64 = 384

- **Digit 2** is in the "tens" place in octal, which corresponds to **8^1** (8 in decimal):

  2 x 8^1 = 2 x 8 = 16

- **Digit 4** is in the "ones" place in octal, which corresponds to **8^0** (1 in decimal):

  4 x 8^0 = 4 x 1 = 4

- **Final result: 624_8** = 384 + 16 + 4 = 404_{10}
    - Thus, **624_8** in octal is equal to **404_{10}** in decimal.

3. Further Example

Let's convert another octal number $\mathbf{135_8}$ into decimal:

- $\mathbf{135_8} = 1 \times 8^2 + 3 \times 8^1 + 5 \times 8^0$

Breaking it down:

- $1 \times 8^2 = 1 \times 64 = 64$
- $3 \times 8^1 = 3 \times 8 = 24$
- $5 \times 8^0 = 5 \times 1 = 5$

Adding them up:

- $\mathbf{135_8} = 64 + 24 + 5 = 93_{10}$
- **So, 135_8 in octal is equal to 93_{10} in decimal.**

4. Key Takeaways

- **Base 8 (Octal)**: Uses digits 0 through 7.
- **Positional Notation**: The value of each digit is determined by its position and the corresponding power of 8.
- **Conversion To Decimal**: Multiply each digit by its respective power of 8 and sum the results to get the decimal equivalent.

Positional Notation: Base 16 (Hexadecimal System)

The **hexadecimal system** is a base 16 numeral system. It uses 16 symbols: the digits **0-9** and the letters **A-F**, where:

- **0-9** represent values **0-9**, and
- **A** represents **10**, **B** represents **11**, **C** represents **12**, **D** represents **13**, **E** represents **14**, and **F** represents **15**.

1. Example: $6{,}704_{16}$ (Hexadecimal) Converted To Decimal

To convert the hexadecimal number $\mathbf{6{,}704_{16}}$ to its decimal (base 10) equivalent, we expand it by multiplying each digit by a power of 16, according to its position.

- $6704_{16} = 6 \times 16^3 + 7 \times 16^2 + 0 \times 16^1 + 4 \times 16^0$

Place	16^3 **4, 096's Place**	16^2 **256's Place**	16^1 **16's Place**	16^0 **1's Place**
Value	**4, 096**	**256**	**16**	**1**
Evaluate	**6 x 4,096**	**7 x 256**	**0 x 16**	**4 x 1**
Sum for Base 10	24,576	1,792	0	4

2. Step-By-Step Breakdown

- **Digit 6** is in the "thousands" place in hexadecimal, which corresponds to $\mathbf{16^3}$ (4096 in decimal):
  $\mathbf{6 \times 16^3} = 6 \times 4096 = 24{,}576$

- **Digit 7** is in the "hundreds" place in hexadecimal, which corresponds to $\mathbf{16^2}$ (256 in decimal):
  $\mathbf{7 \times 16^2} = 7 \times 256 = 1{,}792$

- **Digit 0** is in the "tens" place in hexadecimal, which corresponds to $\mathbf{16^1}$ (16 in decimal):
  $\mathbf{0 \times 16^1} = 0 \times 16 = 0$

- **Digit 4** is in the "ones" place in hexadecimal, which corresponds to $\mathbf{16^0}$ (1 in decimal):
  $\mathbf{4 \times 16^0} = 4 \times 1 = 4$

- **Final result:** $6704_{16} = 24{,}576 + 1{,}792 + 0 + 4 = 26{,}372_{10}$
  - Thus, $\mathbf{6{,}704_{16}}$ in hexadecimal is equal to $\mathbf{26{,}372_{10}}$ in decimal.

3. Further Example

Let's convert another hexadecimal number **$1A3F_{16}$** into decimal:

- **$1A3F_{16}$** = $1 \times 16^3 + A \times 16^2 + 3 \times 16^1 + F \times 16^0$

Breaking it down:

- $1 \times 16^3 = 1 \times 4{,}096 = 4{,}096$
- A(which is 10 in decimal) $\times 16^2 = 10 \times 256 = 2{,}560$
- $3 \times 16^1 = 3 \times 16 = 48$
- F (which is 15 in decimal) $\times 16^0 = 15 \times 1 = 15$

Adding them up:

- **$1A3F_{16}$** = $4096 + 2560 + 48 + 15 = 6719_{10}$
- **So, $1A3F_{16}$ in hexadecimal is equal to 6719_{10} in decimal.**

4. Key Takeaways

- **Base 16 (Hexadecimal)**: Uses digits 0-9 and letters A-F, where A = 10, B = 11, ..., F = 15.
- **Positional Notation**: The value of each digit is determined by its position and the corresponding power of 16.
- **Conversion To Decimal**: Multiply each digit by its respective power of 16 and sum the results to get the decimal equivalent.

Positional Notation: Base 2 (Binary System)

The **binary system** is a base 2 numeral system, using only two symbols: **0** and **1**. Each digit (also called a **bit**) in a binary number is multiplied by a power of 2 based on its position. The rightmost digit has the lowest value (2^0), and the value increases as you move left (2^1, 2^2, and so on).

1. Example: $1101\ 0110_2$ (Binary) Converted To Decimal (Base 10)

Let's break down the binary number **$1101\ 0110_2$**. To convert it to decimal (base 10), we expand it by multiplying each digit by a power of 2 according to its position from right to left.

- $1101\ 0110_2 = 1 \times 2^7 + 1 \times 2^6 + 0 \times 2^5 + 1 \times 2^4 + 0 \times 2^3 + 1 \times 2^2 + 1 \times 2^1 + 0 \times 2^0$

Place	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
Value	128	64	32	16	8	4	2	1
Evaluate	1 x 128	1 x 64	0 x 32	1 x16	0 x 8	1 x 4	1 x 2	0 x 1
Sum for Base 10	128	64	0	16	0	4	2	0

2. Step-By-Step Breakdown

- **$1 \times 2^7 = 1 \times 128 = 128$**
- **$1 \times 2^6 = 1 \times 64 = 64$**
- **$0 \times 2^5 = 0 \times 32 = 0$**
- **$1 \times 2^4 = 1 \times 16 = 16$**
- **$0 \times 2^3 = 0 \times 8 = 0$**
- **$1 \times 2^2 = 1 \times 4 = 4$**
- **$1 \times 2^1 = 1 \times 2 = 2$**
- **$0 \times 2^0 = 0 \times 1 = 0$**

- **Final result:** $1101\ 0110_2 = 128 + 64 + 0 + 16 + 0 + 4 + 2 + 0 = 214_{10}$
  - Thus, **$1101\ 0110_2$** in binary is equal to **214_{10}** in decimal.

3. Further Example

Let's convert another binary number **1011 1101_2** to decimal:

- $1011\ 1101_2 = 1 \times 2^7 + 0 \times 2^6 + 1 \times 2^5 + 1 \times 2^4 + 1 \times 2^3 + 1 \times 2^2 + 0 \times 2^1 + 1 \times 2^0$

Breaking it down:

- **$1 \times 2^7 = 1 \times 128 = 128$**
- **$0 \times 2^6 = 0 \times 64 = 0$**
- **$1 \times 2^5 = 1 \times 32 = 32$**
- **$1 \times 2^4 = 1 \times 16 = 16$**
- **$1 \times 2^3 = 1 \times 8 = 8$**
- **$1 \times 2^2 = 1 \times 4 = 4$**
- **$0 \times 2^1 = 0 \times 2 = 0$**
- **$1 \times 2^0 = 1 \times 1 = 1$**

Adding them up:

- $1011\ 1101_2 = 128 + 0 + 32 + 16 + 8 + 4 + 0 + 1 = 189_{10}$
- **So, $1011\ 1101_2$ in binary is equal to 189_{10} in decimal.**

4. Key Takeaways

- **Base 2 (Binary)**: Uses only the digits 0 and 1.
- **Positional Notation**: The value of each digit (bit) is determined by its position and the corresponding power of 2.
- **Conversion To Decimal**: Multiply each digit by its respective power of 2 and sum the results to get the decimal equivalent.
- Binary numbers are especially important in computing, as computers operate using binary logic (1s and 0s).

Range of Possible Numbers: Formula $R = B^k$

The **range of possible numbers** in any numeral system depends on three factors:

- **R**: Range (total number of unique values or numbers possible),
- **B**: Base (the radix of the numeral system),
- **K**: Number of digits used to represent the number.

1. The Formula

$$R = B^k$$

- **B** represents the base or radix of the numeral system (e.g., base 10 for decimal, base 2 for binary).
- **K** is the number of digits used to represent the numbers in the system.
- **R** gives the total number of different numbers that can be represented with those K digits in base B.

2. Example #1: Base 10 (Decimal System), 2 Digits

In this example, we are working with the **decimal system** (base 10) and using **2 digits** to represent numbers.

$$R = 10^2 = 100$$

This means with 2 digits in base 10, we can represent **100 different numbers**.

Explanation:

The numbers range from 00 to 99, which is a total of 100 possible values (including 00). This range is:

- **Smallest number**: 00
- **Largest number**: 99
- **Range of possible numbers**: 100 values (from 0 to 99)

3. Example #2: Base 2 (Binary System), 16 Digits

In this example, we are working with the **binary system** (base 2) and using **16 digits** (bits) to represent numbers. This is typical for a **16-bit computer**.

$$R = 2^{16} = 65,536$$

This means with 16 bits in base 2, we can represent **65,536 different numbers**.

Explanation:

The binary number can store values ranging from **$0000\ 0000\ 0000\ 0000_2$** to **$1111\ 1111\ 1111\ 1111_2$**. In decimal, these correspond to:

- **Smallest number**: 0 (binary $0000\ 0000\ 0000\ 0000_2$)
- **Largest number**: 65,53**5** (binary $1111\ 1111\ 1111\ 1111_2$)
- **Range of possible numbers**: 65,53665,53665,536 values, typically written as **64K**.

4. Generalization for Any Base and Number of Digits

For any base B and number of digits K, the formula **R = B^k** will tell you how many different numbers can be represented. The range always starts from **0** (the smallest possible number) and goes up to **B^k**−1 (the largest possible number).

5. Further Examples

Example #3: Base 16 (Hexadecimal System), 4 Digits

$$R = 16^4 = 65,536$$

This is the same number of values as in a 16-bit binary system because both systems are dealing with 4 hexadecimal digits, which can also be represented as 16 bits in binary.

- **Smallest number**: 0000_{16} (which is 0 in decimal)
- **Largest number**: $FFFF_{16}$ (which is 65,535 in decimal)

Example #4: Base 10 (Decimal System), 3 Digits

$$R = 10^3 = 1,000$$

With 3 digits in base 10, we can represent 1,000 different numbers ranging from:

- **Smallest number**: 000 (or just 0)
- **Largest number**: 999

Example #5: Base 2 (Binary System), 8 Digits (8-bit system)

$$R = 2^8 = 256$$

In an **8-bit** system, 256 different numbers can be represented. This is common in early computer systems and microcontrollers.

- **Smallest number**: 000 (binary 0000000020000 $0000_2$000000002)
- **Largest number**: 255255255 (binary 1111111121111 $1111_2$111111112)
- **Range of possible numbers**: 256 values (from 0 to 255)

6. Key Takeaways

- **Base (B)** determines how many different symbols can be used for each digit.
- **Number of digits (K)** determines how many places there are for those symbols.
- The total number of different numbers that can be represented is given by **R = B^k**.

This formula is crucial for understanding the storage capacity of systems, ranging from simple numeric representations to the limits of computer memory, such as **8-bit, 16-bit, and 32-bit** systems.

Decimal Range for Bit Widths

In computing, the **bit width** of a system refers to the number of bits (binary digits) used to represent data, usually in memory or processing. The **range of numbers** that can be represented depends on how many bits are available. The more bits, the larger the range of possible values.

For **unsigned integers** (where all bits represent positive numbers), the **range** of numbers that can be represented by a certain number of bits is calculated by:

$$R = 2^n$$

Where:

- **n** is the number of bits.
- The range is from **0** to $2^n - 1$.

Below is a breakdown of the number of bits, the number of digits in decimal, and the range of numbers that can be represented for different bit widths:

1 Bit

- **Number of Decimal Digits**: 0+
- **Range**: $2^1 = 2$
- **Values**: 0, 1

Explanation:

With 1 bit, you can represent only 2 values: **0** or **1**. This is the simplest binary representation.

4 Bits

- **Number of Decimal Digits**: 1+
- **Range**: $2^4 = 16$
- **Values**: 0 to 15

Explanation:

With 4 bits, you can represent 16 values (from 0 to 15 in decimal). These are commonly used in **nibbles** (half a byte).

8 Bits (1 Byte)

- **Number of Decimal Digits**: 2+
- **Range**: 2^8 = 256
- **Values**: 0 to 255

Explanation:

With 8 bits (one **byte**), you can represent 256 different numbers, from 0 to 255. This is standard for early microprocessors or small data types.

10 Bits

- **Number of Decimal Digits**: 3
- **Range**: 2^{10} = 1,024 (1K)
- **Values**: 0 to 1,023

Explanation:

With 10 bits, you can represent 1,024 values, typically referred to as "1K." This is common in memory addressing.

16 Bits (2 Bytes)

- **Number of Decimal Digits**: 4+
- **Range**: 2^{16} = 65,536 (64K)
- **Values**: 0 to 65,535

Explanation:

With 16 bits, you can represent **65,536** values, often referred to as **64K**. This was typical for early computers, like **16-bit systems**.

20 Bits

- **Number of Decimal Digits**: 6
- **Range**: 2^{20} = 1,048,576 (1M)
- **Values**: 0 to 1,048,575

Explanation:

With 20 bits, you can represent **1,048,576** values, typically called **1 megabit (1M)**.

32 Bits (4 Bytes)

- **Number of Decimal Digits**: 9+
- **Range**: 2^{32} = 4,294,967,296 (4G)
- **Values**: 0 to 4,294,967,295

Explanation:

With 32 bits, you can represent **4,294,967,296** values, often referred to as **4 gigabits (4G)**. This is typical for **32-bit systems**, widely used in modern computing.

64 Bits (8 Bytes)

- **Number of Decimal Digits**: 19+
- **Range**: $2^{64} \approx 1.8 \times 10^{19}$
- **Values**: From 0 to approximately **18.4 quintillion** (exact value: 18,446,744,073,709,551,615)

Explanation:

A **64-bit** system can represent a vast number of values, roughly **1.8×10^{19}** (18 quintillion). These are typical in modern 64-bit architectures, allowing for huge amounts of data storage and processing.

128 Bits

- **Number of Decimal Digits**: 38+
- **Range**: $2^{128} \approx 3.4 \times 10^{38}$
- **Values**: From 0 to approximately **340 undecillion** (exact value: 340,282,366,920,938,463,463,374,607,431,768,211,455)

Explanation:

With 128 bits, the range is mind-bogglingly large, around **3.4×10^{38}**. This is far beyond what most applications need, and is used in advanced cryptography and very large number computations.

Key Takeaways

- The number of different values that can be represented with **n bits** is 2^{n}.
- **1 bit** can represent only 2 values (0 or 1), while **32 bits** can represent over 4 billion values.
- The more bits available, the more digits you can represent and the larger the possible range of numbers.

Example of Range by Bit Width

Example 1: 8 bits	Example 3: 32 bits
Example 1: 8 bits ● With 8 bits, you can represent 256 values. ● Decimal range: 0 to 255. **Example 2: 16 bits** ● With 16 bits, you can represent **65,536** values. ● Decimal range: 0 to 65,535 (commonly known as **64K**).	**Example 3: 32 bits** ● With 32 bits, you can represent over 4 billion values. ● Decimal range: 0 to **4,294,967,295**. **Example 4: 64 bits** ● With 64 bits, the range is over **18 quintillion** values. ● Decimal range: 0 to **18,446,744,073,709,551,615**.

Base or Radix

The **base** (also called the **radix**) of a number system is the number of unique symbols (or digits) used to represent numbers in that system. The value of each digit is determined by its position and the base of the system. The larger the base, the more symbols are required to represent any given number.

1. Key Points

- **Base** is the number of different symbols or digits used in a numeral system.
- The **place value** of each digit depends on its position and the base.
- For larger bases, more symbols are needed to represent numbers.

2. Examples of Different Bases

1. Base 10 (Decimal System)

- **Symbols**: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9
- This system uses 10 symbols (0 to 9) and is the most commonly used numbering system in everyday life.

Example:

The number **527_{10}** in base 10 can be expanded as:

- $527_{10} = 5 \times 10^2 + 2 \times 10^1 + 7 \times 10^0 = 500 + 20 + 7 = 527$

Here, each digit is multiplied by powers of **10** (the base), depending on its position.

2. Base 2 (Binary System)

- **Symbols**: 0, 1
- This system uses only 2 symbols, 0 and 1, and is fundamental in computing because digital electronics operate in binary (on/off, high/low).

Example:

The number **1101_2** in base 2 can be expanded as:

- $1101_2 = 1 \times 2^3 + 1 \times 2^2 + 0 \times 2^1 + 1 \times 2^0 = 8 + 4 + 0 + 1 = 13_{10}$

In binary, each digit is multiplied by powers of **2**.

3. Base 8 (Octal System)

- **Symbols**: 0, 1, 2, 3, 4, 5, 6, 7
- The octal system uses 8 symbols, and it is often used in computing as a shorthand for binary (because 1 octal digit equals 3 binary digits).

Example:

The number **624_8** in base 8 can be expanded as:

- $624_8 = 6 \times 8^2 + 2 \times 8^1 + 4 \times 8^0 = 384 + 16 + 4 = 404_{10}$

In octal, each digit is multiplied by powers of **8**.

4. Base 16 (Hexadecimal System)

- **Symbols**: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F
- Hexadecimal uses 16 symbols: the numbers 0-9 and the letters A-F (where A = 10, B = 11, ..., F = 15). It is often used in computing as a more compact way of representing binary data.

Example:

The number **$6A3_{16}$** in base 16 can be expanded as:

- $6A3_{16} = 6 \times 16^2 + A \times 16^1 + 3 \times 16^0$

Where **A** = 10, so:

- $6A3_{16} = (6 \times 256) + (10 \times 16) + (3 \times 1) = 1536 + 160 + 3 = 1699_{10}$

3. How Base Affects Numerals

- **Base 10 (Decimal)**: Uses 10 symbols, meaning that each position can have one of 10 values (0 through 9).
- **Base 2 (Binary)**: Uses only 2 symbols, 0 and 1. To represent larger numbers, you need more digits than in base 10.
- **Base 8 (Octal)**: Uses 8 symbols, so it requires fewer digits than binary but more digits than decimal for the same number.
- **Base 16 (Hexadecimal)**: Uses 16 symbols, making it more compact than base 2 or base 10 for representing large numbers.

4. Further Examples

1. Base 10 Example:

The number **123_{10}** means:

- 123_{10} = (1 x 10^2) + (2 x 10^1) + (3 x 10^0) = 100 + 20 + 3 = 123

2. Base 2 Example:

The number **1010_2** means:

- 1010_2 = (1 x 2^3) + (0 x 2^2) + (1 x 2^1) + (0 x 2^0) = 8 + 0 + 2 + 0 = 10_{10}

3. Base 8 Example:

The number **157_8** means:

- 157_8 = (1 x 8^2) + (5 x 8^1) + (7 x 8^0) = 64 + 40 + 7 = 111_{10}

4. Base 16 Example:

The number **$1F4_{16}$** means:

- $1F4_{16}$ = (1 x 16^2) + (F x 16^1) + (4 x 16^0)

Where **F** = 15:

- $1F4_{16}$ = (1 x 256) + (15 x 16) + (4 x 1) = 256 + 240 + 4 = 500_{10}

5. Key Takeaways

- **Base (or radix)** refers to the number of symbols in the numeral system.
- Larger bases require more symbols.
- Base 10 (decimal) is the most familiar, but base 2 (binary), base 8 (octal), and base 16 (hexadecimal) are crucial in computing.
- Each base uses the powers of the base to determine the value of each digit in a number.

Number of Symbols vs. Number of Digits

For a given number, the **base (or radix)** determines how many **symbols** (digits) are used in that number system. The **larger the base**, the more symbols are required to represent individual digits, but **fewer digits** are needed to represent the same number when compared to smaller bases.

In lower bases, like **binary (base 2)**, more digits are needed to represent the same value than in **hexadecimal (base 16)**, which has more available symbols.

1. Key Concepts

- **Number of Symbols**: The number of different characters (or digits) that can be used in a given base.
  - **Base 2** (Binary): 2 symbols (0, 1)
  - **Base 8** (Octal): 8 symbols (0, 1, 2, 3, 4, 5, 6, 7)
  - **Base 10** (Decimal): 10 symbols (0 to 9)
  - **Base 16** (Hexadecimal): 16 symbols (0 to 9, A to F)

- **Number of Digits**: The number of places (or length) required to represent a number. As the base increases, fewer digits are required to represent the same value.

2. Example #1

Consider the decimal number $\mathbf{101_{10}}$. This number can be represented in different bases (hexadecimal, binary, and octal).

- **Hexadecimal (Base 16)**
  - Representation: $\mathbf{65_{16}}$
  - Explanation: In base 16, $\mathbf{65_{16}}$ equals $\mathbf{101_{10}}$.
  - $65_{16} = (6 \times 16^{1}) + (5 \times 16^{0}) = 96 + 5 = 101_{10}$
  - Number of digits: 2

- **Decimal (Base 10)**
  - Representation: $\mathbf{101_{10}}$
  - Number of digits: 3

- **Octal (Base 8)**
  - Representation: **145_8**
  - Explanation: In base 8, **145_8** equals **101_{10}**.
  - 145_8 = (1 x 8^2) + (4 x 8^1) + (5 x 8^0) = 64 + 32 + 5 = 101_{10}
  - Number of digits: 3

- **Binary (Base 2)**
  - Representation: **110 0101_2**
  - Explanation: In base 2, **110 0101_2** equals **101_{10}**.
  - 1100 101_2 = (1 x 2^6) + (1 x 2^5) + (0 x 2^4) + (0 x 2^3) + (1 x 2^2) + (0 x 2^1) + (1 x 2^0) = 64 + 32 + 4 + 1 = 101_{10}
  - Number of digits: 7

3. Example #2

Consider the decimal number **284_{10}**. This number can also be represented in various bases.

- **Hexadecimal (Base 16)**
  - Representation: **$11C_{16}$**
  - Explanation: In base 16, **$11C_{16}$** equals **284_{10}**.
  - $11C_{16} = (1 \times 16^2) + (1 \times 16^1) + (C \times 16^0)$

    Here, **C = 12**:

    $11C_{16} = (1 \times 256) + (1 \times 16) + (12 \times 1) = 256 + 16 + 12 = 284_{10}$
  - Number of digits: 3

- **Decimal (Base 10)**
  - Representation: **284_{10}**
  - Number of digits: 3

- **Octal (Base 8)**
  - Representation: **434_8**
  - Explanation: In base 8, **434_8** equals **284_{10}**.
  - $434_8 = (4 \times 8^2) + (3 \times 8^1) + (4 \times 8^0) = 256 + 24 + 4 = 284_{10}$
  - Number of digits: 3

- **Binary (Base 2)**
  - Representation: **$1\ 0001\ 1100_2$**
  - Explanation: In base 2, **$1\ 0001\ 1100_2$** equals **284_{10}**.
  - $1\ 0001\ 1100_2 = (1 \times 2^8) + (0 \times 2^7) + (0 \times 2^6) + (0 \times 2^5) + (1 \times 2^4) + (1 \times 2^3) + (1 \times 2^2) + (0 \times 2^1)$

    $+ (0 \times 2^0) = 256$
  - Number of digits: 9

4. Key Observations:

- **Larger bases use more symbols**: Base 16 (hexadecimal) uses digits 0-9 and letters A-F, while base 2 (binary) uses only 0 and 1.
- **Smaller bases require more digits**: Base 2 requires many more digits to represent a number than base 16 or base 10.
- **Compact representation in larger bases**: The same number (e.g., **284_{10}**) requires only **3 digits** in hexadecimal but **9 digits** in binary.

5. Further Breakdown:

Decimal 101_{10} Across Bases:

- **Base 16 (Hexadecimal)**: **65_{16}** → 2 digits
- **Base 10 (Decimal)**: **101_{10}** → 3 digits
- **Base 8 (Octal)**: **145_{8}** → 3 digits
- **Base 2 (Binary)**: **$110\ 0101_{2}$** → 7 digits

Decimal 284_{10} Across Bases:

- **Base 16 (Hexadecimal)**: **$11C_{16}$** → 3 digits
- **Base 10 (Decimal)**: **284_{10}** → 3 digits
- **Base 8 (Octal)**: **434_{8}** → 3 digits
- **Base 2 (Binary)**: **$1\ 0001\ 1100_{2}$** → 9 digits

Conclusion

- **Higher bases (like hexadecimal)** are more **efficient** in representing numbers with fewer digits.
- **Lower bases (like binary)** require more digits, but they are easier to implement in digital systems (like computers) because of their simplicity.
- The choice of base affects both the number of **symbols** and the **number of digits** used to represent a number. For **compactness**, higher bases are preferred, but for simplicity, lower bases are used.

Counting in Base 2

Binary Number	Equivalent				Decimal Number
	8's (2^3)	4's (2^2)	2's (2^1)	1's (2^0)	
0				**0×2^0**	0
1				**1×2^0**	1
10			**1×2^1**	**0×2^0**	2
11			**1×2^1**	**1×2^0**	3
100		**1×2^2**			4
101		**1×2^2**		**1×2^0**	5
110		**1×2^2**	**1×2^1**		6
111		**1×2^2**	**1×2^1**	**1×2^0**	7
1000	**1×2^3**				8
1001	**1×2^3**			**1×2^0**	9
1010	**1×2^3**		**1×2^1**		10

This table shows how binary numbers (base 2) are converted to their equivalent decimal numbers (base 10). Here's an explanation for each part of the table:

1. Columns

- **Binary Number**: A number expressed in the binary system, which uses only two digits: 0 and 1. Each binary digit (or "bit") represents an increasing power of 2, starting from the right.

- **Equivalent**: This breaks down how the binary number can be expressed as a sum of powers of 2. Each binary digit is multiplied by a corresponding power of 2.
  - The powers of 2 are labeled as 1's (2^0), 2's (2^1), 4's (2^2), and 8's (2^3) in this table.

- **Decimal Number**: The base 10 equivalent of the binary number, calculated by summing the results of the products of binary digits and their respective powers of 2.

2. Explanation of Each Row

1. Binary: 0

- **Equivalent**: 0×2^{0}
- **Decimal**: 0
- **Explanation**: The binary number 0 is simply 0 in decimal.

2. Binary: 1

- **Equivalent**: 1×2^{0}
- **Decimal**: 1
- **Explanation**: The rightmost digit 1 represents $2^{0} = 1$.

3. Binary: 10 (which means "two" in binary)

- **Equivalent**: $1 \times 2^{1} + 0 \times 2^{0}$
- **Decimal**: 2
- **Explanation**: The 1 in the second position from the right represents $2^{1} = 2$, and the 0 in the first position represents 0.

4. Binary: 11 (which means "three" in binary)

- **Equivalent**: $1 \times 2^{1} + 1 \times 2^{0}$
- **Decimal**: 3
- **Explanation**: $1 \times 2^{1} = 2$ and $1 \times 2^{0} = 1$, so the sum is $2 + 1 = 3$.

5. Binary: 100 (which means "four" in binary)

- **Equivalent**: 1×2^{2}
- **Decimal**: 4
- **Explanation**: The 1 in the third position from the right represents $2^{2} = 4$.

6. Binary: 101 (which means "five" in binary)

- **Equivalent**: $1 \times 2^{2} + 0 \times 2^{1} + 1 \times 2^{0}$
- **Decimal**: 5
- **Explanation**: $1 \times 2^{2} = 4$ and $1 \times 2^{0} = 1$, so the sum is $4 + 1 = 5$.

7. Binary: 110 (which means "six" in binary)

- **Equivalent**: $1 \times 2^2 + 1 \times 2^1$
- **Decimal**: 6
- **Explanation**: $1 \times 2^2 = 4$ and $1 \times 2^1 = 2$, so the sum is $4 + 2 = 6$.

8. Binary: 111 (which means "seven" in binary)

- **Equivalent**: $1 \times 2^2 + 1 \times 2^1 + 1 \times 2^0$
- **Decimal**: 7
- **Explanation**: $1 \times 2^2 = 4$, $1 \times 2^1 = 2$, and $1 \times 2^0 = 1$, so the sum is $4 + 2 + 1 = 7$.

9. Binary: 1000 (which means "eight" in binary)

- **Equivalent**: 1×2^3
- **Decimal**: 8
- **Explanation**: The 1 in the fourth position from the right represents $2^3 = 8$.

10. Binary: 1001 (which means "nine" in binary)

- **Equivalent**: $1 \times 2^3 + 0 \times 2^2 + 0 \times 2^1 + 1 \times 2^0$
- **Decimal**: 9
- **Explanation**: $1 \times 2^3 = 8$ and $1 \times 2^0 = 1$, so the sum is $8 + 1 = 9$.

11. Binary: 1010 (which means "ten" in binary)

- **Equivalent**: $1 \times 2^3 + 0 \times 2^2 + 1 \times 2^1 + 0 \times 2^0$
- **Decimal**: 10
- **Explanation**: $1 \times 2^3 = 8$ and $1 \times 2^1 = 2$, so the sum is $8 + 2 = 10$.

3. Further Example

Binary: 1101 (which means "thirteen" in binary)

- **Equivalent**: $1 \times 2^3 + 1 \times 2^2 + 0 \times 2^1 + 1 \times 2^0$
- **Decimal**: 13
- **Explanation**:
    - $1 \times 2^3 = 8$
    - $1 \times 2^2 = 4$
    - $0 \times 2^1 = 0$
    - $1 \times 2^0 = 1$
    - Sum = 8 + 4 + 0 + 1 = 13

Binary: 1111 (which means "fifteen" in binary)

- **Equivalent**: $1 \times 2^3 + 1 \times 2^2 + 1 \times 2^1 + 1 \times 2^0$
- **Decimal**: 15
- **Explanation**:
    - $1 \times 2^3 = 8$
    - $1 \times 2^2 = 4$
    - $1 \times 2^1 = 2$
    - $1 \times 2^0 = 1$
    - Sum = 8 + 4 + 2 + 1 = 15

This pattern continues as you move up in binary digits, with each new bit representing a higher power of 2. The decimal value is calculated by summing the products of binary digits and their respective powers of 2.

Base 10 Addition Table

$3_{10} \times 6_{10} = 18_{10}$

+	0	1	2	3	4	5	6	7	8	9
0	0	1	2	3	4	5	6	7	8	9
1	1	2	3	4	5	6	7	8	9	10
2	2	3	4	5	6	7	8	9	10	11
3	3	4	5	6	7	8	9	10	11	12
4	4	5	6	7	8	9	10	11	12	13

etc

The table you've provided shows an **addition table in base 10** (decimal), demonstrating how two numbers are added together.

1. Key Parts of the Table

- **Numbers Along The Horizontal And Vertical Axes**:
  - The top row (horizontal) and the first column (vertical) represent numbers in base 10 (decimal). These numbers range from 0 to 9.
  - The first column (vertical) shows the first addend, and the top row (horizontal) shows the second addend in the addition operation.

- **Addition Results In The Cells**:
  - Each cell in the table represents the sum of the corresponding numbers from the top row and the first column.
  - For example, the cell at the intersection of row "3" and column "6" shows "9" (i.e., 3 + 6 = 9 in base 10).

- **Highlighted Example**:
    - The table highlights the specific addition of $3_{10} + 6_{10} = 9_{10}$ with the number "9" being highlighted.
    - The highlighted path traces the numbers "3" from the first column and "6" from the top row to their intersection, where the sum "9" appears.

- **Color-Coding**:
    - The blue colouring emphasizes the result of adding numbers in the specific example.
    - The orange lines guide you from the row and column of "3" and "6" to their sum.

2. Detailed Example

Let's break down the highlighted example, 3 + 6 = 9:

- **First Addend**: 3 (vertical axis).
- **Second Addend**: 6 (horizontal axis).
- **Result**: The cell where the row labeled "3" and the column labeled "6" meet contains "9," which is the sum of the two numbers in base 10.

3. Further Examples from the Table

- **Example: 2 + 4**
    - Look for the row starting with 2 and the column starting with 4.
    - At the intersection, you will find the number "6," so 2 + 4 = 6.
- **Example: 7 + 8**
    - Look for the row starting with 7 and the column starting with 8.
    - At the intersection, the result is 15, so 7 + 8 = 15 in base 10.

4. How the Table Works

- Each intersection of a row and column gives the sum of the corresponding numbers.
- The top row and first column help you locate the two numbers being added, and the corresponding cell shows the result.

5. Application of the Addition Table

- **Educational Tool**: This type of addition table is useful for learning addition in base 10, particularly for those who are new to arithmetic or children learning basic math.
- **Pattern Recognition**: As you study the table, you can recognize patterns, such as how sums increase as you move right along a row or down a column.

Base 8 Addition Table

$3_8 + 6_8 = 11_8$

+	0	1	2	3	4	5	6	7
0	0	1	2	3	4	5	6	7
1	1	2	3	4	5	6	7	10
2	2	3	4	5	6	7	10	11
3	3	4	5	6	7	10	11	12
4	4	5	6	7	10	11	12	13
5	5	6	7	10	11	12	13	14
6	6	7	10	11	12	13	14	15
7	7	10	11	12	13	14	15	16

(no 8 or 9, of course)

The image you've provided shows a partial addition table in base 8 (octal system), with a highlighted calculation: $3_8 + 6_8 = 11_8$.

1. Octal (Base 8) System

- **Base 8** uses digits from 0 to 7, meaning no digit in a base 8 number can be 8 or 9.
- Every number is expressed using only the digits 0, 1, 2, 3, 4, 5, 6, and 7.

2. Explanation of $3_8 + 6_8 = 11_8$

- **Octal Digits**:
  - In base 8, 3_8 it represents three and 6_8 represents six.
  - These values are similar to their decimal (base 10) equivalents because they are less than 8.
- **Addition in Base 8**:
  - If we perform the addition 3 + 6 in decimal (base 10), the sum is 9.
  - Since 9 is not a valid digit in base 8, we must convert it:
    - 9_{10} is equal to 11_8 because in base 8, the number 9 is written as $1 \times 8^1 + 1 \times 8^0$, or 11_8.

Thus, $3_8 + 6_8 = 11_8$.

3. Key Concept

- **Carrying in Base 8**:
  - When adding numbers in base 8, you "carry over" once you reach 8 (just like you carry over after 9 in base 10).
  - For example, in base 10, 7 + 5 = 12, so you write down 2 and carry over 1.
  - In base 8, $7_8 + 3_8 = 12_8$ (write down 2 and carry over 1 to the next column).

4. Addition Table in the Image

- The table provided helps illustrate the addition of octal numbers.
- The numbers on the top and left of the table represent the digits to be added, and their sum is found at the intersection.
  - For example, find 3 on the left column and 6 on the top row. The intersection gives you the result, 11_8.

5. Example for Further Understanding

- Let's take another example from the table:
  - $5_8 + 7_8$:
    - 5 + 7 = 12_{10}, which is 14_8.
    - Therefore, $5_8 + 7_8 = 14_8$.

6. Octal to Decimal Conversion

- **To Convert Octal to Decimal**:
  - For example, 11_8:
    - The value of 11811_8118 in decimal is calculated as:
      - $1 \times 8^1 + 1 \times 8^0 = 8 + 1 = 9_{10}$.
    - So $11_8 = 9_{10}$.

- **To Convert Decimal to Octal**:
  - Convert decimal 9 to octal:
    - Divide 9 by 8, which gives 1 remainder 1.
    - Therefore, $9_{10} = 11_{88}$.

By understanding how to add in base 8 and how octal values map to decimal, you can apply these principles to perform other operations like multiplication, subtraction, and division in base 8.

Base 10 Multiplication Table

3_{10} x 6_{10} = 18_{10}

x	0	1	2	3	4	5	6	7	8	9
0				0						
1		1	2	3	4	5	6	7	8	9
2		2	4	6	8	10	12	14	16	18
3		3	6	9	12	15	18	21	24	27
4	0	4	8	12	16	20	24	28	32	36
5		5	10	15	20	25	30	35	40	45
6		6	12	18	24	30	36	42	48	54
7		7	14	21	28	35	42	49	56	63

etc.

The image provided shows a **Base 10 (decimal) multiplication table** with the highlighted operation $3_{10} \times 6_{10} = 18_{10}$.

1. Base 10 (Decimal) System

- **Base 10** uses digits from 0 to 9. This is the system most commonly used in everyday arithmetic.
- Every number in base 10 is expressed using the digits 0 to 9.

2. Explanation of $3_{10} \times 6_{10} = 18_{10}$

- **Decimal Digits**:
  - In base 10, 3_{10} represents three and 6_{10} represents six, which are their actual values (since base 10 is the standard system).

- **Multiplication in Base 10**:
  - When multiplying 3 × 6, the result is 18_{10}.

3. Structure of the Table:

- **Multiplication Table**: The numbers on the top row and left column represent the numbers to be multiplied. The result of their product is found at the intersection of these values.
  - For example, find 3 on the left column and 6 on the top row. The intersection gives the result, 18_{10}, which is the product of 3 × 6.

4. Key Concept

- **Multiplication** is repeated addition. For example:
  - 3 × 6 can be understood as adding 3 six times: 3 + 3 + 3 + 3 + 3 + 3 = 18.

5. Example for Further Understanding

- Let's take another example from the table:
  - $5_{10} \times 7_{10}$:
    - The product of 5 × 7 = 35.
    - Therefore, $5_{10} \times 7_{10} = 35_{10}$.

6. Properties of Multiplication

- **Commutative Property**:
  - a × b = b × a.
  - For example, 3 × 6 = 6 × 3, and both give 18.

- **Associative Property**:
  - When multiplying three or more numbers, the grouping of the numbers doesn't affect the result.
  - For example, (2×3) × 4 = 2 × (3×4), both result in 24.

- **Distributive Property**:
  - a × (b+c) = a × b + a × c.
  - For example, 3 × (5+6) = 3 × 5 + 3 × 6 = 15 + 18 = 33.

7. Why Use the Multiplication Table?

- The table helps with quick mental multiplication and is often used in elementary education to teach the concept of multiplication.
- It also reinforces the relationships between numbers and shows patterns (such as how multiples of 5 always end in 0 or 5).

8. Another Detailed Example:

- $4_{10} \times 9_{10}$:
  - From the table, find 4 on the left column and 9 on the top row.
  - The intersection gives 36_{10}, which is the product of 4 × 9.

Conclusion

- The base 10 multiplication table is a fundamental tool for learning how numbers relate to each other through multiplication.
- The example $3_{10} \times 6_{10} = 18_{10}$ is straightforward, as you directly perform multiplication in the decimal system without the need for conversion (unlike in other bases such as base 8 or base 2).

Base 8 Multiplication Table

3_8 x 6_8 = 22_8

x	0	1	2	3	4	5	6	7
0				0				
1		1	2	3	4	5	6	7
2		2	4	6	10	12	14	16
3	0	3	6	11	14	17	22	25
4		4	10	14	20	24	30	34
5		5	12	17	24	31	36	43
6		6	14	22	30	36	44	52
7		7	16	25	34	43	52	61

The table you've provided shows a **multiplication table in base 8** (octal), demonstrating how two numbers in base 8 are multiplied together.

1. Key Parts of the Table

- **Numbers Along The Horizontal And Vertical Axes**:
  - The top row (horizontal) and the first column (vertical) represent numbers in base 8, ranging from 0 to 7.
  - The first column (vertical) represents the first factor in the multiplication operation, and the top row (horizontal) represents the second factor.

- **Multiplication Results In The Cells**:
  - Each cell in the table represents the product of the corresponding numbers from the top row and the first column.
  - For example, the cell at the intersection of row "3" and column "6" shows "22", meaning $3_8 \times 6_8 = 22_8$ in base 8.

- **Highlighted Example**:
  - The table highlights the specific multiplication of $3_8 \times 6_8 = 22_8$, with the number "22" highlighted in blue.
  - The highlighted path traces the numbers "3" from the first column and "6" from the top row to their intersection, where the product "22" (in base 8) appears.

- **Color-Coding**:
  - The blue shading highlights the result of multiplying numbers in the specific example.
  - The orange lines guide you from the row and column of "3" and "6" to their product.

2. Explanation of the Highlighted Example

In the octal number system (base 8), the digits range from 0 to 7. The highlighted example shows the multiplication of two base 8 numbers, $3_8 \times 6_8$:

- **First factor**: 3 (vertical axis, in base 8).
- **Second factor**: 6 (horizontal axis, in base 8).
- **Product**: The cell where the row labeled "3" and the column labeled "6" meet contains "22", which represents $3_8 \times 6_8 = 22_8$.

In base 10, the values of the base 8 numbers are:

- $3_8 = 3_{10}$
- $6_8 = 6_{10}$

Multiplying in base 10:

- $3_{10} \times 6_{10} = 18_{10}$

Converting 181018_{10}1810 to base 8:

- $18_{10} = 22_8$

So, $3_8 \times 6_8 = 22_8$.

3. Further Examples from the Table

- **Example**: $2_8 \times 4_8$
  - Look for the row starting with 2 and the column starting with 4.
  - At the intersection, the result is "10", so $2_8 \times 4_8 = 10_8$.

- In base 10:
  - $2_8 = 2_{10}$ and $4_8 = 4_{10}$
  - $2_{10} \times 4_{10} = 8_{10}$
  - Converting 8_{10} to base 8 gives 10_8.

- **Example**: $5_8 \times 7_8$
  - Look for the row starting with 5 and the column starting with 7.
  - The intersection shows "43", so $5_8 \times 7_8 = 43_8$.

- In base 10:
  - $5_8 = 5_{10}$ and $7_8 = 7_{10}$
  - $5_{10} \times 7_{10} = 35_{10}$
  - Converting35_{10} to base 8 gives 43_8.

4. How to Use the Base 8 Multiplication Table

- To find the product of two base 8 numbers, locate the first factor in the vertical column and the second factor in the horizontal row. The cell at their intersection gives the result in base 8.

5. Application of the Multiplication Table

- **Educational Tool**: This multiplication table is helpful for learning base 8 multiplication, especially when working with number systems other than decimal.
- **Octal Arithmetic**: It can be used in computer science, where base 8 (octal) is sometimes used for addressing or representing data more compactly than binary.

Converting From Base 10

Power / Base	8	7	6	5	4	3	2	1	0
2	256	128	64	32	16	8	4	2	1
8				32,768	4,096	512	64	8	1
16					65,536	4,096	256	16	1

The table you've shared seems to be a "Powers Table" for converting numbers from Base 10 to other bases, such as binary (Base 2), octal (Base 8), and hexadecimal (Base 16). Here's a breakdown of the table and an explanation for each part:

1. Understanding the Powers Table

This table shows the powers of different base numbers (2, 8, and 16) raised to different exponents, typically used in number systems and conversions.

1A. Columns (Power 8 to Power 0)

Each column represents the **exponent** that the base is raised to. The power ranges from 888 down to 000.

- The first column is $Base^8$ (base raised to the power of 8)
- The next column is $Base^7$ (base raised to the power of 7), and so on.
- The last column is $Base^0$, which is always 1, because any number raised to the power of 0 is 1.

1B. Rows (Bases 2, 8, and 16)

Each row represents a different base (2, 8, and 16) used for the conversion. Each base represents a different numeral system:

- **Base 2** (binary) is a numeral system that uses only two digits: 0 and 1.
- **Base 8** (octal) is a numeral system that uses eight digits: 0-7.
- **Base 16** (hexadecimal) is a numeral system that uses sixteen digits: 0-9, followed by A, B, C, D, E, F (representing values 10-15).

2. Explanation of Each Row

- **Base 2 (Binary)**:
  - The powers of 2 are important in computing, where binary is the foundation of all computer systems.
  - The values represent 2^8 = 256, 2^7 = 128, down to 2^0 = 1.
  - Example: If you wanted to represent the number 150 in binary, you would break it down into sums of powers of 2:
    - 150 = 128(2^7) + 16(2^4) + 4(2^2) + 2(2^1)
    - In binary, this would be represented as 1001 0110.

- **Base 8 (Octal)**:
  - The values represent powers of 8: 8^8, 8^7, etc. For practical purposes, we often stop at 8^5 and lower, as the values quickly become large.
  - Example: To represent 512 in octal, we use 512 = 8^3, which in octal would be written as 1000.

- **Base 16 (Hexadecimal)**:
  - Hexadecimal is widely used in computing, particularly for memory addresses and color codes in HTML and CSS.
  - The powers of 16 are: 16^8 = 65536, 16^4=4,096 etc.
  - Example: To convert the decimal number 255 into hexadecimal:
    - 255 = $16^1 \times 15 + 16^0 \times 15 \rightarrow$ which becomes F in hexadecimal (since F represents 15).

3. Detailed Examples for Further Explanation

Example 1: Converting 45 (Base 10) to Base 2 (Binary):

- Break down 45 into sums of powers of 2:
  - 45 = 32 + 8 + 4 + 1
  - This corresponds to $2^5 + 2^3 + 2^2 + 2^0$
- In binary, this becomes: **101101**.

Example 2: Converting 64 (Base 10) to Base 8 (Octal):

- Divide 64 by 8 to find the coefficients:
  - 64 ÷ 8 = 8, remainder 0
  - 8 ÷ 8 = 1, remainder 0
  - So, 64 = 100 in octal.

Example 3: Converting 43981 (Base 10) to Base 16 (Hexadecimal):

- Divide 43981 by 16:
  - 43981 ÷ 16 = 2748 remainder 13 (D in hex)
  - 2748 ÷ 16 = 171 remainder 12 (C in hex)
  - 171 ÷ 16 = 10 remainder 11 (B in hex)
  - 10 ÷ 16 = 0 remainder 10 (A in hex)
- So, 43981 in hexadecimal is **ABCD**.

Conclusion

This table is highly useful for converting decimal (Base 10) numbers to other bases, which is a common task in digital electronics, networking (IP addresses), and software development. The table helps visualize how numbers break down in these systems, especially in binary (Base 2), octal (Base 8), and hexadecimal (Base 16).

From Base 10 To Base 2

$42_{10} = 101010_2$

Power / Base	6	5	4	3	2	1	0
2	64	32	16	8	4	2	1
		1	0	1	0	1	0
Integer		42/32 = 1	10/16 = 0	10/8 = 1	2/4 = 0	2/2 = 1	0/1 = 0
Remainder		10	10	2	2	0	0

The image provided shows the conversion of the decimal number 42 (base 10) to its binary equivalent (base 2). Let's break it down in detail:

1. Decimal to Binary Conversion Process

- **Base 10 (Decimal)**:
  - Decimal (base 10) is the standard numbering system most commonly used by humans. It consists of 10 digits, ranging from 0 to 9.

- **Base 2 (Binary)**:
  - Binary (base 2) is a numbering system that uses only two digits: 0 and 1. It is used in computing because computers operate using a binary system of logic (on and off, or 1 and 0).

2. Converting 42 (Base 10) To Binary

The goal is to express the number 421042_{10}4210 in binary form, which becomes 101010_2.

Steps to Convert from Decimal to Binary:

- **Determine the Powers of 2**:
  - Write down the powers of 2 from the largest possible power that is less than or equal to 42.
    - $2^6 = 64$
    - $2^5 = 32$
    - $2^4 = 16$
    - $2^3 = 8$
    - $2^2 = 4$
    - $2^1 = 2$
    - $2^0 = 1$

- **Divide the Decimal Number by Successive Powers of 2**:
  - Start with the largest power of 2 that fits into the number, and keep track of both the integer quotient and the remainder at each step.

- **Step-by-Step Division**:
  - **42 ÷ 32 = 1** (integer quotient = 1, remainder = 42 - 32 = 10)
  - **10 ÷ 16 = 0** (integer quotient = 0, remainder = 10)
  - **10 ÷ 8 = 1** (integer quotient = 1, remainder = 10 - 8 = 2)
  - **2 ÷ 4 = 0** (integer quotient = 0, remainder = 2)
  - **2 ÷ 2 = 1** (integer quotient = 1, remainder = 2 - 2 = 0)
  - **0 ÷ 1 = 0** (integer quotient = 0, remainder = 0)

- **Write the Binary Result**:
  - The quotients from each division give you the binary digits from left to right (from the largest power of 2 to the smallest).
  - So, $42_{10} = 101010_2$.

3. Explanation of the Table

- The table provided shows this step-by-step division process:
  - The first row lists the powers of 2: $2^6, 2^5, \ldots, 2^0$.
  - The second row shows the values of these powers (64, 32, 16, 8, 4, 2, 1).
  - The third row (labeled "Integer") shows the result of dividing the current number by the corresponding power of 2 (either a 1 or 0, depending on whether the division results in a quotient of 1 or not).
  - The fourth row (labeled "Remainder") shows the remainder after subtracting the largest power of 2 used in each step.

4. Binary Representation

From the table:

- The binary equivalent of 42_{10} = 101010_2, which corresponds to:
  - 1×32 + 0×16 + 1×8 + 0×4 + 1×2 + 0×1 = 32 + 8 + 2 = 42.

5. Another Example

Let's convert 19_{10} to binary:

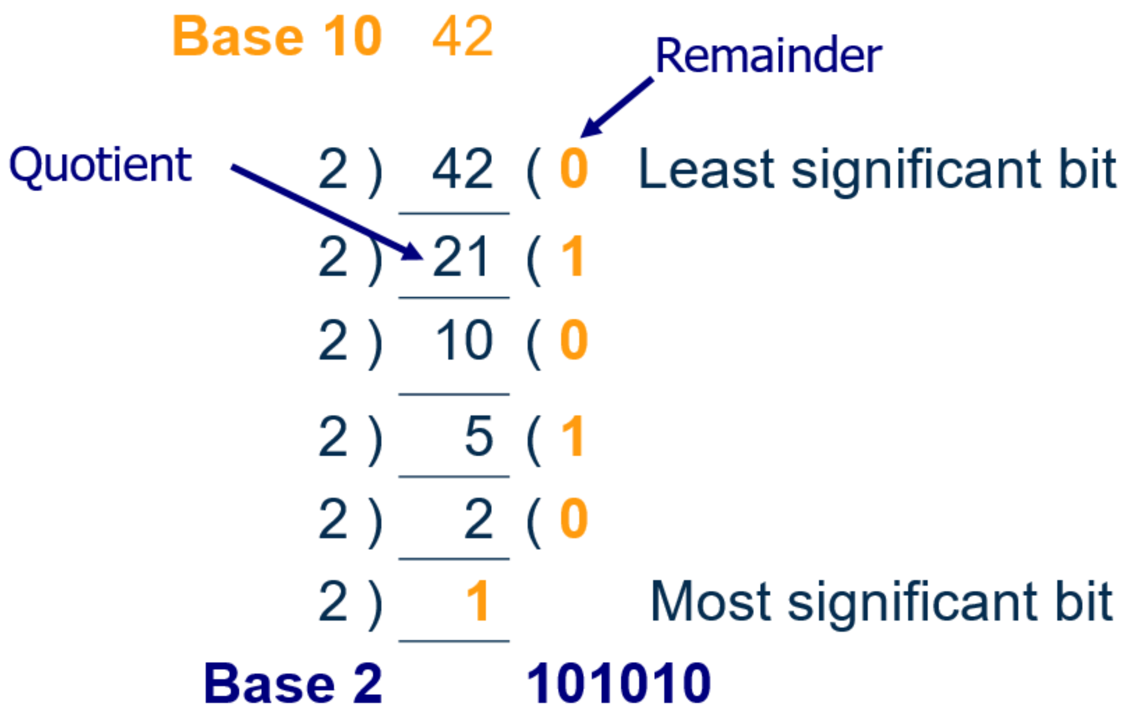
- List the powers of 2: 2^4 = 16, 2^3 = 8, 2^2 = 4, 2^1 = 2, 2^0 = 1.

- Divide 19 by the largest power of 2:
  - 19 ÷ 16 = 1 (remainder = 19 - 16 = 3)
  - 3 ÷ 8 = 0
  - 3 ÷ 4 = 0
  - 3 ÷ 2 = 1 (remainder = 3 - 2 = 1)
  - 1 ÷ 1 = 1 (remainder = 0)

- The binary representation of 19 is 10011_2:
  - 1×16 + 0×8 + 0×4 + 1×2 + 1×1 = 16 + 2 + 1 = 19.

By using this process, any decimal number can be converted into its binary equivalent.

From Base 10 To Base 2



The image shows the process of converting a number from **Base 10 (Decimal)** to **Base 2 (Binary)**. Here, the example is the conversion of the decimal number 42 to its binary equivalent (101010). Let's break it down step by step:

1. Step-By-Step Explanation Of Base 10 To Base 2 Conversion

- **Start with the Decimal Number (Base 10)**:
  - The given number is **42** in base 10 (decimal).
- **Divide by 2**:
  - To convert the number to binary (base 2), continuously divide the number by 2, keeping track of both the **quotient** and the **remainder** at each step.
- **Write the Remainder**:
  - At each step, record the remainder (0 or 1). This remainder is a binary digit, either **0** or **1**. The remainder at each division will be part of the binary number.
- **Repeat until the Quotient is 0**:
  - Continue dividing the quotient by 2 until you reach 0. Once the quotient becomes zero, stop the process.
- **Read the Binary Digits from Bottom to Top**:
  - The binary digits collected from the remainder should be read **in reverse order**, starting from the last division (which gives the most significant bit) to the first division (which gives the least significant bit).

2. Breakdown Of The Example In The Image

- **Step 1:**
  - **42 ÷ 2** = 21, remainder **0**
  - The remainder is **0** (which will be the least significant bit of the binary number).
- **Step 2:**
  - **21 ÷ 2** = 10, remainder **1**
  - The remainder is **1**.
- **Step 3:**
  - **10 ÷ 2** = 5, remainder **0**
  - The remainder is **0**.
- **Step 4:**
  - **5 ÷ 2** = 2, remainder **1**
  - The remainder is **1**.
- **Step 5:**
  - **2 ÷ 2** = 1, remainder **0**
  - The remainder is **0**.
- **Step 6:**
  - **1 ÷ 2** = 0, remainder **1**
  - The remainder is **1** (this will be the most significant bit).
- **Resulting Binary Number:**
  - The remainders are: 1, 0, 1, 0, 1, 0.
  - Read from bottom to top, the binary representation of **42** is **101010** in base 2.

3. General Rules

- The remainders will always be **0** or **1** because we are dividing by 2.
- The binary number is constructed by reading the remainders from the last step (most significant bit) to the first step (least significant bit).
- Each remainder corresponds to a binary digit, either 0 or 1, which together form the binary number.

3. Example 2: Convert 25 from Base 10 to Base 2:

- **25 ÷ 2** = 12, remainder **1**
- **12 ÷ 2** = 6, remainder **0**
- **6 ÷ 2** = 3, remainder **0**
- **3 ÷ 2** = 1, remainder **1**
- **1 ÷ 2** = 0, remainder **1**

The binary representation of **25** is **11001** (reading remainders from bottom to top).

Summary

- **Decimal to binary conversion** involves repeated division by 2.
- The remainders form the binary digits, read from bottom to top.

From Base 10 To Base 16

$5,735_{10} = 1667_{16}$

Power / Base	4	3	2	1	0
16	65,536	4,096	256	16	1
		1	**6**	**6**	**7**
Integer		5,735 /4,096 = 1	1,639 / 256 = 6	103 /16 = 6	7
Remainder		5,735 - 4,096 = 1,639	1,639 –1,536 = 103	103 – 96 = 7	

The image demonstrates how to convert a decimal number (base 10) to a hexadecimal number (base 16). In this case, the number **5,735** in base 10 is being converted to **1,667** in base 16. Let's break this process down step by step:

1. Steps to Convert Decimal to Hexadecimal (Base 10 to Base 16):

- **Identify the Base Powers of 16**:
  - In hexadecimal, each digit represents a power of 16. The columns in the table represent powers of 16, starting from the largest power down to the smallest:
    - $16^4 = 65,536$
    - $16^3 = 4,096$
    - $16^2 = 256$
    - $16^1 = 16$
    - $16^0 = 1$

- **Divide the Decimal Number by the Largest Power of 16**:
  - Start by dividing the decimal number (5,735) by the largest power of 16 that fits into it. In this case, we start with $16^3 = 4,096$ since $16^4 = 65,536$ is too large.

2. Detailed Steps From The Example

Power of 16^3 = 4,096:

- **Step 1:**
  - **Divide 5,735 by 4,096**:
    5,735 ÷ 4,096 = 1 (integer division)
  - So, the digit in the 16^3 place is **1**.
- **Step 2:**
  - **Find the remainder**:
    5,735 − (1×4,096) = 5,735 − 4,096 = 1,639
    We will now work with this remainder (1,639) for the next power of 16.

Power of 16^2 = 256:

- **Step 3:**
  - **Divide 1,639 by 256**:
    1,639 ÷ 256 = 6 (integer division)
  - So, the digit in the 16^2 place is **6**.
- **Step 4:**
  - **Find the remainder**:
    1,639 − (6×256) = 1,639 − 1,536 = 103
    We will now work with this remainder (103) for the next power of 16.

Power of 16^1 = 16:

- **Step 5:**
  - **Divide 103 by 16**:
    103 ÷ 16 = 6 (integer division)
  - So, the digit in the 16^1 place is **6**.
- **Step 6:**
  - **Find the remainder**:
    103 − (6×16) = 103 − 96 = 7
    This remainder (7) will now be the digit for the next power of 16, which is 16^0 **= 1**.

Power of 16^0 = 1:

- **Step 7:**
  - The remainder is **7**, so the digit in the 16^0 place is **7**.

Final Result:

- The hexadecimal equivalent of 5,735 in base 10 is **1,667** in base 16.

3. General Approach For Base 10 To Base 16 Conversion

- Start with the decimal number and divide it by the largest power of 16 that fits into it.
- Record the quotient (integer result) as the digit in the corresponding hexadecimal place.
- Subtract the product of the quotient and the power of 16 from the decimal number to get the remainder.
- Repeat the process for the next lower power of 16 using the remainder.
- Continue until you reach 16^0, which gives the final digit.

4. Example 2: Convert 2,578 From Base 10 To Base 16

- **Power 16^2 = 256**:
  2,578 ÷ 256 = 10
  So, the digit in the 16^2 place is **A** (in hexadecimal, 10 is represented as 'A').

- **Find The Remainder**:
  2,578 − (10×256) = 2,578 − 2,560 = 18

- **Power 16^1 = 16**:
  18 ÷ 16 = 1
  So, the digit in the 16^1 place is **1**.

- **Find The Remainder**:
  18 − (1×16) = 18 − 16 = 2

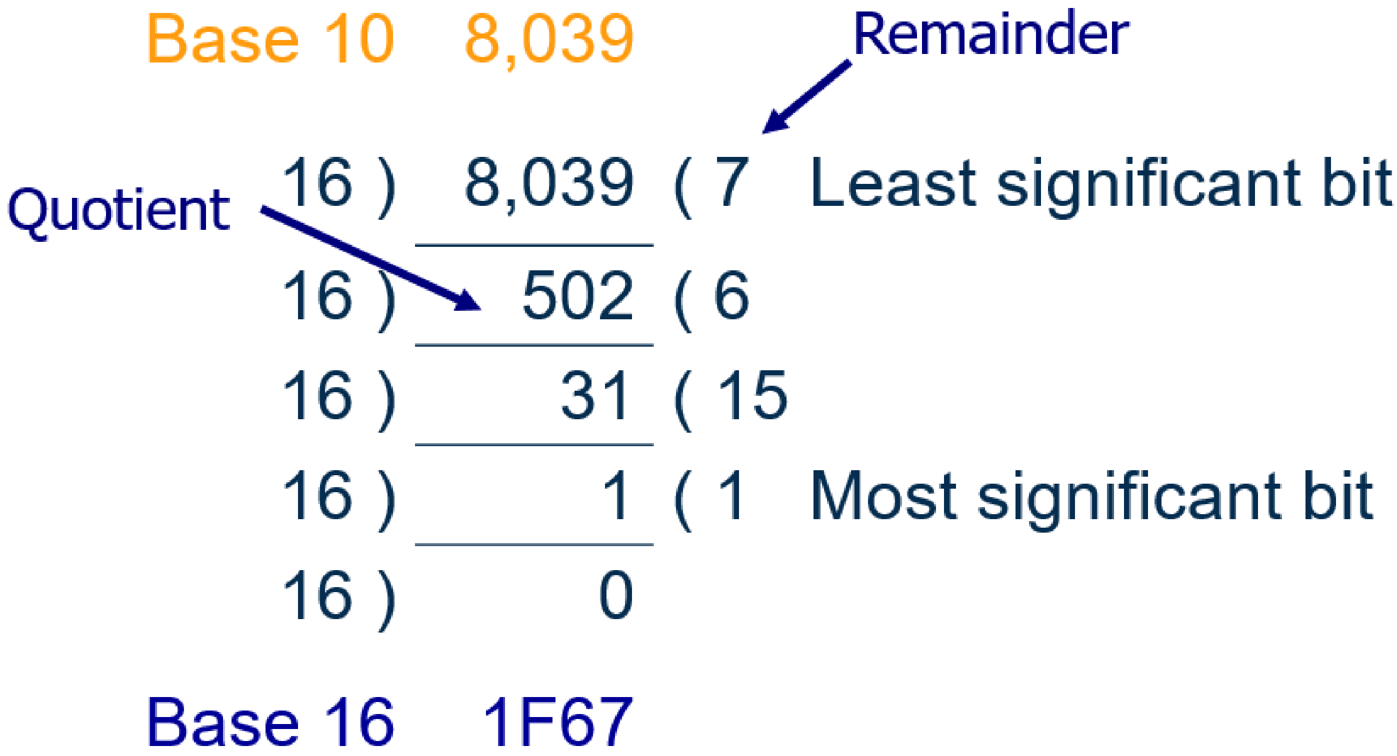
- **Power 16^0 = 1**:
  The remainder is **2**, so the digit in the 16^0 place is **2**.

Final result: **2,578** in base 10 is **A12** in base 16.

Summary

- Base 10 to base 16 conversion uses powers of 16 and repeated division, with remainders forming the digits of the hexadecimal number.
- Larger powers of 16 handle larger chunks of the number, and the process continues until no further division is possible.

From Base 10 To Base 16



The image demonstrates the process of converting the decimal number **8,039** (in base 10) to hexadecimal (base 16), resulting in the hexadecimal value **1F67**. Let's explain this step-by-step and generalize the method for converting base 10 to base 16.

1. Step-By-Step Explanation

1. Start with the Decimal Number

- The given number is **8,039** in base 10.

2. Divide by 16

- To convert from base 10 to base 16, repeatedly divide the decimal number by 16, keeping track of both the quotient and the remainder at each step.

3. Record the Remainders

- The remainder of each division will be a digit in the hexadecimal number. This remainder could be between 0 and 15:
  - If the remainder is between 0 and 9, it corresponds directly to the same digit in hexadecimal.
  - If the remainder is between 10 and 15, it corresponds to the letters A–F in hexadecimal:
    - 10 = A, 11 = B, 12 = C, 13 = D, 14 = E, 15 = F

4. Repeat until the Quotient is 0

- Continue dividing the quotient by 16 until you reach 0.

5. Read the Digits from Bottom to Top

- The hexadecimal digits are formed by reading the remainders in reverse order (from the last step to the first). The first remainder will be the **least significant bit (LSB)**, while the last quotient gives the **most significant bit (MSB)**.

2. Breakdown Of The Example In The Image

Step 1:

- **Divide 8,039 by 16**:
  8,039 ÷ 16 = 502 with a remainder of **7**.
  - So, the remainder **7** is the **least significant digit** in the hexadecimal representation.

Step 2:

- **Divide 502 by 16**:
  502 ÷ 16 = 31 with a remainder of **6**.
  - So, the remainder **6** is the next hexadecimal digit.

Step 3:

- **Divide 31 by 16**:
  31 ÷ 16 = 1 with a remainder of **15**.
  - The remainder **15** corresponds to the hexadecimal digit **F** (since 15 = F in hexadecimal).

Step 4:

- **Divide 1 by 16**:
  1 ÷ 16 = 0 with a remainder of **1**.
  - This quotient is now 0, and the remainder **1** is the **most significant digit** in the hexadecimal representation.

Final Hexadecimal Number:

- Reading the remainders from bottom to top, the hexadecimal equivalent of **8,039** in base 10 is **1F67** in base 16.

3. General Procedure for Base 10 to Base 16 Conversion

- **Divide the number by 16** repeatedly.
- **Record the remainder** at each step.
- **Stop when the quotient is 0**.
- **Read the remainders in reverse order** (from the last remainder to the first).

4. Example 2: Convert 2,955 From Base 10 To Base 16

Step 1:

- **Divide 2,955 by 16**:
  2,955 ÷ 16 = 184, remainder **11**.
  - Remainder 11 = **B** in hexadecimal.

Step 2:

- **Divide 184 by 16**:
  184 ÷ 16 = 11, remainder **8**.

Step 3:

- **Divide 11 by 16**:
  11 ÷ 16 = 0, remainder **11**.
  - Remainder 11 = **B**.

The hexadecimal representation of **2,955** is **B8B**.

Summary

- The process involves dividing by 16, recording the remainders, and then reading the remainders from bottom to top.
- Remainders above 9 convert to the letters A–F in hexadecimal.

From Base 8 To Base 10

$7263_8 = 3{,}763_{10}$

Power	8^3	8^2	8^1	8^0
	512	64	8	1
	x 7	x 2	x 6	x 3
Sum for Base 10	3,584	128	48	3

The image demonstrates how to convert a number from **base 8 (octal)** to **base 10 (decimal)**. In this example, the octal number **7263_8** is being converted to its decimal equivalent, **$3{,}763_{10}$**. Let's explain this process step-by-step and provide additional examples.

1. Understanding Base 8 (Octal)

- Base 8 uses the digits **0 to 7**, meaning each digit in the octal system represents a power of 8.
- To convert an octal number to a decimal number, each digit of the octal number is multiplied by the appropriate power of 8, based on its position.

2. Steps To Convert From Base 8 To Base 10

- **Identify the Powers of 8**:
  - For a number written in base 8, starting from the right, the position of each digit represents a power of 8:
    - The first digit (rightmost) is multiplied by 8^0,
    - The second digit is multiplied by 8^1,
    - The third digit is multiplied by 8^2,
    - The fourth digit is multiplied by 8^3, and so on.

- **Multiply Each Digit by the Corresponding Power of 8**:
  - Multiply each digit by the corresponding power of 8, based on its position.

- **Sum the Products**:
  - Add the results of all the multiplications to get the equivalent decimal (base 10) number.

3. Detailed Breakdown Of The Example (7263_8 To $3,763_{10}$)

The octal number is **7263_8**. Let's break it down:

Step 1: Break the number down by powers of 8

$7263_8 = (7\times8^3) + (2\times8^2) + (6\times8^1) + (3\times8^0)$

- **Power 8^3 = 512:**
  - $7 \times 512 = 3,584$
  - So, the first term contributes **3,584** to the decimal number.

- **Power 8^2 = 64:**
  - $2 \times 64 = 128$
  - So, the second term contributes **128** to the decimal number.

- **Power 8^1 = 8:**
  - $6 \times 8 = 48$
  - So, the third term contributes **48** to the decimal number.

- **Power 8^0 = 1:**
  - $3 \times 1 = 3$
  - So, the fourth term contributes **3** to the decimal number.

Step 2: Add the products together

$3,584 + 128 + 48 + 3 = 3,763$

Thus, the decimal equivalent of **7263_8** is **$3,763_{10}$**.

4. General Approach To Convert From Base 8 To Base 10

- Identify the position of each digit in the octal number.
- Multiply each digit by 8^n, where **n** is the position of the digit, starting from 0 on the right.
- Sum the results of the multiplication.

5. Example 2: Convert 157_8 To Base 10

- **Break the number down by powers of 8:**

$157_8 = (1×8^2) + (5×8^1) + (7×8^0)$

  - $1×8^2 = 1×64 = 64$
  - $5×8^1 = 5×8 = 40$
  - $7×8^0 = 7×1 = 7$

- **Add the products together:**
  - 64 + 40 + 7 =111
  - So, **157_8** in base 8 is equivalent to **111_{10}** in base 10.

6. Example 3: Convert 745_8 To Base 10

- **Break the number down by powers of 8:**

$745_8 = (7×8^2) + (4×8^1) + (5×8^0)$

  - 7 × 64 = 448
  - 4 × 8 = 32
  - 5 × 1 = 5

- **Add the products together:**
  - 448 + 32 + 5 = 485
  - Thus, **745_8** in base 8 is **485_{10}** in base 10.

Summary

- To convert a number from base 8 to base 10, multiply each digit by the corresponding power of 8 (starting with 8^0 for the rightmost digit).
- Sum the results to obtain the decimal (base 10) equivalent.
- Octal numbers are easier to handle because they only involve powers of 8 and use digits from 0 to 7.

From Base 8 To Base 10

$$7263_8 = 3{,}763_{10}$$

$$\begin{array}{r} \mathbf{7} \\ \underline{\times\ 8} \\ 56 \end{array} \quad + \mathbf{2} = 58$$

$$\begin{array}{r} 58 \\ \underline{\times\ 8} \\ 464 \end{array} \quad + \mathbf{6} = 470$$

$$\begin{array}{r} 470 \\ \underline{\times\ 8} \\ 3760 \end{array} \quad + \mathbf{3} = 3{,}763$$

The image demonstrates another method of converting the octal number **7263_8** to its decimal (base 10) equivalent, **$3{,}763_{10}$**. This method relies on iterative multiplication, where you process each digit of the octal number step-by-step.

Let's go through the detailed steps of this process and explain how it works.

1. Step-By-Step Explanation

Starting with the leftmost digit:

- **Begin with the first digit:**
  - The first digit of the octal number is **7**.
  - Start by multiplying **7** by **8** (since the base is 8): $7 \times 8 = 56$

- **Add the second digit:**
  - The next digit in the octal number is **2**.
  - Take the result from the previous multiplication (**56**) and add **2** to it: $56 + 2 = 58$

- **Multiply by 8 again:**
  - Next, multiply the result (**58**) by **8**: $58 \times 8 = 464$

- **Add the third digit:**
  - The third digit in the octal number is **6**.
  - Add **6** to the result from the previous multiplication: $464 + 6 = 470$

- **Multiply by 8 again:**
  - Multiply the result (**470**) by **8**: $470 \times 8 = 3{,}760$

- **Add the last digit:**
  - The last digit in the octal number is **3**.
  - Add **3** to the result from the previous multiplication: $3{,}760 + 3 = 3{,}763$

Final Decimal Result:

Thus, $\mathbf{7263_8}$ in base 8 is equivalent to $\mathbf{3{,}763_{10}}$ in base 10.

2. General Method For Converting From Base 8 To Base 10 Using Iterative Multiplication

- **Start with the leftmost digit** of the octal number.
- **Multiply by 8**, then **add the next digit**.
- **Repeat** this process: multiply by 8 and add the next digit, until you reach the last digit of the octal number.
- The result after processing the final digit is the decimal equivalent.

3. Example 2: Convert 174_8 To Base 10 Using Iterative Multiplication

- **Start with 1** (the leftmost digit).
  - Multiply by 8: 1×8 = 8

- **Add the second digit, 7**:
  8 + 7 = 15

- **Multiply by 8**:
  15 × 8 = 120

- **Add the third digit, 4**:
  120 + 4 =124

Thus, **174_8** in base 8 is **124_{10}** in base 10.

4. Example 3: Convert 536_8 To Base 10

- **Start with 5**:
  5×8 = 40

- **Add the second digit, 3**:
  40 + 3 = 43

- **Multiply by 8**:
  43 × 8 = 344

- **Add the third digit, 6**:
  344 + 6 = 350

Thus, **536_8** in base 8 is **350_{10}** in base 10.

Summary

- This iterative multiplication method is a quick and efficient way to convert octal numbers to decimal.
- Start from the leftmost digit, multiply by 8, and add the next digit at each step until you process all digits.
- This process is ideal for hand calculations or mental conversions.

From Base 16 To Base 2

1. Nibbles, Hexadecimal, and Binary: How They Relate

Let's break down the concept of a **nibble**, **hexadecimal**, and **binary**, and understand why hexadecimal is often used in computing:

1A. The Nibble Approach

- **Nibble**: A nibble consists of **4 bits** (binary digits). In binary, a nibble can represent numbers from 0 to 15, which requires four binary digits:
    - 0000_2 to 1111_2

- **Example of Nibbles**:
    - 0001_2 represents 1_{10}
    - 1111_2 represents 15_{10}
    - 0110_2 represents 6_{10}
    - 0111_2 represents 7_{10}

The nibble approach is significant because **4 bits** perfectly map to **1 hexadecimal digit**. That makes hexadecimal a compact way of representing binary data.

2. Hexadecimal (Base 16)

Hexadecimal, or **base 16**, is a number system that uses 16 unique symbols:

- **Digits from 0 to 9**: They represent values 0 to 9 (just like in decimal).
- **Letters from A to F**: They represent values 10 to 15.

Thus, a single hexadecimal digit can represent four binary digits (1 nibble).

2A. Base 16 and Base 2 Comparison

Base 16 (Hex)	Base 2 (Binary)	Base 10 (Decimal)
0	0000	0
1	0001	1
2	0010	2
3	0011	3
4	0100	4
5	0101	5
6	0110	6
7	0111	7
8	1000	8
9	1001	9
A	1010	10
B	1011	11
C	1100	12
D	1101	13
E	1110	14
F	1111	15

3. Why Hexadecimal?

Base 16	1	F	6	7
Base 2	0001	1111	0110	0111

Hexadecimal is used because it provides a more human-readable format for binary data. Here's why:

Benefits:

- **Compact Representation**:
  - Binary can get lengthy, and hard to read and manage when dealing with large numbers. Every 4 bits in binary can be replaced by a single hex digit. This makes data easier to interpret.
  - For example:
    - $1011\ 1010_2$ (binary) = BA_{16} (hexadecimal)

- **Ease of Conversion**:
  - Hexadecimal easily converts to binary and vice versa. You can take each hex digit and translate it directly into its 4-bit binary equivalent.
  - Example:
    - $1F67_{16}$ (hexadecimal) can be written as:
      - 1 = 0001_2
      - F = 1111_2
      - 6 = 0110_2
      - 7 = 0111_2
    - So, $1F67_{16}$ = $0001\ 1111\ 0110\ 0111_2$.

- **Used in Modern Computing**:
  - Many programming languages, memory addresses, machine-level code, and error codes represent data in hexadecimal.
  - It's also used in color coding (RGB color values), memory dumps, error messages, and network addresses.
  - For example:
    - **Memory Address**: $0x3F2C$ represents a memory address in hexadecimal, which is much more manageable compared to binary.
    - **MAC Address**: A device's MAC address might look like $AA: BB: CC: 11: 22: 33$, which is in hex format.

4. Example of Hexadecimal Conversion

Let's convert the hexadecimal number **1F67** to binary and then back to decimal.

Step 1: Convert Hex to Binary

- 1 = 0001_2
- F = 1111_2
- 6 = 0110_2
- 7 = 0111_2

Thus, $1F67_{16}$ = $0001\ 1111\ 0110\ 0111_2$.

Step 2: Convert Binary to Decimal

Now we'll convert the binary number $0001\ 1111\ 0110\ 0111_2$ to decimal:

$(1\times2^{12}) + (1\times2^{11}) + (1\times2^{10}) + (1\times2^{9}) + (1\times2^{8}) + (0\times2^{7}) + (1\times2^{6}) + (1\times2^{5}) + (0\times2^{4}) + (0\times2^{3}) + (1\times2^{2}) + (1\times2^{1}) + (1\times2^{0}) = 4096 + 2048 + 1024 + 512 + 256 + 64 + 32 + 4 + 2 + 1 = 8039_{10}$

5. Additional Example

Let's convert $5A3_{16}$ to binary and decimal.

Step 1: Convert Hex to Binary

- 5 = 0101_2
- A = 1010_2
- 3 = 0011_2

So, $5A3_{16}$ = $0101\ 1010\ 0011_2$.

Step 2: Convert Binary to Decimal

Now, convert $0101\ 1010\ 0011_2$ to decimal:

$(0\times2^{11}) + (1\times2^{10}) + (0\times2^{9}) + (1\times2^{8}) + (1\times2^{7}) + (0\times2^{6}) + (1\times2^{5}) + (0\times2^{4}) + (0\times2^{3}) + (0\times2^{2}) + (1\times2^{1}) + (1\times2^{0}) =$ $1024 + 256 + 128 + 32 + 2 + 1 = 1443_{10}$.

Thus, $5A3_{16}$ = 1443_{10}.

Summary Of Key Points

- **Nibble**: A 4-bit binary number.
- **Hexadecimal**: Base-16 number system, where each digit represents 4 bits.
- **Why Use Hex?**: It’s more compact and easier to read/write compared to binary. It's widely used in computing for memory addresses, error codes, and network data representation.

By understanding how binary and hex relate, you can better interpret computer data and perform conversions between number systems easily.

Questions

1. Convert 15(Dec) to base 8(Oct).

(a) 17

(b) 1111

(c) F

(d) 45

Answer: **(a) 17**

Solution: To convert from decimal to octal, divide the number by 8 and keep track of the remainder.

15 ÷ 8 = 1 remainder 7 (1st remainder: 7)

1 ÷ 8 = 0 remainder 1 (2nd remainder: 1)

2. What is a Signal?

Answer:

A signal is a function that conveys information about the behaviour or attributes of some phenomenon. It can be used to encode data and can be either analog or digital. A signal is a gesture or message that people use to communicate with each other. The wave you give a good friend to call her over from across the room and the impulse that transmits your voice through the telephone to your mother are both signals.

3. What is a Bit?

(a) 8 bytes

(b) 1024 bytes

(c) 4 bytes

(d) Binary digit as a single 1 or 0

Answer: **(d) Binary digit as a single 1 or 0**

4. A signal can be used to encode a message. In real-life, Sound, Light and Electricity are all examples of what type of signal?

(a) Binary

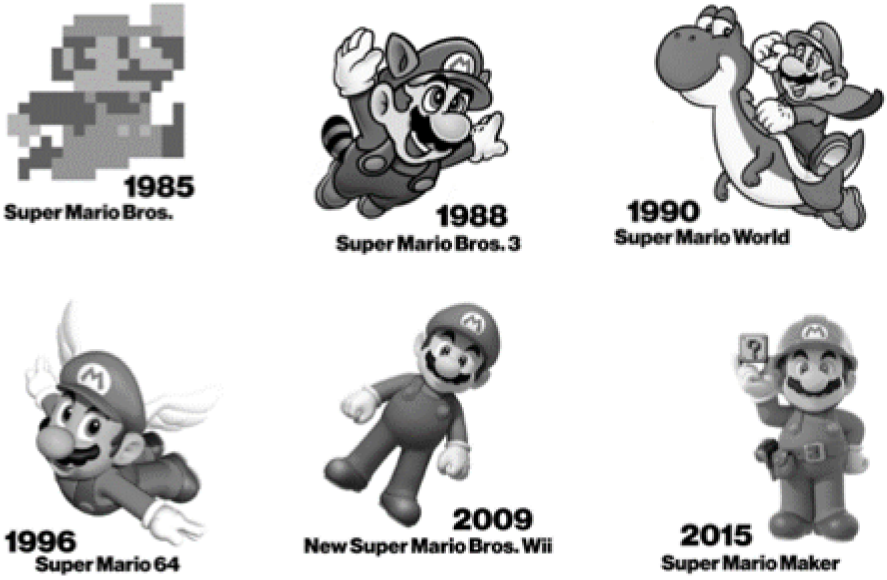
(b) Analog

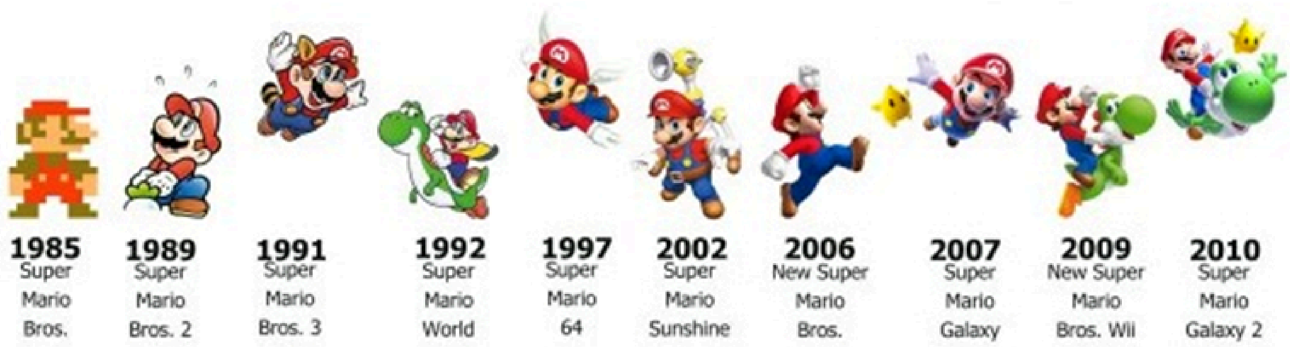
(c) Digital

(d) Finite

Answer: **(b) Analog**

5. With reference to the image below, which of the following statements is true?





(a) Computers represent all information including text and graphics with binary digits (0,1)

(b) The greater the resolution of an image, the greater the number of bits needed for its representation

(c) The more the bits used to represent an image, the better the quality of the image

(d) All of the options

Answer: **(d) All of the options**

6. Convert 58(Dec) to base 16(Hex).

(a) 72

(b) 3A

(c) 97

(d) 111010

Answer: **(b) 3A**

7. What is the range of values for a 4-bit octal system?

(a) 65,536

(b) 256

(c) 16

(d) 4,096

Answer: **(d) 4,096**

Explanation:

- **Octal System:**
  - The octal system is base 8, meaning each digit can range from 0 to 7.
  - A 4-digit octal number would look like this: $xxxx_8$, where each 'x' can be any value from 0 to 7.
- **Range of Values:**
  - The smallest 4-digit octal number is 0000_8, which is 0 in decimal.
  - The largest 4-digit octal number is 7777_8.
- **Convert 7777_8 to Decimal:**
  - Each digit is multiplied by $8^{Position}$ and summed:
    - $7 \times 8^3 + 7 \times 8^2 + 7 \times 8^1 + 7 \times 8^0$ = 7x512 + 7x64 +7x8 +7x1 = 4095
    - So, the range of values is from 0 to 4095.
    - The total count of values from 0 to 4095 is 4096

8. The abbreviation 'MB' means _________ and has the value ___________

(a) MegaBits, 1,024x1024

(b) Mega Bits, 1,000,000

(c) MegaBytes, 1,024x1,000

(d) Mega Bytes, 1,024x1024

Answer: **(d) Mega Bytes, 1,024x1024**

9. Computers work best with Digital signals primarily because?

(a) Digital signals are easier to correct in case of errors

(b) Digital signals can be compressed

(c) Digital signals are easier to work with

(d) All of the options

Answer: **(d) All of the options**

10. Identify 3 differences between Analog and Digital signals.

Answer:

Parameter	Analog Signal	Digital Signal
Definition	A signal for conveying information which is a continuous function of time is known as an analog signal.	A signal which is a discrete function of time, i.e. non-continuous signal, is known as a digital signal.
Typical representation	An analog signal is typically represented by a sine wave function. There are many more representations for analog signals also.	The typical representation of a signal is given by a square wave function.
Signal values	Analog signals use a continuous range of values to represent the data and information.	Digital signals use discrete values (or discontinuous values), i.e. discrete 0 and 1, to represent the data and information.
Signal bandwidth	The bandwidth of an analog signal is low.	The bandwidth of a digital signal is relatively high.
Suitability	The analog signals are more suitable for transmission of audio, video and other information through the communication channels.	The digital signals are suitable for computing and digital electronic operations such as data storage, etc.
Effect of electronic noise	Analog signals are easily affected by electronic noise easily.	The digital signals are more stable and less susceptible to noise than the analog signals.
Accuracy	Due to more susceptibility to noise, the accuracy of analog signals is less.	The digital signals have high accuracy because they are immune from the noise.
Power	Analog signals use more power for data	Digital signals use less power than analog

consumption	transmission.	signals for conveying the same amount of information.
Circuit components	Analog signals are processed by analog circuits whose major components are resistors, capacitors, inductors, etc.	Digital circuits are required for the processing of digital signals whose main circuit components are transistors, logic gates, ICs, etc.
Observational errors	The analog signals give observational errors.	The digital signals do not given observational errors.
Examples	The common examples of analog signals are temperature, current, voltage, voice, pressure, speed, etc.	A common example of a digital signal is the data stored in a computer's memory.
Applications	The analog signals are used in landline phones, thermometers, electric fans, volume knobs of radio, etc.	The digital signals are used in computers, keyboards, digital watches, smartphones, etc.